IBM® Storage

Red Hat OpenShift deployment with IBM Storage Enabler for Containers

Version 1 Release 1

IBM Storage Team

Contents

Executive summary

Most organizations will soon be operating in a hybrid multicloud environment. Container technology will help drive this rapid evolution from applications and data anchored on-premises in siloed systems, to applications and data easily moving when and where needed to gain the most insight and advantage.

IBM® Storage unifies traditional and container-ready storage, and provides cloud-native agility with the reliability, availability, and security to manage enterprise containers in production. As clients scale containerized applications beyond experimental or departmental use, IBM's award-winning storage solutions enable mission-critical infrastructure that delivers shared-storage operational efficiency, price-performance leadership, and container data protection.

Through integration with the automation capabilities of Kubernetes and IBM Cloud™ Paks, IBM enables IT infrastructure and operations to improve developer speed and productivity, while delivering data reduction, disaster recovery, and data availability with enterprise storage.

IBM Storage for Red Hat OpenShift Container Platform is a comprehensive, container-ready solution that includes all of the elements and expertise needed for implementing the technologies that are driving business in the 21st century.

Scope

This document is intended to show the proof of concept environment created in a lab environment, while keeping the Red Hat OpenShift cluster prerequisites and requirements in mind. The document describes the setup of IBM Storage Enabler for Containers for the installation used in the lab environment. For guidance on configuring worker nodes for storage connectivity, IBM Storage, and IBM Spectrum® Connect, refer to IBM Storage Solutions for IBM Cloud Private Blueprint.

This document does not illustrate the Red Hat OpenShift installation component.

The setup instructions provided in this document are not a replacement for any official documentation released by Red Hat OpenShift or Linux operating system providers.

Prerequisites

The lab setup of OpenShift was created as a user-provisioned infrastructure using Red Hat OpenShift Enterprise V3.11.

Create ubiquity namespace

The ubiquity namespace is the default installation location for IBM Storage Enabler for Containers. The **patch** command (Example 1) is run so that the *DaemonSet* creates pods on the master nodes, which is required for master controlled attach and detach operations.

Example 1 The patch command and ubiquity namespace

```
oc new-project ubiquity
oc project ubiquity
oc patch namespace ubiquity -p '{"metadata": {"annotations":
{"openshift.io/node-selector": ""}}}'
```

Set up Red Hat OpenShift SecurityContextConstraint

The deployment of IBM Storage Enabler for Containers requires setting up `SecurityContextConstraint` (SCC) to support Red Hat OpenShift, as shown in Table 1 and Example 2.

Table 1 Parameters to set up SCC

Service Account	SecurityContextConstraint	Description
default	anyuid	Default service account
ubiquity	anyuid	Ubiquity server
ubiquity-k8s-provisioner	anyuid	Kubernetes Provisioner
ubiquity-helm-hook	anyuid	Helm Hooks
ubiquity-k8s-flex	privileged	Kubernetes Flex Driver

Example 2 Setting up SCC

```
oc adm policy add-scc-to-user anyuid system:serviceaccount:ubiquity:default
oc adm policy add-scc-to-user anyuid system:serviceaccount:ubiquity:ubiquity
oc adm policy add-scc-to-user anyuid
system:serviceaccount:ubiquity:ubiquity-k8s-provisioner
oc adm policy add-scc-to-user anyuid
system:serviceaccount:ubiquity:ubiquity-helm-hook
oc adm policy add-scc-to-user privileged
system:serviceaccount:ubiquity:ubiquity-k8s-flex
```

Download the IBM Storage Enabler for Containers helm chart

Use the following steps to download and extract the IBM Storage Enabler for Containers helm chart:

1. Download the helm chart from GitHub:

   ```
   curl -O
   https://raw.githubusercontent.com/IBM/charts/master/repo/stable/ibm-storage-ena
   bler-for-containers-1.0.1.tgz
   ```

2. Extract the TAR files:

   ```
   tar xzvf ibm-storage-enabler-for-containers-1.0.1.tgz
   ```

Download the helm executable

Use the following steps to download, extract, and initialize the helm executables:

