Multi-Tenancy for Kubernetes and Lightbits: Implementation Framework

Author: Rob Bloemendal, Principal Solution Consultant
Date: May 2025

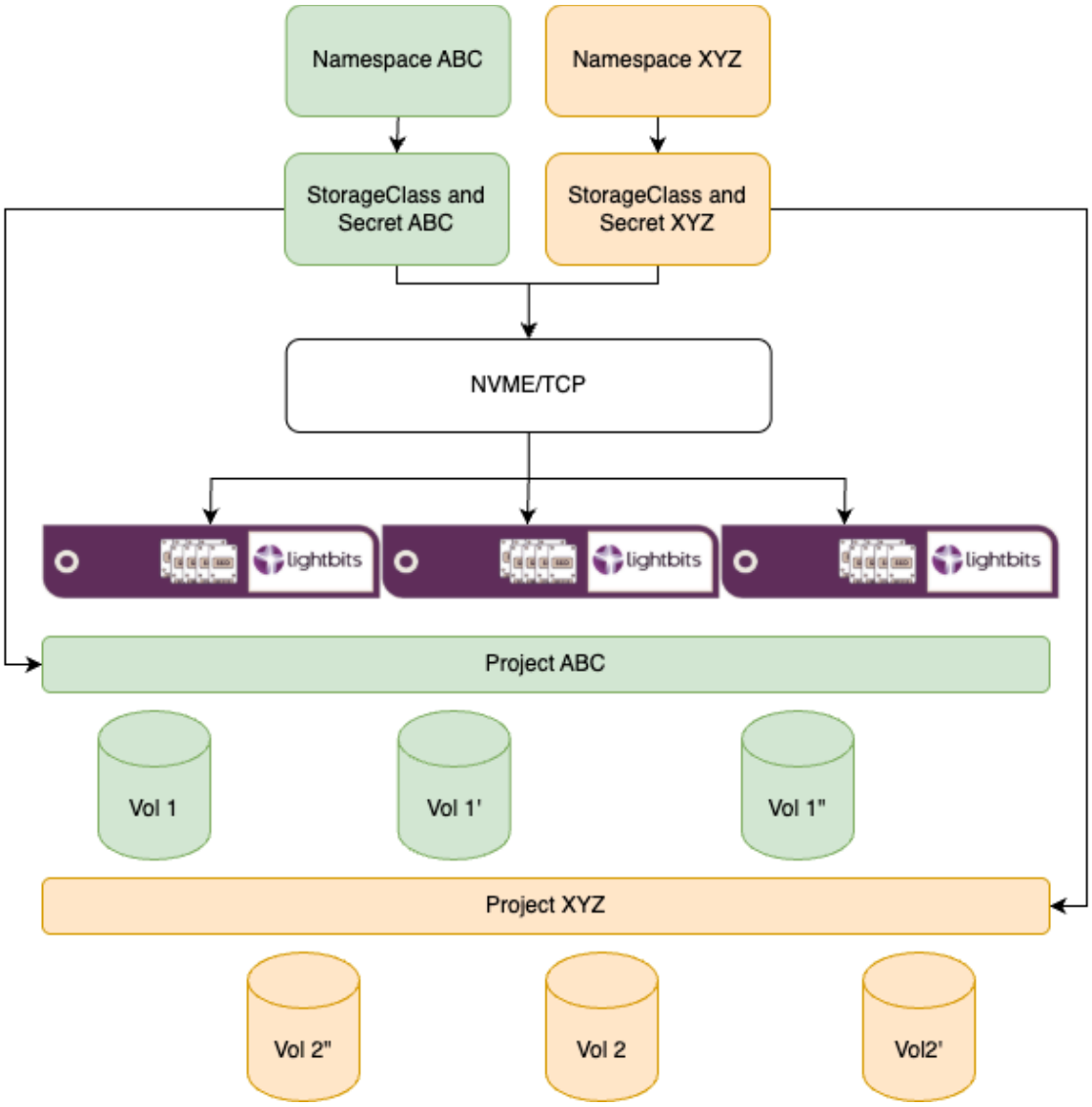
Abstract:

In today's dynamic cloud landscape, multi-tenancy is the key to maximizing resource efficiency, ensuring tenant isolation, and optimizing operational costs. By integrating Kubernetes powerful cloud orchestration with Lightbits' high-performance, software-defined storage, organizations can unlock seamless scalability, ultra-low latency, and enterprise-grade data resiliency. This white paper provides a comprehensive framework for implementing multi-tenancy in a Kubernetes-Lightbits environment—empowering businesses to deliver secure, efficient, and high-performance cloud services while maintaining full control over storage and compute resources.

