Lightbits Storage Integration with Red Hat Openstack Services on Openshift (RHOSO)

A Step-by-Step Technical Reference Guide

Version 1.0 | Tested on OpenShift 4.18.32 + RHOSO 18

Author: Sagy Volkov, Field CTO

Table of Contents

1. Executive Summary

Modern cloud infrastructure demands storage that is fast, scalable, and tightly integrated with compute orchestration layers. This white paper describes a validated integration between Lightbits NVMe/TCP disaggregated storage and Red Hat OpenStack Services on OpenShift (RHOSO) — one of the most sophisticated OpenStack deployment models available today.

RHOSO 18 runs the OpenStack control plane as pods on the OpenShift Container Platform (OCP), while compute workloads are managed as External Data Plane Management (EDPM) nodes running RHEL 9.4. Integrating a high-performance storage backend into this hybrid architecture requires orchestration and integration between the Kubernetes control plane, the bare-metal compute nodes and the Lightbits cluster.

This document provides complete, production-oriented guidance for integrating Lightbits storage as the Cinder block storage backend and as persistent volume storage for Glance image services — including sample YAML manifests, permission requirements, and multi-node scaling considerations.

2. Architecture Overview

2. 1 Component Roles

The integration spans three distinct layers:

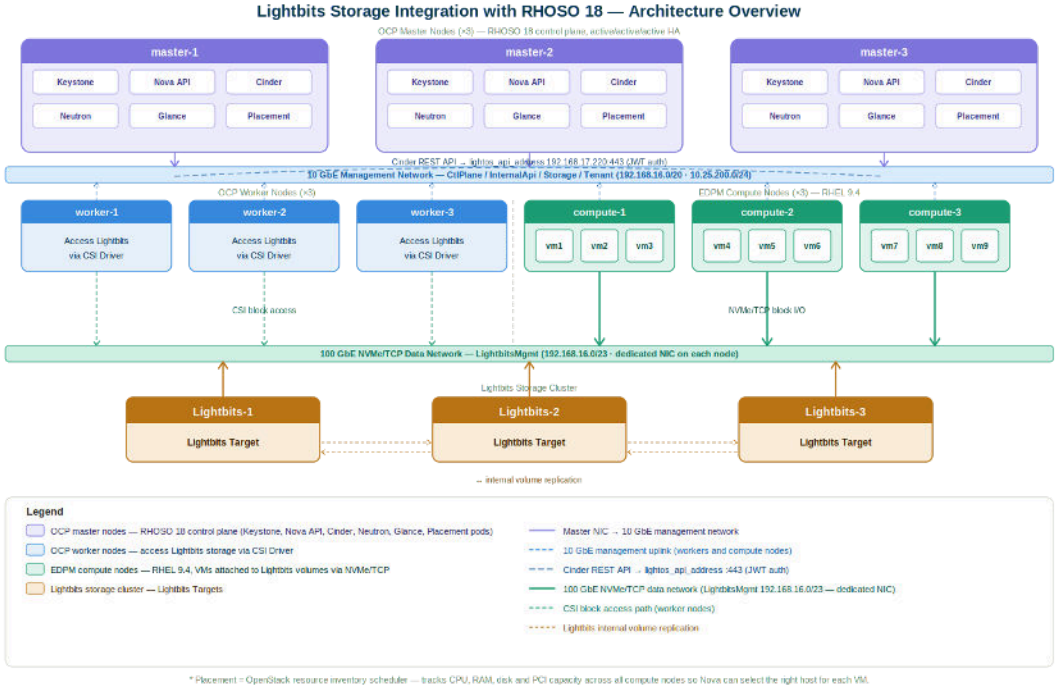
- **OpenShift Control Plane (OCP):** Hosts the RHOSO control plane pods — Cinder, Nova API, Keystone, Glance, Neutron, Placement.
- **EDPM (External Data Plane Management) Compute Nodes (RHEL 9.4):** Pre-provisioned bare-metal or virtual machines managed by the OpenStack Data Plane Management operator—these run nova_compute as a Podman-managed container.
- **Lightbits Cluster:** A dedicated NVMe/TCP storage cluster accessed over a separate management/data network. Cinder communicates with Lightbits via its REST API, while compute nodes connect to volumes using the NVMe/TCP protocol through the Lightbits discovery client.
- **OpenShift worker nodes:** For running regular container-based workloads in Kubernetes, not participating in the RHOSO architecture. Pods on these nodes will use the Lightbits CSI driver to create Persistent Volume Claims (PVCs) for their workloads.