1. Download the helm TAR file:

   ```
   curl -O
   https://storage.googleapis.com/kubernetes-helm/helm-v2.9.1-linux-amd64.tar.gz
   ```

2. Extract the helm files:

   ```
   tar xzvf linux-amd64/helm-v2.9.1-linux-amd64.tar.gz
   ```

3. Move the helm executable into `$PATH`:

   ```
   mv helm /usr/local/bin
   ```

4. Initialize helm for use:

   ```
   helm init --client-only
   ```

Update the templates/_helpers.tpl helm chart

Use the following steps to update the helm chart `templates/_helpers.tpl` file:

1. Open the `templates/_helpers.tpl` file to edit:

   ```
   vi ibm-storage-enabler-for-containers/templates/_helpers.tpl
   ```

2. Comment out the following lines (highlighted in red) by adding a hash sign (#) to the beginning of the line:

   ```
   {{- define "ibm_storage_enabler_for_containers.securityContext" -}}
   securityContext:
     privileged: false
     allowPrivilegeEscalation: false
     readOnlyRootFilesystem: false
     runAsNonRoot: false
     runAsUser: 0
     capabilities:
       drop:
      - ALL
       add:
   #    - CHOWN
   #    - FSETID
   #    - FOWNER
   #    - SETGID
   #    - SETUID
   #    - DAC_OVERRIDE
   {{- end -}}
   ```

3. Add the following new lines:

   ```
   {{- define "ibm_storage_enabler_for_containers.securityContext-k8s-flex" -}}
   securityContext:
     privileged: false
     allowPrivilegeEscalation: true
     readOnlyRootFilesystem: false
     runAsNonRoot: false
     runAsUser: 0
     capabilities:
       drop:
      - ALL
   ```

```
        add:
       - CHOWN
       - FSETID
       - FOWNER
       - SETGID
       - SETUID
       - DAC_OVERRIDE
   {{- end -}}
```

4. Save the file:

   ```
   :x
   ```

Update the templates/ubiquity-k8s-flex-daemonset.yaml helm chart

Use the **`sed`** command to replace `securityContext` with `securityContext-k8s-flex` to match the definition shown in step 3 previously:

```
sed -i -e 's/securityContext/securityContext-k8s-flex/g'
ibm-storage-enabler-for-containers/templates/ubiquity-k8s-flex-daemonset.yaml
```

Create a Secret for the IBM Storage Enabler for Containers database

There are two options for creating the Secret.

Option 1: Create the Secret from a YAML file

> **Important Note:** Data values need to be encoded as base64 for entry into the YAML file. The output from base64 will be entered in the `Data.username`, `Data.Password`, and `Data.dbname` fields.

Example: `echo -n ubiquity | base64`

Output: `dWJpcXVpdHk=`

File: `ubiquity-db-credentials-secret.yml`

```
apiVersion: v1
kind: Secret
metadata:
  name: ubiquity-db-credentials
  namespace: ubiquity
  labels:
    product: ibm-storage-enabler-for-containers
    # Ubiquity database credentials needed for ubiquity and
    # ubiquity-db deployments
    # Attention:
    #      These settings will configure the database properties
    #      during the initial installation.
    #      If these settings need to be changed after installation,
    #.     configure them manually in the ubiqutiy-db postgres as well.
type: Opaque
```

```
data:
   # Base64-encoded username to be set for the ubiquity-db deployment.
   username: "dWJpcXVpdHk="
   # Base64-encoded password to be set for the ubiquity-db deployment.
   password: "dWJpcXVpdHk="
   # Base64-encoded database name ("dWJpcXVpdHk=" base64 is "ubiquity")
   # to be created for the ubiquity-db deployment.
   dbname: "dWJpcXVpdHk="
```

Apply the new secret:

```
oc create -n ubiquity -f ubiquity-db-credentials-secret.yml
```

Option 2: Create the Secret using the oc command

Use the following command:

```
oc create secret generic ubiquity-db-credentials --from-literal=dbname=ubiquity
--from-literal=username=ubiquity --from-literal=password=ubiquity  -n ubiquity
```

Configure IBM Block Storage (IBM Spectrum Connect)

This section describes the installation sequence and configuration procedure followed for IBM Storage Enabler for Containers for IBM Block Storage.