1. Introduction

This white paper provides a step-by-step guide to seamlessly integrating **Lightbits storage** with **Kubernetes** in a multi-tenant environment. You'll learn how to **create a project in Lightbits**, configure secure access credentials, and set up the **CSI driver in Kubernetes** for optimal performance. We'll walk you through **creating a new namespace**, configuring **storage classes** to ensure isolated and efficient storage provisioning. Finally, you'll see how to **claim a PVC for a pod** attached to a **secure, high-performance volume** and leverage the CSI driver — reinforcing security and tenant isolation.

To have an understanding of the environment layout the drawing below gives an overview of the multi-tenant environment.



2. Prerequisites

Before diving into seamless multi-tenancy with **Kubernetes and Lightbits**, a solid foundation is essential. You'll need a **fully functional Kubernetes deployment** with administrative access, ensuring smooth orchestration and resource management. A **Lightbits cluster** must be up and running, ready to deliver high-performance, software-defined storage to Kubernetes services. Lastly, a **well-configured network** is critical—enabling secure, efficient communication between Kubernetes and Lightbits for optimal performance and scalability. With these key prerequisites in place, you're set to unlock the full potential of multi-tenant cloud storage.

Furthermore, there will be three variables we are going to work with, and please change those to what is suitable for your environment:

- Namespace = lightkube
- Project = lightkube

3. Lightbits Project Setup

3.1 Create the project

The first step we take is to create a project in Lightbits. Log in to your Lightbits cluster and list the existing project with the following command:

```
Unset
lbcli list projects
```

The result could be like:

```
Name    UUID                                    Description
default dc4c46e4-cb83-5df6-b84a-8beae38ac43c    This is the default project description
acme    f26153be-e9a4-4808-9ab8-3d2e39db837d    This is for customer acme.
```

To create a new project:

```
Unset
lbcli --endpoint https://localhost:443 create project --name lightkube
--description "Project for Lightbits and Kubernetes integration"
```

Output:

```
Name      UUID                                    Description
lightkube e200fb6f-0f2b-4b89-a01f-2b92c9ec9770    Project for Lightbits and Kubernetes integration
```

3.2 Set the security for the project in Lightbits

For Kubernetes to communicate with Lightbits via the CSI driver, a credential needs to be created. This will create a JWT at the last step, and that JWT is required with the configuration of the storage secret in Kubernetes. The first step we have to take is to create the keys. In this example, we are using *lightkube* as our project name. Please make sure you change *lightkube* with your name to generate the keys.

3.2.1 Create the RSA keys

```
Unset
mkdir -p ${HOME}/.lightkube_keys && ssh-keygen -t rsa -f
${HOME}/.lightkube_keys/lightkubekey -q -N "" -m PKCS8 && openssl rsa
-in ${HOME}/.lightkube_keys/lightkubekey -pubout -out
${HOME}/.lightkube_keys/lightkubekey.pem
```

The output will be like:

```
writing RSA key
```

Please verify that the keys have been created.

```
Unset
ls .lightkube_keys
```

It will show you three keys:

```
lightkubekey  lightkubekey.pem  lightkubekey.pub
```

(It should show you your project name instead of *lightkube*.)

3.2.2 Create the credential

First, list the credential for the project

```
Unset
lbcli list credentials --project-name lightkube
```

The output will show that no credential exists for this new project

```
ID    Type    Usage Kind  Size
```

Create the credential

```
Unset
lbcli --endpoint https://localhost:443 create credential --id=cred1
--project-name lightkube --type rsa256pubkey
${HOME}/.lightkube_keys/lightkubekey.pem
```

Output created

```
ID              Type                            Usage Kind   Size
cred1           RSA Public Key                  User         625 B
```

3.3.3 Create the JWT

We now need to create the JWT

```
Unset
lbcli create jwt --key-id lightkube:cred1 --key
${HOME}/.lightkube_keys/lightkubekey --role lightkube:admin
--issuer=root@example.com --subject=tenant
```

