VAST COSI Driver 2.6 Administrator's Guide

22 April 2025

Always check support.vastdata.com for the latest

Contents

About VAST COSI Driver

VAST COSI Driver Overview

VAST COSI Driver allows container orchestration frameworks such as Kubernetes to dynamically provision object storage on a VAST cluster using the Container Object Storage Interface (COSI).

With dynamic provisioning, you do not need to create container-specific persistent volumes. Instead, you deploy a dynamic provisioner and a YAML class definition that specifies the dynamic provisioner and provisioning options. When an application makes a claim for storage, the Kubernetes framework employs the dynamic provisioner to dynamically create a persistent volume. Later, when a containerized application (via its YAML) references this claim, Kubernetes provides the storage to the container.

VAST COSI Driver is deployed using Helm charts. Check the deployment requirements and follow these steps to deploy the driver and implement dynamic provisioning with VAST COSI Driver.

View per Bucket

VAST COSI Driver creates a VAST Cluster view for each volume or bucket being provisioned. Such a view is automatically deleted when its volume or bucket is deleted.

The views created by VAST COSI Driver can be monitored using VAST Cluster's Web UI or CLI.

To control access permissions for storage exposed through these views, VAST COSI Driver uses a view policy that is attached to the view. This is the view policy specified for the bucket class in the VAST COSI Driver's Helm chart configuration file.

Handling Deletions

VAST COSI Driver gains a substantial performance boost during bucket deletions through the use of *Trash Folder Access*, a feature of VAST Cluster.

Note

VAST Cluster 4.7.0-SP6 or later is required for VAST COSI Driver to be able to use the Trash Folder Access feature.

When deleting a bucket that resides on VAST Cluster 4.7.0-SP6 and later where the Trash Folder Access feature is enabled, the delete request is processed using the VAST REST API's `/folders/delete_folder/` endpoint. This means that the processing takes place locally on the VAST cluster.

SSL Encryption

VAST COSI Driver supports SSL encryption. To secure the connection to the VAST cluster with SSL, follow the guidelines in Configuring SSL Encryption for VAST COSI Driver.

Upgrading VAST COSI Driver

Upgrade VAST COSI Driver

Complete these steps to upgrade VAST COSI Driver:

1. Refresh Helm repository information to see which VAST COSI Driver versions are available for deployment:

   ```
   helm repo update
   ```

2. Update the VAST COSI Driver's Helm chart configuration file (`values.yaml`) to specify the target release of VAST COSI Driver:

   ```
   image:
     csiVastPlugin:
       repository: <repo>
       tag: <new release>
       imagePullPolicy: IfNotPresent
   ```

   Where:

   - `<repo>` is the name of the VAST COSI Driver Helm repository.
   - `<new release>` is the VAST COSI Driver release to be upgraded to.

   For example:

   ```
   image:
     csiVastPlugin:
       repository: vastdataorg/csi
       tag: v2.6.0
       imagePullPolicy: IfNotPresent
   ```

3. Run Helm upgrade:

   ```
   helm upgrade <old release> <repo>/vastcosi -f <filename>.yaml
   ```

   Where:

   - `<old release>` is the VAST COSI Driver release to be upgraded from.
   - `<repo>` is the name of the VAST COSI Driver Helm repository.
   - `vastcosi` is the name of the VAST COSI Driver Helm chart.
   - `<filename>.yaml` is the Helm chart configuration file.

4. Verify the status of the new VAST COSI Driver chart release:

   ```
   helm ls
   ```

Deploying VAST COSI Driver

VAST COSI Driver Requirements

Ensure that your environment meets the following requirements:

- A Kubernetes cluster is up and running.
- A VAST cluster is up and running.
- All nodes of the Kubernetes cluster are networked with the VAST cluster.
- At least one node can communicate with the VAST Cluster management virtual IP.
- A host is available with a Helm client installed, preferably in the VMS network.

Supported Versions

VAST COSI Driver	Kubernetes	VAST Cluster	Helm
2.6.x	1.25 - 1.28.1	4.7.0-SP6 or later with the *Trash Folder Access* feature enabled	3.x.x

Required Permissions

VAST COSI Driver requires root (sysadmin) privileges.

Steps to Deploy VAST COSI Driver

Before you begin, ensure that your environment meets the requirements.