Option 1: Create a Secret for IBM Spectrum Connect from a YAML file

Important Note: Data Values need to be encoded as base64 for entry into the YAML file.

Example: `echo -n ubiquity | base64`

Output: `dWJpcXVpdHk=`

File: `scbe-credentials.yaml`

```
apiVersion: v1
kind: Secret
metadata:
  name: scbe-credentials
  namespace: ubiquity
  labels:
    product: ibm-storage-enabler-for-containers
# Spectrum Connect (previously known as SCBE) credentials needed for ubiquity,
# ubiquity-k8s-provisioner deployments, And ubiquity-k8s-flex daemonset.
type: Opaque
data:
   # Base64-encoded username defined for the IBM Storage Enabler
   # for Containers interface in Spectrum Connect.
   username: "dWJpcXVpdHk="
   # Base64-encoded password defined for the IBM Storage Enabler
   # for Containers interface in Spectrum Connect.
   password: "cGFzc3cwcmQ="
```

Apply the new secret:

```
oc create -n ubiquity -f scbe-credentials.yaml
```

Option 2: Create a Secret for IBM Spectrum Connect using the oc command

Use the following command:

```
oc create secret generic scbe-credentials --from-literal=username=ubiquity
--from-literal=password=passw0rd  -n ubiquity
```

Update the values.yaml file for IBM Block Storage (IBM Spectrum Connect)

> **Note:** The full `values.yaml` file is shown in “Appendix A” on page 12. Only relevant values that need to be modified are captured here and highlighted in red.

File:

```
backend: spectrumConnect
spectrumConnect:
  connectionInfo:
    fqdn: flashse-scb.flashse-ad.ibm.local
    port: 8440
    existingSecret: scbe-credentials

  backendConfig:
    instanceName: openshift
    defaultStorageService: ibmc-block-gold
    newVolumeDefaults:
      fsType: ext4
      size: 1

  storageClass:
    storageService: ibmc-block-gold
    fsType: ext4

ubiquityDb:
  dbCredentials:
    existingSecret: ubiquity-db-credentials

  persistence:
    useExistingPv: false
    pvName: ibm-ubiquity-db
    pvSize: 20Gi

    storageClass:
      storageClassName: ibmc-block-gold
      existingStorageClass:
      defaultClass: false
```

Configure for IBM Spectrum Scale

This section describes the installation sequence and configuration procedure used for IBM Storage Enabler for Containers for IBM Spectrum Scale.

Option 1: Create the Secret for IBM Spectrum Scale from a YAML file

Important Note: Data values need to be encoded as base64 for entry into the YAML file.

Example: `echo -n ubiquity | base64`

Output: `dWJpcXVpdHk=`

File: `spectrumscale-credentials.yaml`

```
apiVersion: v1
kind: Secret
metadata:
  labels:
    product: ibm-storage-enabler-for-containers
  name: spectrumscale-credentials
  namespace: ubiquity
type: Opaque
data:
  password: YWRtaW4wMDE=
  username: YWRtaW4=
```

Apply the new secret:

```
oc create -n ubiquity -f spectrumscale-credentials.yaml
```

Option 2: Create the Secret for IBM Spectrum Scale using the oc command

Use the following commands:

```
oc create secret generic spectrumscale-credentials --from-literal=username=admin
--from-literal=password=admin001  -n ubiquity
```

Update the values.yaml file for IBM Spectrum Scale

Note: The full `values.yaml` file is shown in “Appendix B” on page 15. Only relevant values that need to be modified are captured here and highlighted in Red.