Output

```
eyJhbGciOiJSUzI1NiIsImtpZCI6ImxpZ2h0a3ViZTpjcmVkMSIsInR5cCI6IkpXVCJ9.eyJpc3M
iOiJyb290QGV4YW1wbGUuY29tIiwic3ViIjoidGVuYW50IiwiYXVkIjoiTGlnaHRPUyIsImV4cCI
6MTc1MDUwMDI2NywibmJmIjoxNzQ3OTA4MjY3LCJpYXQiOjE3NDc5MDgyNjcsImp0aSI6IjhiYjJ
lMzdiLTY2ZjktNGI4ZS1hMzIzLTgwMzIyNjk3NGNhNyIsInJvbGVzIjpbImxpZ2h0a3ViZTphZG1
pbiJdfQ.VBuD0Ep_KysJ2aUXt5aVvOC_33o3ZyKD1qxpk2I9sy3LVNaMuE3joc_DxqToIQm8gFgg
-JhG52p-cc659GnMKLyK-hr_-Gt4728N4S9e6m9FT62PIXecceHFMB0ktpmkmLDVNcJo8rNPkmRL
bORfS5DffeHOTKpPwo05TPPgx0OM7U6hcOyDCKKC6uteg2ab-pk3LqqnyKfRuPJyIeo4mfJOOCd0
y9wNb-irM1II4d1iM-FBpv16pbiChREpXeYe2rt36SnWWlHJhjW_qJx1aPNyOInVYO9nhl0-DwIw
Omou_e6PzSnB4xs6CDjaO8-daNl6pVH8i1qSejrXnZ66WjCO7muYX0kWKr9GuYCHYhPF3nPgGRxU
iT6dn8P-C4sOdyg5heJrSdrbrKmVotw0XU5D-SCzS-88rCwt-y6wkVHIF25-45INSxeQYMB2UqAQ
gVMuO6bP5cjm7BZ-_IKkb44d04fHXbF55A9GdlWcoDwTxDVu4qvI8SNz5M7MrftL
```

3.3.4 Verification of the JWT

Now, let us verify that the JWT we just created is working for the new project.

```
Unset
export
lightkube=eyJhbGciOiJSUzI1NiIsImtpZCI6ImxpZ2h0a3ViZTpjcmVkMSIsInR5cCI6Ik
pXVCJ9.eyJpc3MiOiJyb290QGV4YW1wbGUuY29tIiwic3ViIjoidGVuYW50IiwiYXVkIjoiT
GlnaHRPUyIsImV4cCI6MTc1MDUwMDI2NywibmJmIjoxNzQ3OTA4MjY3LCJpYXQiOjE3NDc5M
DgyNjcsImp0aSI6IjhiYjJlMzdiLTY2ZjktNGI4ZS1hMzIzLTgwMzIyNjk3NGNhNyIsInJvb
GVzIjpbImxpZ2h0a3ViZTphZG1pbiJdfQ.VBuD0Ep_KysJ2aUXt5aVvOC_33o3ZyKD1qxpk2
I9sy3LVNaMuE3joc_DxqToIQm8gFgg-JhG52p-cc659GnMKLyK-hr_-Gt4728N4S9e6m9FT6
2PIXecceHFMB0ktpmkmLDVNcJo8rNPkmRLbORfS5DffeHOTKpPwo05TPPgx0OM7U6hcOyDCK
KC6uteg2ab-pk3LqqnyKfRuPJyIeo4mfJOOCd0y9wNb-irM1II4d1iM-FBpv16pbiChREpXe
Ye2rt36SnWWlHJhjW_qJx1aPNyOInVYO9nhl0-DwIwOmou_e6PzSnB4xs6CDjaO8-daNl6pV
H8i1qSejrXnZ66WjCO7muYX0kWKr9GuYCHYhPF3nPgGRxUiT6dn8P-C4sOdyg5heJrSdrbrK
mVotw0XU5D-SCzS-88rCwt-y6wkVHIF25-45INSxeQYMB2UqAQgVMuO6bP5cjm7BZ-_IKkb4
4d04fHXbF55A9GdlWcoDwTxDVu4qvI8SNz5M7MrftL
```

Create a new volume

```
Unset
lbcli -J $lightkube create volume --name vol1 --project-name lightkube
--size 10GiB --replica-count 3 --acl ALLOW_ANY --compression true
```