VAST COSI Driver is deployed using Helm charts. A *Helm chart* is an installation template that can be reused to install multiple instances of the software being deployed. Each instance is referred to as a *release*. Helm charts are available from *Helm repositories*.

Steps to deploy VAST COSI Driver include:

1. (Optional) Install CustomResourceDefinitions for COSI.
2. Install CustomResourceDefinitions for snapshots.
3. (Optional) Create a Kubernetes namespace for VAST COSI Driver.

Note

This step is required if you are going to deploy VAST COSI Driver in a Kubernetes namespace other than

`default`. Otherwise, skip to step 4.

4. Configure the VAST cluster.
5. Create a Kubernetes secret with VMS user credentials for VAST COSI Driver.
6. Add the Helm repository that contains the VAST COSI Driver chart.
7. Create a Helm chart configuration file for VAST COSI Driver.
8. Install the Helm chart for VAST COSI Driver.
9. (Optional) Verify the deployment by launching a test application.

Install CRDs for COSI

Run the following commands to install the Custom Resource Definitions (CRDs) for COSI:

```
kubectl create -k github.com/kubernetes-sigs/container-object-storage-interface-api
kubectl create -k github.com/kubernetes-sigs/container-object-storage-interface-controller
```

Install CRDs to Protect Buckets with VAST Snapshots

Complete this step if you are going to use VAST snapshots on your Kubernetes cluster. The Custom Resource Definitions (CRDs) installed in this step are only required for using VAST snapshots.

Run the following commands to install the CRDs for snapshots:

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/v
6.0.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/v
6.0.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/v
6.0.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/v
6.0.1/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/v
6.0.1/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml
```

Create a Kubernetes Namespace for VAST COSI Driver

By default, VAST COSI Driver is deployed to the `default` namespace on the Kubernetes cluster.

If you want to use a different Kubernetes namespace for VAST COSI Driver, create it prior to deployment by running the following command:

```
kubectl create ns <namespace_name>
```

Configure VAST Cluster for VAST COSI Driver

Complete the following steps to make your VAST cluster ready for integration with Kubernetes through the use of VAST COSI Driver:

- Set up virtual IP pools to be used by VAST COSI Driver.

- Configure a view policy to be used for views created by VAST COSI Driver.
- Set up a VMS Manager user to be used by VAST COSI Driver.
- (Optional) Upload your CA-signed SSL certificate to the VAST cluster.
- (Optional): Configure a QoS policy to be associated with views created by VAST COSI Driver.

Set Up Virtual IP Pools

VAST COSI Driver distributes the load among virtual IPs in one or more VAST virtual IP pools.

The virtual IP pool is specified in the Kubernetes bucket class defined in the VAST COSI Driver chart configuration file. The virtual IP pool specified for a bucket class is used when processing read and write operations requested by the application that is using that particular bucket class

To view and manage virtual IP pools in VAST Web UI, log in and choose Network Access -> Virtual IP Pools in the main navigation menu. For more information about VAST Cluster virtual IP pools, see *VAST Cluster Administrator's Guide*.

Configure View Policies

VAST COSI Driver automatically creates a VAST Cluster view for each storage claim being provisioned. These views are controlled using VAST Cluster view policies.

The view policy is specified in the Kubernetes bucket class defined in the VAST COSI Driver chart configuration file. The view policy specified for a bucket class is used when processing read and write operations requested by the application that is using that particular bucket class.

To view and manage existing view policies in VAST Web UI, log in and choose Element Store -> View Policies. For more information about VAST Cluster view policies, see *VAST Cluster Administrator's Guide*.

Set Up a VMS User

Set up a VMS user for VAST COSI Driver to communicate with the VAST Management Service (VMS) via VAST REST API.

You'll need to supply the VMS user credentials in a Kubernetes secret that is specified when creating the VAST COSI Driver's Helm chart configuration file.

VAST COSI Driver can use the default `admin` user that is created by VAST during the initial install, or you can create a new VMS user. The user does not need to have full VMS permissions. The required permissions are Create, View, Edit and Delete permissions in the Logical realm.

Permissions are granted to a VMS user by assigning a role. To view and manage roles in VAST Web UI, log in and choose Administrators -> Roles in the main navigation menu. For more information about managing roles, see *VAST Cluster Administrator's Guide*.

To view and manage VMS users in VAST Web UI, log in and choose Administrators -> Managers in the main navigation menu. For more information about managing VMS Manager users, see *VAST Cluster Administrator's Guide*.

Upload a CA-Signed SSL Certificate to VAST Cluster

If you want to use a Certificate Authority-signed SSL certificate to secure the connection to the VAST cluster, follow the SSL certificate upload procedure in the *VAST Cluster Administrator's Guide* to upload your SSL certificate to the VAST cluster.

For more information about configuring SSL encryption for VAST COSI Driver, see Configuring SSL Encryption for VAST COSI Driver.

Configure a QoS Policy

You can optionally set up a Quality of Service (QoS) policy to be associated with the views that VAST COSI Driver creates. A QoS policy is specified per Kubernetes storage or bucket class configured for the VAST driver.

Notice

This capability requires VAST Cluster 4.6 or later.

To view and manage QoS policies via VAST Web UI, log in and choose Element Store -> QoS Policies. For more information about VAST Cluster QoS policies, see *VAST Cluster Administrator's Guide*.

Create a Kubernetes Secret with VMS User Credentials for VAST COSI Driver

Create a Kubernetes secret to keep VMS user credentials that VAST COSI Driver will use to communicate with the VAST cluster. You need to supply the name of the secret when creating the VAST COSI Driver's Helm chart configuration file.

This Kubernetes secret will be used for all bucket classes defined for the VAST COSI Driver in its Helm chart configuration file. The VAST cluster to connect is specified on the `endpoint` parameter in the configuration file.

1. Create a YAML file with the following content. Note that the VMS user's username and password must be Base64-encoded:

   ```
   apiVersion: v1
   kind: Secret
   metadata:
     name: <secret name>
   type: Opaque
   data:
     username: <VMS user's username>
     password: <VMS user's password>
   ```

2. Apply the YAML file:

   ```
   kubectl apply -f <path to the YAML file>
   ```

Alternatively, you can create a secret with the following command:

```
kubectl create secret generic <secret name> \
    --from-literal=username='<VMS user's username>' \
    --from-literal=password='<VMS user's password>' \
```

Note

If you are creating the secret in a Kubernetes namespace that is different from the namespace used to install the VAST COSI Driver Helm chart, specify the secret's namespace on the command: `-n <secret's namespace>`.

Create a Helm Chart Configuration File for VAST COSI Driver

The driver's Helm chart configuration file lets you override default installation settings provided in the chart with parameters that are specific to your environment.

The configuration file is a YAML file typically named `values.yaml`, although you can use any arbitrary name for it.

Create a Configuration File

Create a YAML file as follows (see also the example below:

Note

For a detailed reference for parameters and values, refer to https://github.com/vast-data/vast-csi/blob/v2.6/charts/vastcosi/values.yaml.

```
secretName: "<secret>"
endpoint: "<endpoint>"
verifySsl: true|false
sslCertsSecretName: "<SSL secret>"

bucketClassDefaults:
    <option 1>
    <option 1>
    ...
    <option n>

bucketClasses:
  <bucket class name 1>:
    <option 1>
    <option 1>
    ...
    <option n>
  <bucket class name 2>:
    <option 1>
    <option 1>
    ...
    <option n>
 ...
  <bucket class name n>:
    <option 1>
    <option 1>
    ...
    <option n>
```

In the YAML file:

1. Set session options:

    - `secretName: "<secret>"` (required): Specify the Kubernetes secret with VMS user credentials to be used by the VAST COSI Driver. The secret must include the VMS user's username and password. For more information, see Create a Kubernetes Secret with VMS User Credentials for VAST COSI Driver.

- `endpoint: "<endpoint>"` (required): Enter the VAST Cluster management hostname.
- `verifySsl: true|false` (optional): Specify `true` to enable SSL encryption for the connection to the VAST cluster. If set to `false` or not specified, SSL encryption is disabled.

  

  Tip

  When enabling SSL encryption, either upload a CA-signed SSL certificate to the VAST cluster (see the procedure in the *VAST Cluster Administrator's Guide*), or supply a self-signed SSL certificate to the driver. The latter can be done either via the `sslCertsSecretName` option, or using `--set-file sslCert` on the Helm chart installation command.

- `sslCertsSecretName: "<SSL secret>"` (optional): Specify the Kubernetes secret that contains the self-signed SSL certificate to be used to secure communications between VAST COSI Driver and the VAST cluster. For more information, see Configuring SSL Encryption for VAST COSI Driver .

2. Set bucket class options:

   - `<bucket class name>` (required): Provide a name to identify the bucket class.

     

     Note

     Define at least one bucket class.

   - `<option 1>...<option n>`: Specify parameters to be used when provisioning storage for buckets with this bucket class. For information on supported options, see Bucket Class Option Reference.

     For each bucket class, the required options are:

     ```
     bucketClasses:
       <bucket class name>:
         storagePath: "<path>"
         vipPool: "<virtual IP pool name>"
         viewPolicy: "<view policy>"
     ```

3. (Optional) Configure registration of the VAST COSI Driver with `kubelet`:

   - `kubeletPath: "<your kubelet root directory>"` (optional): Add this option if you are going to run VAST COSI Driver on a Kubernetes cluster where the kubelet root directory is not `/var/lib/kubelet`.

The following snippet shows a sample configuration file for VAST COSI Driver:

```
secretName: "vast-mgmt"
endpoint: "my.endpoint"

bucketClasses:
  vastdata-bucket:
    storagePath: "/cosi/buckets"
    vipPool: "vippool-1"
    viewPolicy: "cosi"
```

Verify the Configuration File

You can verify the newly created chart configuration file by running the following command:

```
helm template <release name> <repo>/<chart> -f <filename>.yaml -n <namespace>
```

Where:

- `<release name>` identifies the release being deployed.
- `<repo>` is the name of the VAST driver Helm repository.
- `<chart>` is the name of the VAST driver Helm chart (`vastcosi`).
- `<filename>.yaml` is the VAST driver chart configuration file.
- `<namespace>` (optional) determines the Kubernetes namespace to which the release is deployed. If this parameter is not specified, the `default` namespace is used. Otherwise, create a custom namespace prior to installing the VAST driver chart.

For example:

```
helm template cosi-driver vastcosi/vastcosi -f values.yaml
```

Add the Helm Repository for VAST COSI Driver

Add the Helm repository that contains the VAST COSI Driver Helm chart to the list of available repositories:

1. Add the Helm repository for VAST COSI Driver:

   ```
   helm repo add <repo> https://vast-data.github.io/vast-csi
   ```

   Specify any suitable name for `repo`. This name will be used to refer to the VAST COS Driver repository when running Helm commands. For example: *vastcosi*

2. Verify that the repository has been added:

   ```
   helm repo list
   ```

   The output is similar to the following:

   ```
   NAME     URL
   vastcosi https://vast-data.github.io/vast-csi
   ```

Install the VAST COSI Driver Helm Chart

Installing a Helm chart results in deployment of a VAST driver's *release* in your Kubernetes environment. A release is identified with its release name, which you supply during the install.

To install the VAST COSI Driver chart:

1. Refresh Helm repository information:

   ```
   helm repo update
   ```

2. Run the following command to initiate the install:

   ```
   helm install <release name> <repo>/<chart> -f <filename>.yaml -n <namespace> [--set-file s
   slCert=VastCerts/RootCA.crt]
   ```

Where:

- `<release name>` identifies the release being deployed.
- `<repo>` is the name of the VAST COSI Driver Helm repository.
- `<chart>` is the name of the Helm chart to be installed (`vastcosi`).
- `<filename>.yaml` is the Helm chart configuration file for VAST COSI Driver.
- `<namespace>` (optional) determines the Kubernetes namespace to which the release is deployed. If this parameter is not specified, the `default` namespace is used. Otherwise, create a custom namespace prior to installing the Helm chart.
- `--set-file sslCert=VastCerts/RootCA.crt` (optional) specifies the path to a self-signed SSL certificate to secure the connection to the VAST cluster.

For example:

```
helm install cosi-driver vastcosi/vastcosi -f values.yaml
```

The output is similar to the following:

```
NAME: cosi-driver
LAST DEPLOYED: Thu Dec  5 05:24:36 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Thank you for installing vastcosi.

