Deploying the Lightbits CSI Driver on Harvester

TECHNICAL WHITE PAPER

NVMe/TCP shared block storage on an immutable SLE Micro host

LightOS 3.19.1 · CSI plugin v1.23.0 · Harvester v1.8.1 / Kubernetes v1.35

ABSTRACT

This paper documents the end-to-end deployment of the Lightbits Container Storage Interface (CSI) driver on a Harvester cluster, providing Kubernetes and KubeVirt virtual machines with disaggregated NVMe/TCP block storage from a Lightbits cluster. Harvester runs on an immutable, SLE Micro-based operating system whose root filesystem reverts on reboot, which invalidates the conventional package-install workflow that most CSI drivers assume.

We describe a reboot- and upgrade-durable approach built on three pillars: persisting host-side NVMe/TCP prerequisites (the nvme_tcp kernel module and a stable host NQN) through Harvester's CloudInit custom resource rather than direct filesystem edits; deploying the discovery client as an in-pod container via the driver's discovery-client (-dc) manifest variant, eliminating any host package installation; and applying the version-matched static manifests for Kubernetes v1.35 together with a private-registry pull secret, a project-scoped JWT credential, and a CDI-compliant StorageClass.

The procedure covers verification at every stage, the Harvester-specific StorageClass constraints (default-class arbitration and the CDI volume-mode/access-mode annotation), and enabling ReadWriteMany block volumes. The result is a validated, self-contained runbook for delivering high-performance shared block storage to virtualized and containerized workloads on a minimal, immutable-OS footprint.

Audience: Storage/platform engineers deploying Lightbits with Harvester or RKE2 on immutable Linux hosts.

© 2026 Lightbits Labs™

Table of Contents

© 2026 Lightbits Labs™

1 Solution Overview

The Lightbits CSI driver connects a Kubernetes cluster to an external Lightbits storage cluster over NVMe/TCP, presenting cluster-managed volumes to pods and, on Harvester, to virtual machine disks. Unlike hyperconverged defaults such as Longhorn, storage and compute are disaggregated: capacity lives on dedicated Lightbits nodes and is consumed over standard TCP networking, with no special fabric hardware.

On Harvester, the deployment has one defining complication. The host operating system is immutable and SLE Micro-based, so most changes under the root filesystem are reverted on reboot. Any host-side prerequisite must therefore be applied through a persistence mechanism the platform reconciles on every boot, not by editing files directly.

1.1 Component inventory

Component	Role
lb-csi-controller (StatefulSet)	Provisioning, attach/detach, resize, snapshot — one replica.
lb-csi-node (DaemonSet)	Per-node volume staging and mount; runs privileged, hostNetwork.
lb-nvme-discovery-client	In-pod container in the node DaemonSet; maintains NVMe/TCP discovery. No host install.
init-nvme-tcp	Init container that ensures the nvme_tcp module is present before the plugin starts.
snapshot-controller	Cluster-wide CSI snapshot control loop plus the snapshot CRDs.
CSI sidecars	provisioner v5.2.0, attacher v4.8.0, resizer v1.13.2, snapshotter v8.2.0, registrar v2.13.0.

1.2 Target environment

Parameter	Value
Harvester	v1.8.1
Kubernetes (RKE2)	v1.35.6+rke2r1
Host OS	SLE Micro-based, immutable (kernel 6.12.x)
LightOS / CSI	3.19.1 / plugin image v1.23.0
Lightbits mgmt endpoints	192.168.1.41:443, 192.168.1.42:443, 192.168.1.43:443
Lightbits datapath endpoints	10.10.10.41:443, 10.10.10.42:443, 10.10.10.43:443
Container runtime	containerd (RKE2 socket: /run/k3s/containerd/containerd.sock)

© 2026 Lightbits Labs™

2 Host-Side Prerequisites (Reboot-Durable)

The driver requires the nvme_tcp kernel module and a stable host NQN. On an immutable host these are applied through a Harvester CloudInit custom resource, which the node controller writes to /oem and re-applies on every boot. This is durable across reboots and, unlike the installer's after-install-chroot stage, survives Harvester upgrades.

2.1 Persist the nvme_tcp kernel module

Apply the following CloudInit resource with kubectl apply -f. It both writes a persistent modules-load entry and loads the module at the initramfs stage.