Output

```
Name      UUID                                   State      Protection State
NSID      Size      Replicas   Compression   ACL                  Rebuild
Progress
vol1      7572afc3-ad7c-4032-9568-75e09cbf5962   Creating   Unknown
0         10 GiB    3          true          values:"ALLOW_ANY"
```

Delete the just-created volume

```
Unset
lbcli -J $lightkube delete volume --name vol1 --project-name lightkube
```

And check that the volume has been deleted

```
Unset
lbcli list volumes --project-name lightkube
```

Output

Name	UUID	State	Protection State	NSID	Size	Replicas	Compression	ACL	Rebuild Progress

We are now done with Lightbits, the next steps are in Kubernetes

4. Kubernetes configuration

To unlock the full potential of **Lightbits storage** in your **Kubernetes environment**, the first step is to configure the **storage secret** and **storage class**. The storage secret provides access to Lightbits and, more specifically, what kind of access (admin or reviewer) and to what project in Lightbits.

By embracing this approach, you gain **agility, scalability, and control**—ensuring that each tenant gets exactly what they need without unnecessary complexity.

4.1 Create a new namespace

Check the existing namespaces to make sure that the namespace does not already exist.

```
Unset
kubectl get namespaces
```

Output could be

```
NAME              STATUS   AGE
acme              Active   174d
default           Active   208d
kube-node-lease   Active   208d
kube-public       Active   208d
kube-system       Active   208d
```

Create the new namespace with the following command:

```
Unset
kubectl create namespace lightkube
```

Output

```
namespace/lightkube created
```

4.1 Create a storage secret and a storage class

When you created the JWT for the project access, we need to use that same JWT, *lightkube*, for the storage secret. The JWT was exported to $lightkube. To create the base64 encoded secret for Kubernetes, please follow the next steps:

```
Unset
echo -n $lightkube | base64 -w0 -
```

Output:

```
ZXlKaGJHY2lPaUpTVXpJMU5pSXNJbXRwWkNJNkteHBaMmgwYTNWaVpUcGpjbVZrTVNJc0luUjVj
Q0k2SWtwWFZDSjkuZXlKcGMzTWlPaUp5YjI5MFFHVjRZVzF3YkdVdVkyOXRJaXdpYzNWaUlqb2lk
R1Z1WVc1MElpd2lZWFZrSWpvaVRHbG5hSFJQVXlJc0ltVjRjQ0k2TVRjMU1EVXdNREkyTnl3aWJt
Sm1Jam94TnpRM09UQTRNalkzTENKcFlYUWlPakUzTkRjNU1EZ3lOamNzSW1wMGFTSTZJamhpWWpK
bE16ZGlMVFkyWmprdE5HSTRaUzFoTXpJekxUZ3dNekl5TmprM05HTmhOeUlzSW5KdmJHVnpJanBi
SW14cFoyaDBhM1ZpWlRwaFpHMXBiaUpkZlEuVkJ1RDBFcF9LeXNKMmFVWHQ1YVZ2T0NfMzNvM1p5
S0QxcXhwazJJOXN5M0xWTmFNdUUzam9jX0R4cVRvSVFtOGdGZ2ctSmhHNTJwLWNjNjU5R25NS0x5
Sy1ocl8tR3Q0NzI4TjRTOWU2bTlGVDYyUElYZWNjZUhGTUIwa3RwbWttTERWTmNKbzhyTlBrbVJM
Yk9SZlM1RGZmZUhPVEtwUHdvMDVUUFBneDBPTTdVNmhjT3lEQ0tLQzZ1dGVnMmFiLXBrM0xxcW55
S2ZSdVBKeUllbzRtZkpPT0NkMHk5d05iLWlyTTFJSTRkMWlNLUZCcHYxNnBiaUNoUkVwWGVZZTJy
dDM2U25XV2xISmhqV19xSngxYVBOeU9JblZZTzluaGwwLUR3SXdPbW91X2U2UHpTbkI0eHM2Q0Rq
YU84LWRhTmw2cFZIOGkxcVNlanJYblo2NldqQ083bXVZWDBrV0tyOUd1WUNIWWhQRjNuUGdHUnhV
aVQ2ZG44UC1DNHNPZHlnNWhlSnJTZHJickttVm90dzBYVTVELVNDelMtODhyQ3d0LXk2d2tWSElG
MjUtNDVJTlN4ZVFZTUIyVXFBUWdWTXVPNmJQNWNqbTdCWi1fSUtrYjQ0ZDA0ZkhYYkY1NUE5R2Rs
V2NvRHdUeERWdTRxdkk4U056NU03TXJmdEw=
```

This new JWT will be used as the secret for the storage class. Please copy the output to your clipboard.

