


# Documentation

Search Documentation

⌘K



 **Kubernetes**

 Docker

 Linux

 macOS

 Windows

**Upstream**

Redhat Openshift

Amazon Elastic Kubernetes Service

Google Kubernetes Engine

Azure Kubernetes S



# MinIO Object Storage for Kubernetes

MinIO is an object storage solution that provides an Amazon Web Services S3-compatible API and supports all core S3 features. MinIO is built to deploy anywhere - public or private cloud, baremetal infrastructure, orchestrated environments, and edge infrastructure.

This site documents Operations, Administration, and Development of MinIO deployments on Kubernetes platform for the latest stable version of the MinIO Operator: 5.0.10.

MinIO is released under dual license [GNU Affero General Public License v3.0](#) and [MinIO Commercial License](#). Deployments registered through [MinIO SUBNET](#) use the commercial license and include access to 24/7 MinIO support.

You can get started exploring MinIO features using the [MinIO Console](#) and our `play` server at <https://play.min.io>. `play` is a *public* MinIO cluster running the latest stable MinIO server. Any file uploaded to `play` should be considered public and non-protected. For more about connecting to `play`, see [MinIO Console play Login](#).

## Quickstart: MinIO for Kubernetes

This procedure deploys a Single-Node Single-Drive MinIO server onto Kubernetes for early development and evaluation of MinIO Object Storage and its S3-compatible API layer.

Use the [MinIO Operator](#) to deploy and manage production-ready MinIO tenants on Kubernetes.

## Prerequisites

- An existing Kubernetes deployment where *at least* one Worker Node has a locally-attached drive.
- A local `kubectl` installation configured to create and access resources on the target Kubernetes deployment.
- Familiarity with Kubernetes environments
- Familiarity with using a Terminal or Shell environment

## Procedure

### 1. Download the MinIO Object

Download the MinIO Kubernetes Object Definition

Overview of the MinIO Object YAML

---

```

# Deploys a new Namespace for the MinIO Pod
apiVersion: v1
kind: Namespace
metadata:
  name: minio-dev # Change this value if you want a different namespace name
  labels:
    name: minio-dev # Change this value to match metadata.name
---
# Deploys a new MinIO Pod into the metadata.namespace Kubernetes namespace
#
# The `spec.containers[0].args` contains the command run on the pod
# The `/data` directory corresponds to the `spec.containers[0].volumeMounts`
# That mount path corresponds to a Kubernetes HostPath which binds `/data`
#
apiVersion: v1
kind: Pod
metadata:
  labels:
    app: minio
    name: minio
  namespace: minio-dev # Change this value to match the namespace metadata
spec:
  containers:
  - name: minio
    image: quay.io/minio/minio:latest
    command:
    - /bin/bash
    - -c
    args:
    - minio server /data --console-address :9090
    volumeMounts:
    - mountPath: /data
      name: localvolume # Corresponds to the `spec.volumes` Persistent Volume
  nodeSelector:
    kubernetes.io/hostname: kubealpha.local # Specify a node label associated with the node
  volumes:
  - name: localvolume
    hostPath: # MinIO generally recommends using locally-attached volumes
      path: /mnt/disk1/data # Specify a path to a local drive or volume on the node
      type: DirectoryOrCreate # The path to the last directory must exist

```

The object deploys two resources:

- A new namespace `minio-dev`, and
- A MinIO pod using a drive or volume on the Worker Node for serving data

The MinIO resource definition uses Kubernetes [Node Selectors and Labels](#) to restrict the pod to a node with matching hostname label. Use `kubectl get nodes --show-labels` to view all labels assigned to each node in the cluster.

The MinIO Pod uses a [hostPath](#) volume for storing data. This path *must* correspond to a

Users familiar with Kubernetes scheduling and volume provisioning may modify the `spec.nodeSelector`, `volumeMounts.name`, and `volumes` fields to meet more specific requirements.

## 2. Apply the MinIO Object Definition

The following command applies the `minio-dev.yaml` configuration and deploys the objects to Kubernetes:

```
kubectl apply -f minio-dev.yaml
```

The command output should resemble the following:

```
namespace/minio-dev created
pod/minio created
```

You can verify the state of the pod by running `kubectl get pods` :

```
kubectl get pods -n minio-dev
```

The output should resemble the following:

NAME	READY	STATUS	RESTARTS	AGE
minio	1/1	Running	0	77s

You can also use the following commands to retrieve detailed information on the pod status:

```
kubectl describe pod/minio -n minio-dev
```

```
kubectl logs pod/minio -n minio-dev
```

## 3. Temporarily Access the MinIO S3 API and Console

Use the `kubectl port-forward` command to temporarily forward traffic from the MinIO pod to the local machine:

```
kubectl port-forward pod/minio 9000 9090 -n minio-dev
```

The command forwards the pod ports `9000` and `9090` to the matching port on the local

while active in the shell session. Terminating the session closes the ports on the local machine.

