

SSD Management

This section provides an overview to how KumoScale can be used to manage your SSDs with examples for how to perform various activities using custom resource files.

SSD Operations using Custom Resource Definitions

To modify the configuration of an SSD, KumoScale software provides an **SSD Operations CRD**, a template of which is in `kumoscale.kioxia.com_v1_ssoperation_cr.yaml`. The table below shows the parameters supported in the SSD Operations CRD. Using the CRD with the **kubectl create**, **apply**, or **delete** commands, you can:

- [Add an SSD to an SSD group \(pool\)](#)
- [Remove an SSD from an SSD group \(pool\)](#)
- [Remove an SSD that is missing](#)
- [Evict data on an SSD](#) and migrate it to another
- [Upgrade firmware on an SSD](#)

To perform any of the operations, you will need to:

1. Make a copy of `KumoScale_Operator/ks-config-operator/samples/kumoscale_v1_ssoperation_cr.yaml` for editing, and save to a separate directory For example, `deploy/crds/myapp_ssoperation_cr.yaml`.
2. Update your CRD, in this example we use `myapp_ssoperation_cr.yaml`, with values for the parameters listed below:

SSD Operations Parameter Name	Description	Optional/Required
name	The configuration name; must comply with the Name field as defined in KumoScale Field Types .	Required
storageNode	The name of the storage node the SSD is on. The name of storageNode can be obtained by issuing the command kubectl get storagenodes	Required
ssd	The name of the SSD returned in the SSD section when issuing the command kubectl describe storagenodes	Required
operation	The operation to perform. It should be equal to one of the following: AddToGroup , EnableAllocation , Evict , FirmwareUpgrade , RemoveFromGroup , RemoveMissing .	Required
firmwareSecret	The name of the secret file. This is explained in Upgrade firmware on an SSD	Required when operation is FirmwareUpgrade

3. Use **kubectl** with the appropriate command to create, modify, or delete the SSD:

To create a new SSD:

```
kubectl create -f myapp_ssoperation_cr.yaml
```

To modify an existing SSD:

```
kubectl apply -f myapp_ssoperation_cr.yaml
```

To delete an existing SSD:

```
kubectl delete -f myapp_ssoperation_cr.yaml
```

The following sections provide examples of how to use the CRD for the supported operations.

Add an SSD to an SSD Group

To add an SSD, called **ssd1** to a group on **storagenode21**, we will reference **myapp_addssdtogroup_cr.yaml**, a customized copy of the template provided with KumoScale (**kumoscale_v1_ssoperation_cr.yaml**).

1. Edit the CRD **myapp_addssdtogroup_cr.yaml**:

```
kubectl edit -f myapp_addssdtogroup_cr.yaml
```

2. Update and save the file with the **name** (**addssd1**) and required parameters **ssd** (**ssd1**), **storageNode** (**storagenode21**) and **operation** (**AddToGroup**):

```
apiVersion: kumoscale.kioxia.com/v1
kind: SSDOperation
metadata:
  name: addssd1
spec:
  storageNode: storagenode21
  ssd: ssd1
operation: AddToGroup
```

3. Add the **ssd1** to the group:

```
kubectl create -f myapp_addssdtogroup_cr.yaml
```

The pool's allocated capacity can be modified as needed.

4. Confirm the changes were successful with one of the following:

```
kubectl get ssdoperations addssd1 -o wide

kubectl describe ssdoperations addssd1
```

Remove an SSD from an SSD Group

To remove an SSD from the group to which it is currently attached on **storagenode21**, we will reference **myapp_removefromgroup_cr.yaml**, a customized copy of the template provided with KumoScale (**kumoscale_v1_ssdoperation_cr.yaml**).

1. Edit the CRD yaml:

```
kubectl edit -f myapp_removefromgroup_cr.yaml
```

2. Update and save the file with the **name (removessd1)** and required parameters **ssd (ssd1)**, **storageNode (storagenode21)** and **operation (RemoveFromGroup)**:

```
apiVersion: kumoscale.kioxia.com/v1
kind: SSDOperation
metadata:
  name: removessd1
spec:
  storageNode: storagenode21
  ssd: ssd1
operation: RemoveFromGroup
```

3. Remove **ssd1** from the group to which it is attached:

```
kubectl apply -f myapp_removefromgroup_cr.yaml
```

The pool's allocated capacity will be modified as needed.

4. Confirm the changes were successful with one of the following:

```
kubectl get ssdoperations removessd1 -o wide

kubectl describe ssdoperations removessd1
```

Remove an SSD with the State of Missing

When the state of an SSD is equal to **MISSING**, you need to remove it from the storage node. This can happen after ejecting an SSD from a server's drive bay or from a tray of SSDs.

To remove an SSD, **ssd1**, that is missing from **storagenode21**, we will reference **myapp_removemissd_cr.yaml**, a customized copy of the template provided with KumoScale (**kumoscale_v1_ssdoperation_cr.yaml**).