When the CSI driver for Lightbits is installed, there is a subdirectory called examples. One of the examples is secret-and-storage-class.yaml. In this configuration file, the storage secret is declared, and the storage class as well.

Copy this file to lightkube-secret-and-class.yaml, and the file contents look like this:

```
---
# Source: lb-csi-workload-examples/charts/storageclass/templates/secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: example-secret
  namespace: default
  labels:
    helm.sh/chart: "storageclass-0.1.0"
    app.kubernetes.io/instance: "RELEASE-NAME"
    app.kubernetes.io/version: ""
    app.kubernetes.io/managed-by: "Helm"
type: lightbitslabs.com/jwt
data:
  jwt: |-

ZXlKaGJHY2lPaUpTVXpJMU5pSXNJbXRwWkNJNkluTjVjM1JsYlRweWIyOTBJaXdpZEhsd0lqb2lT
bGRVSW4wLmV5SnBjM01pT2lJdmFHOXRaUzlrWlcxdkwyeHBaMmgwYjNNdFkyVnlkR2xtYVdOaGRH
VnpMMk5sY25RdGJHSXRZV1J0YVc0dGEyVjVMbkJsYlNJc0luSnZiR1Z6SWpwYkluTjVjM1JsYlRw
amJIVnpkR1Z5TFdGa2JXbHVJbDBzSW1GMVpDSTZJa3hwWjJoMFQxTWlMQ0p6ZFdJaU9pSnNhV2Rv
ZEc5ekxXTnNhV1Z1ZENJc0ltbGhkQ0k2TVRjeU9UZzNNVFV3TWl3aVpYaHdJam94TnpZeE5EQTNO
VEF5ZlEuVlQtQ3dPMkQwRkhBc3RWZExEM0R5VGxPR0pHb003eE1ub0Q5Nkd0TWNFeVV3ZGJDN3FX
VzR2cXFTR3lKYzZscTlrZjVmX3l5QW1xeUJ2dWxLYVNkUG53a3pWZTN6MURiZkFJcnl6aXl1cjdN
RkdGSVE1cXlleDlyQktXYTF5cVVTcEhTSnl5YVB1S1MxUEt2V3dYLUpCUGRaTUJvLWp6UTI4eE9Y
d0VWU2VpZ2VqQ01OR0ZTZ1ktcTJZWkNaSTdpVjJWZnN3U1hFZnhGYXd4MUtjM3c1cV9OUnRBTGwz
UVFMZVJoclRValMyTl9jZm5lSU9LQ1NVM3JjcDc0d1NOZ2VJaGducEd3cDc4Mi00QzU5bTJLekp3
SThqbzdFbWw1MTc2SkYyVm9VTlptLU5SX01YdTBpY2s1bXhUcEJRS0ZvU3pPdDllWVRaZ20xQ3d2
ZHVNX2Z3
---
# Source:
lb-csi-workload-examples/charts/storageclass/templates/storageclass.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: example-sc
provisioner: csi.lightbitslabs.com
allowVolumeExpansion: true
parameters:
  mgmt-endpoint: 192.168.1.41:443,192.168.1.42:443,192.168.1.43:443
  replica-count: "3"
  compression: enabled
  qos-policy-name: 10-iops-per-gb
```

```
  project-name: default
  mgmt-scheme: grpcs
  csi.storage.k8s.io/controller-publish-secret-name: example-secret
  csi.storage.k8s.io/controller-publish-secret-namespace: default
  csi.storage.k8s.io/controller-expand-secret-name: example-secret
  csi.storage.k8s.io/controller-expand-secret-namespace: default
  csi.storage.k8s.io/node-publish-secret-name: example-secret
  csi.storage.k8s.io/node-publish-secret-namespace: default
  csi.storage.k8s.io/node-stage-secret-name: example-secret
  csi.storage.k8s.io/node-stage-secret-namespace: default
  csi.storage.k8s.io/provisioner-secret-name: example-secret
  csi.storage.k8s.io/provisioner-secret-namespace: default
```

