IBM Spectrum Scale CSI

Release 2.4.0

Mar 15, 2023

Contents

CHAPTER 1

Table of Contents

1.1 Introduction

1.1.1 IBM Spectrum Scale

IBM Spectrum Scale is a clustered file system providing concurrent access to a single file system or set of file systems from multiple nodes. The nodes can be SAN attached, network attached, a mixture of SAN attached and network attached, or a shared nothing cluster configuration. This enables high performance access to this common set of data to support a scale-out solution or to provide a high availability platform.

IBM Spectrum Scale has many features beyond common data access including data replication, policy based storage management, and multi-site operations. You can create a cluster of AIX® nodes, Linux nodes, Windows server nodes, or a mix of all three. IBM Spectrum Scale can run on virtualized instances providing common data access in environments, leverage logical partitioning, or other hypervisors. Multiple IBM Spectrum Scale clusters can share data within a location or across wide area network (WAN) connections.

Please refer to IBM Spectrum Scale Documentation for more information on this product

1.1.2 IBM Spectrum Scale CSI Driver

The IBM Spectrum Scale Container Storage Interface (CSI) driver allows IBM Spectrum Scale to be used as persistent storage for stateful application running in Kubernetes clusters. Through this CSI Driver, Kubernetes persistent volumes (PVs) can be provisioned from IBM Spectrum Scale. Thus, containers can be used with stateful microservices, such as database applications (MongoDB, PostgreSQL etc), web servers (nginx, apache), or any number of other containerized applications needing provisioned storage.

Features

Static provisioning Ability to use existing directories as persistent volumes

Lightweight dynamic provisioning Ability to create directory-based volumes dynamically

Fileset-based dynamic provisioning Ability to create fileset-based volumes dynamically

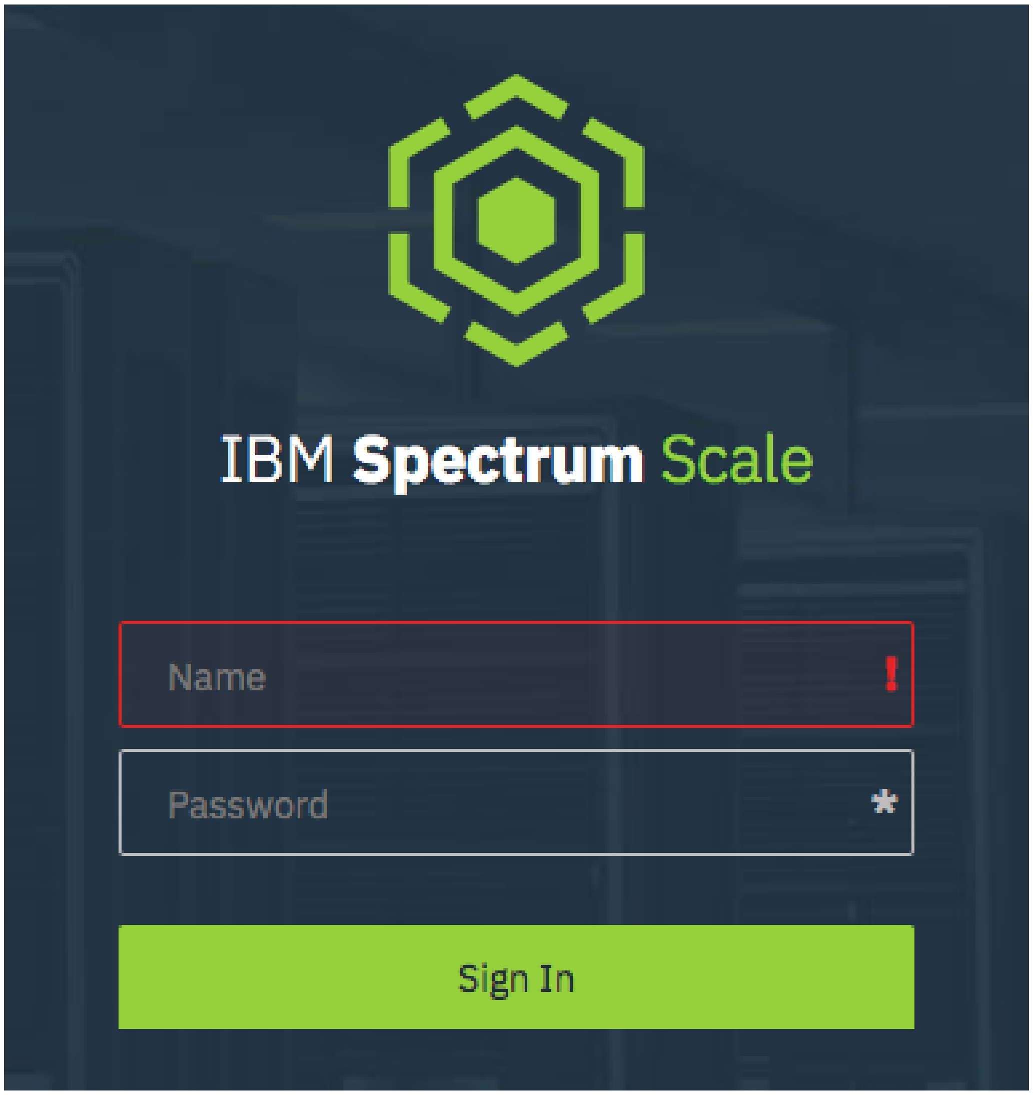
Multiple file systems support Volumes can be created across multiple file systems