2. 2 Network Architecture

This deployment uses the following logical networks — each maps to a Linux bridge interface on the control-plane host:

Note
The network IP addresses in this example are minimal for a small OpenShift/RHOSO cluster. Change these these settings to fit your network subnets and cluster size.

Network	Example Subnet	Purpose
CtlPlane	192.168.16.0/20	Management, Ansible SSH, EDPM node communication
InternalApi	192.168.0.0/19	OpenStack service-to-service API calls
Storage	10.25.200.0/24	Cinder/Glance storage traffic
Tenant	192.168.0.0/18	OVN tunnel (Geneve) between compute and network nodes
External	192.168.0.0/17	Floating IPs, public API endpoints
LightbitsMgmt	192.168.16.0/23	Dedicated NIC for Lightbits API and NVMe/TCP data



3. Prerequisites & Tested Environment

Before beginning, verify that your environment matches the following baseline. All components listed below were used in the validated reference architecture.

Component	Version / Notes
OpenShift Container Platform	4.18.32
RHOSO (Red Hat OpenStack on OpenShift)	18
Compute OS	RHEL 9.4 (EDPM-managed, pre-provisioned)
Kernel (reference)	5.14.0-427.13.1.el9_4.x86_64
Nova Compute Image	registry.redhat.io/rhoso/openstack-nova-compute-rhel9 (sha256:1a9c15f…)
Lightbits Discovery Client	discovery-client-3.13.1-1~.x86_64.rpm
Glance Storage	Using Lightbits PVC via CSI.

Note on Glance + CSI
Using a CSI-based PVC as Glance storage is functional, but is not officially supported by Red Hat. It is included in this guide as a working example. For production deployments, consult Red Hat for supported Glance storage configurations.

4. Part 1 — Control Plane Setup

The RHOSO control plane runs as Kubernetes pods on the OpenShift control plane nodes. Before applying the Lightbits-specific manifests, the OpenStack networking, IP address pools, and MetalLB configuration must be established. Follow the Red Hat RHOSO installation documentation as your primary reference; the YAML files in the control-plane-setup archive serve as a working reference adapted to the environment described here.

File Execution Order
Apply the control-plane manifests from the control-plane-setup.tgz compressed tarball in the following order:
1. controlplane-nncp.yaml
2.1.ctlplane-IPAddressPool.yaml
2.2.openstack-ipAddressPools.yaml
3.openstack_netconfig.yaml
4.openstack_control_plane_glance_using_pvc.yaml

4. 1 Step 1 — Node Network Configuration Policy (NNCP)

The NodeNetworkConfigurationPolicy (NNCP) configures the Linux bridge interfaces on the OCP control-plane node using the nmstate operator. Each OpenStack network segment maps to one bridge:

```
# 1.controlplane-nncp.yaml (excerpt)
apiVersion: nmstate.io/v1
kind: NodeNetworkConfigurationPolicy
metadata:
  name: osp-flat-networks
spec:
  nodeSelector:
    kubernetes.io/hostname: <your-control-plane-node>
  desiredState:
    interfaces:
      - name: internalapi
        type: linux-bridge
        ipv4:
          address: [{ip: 192.168.25.34, prefix-length: 20}]
          enabled: true
      - name: storage
        type: linux-bridge
        ipv4:
          address: [{ip: 10.25.200.60, prefix-length: 24}]
          enabled: true
      # ... tenant, designate, octavia bridges follow the same pattern
```

Adjust the node name under nodeSelector and all IP addresses to match your environment before applying.