Shell

```
apiVersion: node.harvesterhci.io/v1beta1
kind: CloudInit
metadata:
  name: lightbits-nvme-tcp
spec:
  matchSelector:
    harvesterhci.io/managed: "true"
  filename: 99-lightbits-nvme-tcp.yaml
  contents: |
       stages:
       initramfs:
       - name: "load nvme-tcp"
          commands:
              - modprobe nvme-tcp
       files:
             - path: /etc/modules-load.d/nvme-tcp.conf
              content: "nvme-tcp\n"
              permissions: 0644
  paused: false
```

Verify, then reboot once and re-check to confirm persistence:

Shell

```
kubectl get cloudinit
sudo cat /oem/99-lightbits-nvme-tcp.yaml
lsmod | grep nvme_tcp
# after: sudo reboot
lsmod | grep nvme_tcp
cat /etc/modules-load.d/nvme-tcp.conf
```


© 2026 Lightbits Labs™

VERIFIED On the reference system the module dependency chain (nvme_tcp, nvme_fabrics, nvme_core, nvme_keyring) remained loaded after a full reboot.

2.2 Confirm a stable host NQN

The discovery client and node plugin key off the host NQN and host ID under /etc/nvme. Confirm both exist and are stable across a reboot:

```shell
cat /etc/nvme/hostnqn      # e.g. nqn.2014-08.org.nvmexpress:uuid:...
cat /etc/nvme/hostid
```

NOTE If the NQN changes across reboots it must be pinned via an additional CloudInit stage that writes /etc/nvme/hostnqn explicitly, because the Lightbits cluster ACLs are keyed to it. On the reference system the NQN was stable and required no pinning.

3 Runtime and Registry Reachability

Harvester's RKE2 containerd uses a non-default socket, so the bundled crictl must be pointed at it before manual image pulls will work. This check is optional but recommended; it proves image egress before deployment.

```shell
sudo tee /etc/crictl.yaml >/dev/null <<'EOF'
runtime-endpoint: unix:///run/k3s/containerd/containerd.sock
image-endpoint: unix:///run/k3s/containerd/containerd.sock
timeout: 10
EOF

sudo /var/lib/rancher/rke2/bin/crictl pull \
  registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0
```

NOTE /etc/crictl.yaml is on the immutable root and will not persist a reboot. It is needed only for manual pulls; kubelet talks to containerd directly and does not depend on it.

© 2026 Lightbits Labs™

4 Private Registry Pull Secret

The Lightbits images are served from docker.lightbitslabs.com and require credentials obtained from your Lightbits account or support channel. Create the pull secret in kube-system, where the plugin runs:

```
Shell
export KUBECONFIG=/etc/rancher/rke2/rke2.yaml

kubectl create secret docker-registry lb-docker-reg-cred \
  --docker-server=docker.lightbitslabs.com \
  --docker-username='<user>' \
  --docker-password='<pass>' \
  -n kube-system
```

Validate credentials and egress by pulling the private plugin image directly:

```
Shell
sudo /var/lib/rancher/rke2/bin/crictl pull \
  docker.lightbitslabs.com/lightos-csi/lb-csi-plugin:v1.23.0
```

A 401/403 indicates bad credentials; a timeout indicates the node cannot reach the registry, in which case the three Lightbits images (lb-csi-plugin and lb-nvme-discovery-client) must be mirrored to a reachable registry and the manifest image references rewritten.

5 Prepare the CSI Manifest

The shipped v1.35 discovery-client manifest (lb-csi-plugin-k8s-v1.35-dc.yaml) comments out the imagePullSecrets clause in both workloads. Because all Lightbits images use an Always pull policy, the secret must be wired in or pods will fail with ImagePullBackOff. Copy the file to a writable path and activate both blocks:

```
Shell
cp lb-csi-plugin-k8s-v1.35-dc.yaml ~/lb-csi-plugin.yaml

sed -i \
  's/^       #imagePullSecrets:/       imagePullSecrets:/; \
```


© 2026 Lightbits Labs™

```
   s/^       #- name: my-docker-registry-credentials-secret/      - name:
lb-docker-reg-cred/' \
  ~/lb-csi-plugin.yaml

grep -n 'imagePullSecrets:' ~/lb-csi-plugin.yaml     # expect 2 hits
grep -n 'name: lb-docker-reg-cred' ~/lb-csi-plugin.yaml # expect 2 hits
```