File:

```
backend: spectrumScale

spectrumScale:
  connectionInfo:
    # IP\FQDN and port of Spectrum Scale RESTful API server.
    fqdn: scale-node-01.flashse-ad.ibm.local
    port: 443
    # Set this param with an existing Spectrum Scale secret object if one
exist.
    existingSecret: spectrumscale-credentials

  backendConfig:
    # Default Spectrum Scale filesystem to be used.
    defaultFilesystemName: filesystem_1

ubiquityDb:
  dbCredentials:
    existingSecret: ubiquity-db-credentials

  persistence:
    useExistingPv: false
    pvName: ibm-ubiquity-db
    pvSize: 20Gi
    storageClass:

      storageClassName:
      existingStorageClass: ibmc-file-gold
      defaultClass: false
```

Install IBM Storage Enabler for Containers

This section describes the installation sequence and configuration procedures used for IBM Storage Enabler for Containers.

Option 1: Install IBM Storage Enabler for Containers using helm tiller

This section describes the installation sequence and configuration procedure followed for IBM Storage Enabler for Containers using helm and tiller with Red Hat OpenShift:

1. Tiller installation in Openshift V3.11:

```
# oc new-project tiller
# oc project tiller
# export TILLER_NAMESPACE=tiller
```

2. Install the tiller services from GitHub:

```
# oc process -f
https://github.com/openshift/origin/raw/master/examples/helm/tiller-template.ya
ml -p TILLER_NAMESPACE="${TILLER_NAMESPACE}" -p HELM_VERSION=v2.9.1 | oc create
-f -
```

3. Wait for tiller to complete the installation:

```
# oc rollout status deployment tiller
```

4. Verify the helm version that was installed:

```
# helm version
Client: &version.Version{SemVer:"v2.9.1",
GitCommit:"20adb27c7c5868466912eebdf6664e7390ebe710", GitTreeState:"clean"}

Server: &version.Version{SemVer:"v2.9.1 ",
GitCommit:"20adb27c7c5868466912eebdf6664e7390ebe710", GitTreeState:"clean"}
```

5. Add `ClusterRolebinding` to `cluster-admin ClusterRole` for the tiller service account:

   ```
   # oc adm policy add-cluster-role-to-user cluster-admin
   "system:serviceaccount:tiller:tiller"
   ```

6. After `` `helm init --client-only` `` is done, you need to use `--upgrade` to use the tiller serviceAccount:

   ```
   # helm init --upgrade --service-account tiller
   ```

7. Repackage the helm chart for installation:

   ```
   # helm package ibm-storage-enabler-for-containers/
   ```

8. Install the helm chart using the updated `values.yaml` file:

   ```
   # oc project ubiquity
   # helm install ./ibm-storage-enabler-for-containers-1.0.1.tgz -f
   ibm-storage-enabler-for-containers/values.yaml --name ubiquity --namespace
   ubiquity
   ```

9. Validate the IBM Storage Enabler for Containers installation:

   ```
   # helm list ubiquity

   NAME REVISION UPDATED STATUS CHART NAMESPACE
   ubiquity 1  Tue Mar 12 13:10:44 2019 DEPLOYED
   storage-enabler-for-containers-1.0.1 ubiquity
   ```

Option 2: Install IBM Storage Enabler for Containers using a helm template

This section describes the installation sequence and configuration procedure followed for IBM Storage Enabler for Containers using helm to create a template with Red Hat OpenShift:

1. Create the helm template using the updated `values.yaml` file:

   ```
   # oc project ubiquity
   # helm template ibm-storage-enabler-for-containers -f
   ibm-storage-enabler-for-containers/values.yaml --name ubiquity --namespace
   ubiquity > ubiquity.yaml
   ```