4. 2 Step 2 — MetalLB IP Address Pools

MetalLB provides LoadBalancer-type services for OpenStack API endpoints. Two manifest files are needed: 2.1 creates the control-plane address pool; 2.2 creates pools for all OpenStack networks plus L2Advertisement objects that bind pools to their corresponding bridge interfaces.

```
# 2.2.openstack-ipAddressPools.yaml (excerpt)
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: storage
  namespace: metallb-system
spec:
  addresses:
    - 10.25.200.61-10.25.200.65
  autoAssign: true
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: osp-storage-adv
  namespace: metallb-system
spec:
  ipAddressPools:
    - storage
  interfaces:
    - storage
```

4. 3 Step 3 — OpenStack Network Configuration

The NetConfig resource tells RHOSO about all network segments, their DNS domains, CIDR ranges, and allocation pools. A critical addition for Lightbits is the LightbitsMgmt network:

```
# 3.openstack_netconfig.yaml (excerpt — Lightbits network)
  - name: LightbitsMgmt
    dnsDomain: lightbits.rhoso
    subnets:
    - name: subnet-lightbits
      allocationRanges:
        - {start: 192.168.17.100, end: 192.168.17.120}
      cidr: 192.168.16.0/23
```

This network will be referenced as a networkAttachment in both the Cinder volume service pod and in the EDPM node set.

4. 4 Step 4 — OpenStack Control Plane with Lightbits Cinder Backend

The OpenStackControlPlane resource is the central manifest that defines all OpenStack services. The Lightbits integration requires specific additions to the Cinder and Glance sections.

4. 4.1 Cinder Volume Backend Configuration

A dedicated cinderVolumes backend named "lightbits" is defined with the LightOS volume driver. Key parameters to customize:

- storageClass: The Kubernetes StorageClass backed by Lightbits (e.g., lb-replica1-xfs). Used for Cinder, Galera, and RabbitMQ persistent volumes.
- networkAttachments: Must include both "storage" and "lightbitsmgmt" so the Cinder pod can reach the Lightbits API.
- nodeSelector: Pin the Cinder volume pod to the control-plane node that has the Lightbits NIC.
- lightos_api_address: IP address of the Lightbits cluster REST API endpoint.
- lightos_jwt: JWT token for authenticating to the Lightbits cluster API.
- lightos_default_num_replicas: Replication factor for volumes (1 in this lab; use 3 for production HA).

```
# 4.openstack_control_plane_glance_using_pvc.yaml (Cinder excerpt)
cinderVolumes:
  lightbits:
    template:
      storageClass: lb-replica1-xfs
      replicas: 1
      networkAttachments:
        - storage
        - lightbitsmgmt
      nodeSelector:
        kubernetes.io/hostname: <your-control-plane-node>
      customServiceConfig: |
        [lightos]
        volume_driver = cinder.volume.drivers.lightos.LightOSVolumeDriver
        volume_backend_name = lightos
        lightos_api_address = 192.168.17.220
        lightos_api_port = 443
        lightos_jwt = <your-jwt-token>
        lightos_default_num_replicas = 1
        lightos_default_compression_enabled = False
        lightos_api_service_timeout = 30
```

4. 4.2 Glance with Lightbits PVC Storage

Glance image storage is configured to use a PersistentVolumeClaim backed by the Lightbits storage class. The PVC is mounted inside the GlanceAPI pod at /var/lib/glance/images and referenced via the filesystem store backend:

```
# Glance storage section
glance:
  template:
    storage:
      storageClass: lb-replica1-xfs
      storageRequest: 10Gi
    extraMounts:
      - extraVol:
          - propagation: [GlanceAPI]
            mounts:
              - name: glance-images
                mountPath: /var/lib/glance/images
            volumes:
              - name: glance-images
                persistentVolumeClaim:
                  claimName: glance-images-pvc
    glanceAPIs:
      single:
        customServiceConfig: |
          [DEFAULT]
          enabled_backends = file:file
          [glance_store]
          default_backend = file
          [file]
          filesystem_store_datadir = /var/lib/glance/images
```