6 Deploy the Snapshot Controller

The snapshot controller and its CRDs are cluster-wide prerequisites for the controller's snapshotter sidecar. Deploy them first:

Shell

```
kubectl apply -f snapshot-controller.yaml
kubectl get crd | grep snapshot.storage.k8s.io
```

7 Create the JWT Credential

The driver authenticates to the Lightbits API with a project-scoped JWT. Generate a fresh token on a Lightbits node, base64-encode it without a trailing newline, and store it as a typed secret:

Shell

```
echo -n '<reissued-jwt>' | base64 -w0
# lb-jwt-secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: lb-jwt-secret
  namespace: kube-system
type: lightbitslabs.com/jwt
data:
  jwt: <BASE64_TOKEN>
kubectl apply -f lb-jwt-secret.yaml
```


© 2026 Lightbits Labs™

SECURITY Treat any JWT or kubeconfig that has been copied outside the cluster as compromised and reissue it. A Lightbits system:cluster-admin token grants full control of the storage cluster.

8 Deploy the CSI Plugin

```
Shell
kubectl create -f ~/lb-csi-plugin.yaml
```

Verify that the controller and node pods reach a fully ready state. On a single node, expect one controller pod at 5/5 and one node pod at 3/3 (plugin, registrar, discovery-client):

```
Shell
kubectl get statefulset -n kube-system lb-csi-controller
kubectl get daemonset  -n kube-system lb-csi-node
kubectl get pods -n kube-system -l app=lb-csi-plugin -o wide
```

If any pod is in ImagePullBackOff, revisit Sections 4-5; kubectl describe pod -n kube-system <pod> identifies the failing image or missing secret.

9 StorageClass and Harvester Constraints

Two Harvester-specific validations gate StorageClass creation, encountered in sequence:

9.1 Default-class arbitration

Harvester ships harvester-longhorn as the cluster default, and Kubernetes permits only one default class. Either omit the default annotation on the Lightbits class (recommended — Longhorn continues to back Harvester's internal machinery), or explicitly demote Longhorn first:

```
Shell
kubectl patch storageclass harvester-longhorn --type merge \
  -p
'{"metadata":{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"
}}}'
```


© 2026 Lightbits Labs™

9.2 CDI volume-mode / access-mode annotation

Harvester's CDI integration requires the cdi.harvesterhci.io/storageProfileVolumeModeAccessModes annotation on any non-Longhorn class used for VM images; without it the class is rejected. The value maps each volume mode to its supported access modes. Harvester VM disks use Block mode. Advertise ReadWriteMany only if RWX is enabled on the plugin (Section 10).

```
Shell
# lb-storageclass.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: lb-sc
  annotations:
    cdi.harvesterhci.io/storageProfileVolumeModeAccessModes:
'{"Block":["ReadWriteOnce","ReadWriteMany"]}'
provisioner: csi.lightbitslabs.com
allowVolumeExpansion: true
parameters:
  mgmt-endpoint: 10.10.10.41:443,10.10.10.42:443,10.10.10.43:443
  replica-count: "3"
  compression: enabled
  project-name: default
  mgmt-scheme: grpcs
  csi.storage.k8s.io/controller-publish-secret-name: lb-jwt-secret
  csi.storage.k8s.io/controller-publish-secret-namespace: kube-system
  csi.storage.k8s.io/controller-expand-secret-name: lb-jwt-secret
  csi.storage.k8s.io/controller-expand-secret-namespace: kube-system
  csi.storage.k8s.io/node-publish-secret-name: lb-jwt-secret
  csi.storage.k8s.io/node-publish-secret-namespace: kube-system
  csi.storage.k8s.io/node-stage-secret-name: lb-jwt-secret
  csi.storage.k8s.io/node-stage-secret-namespace: kube-system
  csi.storage.k8s.io/provisioner-secret-name: lb-jwt-secret
  csi.storage.k8s.io/provisioner-secret-namespace: kube-system
kubectl apply -f lb-storageclass.yaml
kubectl get storageclass
```

NOTE Use {"Block":["ReadWriteOnce"]} if RWX is not enabled.