2. Remove all references to the following component from `ubiquity.yaml`:

   ```
   imagePullSecrets:
    - name: sa-ubiquity
   ```

3. Delete the following sections and their associated content:

   ```
   # Source: ibm-storage-enabler-for-containers/templates/tests/sanity-test.yaml
   apiVersion: v1
   kind: Pod
   metadata:
     name: ubiquity-sanity-test
     annotations:
       “helm.sh/hook”: test-success
     labels:
       app.kubernetes.io/name: ibm-storage-enabler-for-containers
       helm.sh/chart: ibm-storage-enabler-for-containers-1.0.1
       release: ubiquity
       app.kubernetes.io/instance: ubiquity
       app.kubernetes.io/managed-by: Tiller
   ```

```
spec:
  hostNetwork: false
  hostPID: false
  hostIPC: false
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
         - matchExpressions:
             - key: beta.kubernetes.io/arch
                operator: In
                values:
                 - amd64
                 - ppc64le
                 - s390x
  containers:
   - name: sanity-test
      securityContext:
        privileged: false
        allowPrivilegeEscalation: false
        readOnlyRootFilesystem: false
        runAsNonRoot: false
        runAsUser: 0
        capabilities:
          drop:
         - ALL
          add:
         - CHOWN
         - FSETID
         - FOWNER
         - SETGID
         - SETUID
         - DAC_OVERRIDE
      image: “ibmcom/ibm-storage-enabler-for-containers-helm-utils:2.1.0”
      imagePullPolicy: IfNotPresent
      command: [‘/usr/bin/hook-executor’, ‘sanity’]
      env:
       - name: NAMESPACE
          value: ubiquity
       - name: STORAGE_CLASS
          value:
  restartPolicy: Never
  serviceAccountName: ubiquity-helm-hook
---

# Source: ibm-storage-enabler-for-containers/templates/pre-delete-job.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: pre-delete
  annotations:
    “helm.sh/hook”: “pre-delete”
    “helm.sh/hook-delete-policy”: “hook-succeeded”
  labels:
    app.kubernetes.io/name: ibm-storage-enabler-for-containers
    helm.sh/chart: ibm-storage-enabler-for-containers-1.0.1
    release: ubiquity
    app.kubernetes.io/instance: ubiquity
    app.kubernetes.io/managed-by: Tiller
spec:
  template:
    spec:
      hostNetwork: false
      hostPID: false
      hostIPC: false
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
             - matchExpressions:
                 - key: beta.kubernetes.io/arch
                    operator: In
                    values:
                     - amd64
                     - ppc64le
                     - s390x
      containers:
       - name: pre-delete-hook
          securityContext:
            privileged: false
            allowPrivilegeEscalation: false
            readOnlyRootFilesystem: false
            runAsNonRoot: false
            runAsUser: 0
            capabilities:
              drop:
             - ALL
              add:
             - CHOWN
             - FSETID
             - FOWNER
             - SETGID
             - SETUID
             - DAC_OVERRIDE
          image: “ibmcom/ibm-storage-enabler-for-containers-helm-utils:2.1.0”
          imagePullPolicy: IfNotPresent
          command: [“/usr/bin/hook-executor”]
          args: [“predelete”]
          env:
           - name: NAMESPACE
              value: ubiquity
           - name: UBIQUITY_DB_PV_NAME
              value: “ibm-ubiquity-db”
           - name: UBIQUITY_DB_STORAGECLASS
              value:
      restartPolicy: Never
      serviceAccountName: ubiquity-helm-hook
  backoffLimit: 1
```

4. Delete any empty sections from the end of the file:

   Example:

   ```
   # Source:
   ibm-storage-enabler-for-containers/templates/ubiquity-k8s-psp-role-binding.yaml
   ```