5. Part 2 — Nova Compute Node Setup

EDPM compute nodes are pre-installed RHEL 9.4 hosts that run nova_compute as a Podman container. The Lightbits discovery client must be installed and configured on each compute node before the data plane manifests are applied. The discovery client manages the NVMe/TCP connection lifecycle, allowing Nova Compute to discover and connect to Lightbits volumes.

For steps on how to install and set up the Lightbits Discovery Client (or DC for short), please refer to the discovery client deployment guide.

5. 1 Installing the Lightbits Discovery Client

Install the RPM on each compute node:

```
# Install the discovery client RPM
rpm -ivh discovery-client-3.13.1-1~.x86_64.rpm
```

5. 2 Required Files and Permissions

After installation, verify that the following paths exist and have the correct ownership. The nova user and group are created during the standard EDPM compute node setup:

Path	Owner	Permissions
/usr/bin/discovery-client	root:root	0755 (executable)
/etc/discovery-client/	nova:nova	0755
/etc/discovery-client/discovery.d/	nova:nova	drwxrwxrwx (0777)
/etc/discovery-client/discovery-client.yaml	nova:nova	0644
/run/lightos/	root:root	0755

The wide permissions on discovery.d/ (0777) allow nova_compute running as the nova user inside the container to write NVMe discovery entries without requiring elevated privileges.

6. Part 3 — RHOSO Data Plane Setup

With the control plane operational and discovery clients installed on compute nodes, the EDPM data plane can be configured. The five YAML manifests below are applied in order using oc apply -f. They automate the configuration of nova_compute containers to mount Lightbits binaries and directories, configure sudoers for privilege escalation, and deploy the EDPM node set.

Single vs. Multi-Node Deployments
The example files configure a single compute node (192.168.16.195 / rack03-server59). For multiple compute nodes: (1) add additional entries under the "nodes" key in 03-nova-lightbits-nodeset.yaml, and (2) move the vars.yaml content out of 00-edpm-ansible-vars-configmap.yaml and into each node's section in the nodeset manifest so per-node IPs are set correctly.

6. 1 File 00 — Ansible Variables ConfigMap

This ConfigMap holds node-specific network variables consumed by Ansible during EDPM deployment. Each variable maps a logical role (tenant, storage, ctlplane, etc.) to the node's actual IP address on that network:

```
# 00-edpm-ansible-vars-configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: edpm-ansible-vars
  namespace: openstack
data:
  vars.yaml: |
    tenant_ip: 192.168.16.195
    local_ip: 192.168.16.195
    edpm_ovn_tunnel_ip: 192.168.16.195
    internal_api_ip: 192.168.16.195
    storage_ip: 10.25.200.59
    ctlplane_ip: 192.168.16.195
    edpm_tls_certs_enabled: "false"
    ovn_sb_connection: ssl:ovsdbserver-sb.openstack.svc:6642
```

6. 2 File 01 — Nova Lightbits ConfigMap

This ConfigMap injects a nova.conf fragment into the nova_compute container. It configures the privsep_osbrick helper command, which enables the nova process to execute privileged storage operations via sudo without requiring the full container to run as root:

```
# 01-nova-lightbits-configmap.yaml
data:
  nova-20-lightbits.conf: |-
    [privsep_osbrick]
    helper_command = sudo -E /usr/bin/privsep-helper \
      --config-file /etc/nova/nova.conf \
      --config-dir /etc/nova/nova.conf.d
```

6. 3 File 02 — Nova Lightbits Discovery Client Service

This OpenStackDataPlaneService resource includes an Ansible playbook that runs on each compute node after the standard EDPM services are started. It performs the following operations in sequence:

1. Creates a sudoers file (/etc/sudoers.d/nova-lightbits) granting the nova user passwordless access to nova-rootwrap and privsep-helper, with SETENV to preserve environment variables.