Remote mount support Volumes can be created on a remotely mounted file system

1.2 Getting Started

1.2.1 Prerequisites

1. Ensure the Spectrum Scale GUI is running by pointing your browser to the GUI IP address:



If you do not see a login or on-screen instructions, review the GUI Documentation here.

2. Create a `CsiAdmin` group account.

```
export USERNAME="SomeUser"
export PASSWORD="SomePassword"
/usr/lpp/mmfs/gui/cli/mkuser ${USERNAME} -p ${PASSWORD} -g CsiAdmin
```

Tip: If the user already exists, use `chuser` command to add the group to the existing user

3. Create a Kubernetes secret for the `CsiAdmin` user:

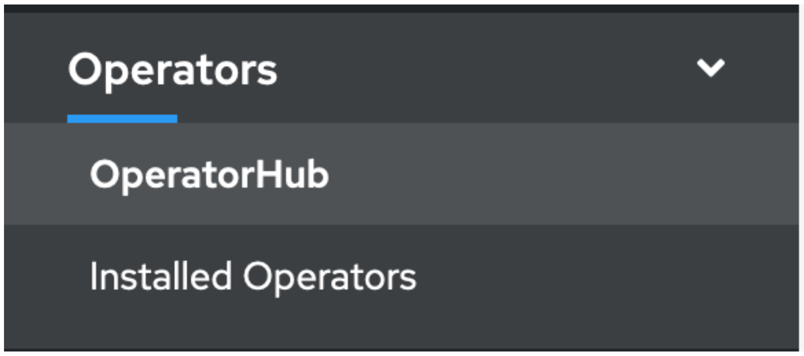
```
export USERNAME_B64=$(echo $USERNAME | base64)
export PASSWORD_B64=$(echo $PASSWORD | base64)

# Set the following to the target namespace to deploy the operator in.
export OPERATOR_NAMESPACE="SomeNamespace"

cat << EOF > /tmp/csisecret.yaml
apiVersion: v1
data:
  password: ${PASSWORD_B64}
  username: ${USERNAME_B64}
kind: Secret
type: Opaque
metadata:
  name: csisecret    # This should be in your CSIScaleOperator definition
  namespace: ${OPERATOR_NAMESPACE}
  labels:
    app.kubernetes.io/name: ibm-spectrum-scale-csi-operator # Used by the operator to␣
↪detect changes, set on load of CR change if secret matches name in CR and namespace.
EOF


kubectl create -f /tmp/csisecret.yaml
rm -f /tmp/csisecret.yaml
```

1.2.2 Installing the Operator

The recommended method of deploying/managing the IBM Spectrum Scale CSI Plugin is through the use of Operators. The IBM Spectrum Scale CSI Operator can be installed from the OperatorHub with Operator Lifecycle Manager (OLM). OLM is part of the Operator Framework. For more information, see: How to install an Operator from OperatorHub

OpenShift

1. Log into the OpenShift Console. On the right sidebar, under **"Operators"**, click **"OperatorHub"**



2. Search **"IBM Spectrum Scale CSI"** and click to **"Install"** to install the Operator.

Tip: Some operators may have multiple icons appear in OperatorHub. We recommend to filter on "Certified" Operators.