10 Enabling ReadWriteMany (Optional)

By default, the plugin rejects RWX with unsupported mode: 'MULTI_NODE_MULTI_WRITER'. RWX requires both back-end multi-attach support and the RWX flag set on the plugin containers in the controller

© 2026 Lightbits Labs™

StatefulSet and node DaemonSet. After enabling, restart the workloads and confirm the CDI annotation advertises RWX:

```
Shell
kubectl rollout restart statefulset/lb-csi-controller daemonset/lb-csi-node -n
kube-system
kubectl rollout status  statefulset/lb-csi-controller -n kube-system
kubectl rollout status  daemonset/lb-csi-node -n kube-system
```

11 End-to-End Validation

Bind a test claim to confirm the full path — controller provisioning, JWT authentication, Lightbits volume creation, discovery, and attach:

```
Shell
# lb-test-pvc.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: lb-test-pvc
  namespace: default
spec:
  accessModes: ["ReadWriteOnce"]
  storageClassName: lb-sc
  resources:
       requests:
       storage: 10Gi
kubectl apply -f lb-test-pvc.yaml
kubectl get pvc -n default lb-test-pvc  # expect: Bound
```

A Bound status confirms the deployment is operational. Volumes can now be consumed by pods and by Harvester VM disks that select the lb-sc StorageClass.

12 Persistence and Upgrade Notes

Because the host OS is immutable and the manifest was customized in place, keep the following in mind for the lifecycle of the deployment:

- **Host prerequisites** (module, host NQN) persist through the CloudInit resource and are reconciled on every boot, including after Harvester upgrades.

© 2026 Lightbits Labs™

- **Discovery client** runs as an in-pod container; there is no host package to reinstall or re-persist.
- **Customized manifest.** The edited ~/lb-csi-plugin.yaml (pull secret and RWX flag if set) is the source of truth. Re-applying the stock -dc manifest from the bundle would drop these customizations. Store the customized manifest, StorageClass, JWT secret, and CloudInit resources in version control.
- **Credentials.** Reissue any JWT or kubeconfig exposed during setup.

Appendix A Deployment Checklist

- ☐ nvme_tcp CloudInit applied; module survives reboot
- ☐ Host NQN present and stable (/etc/nvme/hostnqn)
- ☐ registry.k8s.io reachable (crictl pull test)
- ☐ lb-docker-reg-cred secret created; private image pulls
- ☐ imagePullSecrets activated in both workloads (2 + 2 grep hits)
- ☐ snapshot-controller and CRDs applied
- ☐ lb-jwt-secret created with reissued token
- ☐ CSI plugin deployed: controller 5/5, node 3/3
- ☐ Longhorn default handled; lb-sc created with CDI annotation
- ☐ RWX enabled only if required and back end supports it
- ☐ Test PVC reaches Bound

Appendix B Image Reference

Image	Tag
docker.lightbitslabs.com/lightos-csi/lb-csi-plugin	v1.23.0
docker.lightbitslabs.com/lightos-csi/lb-nvme-discovery-client	v1.23.0
registry.k8s.io/sig-storage/csi-provisioner	v5.2.0
registry.k8s.io/sig-storage/csi-attacher	v4.8.0
registry.k8s.io/sig-storage/csi-resizer	v1.13.2
registry.k8s.io/sig-storage/csi-snapshotter	v8.2.0
registry.k8s.io/sig-storage/csi-node-driver-registrar	v2.13.0
registry.k8s.io/sig-storage/snapshot-controller	v8.2.0

© 2026 Lightbits Labs™

About Lightbits Labs™

Lightbits Labs (Lightbits) is the inventor of the NVMe over TCP storage protocol, which is natively built into its industry-leading block storage, and the first KV cache prefetch engine acceleration for AI. Lightbits data storage solutions are engineered to deliver unmatched high performance and maximum hardware efficiency for LLM inference, real-time analytics, and transactional workloads at scale. Lightbits is backed by enterprise technology leaders [Cisco Investments, Dell Technologies Capital, Intel Capital, Lenovo, and Micron] and is on a mission to deliver best-in-class, cost-efficient storage systems 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

The information in this document and any document referenced herein is provided for informational purposes only, is provided as is and with all faults and cannot be understood as substituting for customized service and information that might be developed by Lightbits Labs ltd for a particular user based upon that user's particular environment. Reliance upon this document and any document referenced herein is at the user's own risk.

The software is provided "As is", without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. In no event shall the contributors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or other dealings with the software.

Unauthorized copying or distributing of included software files, via any medium is strictly prohibited.

COPYRIGHT© 2026 LIGHTBITS LABS LTD. - ALL RIGHTS RESERVED

LBWP26/2026/8