2. Patches the nova_compute.json container definition using jq to add the four required Lightbits volume mounts (discovery-client binary, config directory, run/lightos socket directory, sudoers file).
3. Updates the systemd unit file (edpm_nova_compute.service) to use the EDPM wrapper script for container start, ensuring volume mounts are applied correctly at startup.
4. Stops and removes the existing nova_compute container, then recreates it from the patched JSON specification using a Python3 script that translates the JSON definition to a podman create command.
5. Restarts the edpm_nova_compute systemd service and waits up to 150 seconds for the container to reach the running state.
6. Verifies that all four required Lightbits mounts are present in the running container using podman inspect.

Why Patch nova_compute.json?
The EDPM operator generates the nova_compute container definition from templates that do not natively expose Lightbits-specific volume mounts. Patching the JSON at the Ansible level is the prescribed approach for injecting custom mounts into EDPM-managed containers without forking operator code.

6. 4 File 03 — Nova Lightbits NodeSet

The OpenStackDataPlaneNodeSet defines the compute node inventory and the ordered list of services to deploy. The Lightbits-specific additions are:

- networkAttachments: lightbitsmgmt — attaches the Lightbits management network to the node set, enabling network-level reachability to the Lightbits cluster.
- extraMounts — mounts both ConfigMaps (edpm-ansible-vars and nova-compute-lightbits-config) into the Ansible runner pod at the correct paths.
- services list — nova-lightbits-discovery-client is appended after the standard nova service, ensuring the patching playbook runs only after nova_compute is fully initialized.

```
# 03-nova-lightbits-nodeset.yaml (key sections)
spec:
  preProvisioned: true
  networkAttachments:
    - lightbitsmgmt
  nodes:
    rack03-server59:
      hostName: rack03-server59
      networks:
        - name: CtlPlane
          subnetName: subnet-ctlplane
          fixedIP: 192.168.16.195   # replace with your node IP
  services:
    - bootstrap
    - download-cache
```

```
    - configure-os
    - ssh-known-hosts
    - install-certs
    - ovn
    - libvirt
    - nova
    - nova-lightbits-discovery-client   # Lightbits-specific
```

6. 5 File 04 — EDPM Deployment

The OpenStackDataPlaneDeployment triggers the Ansible-based deployment across the node set. It references the node set by name and causes the operator to run all listed services in order:

```
# 04-edmp-deployment.yaml
apiVersion: dataplane.openstack.org/v1beta1
kind: OpenStackDataPlaneDeployment
metadata:
  name: edpm-deployment-rack03
  namespace: openstack
spec:
  nodeSets:
    - openstack-edpm-ipam
```

7. Verification & Troubleshooting

7. 1 Verifying Cinder Backend

After the control plane is running, verify the Lightbits backend is registered and operational:

```
# Check Cinder volume service list
openstack volume service list

# Verify backend details
openstack volume backend pool list

# Create a test volume on the Lightbits backend
openstack volume create --size 10 --type lightos test-vol
```

7.2 Verifying Discovery Client Mounts

On each compute node, confirm the mounts are present in the running nova_compute container:

```
# Check container mounts
podman inspect nova_compute | python3 -c '
import json, sys
binds = json.load(sys.stdin)[0]["HostConfig"]["Binds"]
for b in binds: print(b)
'

# Expected entries:
# /usr/bin/discovery-client:/usr/bin/discovery-client:ro
# /etc/discovery-client:/etc/discovery-client:rw,z
# /run/lightos:/run/lightos:shared,z
# /etc/sudoers.d/nova-lightbits:/etc/sudoers.d/nova:ro
```