3. Validate the options for the operator and click **"Subscribe"** to complete the install of the Operator.

Kubernetes

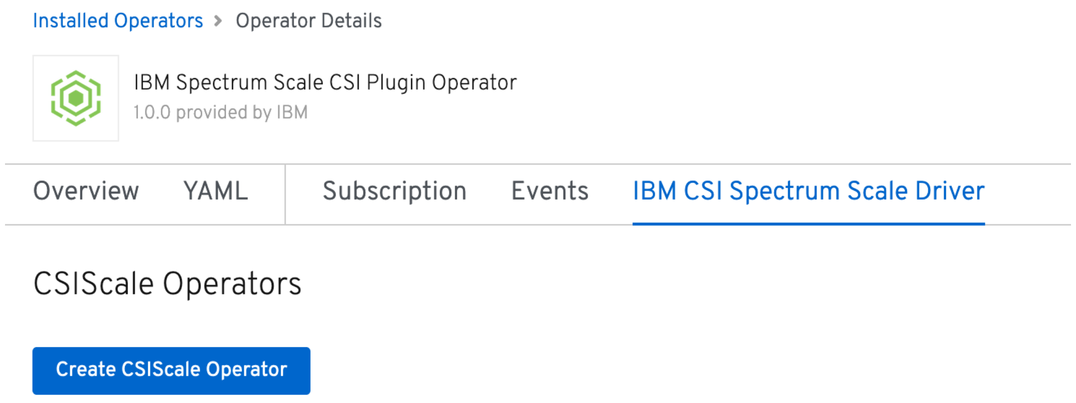
1. Navigate to ibm-spectrum-scale-csi-operator and follow the instructions that appear when you click on the **"Install"** button.

1.2.3 Deploying the Driver

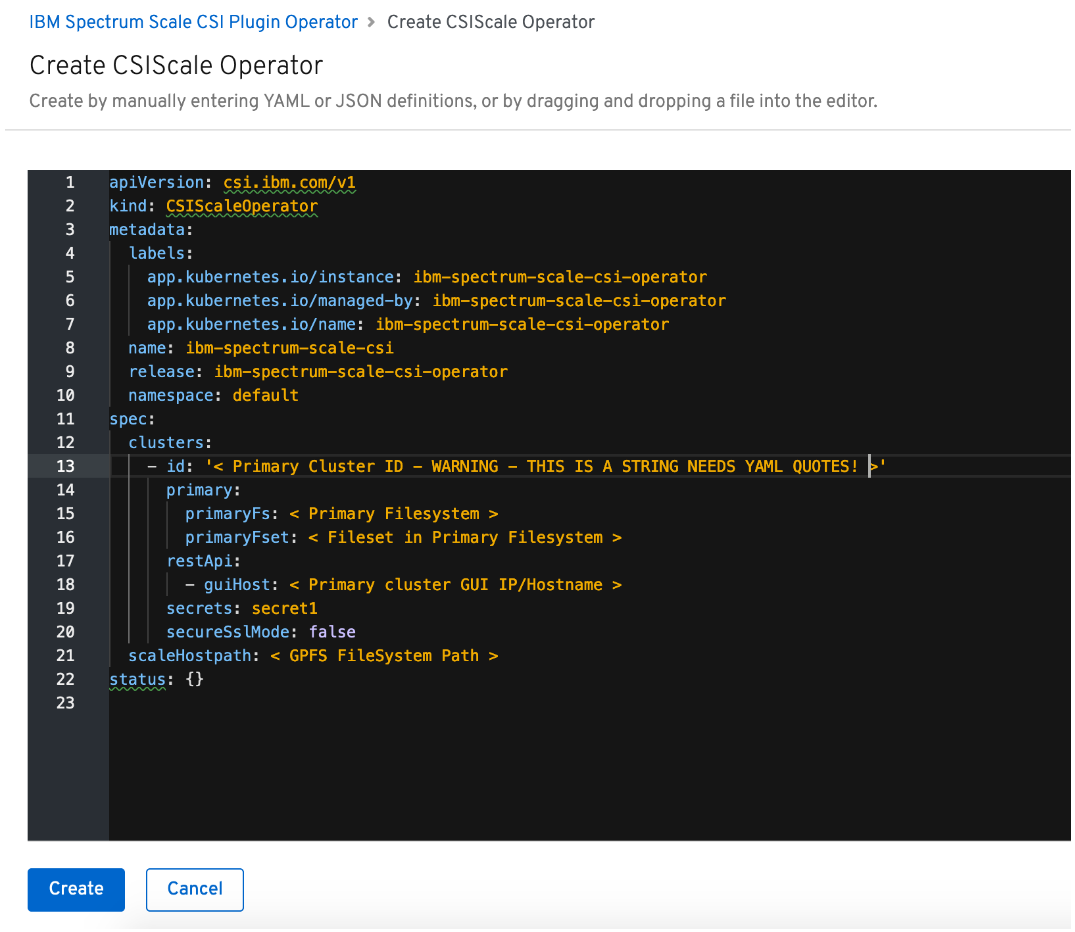
Before you can deploy the driver, you need to modify the Custom Resource (CR) and set the properties matching your IBM Spectrum Scale install.