5. Apply the template:

   ```
   oc apply -n ubiquity -f ubiquity.yaml
   ```

Appendix A

Contents of the `values.yaml` file to support IBM Block Storage with IBM Spectrum Connect:

```
# ----------------------------------------------------------
# Helm chart to install IBM Storage Enabler for Containers.
# Enables IBM Storage with Kubernetes by implementing Kubernetes Dynamic
Provisioner and FlexVolume.
# IBM Storage Enabler for Containers includes the following main images:
#   - deployment/ubiquity                : A mediator between IBM Storage and k8s
FlexVolume \ Dynamic Provisioner.
#   - deployment/ubiquity-db              : Stores meta-data for the dynamic
provisioned volumes.
#   - deamonset/ubiquity-k8s-flex         : Implements k8s FlexVolume driver.
#   - deployment/ubiquity-k8s-provisioner : Implements k8s Dynamic Provisioner
# ----------------------------------------------------------

# Backend for Provisioner and Flex volume.
# IBM Storage Enabler for Containers supports one of the following backend types:
spectrumConnect OR spectrumScale.
backend: spectrumConnect
# IBM Storage Enabler for Containers supports one of the following backend types:
spectrumConnect OR spectrumScale.
# Select a backend that you intend to use and comment out the other backend
section.
spectrumConnect:
  connectionInfo:
    # IP\FQDN and port of Spectrum Connect server.
    fqdn: flashsse-scb.flashse-ad.ibm.local
    port: 8440
    # Set this param with an existing spectrum connect secret object if one exist.
    existingSecret: scbe-credentials
  backendConfig:
    # A prefix for any new volume created on the storage system.
    instanceName: openshift
    # Default Spectrum Connect storage service to be used, if not specified by the
storage class.
    defaultStorageService: ibmc-block-gold
    newVolumeDefaults:
      # The fstype of a new volume if not specified by the user in the storage
class.
      # File system type. Allowed values: ext4 or xfs.
      fsType: ext4
      # The default volume size (in GB) if not specified by the user when creating
a new volume.
```

```
      size: 1
  ## storageClass parameters for ubiquity-db PVC
  storageClass:
    # Storage Class profile parameter must point to the Spectrum Connect storage
service name.
    storageService: ibmc-block-gold
    # Storage Class filesystem type. Allowed values: ext4 or xfs.
    fsType: ext4
# IBM Storage Enabler for Containers supports one of the following backend types:
spectrumConnect OR spectrumScale.
# Select a backend that you intend to use and comment out the other backend
section.
spectrumScale:
  connectionInfo:
    # IP\FQDN and port of Spectrum Scale RESTful API server.
    fqdn:
    port: 443
    # Set this param with an existing Spectrum Scale secret object if one exist.
    existingSecret:
  backendConfig:
    # Default Spectrum Scale filesystem to be used.
    defaultFilesystemName:
ubiquityDb:
  image:
    repository: ibmcom/ibm-storage-enabler-for-containers-db
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
  nodeSelector: {}
  dbCredentials:
    # Set this param with an existing secret object if one exist. . Note: Do not
use the "postgres" username, because it already exists.
    existingSecret: ubiquity-db-credentials
  # The Helm installation has automatic boot strap of the ubiquity-db volume (PVC
named ibm-ubiquity-db).
  # The boot strap creates a storage class (see details below) and the PVC.
  persistence:
    # Set this param to true if you want to use an existing PV as Ubiquity
database PV.
    # Use it only when you want to upgrade Ubiquity from old version installed by
script to the latest version.
    useExistingPv: false
    # Ubiquity database PV name. For Spectrum Virtualize, Spectrum Accelerate and
Spectrum Scale, use default value "ibm-ubiquity-db".
    # For DS8000 Family, use "ibmdb" instead and make sure
UBIQUITY_INSTANCE_NAME_VALUE value length does not exceed 8 chars.
    pvName: ibm-ubiquity-db
    pvSize: 20Gi
    storageClass:
      # Parameters to create the first storage class that is also to be used by
Ubiquity for ibm-ubiquity-db PVC.
      # Note: The default reclaimPolicy is Delete. Can be changed manually if
needed.
      storageClassName: ibmc-block-gold
      # Set this param with an existing storageclass object if one exist.
      existingStorageClass:
      # Set StorageClass as the default StorageClass. Ignored if
storageClass.create is false.
      defaultClass: false
ubiquity:
  image:
    repository: ibmcom/ibm-storage-enabler-for-containers
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
ubiquityK8sFlex:
  image:
    repository: ibmcom/ibm-storage-flex-volume-for-kubernetes
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
  # By default, the toleration is set to run the Flex DeamonSet on all worker and
master nodes. To define a different toleration, uncomment and apply the relevant
toleration value.
  tolerations: {}
  # Flex log directory. If the default value is changed, make sure that the new
path exists on all the nodes and update the Flex DaemonSet hostpath accordingly.
  flexLogDir: /var/log
ubiquityK8sFlexInitContainer:
  resources: {}
ubiquityK8sFlexSidecar:
  image:
    repository: ibmcom/ibm-storage-flex-volume-sidecar-for-kubernetes
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
ubiquityK8sProvisioner:
  # RBAC and service account are set automatically for the Provisioner.
  image:
    repository: ibmcom/ibm-storage-dynamic-provisioner-for-kubernetes
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
ubiquityHelmUtils:
  image:
    repository: ibmcom/ibm-storage-enabler-for-containers-helm-utils
    tag: "2.1.0"
    pullPolicy: IfNotPresent
# Custom pod security policy. If specified, it is applied to all pods in the
chart.
# New policies cannot be defined. Configure a policy in advance or use existing
ones. Then, attach one or more policies
# to a role or clusterRole, and provide the name for the role or clusterRole.
# Currently, only clusterRole is supported. It will be bound to all
serviceAccounts under the current namespace.
customPodSecurityPolicy:
  # The name of clusterRole that has the required policies attached.
  # Default value for ICP 3.1.1+ is ibm-anyuid-hostpath-clusterrole
  enabled: false
  clusterRole: ibm-anyuid-hostpath-clusterrole
```

```
globalConfig:
  # Log level. Allowed values: debug, info, error.
  logLevel: info
  # SSL verification mode. Allowed values: require (no validation is required) and
verify-full (user-provided certificates).
  # SSL mode is set for all communication paths between
[flex||provisioner]<->ubiquity<->[SpectrumConnect||SpectrumScale].
  sslMode: require
  imagePullSecret:
```