The things which need to be changed are the following, marked in yellow:

```
---
# Source: lb-csi-workload-examples/charts/storageclass/templates/secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: lightkube-secret
  namespace: lightkube
  labels:
    helm.sh/chart: "storageclass-0.1.0"
    app.kubernetes.io/instance: "RELEASE-NAME"
    app.kubernetes.io/version: ""
    app.kubernetes.io/managed-by: "Helm"
type: lightbitslabs.com/jwt
data:
  jwt: |-

ZXlKaGJHY2lPaUpTVXpJMU5pSXNJbXRwWkNJNklteHBaMmgwYTNWaVpUcGpjbVZrTVNJc0luUjVj
Q0k2SWtwWFZDSjkuZXlKcGMzTWlPaUp5YjI5MFFHVjRZVzF3YkdVdVkyOXRJaXdpYzNWaUlqb2lk
R1Z1WVc1MElpd2lZWFZrSWpvaVRHbG5hSFJQVXlJc0ltVjRjQ0k2TVRjMU1EVXdNREkyTnl3aWJt
Sm1Jam94TnpRM09UQTRNalkzTENKcFlYUWlPakUzTkRjNU1EZ3lOamNzSW1wMGFTSTZJamhpWWpK
bE16ZGlMVFkyWmprdE5HSTRaUzFoTXpJekxUZ3dNekl5TmprM05HTmhOeUlzSW5KdmJHVnpJanBi
SW14cFoyaDBhM1ZpWlRwaFpHMXBiaUpkZlEuVkJ1RDBFcF9LeXNKMmFVWHQ1YVZ2T0NfMzNvM1p5
S0QxcXhwazJJOXN5M0xWTmFNdUUzam9jX0R4cVRvSVFtOGdGZ2ctSmhHNTJwLWNjNjU5R25NS0x5
Sy1ocl8tR3Q0NzI4TjRTOWU2bTlGVDYyUElYZWNjZUhGTUIwa3RwbWttTERWTmNKbzhyTlBrbVJM
Yk9SZlM1RGZmZUhPVEtwUHdvMDVUUFBneDBPTTdVNmhjT3lEQ0tLQzZ1dGVnMmFiLXBrM0xxcW55
S2ZSdVBKeUllbzRtZkpPT0NkMHk5d05iLWlyTTFJSTRkMWlNLUZCcHYxNnBiaUNoUkVwWGVZZTJy
dDM2U25XV2xISmhqV19xSngxYVBOeU9JblZZTzluaGwwLUR3SXdPbW91X2U2UHpTbkI0eHM2Q0Rq
YU84LWRhTmw2cFZIOGkxcVNlanJYblo2NldqQ083bXVZWDBrV0tyOUd1WUNIWWhQRjNuUGdHUnhV
aVQ2ZG44UC1DNHNPZHlnNWhlSnJTZHJickttVm90dzBYVTVELVNDelMtODhyQ3d0LXk2d2tWSElG
MjUtNDVJTlN4ZVFZTUIyVXFBUWdWTXVPNmJQNWNqb
---
# Source:
```

```
lb-csi-workload-examples/charts/storageclass/templates/storageclass.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: lightkube-sc
  namespace: lightkube
provisioner: csi.lightbitslabs.com
allowVolumeExpansion: true
parameters:
  mgmt-endpoint: 192.168.1.41:443,192.168.1.42:443,192.168.1.43:443
  replica-count: "3"
  compression: enabled
  qos-policy-name: 10-iops-per-gb
  project-name: lightkube
  mgmt-scheme: grpcs
  csi.storage.k8s.io/controller-publish-secret-name: lightkube-secret
  csi.storage.k8s.io/controller-publish-secret-namespace: lightkube
  csi.storage.k8s.io/controller-expand-secret-name: lightkube-secret
  csi.storage.k8s.io/controller-expand-secret-namespace: lightkube
  csi.storage.k8s.io/node-publish-secret-name: lightkube-secret
  csi.storage.k8s.io/node-publish-secret-namespace: lightkube
  csi.storage.k8s.io/node-stage-secret-name: lightkube-secret
  csi.storage.k8s.io/node-stage-secret-namespace: lightkube
  csi.storage.k8s.io/provisioner-secret-name: lightkube-secret
  csi.storage.k8s.io/provisioner-secret-namespace: lightkube
```