OpenShift

1. To deploy the driver, select the **"IBM Spectrum Scale CSI Driver"** tab and click **"Create CSIScale Operator"**


Installed Operators > Operator Details
IBM Spectrum Scale CSI Plugin Operator
1.0.0 provided by IBM
Overview
YAML
Subscription
Events
IBM CSI Spectrum Scale Driver
CSIScale Operators
Create CSIScale Operator

2. Modify the Custom Resource (CR) to match your running IBM Spectrum Scale properties, then click **"Create"**.


IBM Spectrum Scale CSI Plugin Operator > Create CSIScale Operator
Create CSIScale Operator
Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor.
Create
Cancel

```yaml
apiVersion: csi.ibm.com/v1
kind: CSIScaleOperator
metadata:
  labels:
    app.kubernetes.io/instance: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/managed-by: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/name: ibm-spectrum-scale-csi-operator
  name: ibm-spectrum-scale-csi
  release: ibm-spectrum-scale-csi-operator
  namespace: default
spec:
  clusters:
    - id: '< Primary Cluster ID - WARNING - THIS IS A STRING NEEDS YAML QUOTES! >'
      primary:
        primaryFs: < Primary Filesystem >
        primaryFset: < Fileset in Primary Filesystem >
      restApi:
        - guiHost: < Primary cluster GUI IP/Hostname >
      secrets: secret1
      secureSslMode: false
  scaleHostpath: < GPFS FileSystem Path >
status: {}
```

For a complete sample of valid CR options, see csiscaleoperators.csi.ibm.com.cr.yaml

Kubernetes

TODO

1.3 Maintenance

1.3.1 Secrets

The IBM Spectrum Scale CSI Driver leverages secrets to store API authentication. In the event of an authentication going stale the user will need to update the secret in kubernetes.

Updating a Secret

Note: For OpenShift environments, replace `kubectl` with `oc`

Due to *ansible-operator* constraints when updating a secret *kubectl apply* and *kubectl edit* are not usable at this time. To update the secret and have the operator apply it, please follow the folowing steps:

1. Edit the *json* or *yaml* defining your secret to have the updated authentication information.

   . . . code-block:: bash export SECRET_NAME="mysecret" export NAMESPACE="ibm-spectrum-scale-csi-driver"

   # Note if you still have a json or yaml file you can just edit that. kubectl get secret -n ${NAMESPACE} ${SECRET_NAME} -o yaml > secret.yaml

   # Edit the contents of secret.yaml to be up to date.

2. Ensure the secret has the correct labelling. If the label is not set the operator will not trigger.

   . . . code-block:: yaml

   metadata:

   labels: app.kubernetes.io/name: ibm-spectrum-scale-csi-operator

3. Delete the old secret and apply the updated secret configuration.

   . . . code-block:: bash

   kubectl delete secret -n ${NAMESPACE} ${SECRET_NAME} kubectl apply -f secret.yaml

After running the fresh apply you should see the *spec.trigger* field increment if the secret was sucessfully created. The process may then be monitored in operator logs.