1. Edit the CRD myapp_removemissdyaml:

```
kubectl edit -f myapp_removemissd_cr.yaml
```

2. Update and save the file with the **name (removemissd1)** and required parameters **ssd (ssd1)**, **storageNode (storagenode21)** and **operation (RemoveMissing)**:

```
apiVersion: kumoscale.kioxia.com/v1
kind: SSDOperation
metadata:
  name: removemissd1
spec:
  storageNode: storagenode21
  ssd: ssd1
operation: RemoveMissing
```

3. Remove **ssd1** from the storage node:

```
kubectl apply -f myapp_removemissd_cr.yaml
```

4. Confirm the changes were successful with one of the following:

```
kubectl get ssdoperations removemissd1 -o wide

kubectl describe ssdoperations removemissd1
```

Evict SSD

If you need to evict an SSD from a group, you need to migrate any attached volumes to another SSD. To do this, you need to stop allocation on the first SSD, and enable allocation on the second SSD. The example below shows how to evict **ssd1** from **storagenode21** using **myapp_evictssd1_cr.yaml** and allocate its volumes to **ssd2** using **myapp_allocatessd2_cr.yaml**, copies of the template provided with KumoScale (**kumoscale_v1_ssdoperation_cr.yaml**)

1. Edit the CRD **myapp_evictssd1_cr.yaml**:

```
kubect1 edit -f myapp_evictssd1_cr.yaml
```

2. Update and save the file with the **name (evictssd1)** and required parameters **ssd (ssd1)**, **storageNode (storagenode21)** and **operation (Evict)**:

```
apiVersion: kumoscale.kioxia.com/v1
kind: SSDOperation
metadata:
  name: evictssd1
spec:
  storageNode: storagenode21
  ssd: SSD1
  operation: Evict
```

3. Evict ssd1 from the storage node:

```
kubect1 apply -f myapp_evictssd1_cr.yaml
```

4. Confirm the changes were successful with one of the following:

```
kubect1 get ssdoperations evictssd1 -o wide

kubect1 describe ssdoperations evictssd1
```

5. Edit the CRD **myapp_allocatessd2_cr.yaml** (you could also edit the CRD in steps 1-3):

```
kubect1 edit -f myapp_allocatessd2_cr.yaml
```

6. Update and save the file with the **name (enablealloc2)** and required parameters **ssd (ssd2)**, **storageNode (storagenode21)** and **operation (EnableAllocation)**:

```
apiVersion: kumoscale.kioxia.com/v1
kind: SSDOperation
metadata:
  name: enablealloc2
spec:
  storageNode: storagenode21
  ssd: ssd2
  operation: EnableAllocation
```

7. Allocate volumes to ssd2:

```
kubect1 apply -f myapp_allocatessd2_cr.yaml
```

8. Confirm the changes were successful with one of the following:

```
kubect1 get ssdoperations enablealloc2 -o wide

kubect1 describe ssdoperations enablealloc2
```

Upgrade Firmware on an SSD

To update firmware on an SSD you will need to :

- 1. [Create the firmware secret.](#)
- 2. [Update and apply changes to the SSD Operations](#).custom resource file.

This is detailed in the following sections.

Step 1. Create the Firmware Secret

To create the firmware secret, KumoScale software provides a **Firmware Upgrade Secret CRD**, a template of which is in **KumoScale_Operator/ks-config-operator/deploy/firmwareupgrade-secret_cr.yaml**. The table below shows the parameters supported in the CRD.

Firmware Secret Parameter	Description	Optional/ Required
name	The configuration's name; must comply with the name field as defined in KumoScale Field Types .	Required
URL	The path (ftp, http, https) where the firmware is located.	Required
Username	Username for the ftp site (base64 encoded).	Required for ftp
Password	Password for the ftp site (base 64 encoded).	Required for ftp

The following steps show how to specify the location and credentials for an ftp site hosting the firmware, using a firmware upgrade secret called `myapp_firmwareupgrade-secret_cr.yaml`, a customized copy of the template provided with KumoScale (`kumoscale_v1_firmwareupgrade-secret_cr.yaml`).

1. Edit the CRD `myapp_firmwareupgrade-secret_cr.yaml`:

```
kubectl edit -f myapp_firmwareupgrade-secret_cr.yaml
```

2. Update and save the file with the **name** (`secret-ftpfw`) and required parameters **url** (`ftp://##.###.###.###/ftp/upload/madeupname`), **username** (`myusername` base64 encoded) and **password** (`superstrongpassword` base64 encoded):

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-ftpfw
namespace: default
type: Opaque
stringData:
  url: ftp://##.###.###.###/ftp/upload/madeupname
data:
  username: <base64 encoded my_username>
  password: <base64 encoded superstrongpassword>
```

3. Create the firmware upgrade secret:

```
kubectl create -f myapp_firmwareupgrade-secret_cr.yaml
```

Note: If you need to update the secret - use `kubectl replace` (not `apply`), otherwise sensitive data will be shown when using `kubectl get secret -o yaml`.

Step 2. Update and Apply the SSD Operations CRD

Using the firmware upgrade secret created in [Create the Firmware Secret](#), upgrade the firmware on `ssd1` using `myapp_fwupgradessd1_cr.yaml`, a customized copy of the SSD Operations template provided with KumoScale (`kumoscale_ssdoperation_cr.yaml`).

1. Edit the CRD `myapp_fwupgradessd1yaml`:

```
kubectl edit -f myapp_ fwupgradessd1_cr.yaml
```

2. Update and save the file with the **name** (`upgradefw`) and values for required parameters **ssd** (`ssd1`), **storageNode** (`storagenode21`), **operation** (`FirmwareUpgrade`), and **firmwareSecretName** (`secret-ftpfw`):

```
apiVersion: kumoscale.kioxia.com/v1
kind: SSDOperation
metadata:
  name: upgradefw
spec:
  storageNode: storagenode21
  ssd: ssd1
  operation: FirmwareUpgrade
parameters:
  firmwareSecretName: secret-ftpfw
```

3. Upgrade the firmware:

```
kubectl apply -f myapp_fwupgradessd1_cr.yaml
```

4. Validate the changes were successful with one of the following commands:

```
kubectl get ssdoperations upgradefw -o wide

kubectl describe ssdoperations upgradefw
```

Next: [Telemetry](#)