Create the storage secret and the storage class

```
Unset
kubectl create -f examples/lightkube-secret-and-class.yaml
```

Output

```
secret/lightkube-secret created
storageclass.storage.k8s.io/sc-lightkube created
```

Check the storage secret and the storage class

```
Unset
kubectl get secret,sc -n lightkube
```

Output something like

```
NAME                                                  PROVISIONER
RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
storageclass.storage.k8s.io/local-path (default)   rancher.io/local-path
Delete          WaitForFirstConsumer   false                  209d
storageclass.storage.k8s.io/sc-lightkube           csi.lightbitslabs.com
Delete          Immediate              true                   12s
```

4.2 Create a PVC and a POD on the PVC

In the examples directory, a file called filesystem-workload.yaml which creates a PVC and runs a POD on that PVC. Copy the file to lightkube-pvc-pod.yaml, the contents of the file look like this

```
---
# Source:
lb-csi-workload-examples/charts/filesystem/templates/example-fs-pvc.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: example-fs-pvc
spec:
  storageClassName: "example-sc"
  accessModes:
  - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 10Gi
---
# Source:
lb-csi-workload-examples/charts/filesystem/templates/example-fs-pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: "example-fs-pod"
spec:
  containers:
  - name: busybox-date-container
    imagePullPolicy: IfNotPresent
    image: busybox
    command: ["/bin/sh"]
    args: ["-c", "if [ -f /mnt/test/hostname ] ; then (md5sum -s -c
/mnt/test/hostname.md5 && echo OLD MD5 OK || echo BAD OLD MD5) >>
/mnt/test/log ; fi ; echo $KUBE_NODE_NAME: $(date +%Y-%m-%d.%H-%M-%S) >|
/mnt/test/hostname ; md5sum /mnt/test/hostname >| /mnt/test/hostname.md5 ;
```

```
echo NEW NODE: $KUBE_NODE_NAME: $(date +%Y-%m-%d.%H-%M-%S) >> /mnt/test/log
; while true ; do date +%Y-%m-%d.%H-%M-%S >| /mnt/test/date ; sleep 10 ;
done" ]
    env:
    - name: KUBE_NODE_NAME
      valueFrom:
        fieldRef:
          fieldPath: spec.nodeName
    stdin: true
    tty: true
    volumeMounts:
    - name: test-mnt
      mountPath: "/mnt/test"
  volumes:
  - name: test-mnt
    persistentVolumeClaim:
      claimName: "example-fs-pvc"
```

The things which need to be changed are the following, marked in yellow:

```
---
# Source:
lb-csi-workload-examples/charts/filesystem/templates/example-fs-pvc.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: lightkube-pvc
spec:
  storageClassName: "lightkube-sc"
  accessModes:
  - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 10Gi
---
# Source:
lb-csi-workload-examples/charts/filesystem/templates/example-fs-pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: "lightkube-pod"
  namespace: lightkube
spec:
  containers:
  - name: busybox-date-container
    imagePullPolicy: IfNotPresent
    image: busybox
    command: ["/bin/sh"]
```

```
    args: ["-c", "if [ -f /mnt/test/hostname ] ; then (md5sum -s -c
/mnt/test/hostname.md5 && echo OLD MD5 OK || echo BAD OLD MD5) >>
/mnt/test/log ; fi ; echo $KUBE_NODE_NAME: $(date +%Y-%m-%d.%H-%M-%S) >|
/mnt/test/hostname ; md5sum /mnt/test/hostname >| /mnt/test/hostname.md5 ;
echo NEW NODE: $KUBE_NODE_NAME: $(date +%Y-%m-%d.%H-%M-%S) >> /mnt/test/log
; while true ; do date +%Y-%m-%d.%H-%M-%S >| /mnt/test/date ; sleep 10 ;
done" ]
    env:
    - name: KUBE_NODE_NAME
      valueFrom:
        fieldRef:
          fieldPath: spec.nodeName
    stdin: true
    tty: true
    volumeMounts:
    - name: test-mnt
      mountPath: "/mnt/test"
  volumes:
  - name: test-mnt
    persistentVolumeClaim:
      claimName: "lightkube-pvc"
```

Create the PVC and the POD

```
Unset
kubectl create -f examples/lightkube-pvc-pod.yaml
```

Output

```
persistentvolumeclaim/lightkube-pvc created
pod/lightkube-pod created
```