Additionally, if the operator's custom resource was deployed before the secrets were created the above process may be leveraged to start the operator without deleting the Custom Resource.

1.4 Troubleshooting

1.4.1 operator-sdk

Removing Stuck Operator [operator-sdk/issue/2094]

In cases where deleting the operators `Custom Resource` fails, the following can be executed:

```
# This may need to be customized in OLM environments:
NAMESPACE=ibm-spectrum-scale-csi-driver
kubectl get CSIScaleOperator  -n ${NAMESPACE} -o json | jq '.items[].metadata.
↪finalizers=[] | .items[].status.conditions=[]' > tmp.json
kubectl  apply -f tmp.json
rm -f temp.json
```

Typically this happens when deleting the `Custom Resource Definition` before removing all of the `Custom Resources`.

For more details on this check the following operator-sdk/issue/2094.

1.5 Developers

This section is to help those interested in contributing to the project.

1.5.1 Clone and Build

Clone

Clone down the repository. This repository needs to be accessible in your `GOPATH`. The examples below utilize the `root` user with `GOPATH=/root/go`

```
# Set up some helpful variables
export GOPATH="/root/go"
export IBM_DIR="$GOPATH/src/github.com/IBM"

# Ensure the dir is present then clone.
mkdir -p ${IBM_DIR}
cd ${IBM_DIR}
git clone https://github.com/IBM/ibm-spectrum-scale-csi.git
```

Warning: Due to current constraints in golang, relative paths are not supported. You **must** clone this repository under your `GOPATH`.

Build

Note: Builds requires `docker` 17.05 and later.

Operator

The operator build requires `operator-sdk`.

Tip: To assist in proper configuration of the build environment, a playbook is provided. `ansible-playbook ${IBM_DIR}/ibm-spectrum-scale-csi/tools/ansible/dev-env-playbook.yaml`

1. Navigate to the `operator` directory and use `operator-sdk` to build the operator container image.

```
# IBM_DIR is defined in the previous steps
export REPO_DIR="${IBM_DIR}/ibm-spectrum-scale-csi"
export OPERATOR_DIR="${REPO_DIR}/operator"
cd ${OPERATOR_DIR}

export GO111MODULE="on"

# Build the container image
operator-sdk build ibm-spectrum-scale-csi-operator
```

Driver

1. Navigate to the `driver` directory and use `docker` to build the driver container image.

```
# IBM_DIR is defined in the previous steps
export REPO_DIR="${IBM_DIR}/ibm-spectrum-scale-csi"
export DRIVER_DIR="${REPO_DIR}/driver"
cd ${DRIVER_DIR}

# Build the container image
VERSION="v2.4.0"
docker build -t ibm-spectrum-scale-csi:${VERSION} .

# Save the image into a .tar file
docker save ibm-spectrum-scale-csi:${VERSION} -o ibm-spectrum-scale-csi_${VERSION}.tar
```

1.5.2 Deployment

Container Repository

In order to consume the csi-driver and csi-operator container images built in the previous steps, the images should be pushed to a container repository.

- **Quay.io (recommended)**

  Follow this tutorial to configure quay.io.

  Create two repositories: `ibm-spectrum-scale-csi-operator` and `ibm-spectrum-scale-csi-driver`.

- **Docker**

  Deploying your own Docker registry is an involved process and outside of the scope of this document.

The documentation will assume that the quay.io path is being used.