**Note:**

The following steps of this procedure assume an active `kubectl port-forward` command.

To configure long term access to the pod, configure [Ingress](#) or similar network control components within Kubernetes to route traffic to and from the pod. Configuring Ingress is out of the scope for this documentation.

#### 4. Connect your Browser to the MinIO Server

Access the [MinIO Console](#) by opening a browser on the local machine and navigating to `http://127.0.0.1:9090` .

Log in to the Console with the credentials `minioadmin` | `minioadmin` . These are the default [root user](#) credentials.

**MINIO**

# OBJECT STORE



Username

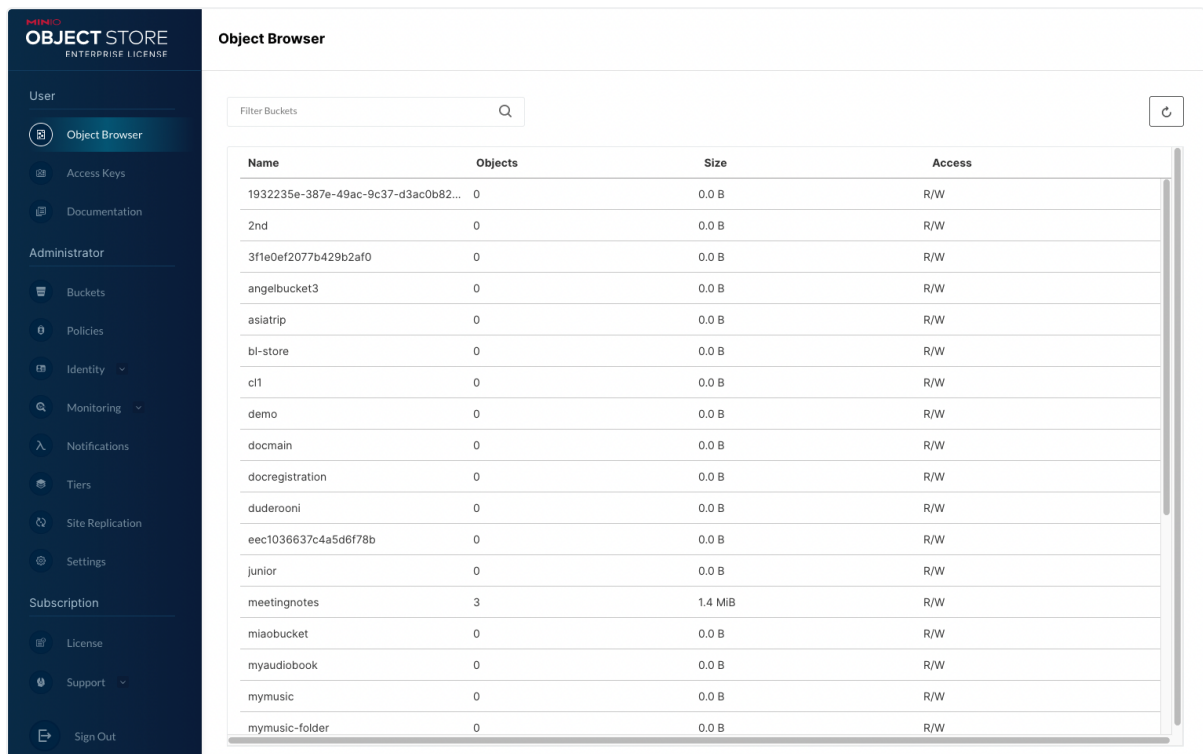


Password

Login

[Use STS](#) →

You can use the MinIO Console for general administration tasks like Identity and Access Management, Metrics and Log Monitoring, or Server Configuration. Each MinIO server includes its own embedded MinIO Console.



For more information, see the [MinIO Console](#) documentation.

## 5. (Optional) Connect the MinIO Client

If your local machine has `mc` installed, use the `mc alias set` command to authenticate and connect to the MinIO deployment:

```
mc alias set k8s-minio-dev http://127.0.0.1:9000 minioadmin minioadmin
mc admin info k8s-minio-dev
```

- The name of the alias
- The hostname or IP address and port of the MinIO server
- The Access Key for a MinIO [user](#)
- The Secret Key for a MinIO [user](#)

## Next Steps

2 Configure Object Retention

---

3 Configure Security

---

4 Deploy MinIO for Production Environments

---

This work is licensed under a Creative Commons Attribution 4.0 International License. 2020-Present, MinIO, Inc.