Check that the PV, PVC and POD are actually there and running

```
Unset
kubectl get pv,pvc,pod -n lightkube
```

Output

```
NAME                                                              CAPACITY
ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM
STORAGECLASS   VOLUMEATTRIBUTESCLASS   REASON   AGE
persistentvolume/pvc-8385cfe9-eb98-44d5-b64e-660beb37fc34   10Gi       RWO
Delete           Bound    lightkube/lightkube-pvc   lightkube-sc   <unset>
26s

NAME                                  STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
persistentvolumeclaim/lightkube-pvc   Bound
pvc-8385cfe9-eb98-44d5-b64e-660beb37fc34   10Gi       RWO
lightkube-sc   <unset>                 26s

NAME                READY   STATUS    RESTARTS   AGE
pod/lightkube-pod   1/1     Running   0          26s
```

The PVC, PV, and POD have been created, and the POD is running on the PVC in the namespace lightkube.

4.3 Verification on Lightbits

The last step is to verify that the PVC named pvc-8385cfe9-eb98-44d5-b64e-660beb37fc34 has the same name in Lightbits and runs in the project lightkube in Lightbits as well. Go to one of the nodes of the Lightbits cluster and provide the following command

```
Unset
lbcli list volumes --project-name lightkube
```

Output

```
Name                                     UUID
State       Protection State   NSID      Size      Replicas   Compression
ACL                                                           Rebuild
Progress
pvc-8385cfe9-eb98-44d5-b64e-660beb37fc34
3281b6cf-f64e-4f3a-b687-1676fb2f867e   Available   FullyProtected     8
10 GiB    3          false
values:"nqn.2019-09.com.lightbitslabs:host:kubernetes.node"   None
```

The volume does exist, the next step is to verify the project the volume has been placed in. Please run the following command

```
Unset
lbcli get volume --project-name lightkube --name
pvc-8385cfe9-eb98-44d5-b64e-660beb37fc34 -o json | grep projectName
```

Output

```
"projectName": "lightkube"
```

5. Conclusion

Effortless CSI Integration for Kubernetes Automation

Lightbits delivers a streamlined, DevOps-friendly storage solution through its native support for the Kubernetes Container Storage Interface (CSI). This allows for automated provisioning, attachment, and lifecycle management of persistent volumes—fully integrated with your CI/CD pipelines and GitOps workflows. With Lightbits, you don't have to manage storage as a separate domain; it becomes an extension of your Kubernetes stack, accelerating deployment times and reducing operational complexity.

Native Multi-Tenancy via StorageClass Mappings

Managing multi-tenant environments just got easier. Lightbits enables direct mapping from Kubernetes StorageClass definitions into isolated Lightbits projects, letting you enforce resource boundaries and performance controls per team, namespace, or workload. This tight integration gives DevOps teams granular control without introducing additional layers of tooling, making it ideal for shared clusters, internal platforms, or managed service environments where tenant isolation and accountability are critical.

Reliable, Scalable, and Built for Ops at Scale

Lightbits brings enterprise-grade resilience and scalability to cloud-native workloads, with features that DevOps teams trust—high availability, consistent performance, and hands-off scaling. Whether you're supporting transactional microservices, stateful data platforms, or distributed ML pipelines, Lightbits delivers rock-solid performance with minimal maintenance. It's storage that fits into your automation mindset, helping you deliver infrastructure that's fast, secure, and always on.

To learn more about Lightbits Labs, visit https://www.lightbitslabs.com.

About Lightbits Labs

Lightbits Labs® (Lightbits) invented the NVMe over TCP protocol and offers best-of-breed software-defined block storage that enables data center infrastructure modernization for organizations building a private or public cloud. Built from the ground up for low consistent latency, scalability, resiliency, and cost-efficiency, Lightbits software delivers the best price/performance for real-time analytics, transactional, and AI/ML workloads. Lightbits Labs is backed by enterprise technology leaders [Cisco Investments, Dell Technologies Capital, Intel Capital, Lenovo, and Micron] and is on a mission to deliver the fastest and most cost-efficient data storage for performance-sensitive workloads at scale.

🌐 www.lightbitslabs.com ✉ info@lightbitslabs.com

US Offices
1830 The Alameda,
San Jose, CA 95126, USA

Israel Office
17 Atir Yeda Street,
Kfar Saba 4464313, Israel

 LBWP14/2025/05