Pushing the image

Once you have a repository ready:

```
#
# Configure some variables
#
# VERSION - a tag version for your image
VERSION="v0.0.1"
# MYUSER  - A user or organization for your container registry
MYUSER="<your-user>"

# Authenticate to quay.io
docker login <credentials> quay.io

# Tag and push the operator image
docker tag ibm-spectrum-scale-csi-operator quay.io/${MYUSER}/ibm-spectrum-scale-csi-
↪operator:${VERSION}
docker push quay.io/${MYUSER}/ibm-spectrum-scale-csi-operator:${VERSION}

# Tag and push the driver image
docker tag ibm-spectrum-scale-csi-driver quay.io/${MYUSER}/ibm-spectrum-scale-csi-
↪driver:${VERSION}
docker push quay.io/${MYUSER}/ibm-spectrum-scale-csi-driver:${VERSION}
```

(continues on next page)

(continued from previous page)

```
# OPERATOR_DIR has been defined in previous steps
cd ${OPERATOR_DIR}
# Use a helper script to update your deployment to point at your operator image
ansible-playbook hacks/change_deploy_image.yml --extra-vars "quay_operator_
↪endpoint=quay.io/${MYUSER}/ibm-spectrum-scale-csi-operator:${VERSION}"
```

Installing the CSI Operator

Note: For OpenShift environments, replace `kubectl` with `oc`

Run the following to deploy the IBM Spectrum Scale CSI operator manually:

```
# OPERATOR_DIR has been defined in the previous steps
kubectl apply -f ${OPERATOR_DIR}/deploy/namespace.yaml
kubectl apply -f ${OPERATOR_DIR}/deploy/operator.yaml
kubectl apply -f ${OPERATOR_DIR}/deploy/role.yaml
kubectl apply -f ${OPERATOR_DIR}/deploy/role_binding.yaml
kubectl apply -f ${OPERATOR_DIR}/deploy/service_account.yaml
kubectl apply -f ${OPERATOR_DIR}/deploy/crds/csiscaleoperators.csi.ibm.com.crd.yaml
```

Installing the CSI Driver

Tip: Before starting the plugin, ensure that any GUI secrets have been added to the appropriate namespace.

A Custom Resource (CR) file is provided csiscaleoperators.csi.ibm.com.cr.yaml. Modify this file to match the properties in your environment.

To start:

```
kubectl apply -f ${OPERATOR_DIR}/deploy/crds/csiscaleoperators.csi.ibm.com.cr.yaml
```

To stop:

```
kubectl delete -f ${OPERATOR_DIR}/deploy/crds/csiscaleoperators.csi.ibm.com.cr.yaml
```

Removing the CSI Operator and Driver

To remove the IBM Spectrum Scale CSI Operator and Driver:

```
# The following removes the csi-driver
kubectl delete -f ${OPERATOR_DIR}/deploy/crds/csiscaleoperators.csi.ibm.com.cr.yaml

# The following removes the csi-operator
kubectl delete -f ${OPERATOR_DIR}/deploy/operator.yaml
kubectl delete -f ${OPERATOR_DIR}/deploy/role.yaml
kubectl delete -f ${OPERATOR_DIR}/deploy/role_binding.yaml
kubectl delete -f ${OPERATOR_DIR}/deploy/service_account.yaml
kubectl delete -f ${OPERATOR_DIR}/deploy/crds/csiscaleoperators.csi.ibm.com.crd.yaml
```

(continues on next page)

(continued from previous page)

```
# The following removes the namespace
kubectl delete -f ${OPERATOR_DIR}/deploy/namespace.yaml
```

This will completely destroy the operator, driver, and all associated resources.

1.5.3 OLM

Using Test Versions of CSV

Due to the nature of Operator Lifecycle Manager (OLM) it is necessary to maintain an application repository to host the most up to date Cluster Service Version (CSV). To assist, two application registries are maintained by the development team:

- Master - https://quay.io/application/ibm-spectrum-scale-dev/ibm-spectrum-scale-csi-master
- Dev - https://quay.io/application/ibm-spectrum-scale-dev/ibm-spectrum-scale-csi-dev

These subscriptions maintain the latest iteration of the CSV on the dev and master branches respectively. To subscribe to these applicaions via OLM, the code repository provides three YAML files:

`tools/olm/operator-source-openshift.yaml`

- Used for both applications on OpenShift.
- Created in the *openshift-marketplace* namespace.