Appendix B

Contents of the `values.yaml` file to support IBM Spectrum Scale:

```
# ---------------------------------------------------------
# Helm chart to install IBM Storage Enabler for Containers.
# Enables IBM Storage with Kubernetes by implementing Kubernetes Dynamic
Provisioner and FlexVolume.
# IBM Storage Enabler for Containers includes the following main images:
#   - deployment/ubiquity                : A mediator between IBM Storage and k8s
FlexVolume \ Dynamic Provisioner.
#   - deployment/ubiquity-db              : Stores meta-data for the dynamic
provisioned volumes.
#   - deamonset/ubiquity-k8s-flex         : Implements k8s FlexVolume driver.
#   - deployment/ubiquity-k8s-provisioner : Implements k8s Dynamic Provisioner
# ---------------------------------------------------------
# Backend for Provisioner and Flex volume.
# IBM Storage Enabler for Containers supports one of the following backend types:
spectrumConnect OR spectrumScale.
backend: spectrumScale
# IBM Storage Enabler for Containers supports one of the following backend types:
spectrumConnect OR spectrumScale.
# Select a backend that you intend to use and comment out the other backend
section.
spectrumConnect:
  connectionInfo:
    # IP\FQDN and port of Spectrum Connect server.
    fqdn:
    port: 8440
    # Set this param with an existing spectrum connect secret object if one exist.
    existingSecret:
  backendConfig:
    # A prefix for any new volume created on the storage system.
    instanceName:
    # Default Spectrum Connect storage service to be used, if not specified by the
storage class.
    defaultStorageService:
    newVolumeDefaults:
      # The fstype of a new volume if not specified by the user in the storage
class.
      # File system type. Allowed values: ext4 or xfs.
      fsType: ext4
      # The default volume size (in GB) if not specified by the user when creating
a new volume.
```

```
      size: 1
  ## storageClass parameters for ubiquity-db PVC
  storageClass:
    # Storage Class profile parameter must point to the Spectrum Connect storage
service name.
    storageService:
    # Storage Class filesystem type. Allowed values: ext4 or xfs.
    fsType: ext4
# IBM Storage Enabler for Containers supports one of the following backend types:
spectrumConnect OR spectrumScale.
# Select a backend that you intend to use and comment out the other backend
section.
spectrumScale:
  connectionInfo:
    # IP\FQDN and port of Spectrum Scale RESTful API server.
    fqdn: scale-node-01.flashse-ad.ibm.local
    port: 443
    # Set this param with an existing Spectrum Scale secret object if one exist.
    existingSecret: spectrumscale-credentials
  backendConfig:
    # Default Spectrum Scale filesystem to be used.
    defaultFilesystemName: filesystem_1
ubiquityDb:
  image:
    repository: ibmcom/ibm-storage-enabler-for-containers-db
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
  nodeSelector: {}
  dbCredentials:
    # Set this param with an existing secret object if one exist. . Note: Do not
use the "postgres" username, because it already exists.
    existingSecret: ubiquity-db-credentials
  # The Helm installation has automatic boot strap of the ubiquity-db volume (PVC
named ibm-ubiquity-db).
  # The boot strap creates a storage class (see details below) and the PVC.
  persistence:
    # Set this param to true if you want to use an existing PV as Ubiquity
database PV.