Your release is named cosi-driver.
The release is installed in namespace default

To learn more about the release, try:

  $ helm status -n default cosi-driver
  $ helm get all -n default cosi-driver

<...>
```

3. Verify that the Helm chart has been installed as follows:
   - Check the release status with the following command:

     ```
     helm status -n <namespace> <release name>
     ```

     For example:

     ```
     helm status -n default cosi-driver
     ```

   - Ensure that the release appears in the list of releases:

     ```
     helm list -n <namespace>
     ```

     For example:

     ```
     helm list -n default
     ```

     The output is similar to the following:

     ```
     NAME            NAMESPACE       REVISION        UPDATED
     ```

```
STATUS          CHART           APP VERSION
cosi-driver     default         1               2024-12-01 04:25:33.783236165 +0000 UTC
deployed        vastcosi-0.1.0  2.6.0
```

Verify VAST COSI Driver Deployment

To verify your VAST COSI Driver deployment, provision a bucket on the VAST cluster and prepare credentials for accessing the bucket:

1. Create a Kubernetes YAML configuration file that defines bucket access:

```
kind: BucketAccessClass
apiVersion: objectstorage.k8s.io/v1alpha1
metadata:
 name: sample-bac
driverName: csi.vastdata.com
authenticationType: KEY

kind: BucketAccess
apiVersion: objectstorage.k8s.io/v1alpha1
metadata:
 name: sample-access
spec:
 bucketClaimName: sample-bucket
 bucketAccessClassName: sample-bac
 credentialsSecretName: my-super-secret

kind: BucketClaim
apiVersion: objectstorage.k8s.io/v1alpha1
metadata:
 name: sample-bucket
spec:
 bucketClassName: vastdata-bucket
 protocols:
 - s3
```

2. Apply the bucket access configuration file:

```
kubectl apply -f <filename>.yaml
```

3. Verify that a newly created bucket `vastdata-bucket` is displayed as a view in the VAST Cluster's Web UI.

4. Verify that the newly created user named `vastdata-bucket` is displayed in the VAST Cluster's Web UI.

5. Verify that a newly created Kubernetes secret `my-super-secret` is listed in the output of the following command:

```
kubectl get secret
```

6. Create a Kubernetes YAML configuration file for a pod that uses the newly created Kubernetes secret:

```
apiVersion: v1
kind: Pod
metadata:
 name: awscli
spec:
 containers:
   - name: awscli
     image: amazon/aws-cli
     command: ["sleep"]
     args: ["9999999999"]
     volumeMounts:
       - name: cosi-secrets
         mountPath: /data/cosi
         readOnly: true
```

```
volumes:
- name: cosi-secrets
  secret:
    secretName: my-super-secret
```

7. Apply the pod configuration file:

   ```
   kubectl apply -f <filename>.yaml
   ```

8. Verify the content of the Kubernetes secret:

   ```
   kubectl exec -it awscli cat /data/cosi/BucketInfo
   ```

   The JSON output lists the VAST Cluster endpoint, the access key and the secret key, which can be used to access the newly created bucket.

Configuring SSL Encryption for VAST COSI Driver

Overview

You can secure the connection between VAST COSI Driver and the VAST cluster with SSL encryption as follows:

1. Enable SSL encryption.
2. Do one of the following to install an SSL certificate:
   - If you want to use a Certified Authority-signed SSL certificate, upload it to the VAST cluster. Follow the SSL certificate upload procedure provided in the *VAST Cluster Administrator's Guide*.
   - If you want to use a self-signed SSL certificate, supply it to the VAST driver.

Enable SSL Encryption

By default, SSL encryption is disabled.

To enable SSL encryption:

1. Add the `verifySsl=true` option to the VAST COSI Driver chart configuration file, for example:

   ```
   secretName: "vast-mgmt"
   endpoint: "my.endpoint"
   verifySsl: true
   <...>
   ```

2. Install or upgrade the VAST COSI Driver Helm chart.

Uploading a CA-Signed SSL Certificate to VAST Cluster

Follow the guidelines provided in the *VAST Cluster Administrator's Guide* to upload a CA-signed SSL certificate to the VAST cluster.

Supplying a Self-Signed SSL Certificate to VAST COSI Driver

You can either point to a file that contains a self-signed SSL certificate file, or specify an existing Kubernetes secret that contains the certificate. These two methods are mutually exclusive.

Do either of the following:

- Install or upgrade the VAST COSI Driver Helm chart with the `--set-file sslCert=<path to certificate file>` option specified, for example:

  ```
  helm install cosi-driver vastcosi/vastcosi -f values.yaml --set-file sslCert=<path to cert
  ificate file>
  ```

OR

- Create a Kubernetes secret with the SSL certificate and specify the secret using the `sslCertsSecretName` option in the VAST COSI Driver Helm chart configuration file:

  1. Create a Kubernetes secret that contains the SSL certificate, for example:

     ```
     kubectl create secret generic vast-ca --from-file=ca-bundle.crt=<path to certificate fi
     le>
     ```

  2. Specify the newly created secret on the `sslCertsSecretName` option in the configuration file, for example:

     ```
     secretName: "vast-mgmt"
     endpoint: "my.endpoint"
     verifySsl: true
     sslCertsSecretName: "vast-ca"
     <...>
     ```

  3. Install or upgrade the VAST driver Helm chart (without specifying `--set-file sslCert`).

Replacing a Self-Signed SSL Certificate for VAST COSI Driver

Choose either of the following, depending on how you supplied the old self-signed SSL certificate:

- If you supplied the old SSL certificate using the `--set-file sslCert` option on the Helm chart install or upgrade command:

  - Upgrade the Helm chart with `--set-file sslCert` pointing to the new SSL certificate file. For example:

    ```
    helm upgrade cosi-driver vastcosi/vastcosi -f values.yaml --set-file sslCert=<path to ne
    w certificate>
    ```

OR

- If the old SSL certificate was supplied via `sslCertsSecretName` in the VAST COSI Driver Helm chart configuration file:

1. Create a new Kubernetes secret with the new SSL certificate:

   ```
   kubectl create secret generic vast-ca-new --from-file=ca-bundle.crt=<path to new certif
   icate file>
   ```

2. Ensure that the new SSL certificate is specified on the `sslCertsSecretName` option in the VAST COSI Driver Helm chart configuration file:

   ```
   secretName: "vast-mgmt"
   endpoint: "my.endpoint"
   verifySsl: true
   sslCertsSecretName: "vast-ca-new"
   <...>
   ```

3. Upgrade the Helm chart, for example:

   ```
   helm upgrade cosi-driver vastcosi/vastcosi -f values.yaml
   ```

Removing a Self-Signed SSL Certificate from VAST COSI Driver

Choose either of the following, depending on how you supplied the self-signed SSL certificate:

- If you used `--set-file sslCert` to supply the SSL certificate:
  - Upgrade the Helm chart without the `--set-file sslCert` option specified. For example:

    ```
    helm upgrade cosi-driver vastcosi/vastcosi -f values.yaml
    ```

OR

- If the old SSL certificate was supplied via `sslCertsSecretName`:
  1. Remove the `sslCertsSecretName` option from the VAST COSI Driver Helm chart configuration file.
  2. Upgrade the Helm chart (without specifying `--set-file sslCert`), for example:

     ```
     helm upgrade cosi-driver vastcosi/vastcosi -f values.yaml
     ```

Provisioning Object Storage with VAST COSI Driver

VAST COSI Driver Workflow

VAST COSI Driver lets you use the Kubernetes API to create and manage S3 buckets, and also to manage access to the buckets. After a bucket is provisioned, end users can perform object operations on the bucket.

Note

COSI buckets can be provisioned on the default VAST Cluster tenant only.

To provision a bucket with VAST COSI Driver:

1. In the VAST COSI Driver chart configuration file, specify a `Bucket Class` that provides parameters and attributes to be used when provisioning buckets through VAST COSI Driver. You can use a predefined bucket class provided with VAST COSI Driver, or you can define one or more custom bucket classes.
2. Create and apply a Kubernetes YAML configuration file with the following definitions:
   - A `BucketClaim`, which is a request to create a bucket of the bucket class specified for the driver.
   - A `BucketAccessClass`, which provides various options and settings for bucket access requests made through the driver.
   - A `BucketAccess` request, which contains credentials that can be used to access the bucket.

Define a Bucket Class

A bucket class (`BucketClass`) acts as a container for various parameters and attributes that VAST COSI Driver uses when creating buckets.

You define a bucket class in the VAST COSI Driver chart configuration file.

A minimal bucket class would specify a virtual IP pool, a storage path, and a view policy, for example:

```
endpoint: <IP>
secretName: <secret>
<...>

bucketClasses:
  vastdata-bucket:
    vipPool: cosi_vippool
    storagePath: /buckets
    viewPolicy: my_s3_policy
<...>
```

Optionally, you can specify parameters that allow you to:

- Determine whether to keep or delete the bucket and the associated VMS user when the bucket object is deleted:

  ```
  deletionPolicy: "Delete|Retain"
  ```

- For an SSL-protected VAST cluster, enable use of HTTPS when creating the bucket endpoint:

  `scheme: "http|https"`

- Set up features and capabilities that can be configured for the bucket using the VAST REST API's `/api/v1/views/` endpoint of your VAST cluster.

For more information about options that can be specified for a bucket class, see Bucket Class Option Reference.

Bucket Class Option Reference

You specify bucket class options in the Helm chart configuration file created for VAST COSI Driver during initial deployment.

Basic Options

Option in Helm chart configuration file	Description
`deletionPolicy: "Delete|Retain"`	Determines the action to be taken when the bucket object is deleted: • `Delete` (default). Delete the underlying bucket and associated VMS user. • `Retain`. Do not delete the underlying bucket and associated VMS user.
`scheme: "http"|"https"`	Determines whether to use HTTP (default) or HTTPS when creating the bucket endpoint. If the VAST Cluster is equipped with SSL, set it to `https`.
`storagePath: "<path>"`	The storage path within VAST Cluster to be used when provisioning storage for COSI buckets. VAST COSI Driver will automatically create a VAST Cluster view for each bucket being provisioned. **Caution** You can specify '/' as the `<path>`. This option is required when defining a bucket class in the Helm chart configuration file.
`viewPolicy: "<policy name>"`	The name of the VAST Cluster view policy to be assigned to VAST Cluster views created by VAST COSI Driver. A view policy defines access settings for storage exposed through a VAST Cluster view. For more information, see Configure View Policies. All view policies used with VAST COSI Driver must have the same security flavor.

Option in Helm chart configuration file	Description
	If you are going to use VAST COSI Driver with VAST Cluster 4.6 or later, a view policy set for a bucket class must belong to the same VAST Cluster tenant as the virtual IP pool(s) specified for that bucket class. This option is required when defining a bucket class in the Helm chart configuration file.
`vipPool: "<virtual IP pool name>"`	The name of the virtual IP pool to be used when provisioning storage for COSI buckets. For more information, see Set Up Virtual IP Pools. If you are going to use VAST COSI Driver with VAST Cluster 4.6 or later, a virtual IP pool that you specify for a bucket class must belong to the same VAST Cluster tenant as the view policy used for this bucket class. This option is required when defining a bucket class in the Helm chart configuration file.

VAST Cluster's `/api/v1/views/` Options

In addition to the VAST COSI Driver options listed above, VAST COSI Driver can pass through parameters that are accepted by the VAST REST API's `/api/v1/views/` endpoint of your VAST cluster. For the endpoint parameter reference, see VAST REST API documentation.

Define a Bucket Claim

A bucket claim (`BucketClaim`) creates a bucket.

A minimal example of a bucket claim would include the bucket name, bucket class, and protocol specification, for example:

```
kind: BucketClaim
apiVersion: objectstorage.k8s.io/v1alpha1
metadata:
  name: sample-bucket
spec:
  bucketClassName: vastdata-bucket
  protocols:
  - s3
```

Define a Bucket Access Class

A bucket access class (`BucketAccessClass`) is a set of common properties that VAST COSI Driver uses to set up bucket access.

A minimal bucket access class would include the driver name and authentication type, as shown below:

```
kind: BucketAccessClass
apiVersion: objectstorage.k8s.io/v1alpha1
metadata:
  name: sample-bac
driverName: csi.vastdata.com
```

```
authenticationType: KEY
```

Define Bucket Access

Bucket access (`BucketAccess`) represents a combination of parameters necessary to get access to a particular bucket.

A minimal bucket access block would include the name of the bucket claim, the name of the bucket access class, and the name of the Kubernetes secret that contains the credentials required for accessing the bucket, for example:

```
kind: BucketAccess
apiVersion: objectstorage.k8s.io/v1alpha1
metadata:
  name: sample-access
spec:
  bucketClaimName: sample-bucket
  bucketAccessClassName: sample-bac
  credentialsSecretName: sample-access-secret
```

After you apply the configuration file with the bucket access definition, verify that the Kubernetes secret exists and can be attached to a Kubernetes pod:

```
kubectl get secret
<...>
NAME                                TYPE                DATA   AGE
sample-access-secret                Opaque              1      11s
```


VAST