`tools/olm/operator-source-k8s-master.yaml`

- Used for OLM subscription to the master stream in raw k8s.
- Created in the *marketplace* namespace.
- **WARNING** : Currently disabled, as master has some issues for upgrade.

`tools/olm/operator-source-k8s-dev.yaml`

- Used for OLM subscription to the master stream in raw k8s.
- Created in the *marketplace* namespace.

This yaml files should be applied against your Kubernetes or OpenShift cluster:

```
kubectl apply -f <operator-source-____.yaml>
```

Note: For OpenShift environments, replace `kubectl` with `oc`

Testing an in development CSV

While modifying a CSV it is conceivable that a developer would want to test their CSV in a local environment. One method for achieving this is to host the CSV on quay.io.

1. Create a new *Application Repository* in quay.io/new.

Tip: Save the name of this repository, because you'll need it in the next steps.

2. Install helm and helm registry:

```
curl -L https://git.io/get_helm.sh | bash
helm init
cd ~/.helm/plugins/ && git clone https://github.com/app-registry/appr-helm-
↪plugin.git registry
```

3. Create a helm project for your application and push it to quay:

```
# Set your variables
QUAY_REPO_NAME="<Your Repo Name>"
QUAY_USER="<Your Quay Username>"
CHANNEL_NAME="test"

# Create the helm project
cd ~
helm create ${QUAY_REPO_NAME}
cd ${QUAY_REPO_NAME}

# Push to quay
helm registry login quay.io
helm registry push --namespace ${QUAY_USER} quay.io
helm registry push --namespace ${QUAY_USER} --channel ${CHANNEL_NAME} quay.io
```

4. Edit the variables for the test playbook (which will push your csv):

```
vi tools/ansible/olm-test-playbook.yaml
```

5. Deploy using *olm-test-playbook.yaml*, you'll need to set the user name and password:

    6. Install operator-courier.

At this point your application is ready to be subscribed to. Use the following templates for k8s and OpenShift respectively.

Kubernetes subscription template

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: olm-crb
subjects:
- kind: ServiceAccount
  name: default
  namespace: kube-system
roleRef:
  kind: ClusterRole
  name: cluster-admin
  apiGroup: ""

---
apiVersion: operators.coreos.com/v1
kind: OperatorSource
```

(continues on next page)

(continued from previous page)

```yaml
metadata:
  name: ibm-spectrum-scale-csi
  namespace: marketplace
spec:
  type: appregistry
  endpoint: https://quay.io/cnr
  registryNamespace:  {{ QUAY_USER }}

---
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: operator-group
  namespace: marketplace
spec:
  targetNamespaces:
  - marketplace

---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: oper-sub
  namespace: marketplace
spec:
  channel: stable
  name: {{ REPO_NAME }}
  source: {{ REPO_NAME }}
  sourceNamespace: marketplace
```

OpenShift subscription template

```yaml
apiVersion: operators.coreos.com/v1
kind: OperatorSource
metadata:
  name: ibm-spectrum-scale
  namespace: openshift-marketplace
spec:
  type: appregistry
  endpoint: https://quay.io/cnr
  registryNamespace:  {{ QUAY_USER }}
  displayName: "CSI Scale Operator"
  publisher: "IBM"
```

1.5.4 Cert Process

Creating the Pull Request

1. Fork **'https://github.ibm.com/IBMPrivateCloud/charts'_**.
2. Clone the forked repository .
3. From the root dir (of this project) execute the following:

```
export CHARTS=<Local Chart Repo Root>

cp -R -L cloudpak/ ${CHARTS}
cd ${CHARTS}/stable

git checkout -b  ibm-spectrum-scale-csi-operator-bundle
git add ibm-spectrum-scale-csi-operator-bundle
git commit -S -m "Some message"
git push origin ibm-spectrum-scale-csi-operator-bundle
```

4. Follow standard Pull Request procedures.