    # Use it only when you want to upgrade Ubiquity from old version installed by
script to the latest version.
    useExistingPv: false
    # Ubiquity database PV name. For Spectrum Virtualize, Spectrum Accelerate and
Spectrum Scale, use default value "ibm-ubiquity-db".
    # For DS8000 Family, use "ibmdb" instead and make sure
UBIQUITY_INSTANCE_NAME_VALUE value length does not exceed 8 chars.
    pvName: ibm-ubiquity-db
    pvSize: 20Gi
    storageClass:
      # Parameters to create the first storage class that is also to be used by
Ubiquity for ibm-ubiquity-db PVC.
      # Note: The default reclaimPolicy is Delete. Can be changed manually if
needed.
      storageClassName: ibmc-file-gold
      # Set this param with an existing storageclass object if one exist.
      existingStorageClass:
      # Set StorageClass as the default StorageClass. Ignored if
storageClass.create is false.
      defaultClass: false
ubiquity:
  image:
    repository: ibmcom/ibm-storage-enabler-for-containers
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
ubiquityK8sFlex:
  image:
    repository: ibmcom/ibm-storage-flex-volume-for-kubernetes
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
  # By default, the toleration is set to run the Flex DeamonSet on all worker and
master nodes. To define a different toleration, uncomment and apply the relevant
toleration value.
  tolerations: {}
  # Flex log directory. If the default value is changed, make sure that the new
path exists on all the nodes and update the Flex DaemonSet hostpath accordingly.
  flexLogDir: /var/log
ubiquityK8sFlexInitContainer:
  resources: {}
ubiquityK8sFlexSidecar:
  image:
    repository: ibmcom/ibm-storage-flex-volume-sidecar-for-kubernetes
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
ubiquityK8sProvisioner:
  # RBAC and service account are set automatically for the Provisioner.
  image:
    repository: ibmcom/ibm-storage-dynamic-provisioner-for-kubernetes
    tag: "2.1.0"
    pullPolicy: IfNotPresent
  resources: {}
ubiquityHelmUtils:
  image:
    repository: ibmcom/ibm-storage-enabler-for-containers-helm-utils
    tag: "2.1.0"
    pullPolicy: IfNotPresent
# Custom pod security policy. If specified, it is applied to all pods in the
chart.
# New policies cannot be defined. Configure a policy in advance or use existing
ones. Then, attach one or more policies
# to a role or clusterRole, and provide the name for the role or clusterRole.
# Currently, only clusterRole is supported. It will be bound to all
serviceAccounts under the current namespace.
customPodSecurityPolicy:
  # The name of clusterRole that has the required policies attached.
  # Default value for ICP 3.1.1+ is ibm-anyuid-hostpath-clusterrole
  enabled: false
  clusterRole: ibm-anyuid-hostpath-clusterrole
```

```
globalConfig:
  # Log level. Allowed values: debug, info, error.
  logLevel: info
  # SSL verification mode. Allowed values: require (no validation is required) and
verify-full (user-provided certificates).
  # SSL mode is set for all communication paths between
[flex||provisioner]<->ubiquity<->[SpectrumConnect||SpectrumScale].
  sslMode: require
  imagePullSecret:
```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

Redbooks (logo) ®
IBM®
IBM Cloud™
IBM Spectrum®

The following terms are trademarks of other companies:

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

September 2019

ISBN 0738458082

REDP-5564-00