7.3 Common Issues

- Cinder pod not starting: Check that the LightbitsMgmt network attachment is correctly defined and that the nodeSelector matches your control-plane node hostname exactly.
- JWT authentication failure: Ensure the lightos_jwt token in the Cinder customServiceConfig is current.
- Volume mounts missing after EDPM deploy: Re-run the Ansible playbook manually by triggering a new OpenStackDataPlaneDeployment resource. Check the Ansible runner pod logs for jq or Python3 errors.
- nova_compute fails to start: Verify that /run/lightos/ exists with root:root ownership before the container starts. The discovery client creates entries here at runtime.


```
# Check container mounts
podman inspect nova_compute | python3 -c '
import json, sys
binds = json.load(sys.stdin)[0]["HostConfig"]["Binds"]
for b in binds: print(b)
'

# Expected entries:
# /usr/bin/discovery-client:/usr/bin/discovery-client:ro
# /etc/discovery-client:/etc/discovery-client:rw,z
# /run/lightos:/run/lightos:shared,z
# /etc/sudoers.d/nova-lightbits:/etc/sudoers.d/nova:ro
```


8. Production Considerations

The reference configuration in this guide is designed for simplicity and validation. For production deployments, consider the following adjustments:

8. 1 Replica Count

The reference uses lightos_default_num_replicas = 1, which means no data redundancy at the storage layer. Production deployments should use a minimum of 3 replicas to protect against drive and node failures within the Lightbits cluster.

8. 2 Management Network Isolation

This guide uses the OCP management network as the RHOSO management network for simplicity. Red Hat recommends a dedicated NIC and subnet for RHOSO management traffic in production to prevent contention and improve security posture.

8. 3 Multi-Node Scaling

For multiple compute nodes, move per-node network variables from the shared 00-edpm-ansible-vars-configmap.yaml into individual node sections within 03-nova-lightbits-nodeset.yaml. Each node requires accurate IP addresses for all networks (ctlplane, storage, tenant, internalapi) to avoid mis-routing of OpenStack traffic.

8. 4 StorageClass and Volume Policy

The lb-replica1-xfs StorageClass shown in the examples maps to a single-replica XFS volume on Lightbits. Define additional StorageClasses with appropriate QoS, compression, and replication settings to match your workload SLAs, and configure Cinder volume types to map to those classes.

9. Summary

This document has walked through a complete, validated integration of Lightbits NVMe/TCP storage with Red Hat OpenStack Services on OpenShift 18. The integration touches every layer of the stack — from nmstate network bridge configuration and MetalLB address pools, through the OpenStack control-plane Cinder driver configuration, down to per-compute-node discovery client installation and Ansible-driven container patching.

The key architectural insight is that Lightbits operates at two levels simultaneously: as a Cinder backend (accessed via the Lightbits REST API from the Cinder volume pod) and as an NVMe/TCP target (connected

directly by nova_compute on each EDPM node via the discovery client). Getting both paths right is essential for full block storage functionality in OpenStack.

The YAML manifests provided are production-oriented starting points. Adjust IP addresses, JWT tokens, node selectors, replica counts, and storage classes to match your specific environment before deploying to production.

10. References

The following YAML files are in compressed tarball format and can be found at:
https://github.com/LightBitsLabs/tech-papers/blob/main/RHOSO/control-plane-setup.tgz
https://github.com/LightBitsLabs/tech-papers/blob/main/RHOSO/data-plane-setup.tgz

10. 1 Control Plane Setup (control-plane-setup.tgz)

- 1.controlplane-nncp.yaml — NMState NodeNetworkConfigurationPolicy for bridge interfaces
- 2.1.ctlplane-IPAddressPool.yaml — MetalLB pool for the CtlPlane network
- 2.2.openstack-ipAddressPools.yaml — MetalLB pools for all OpenStack networks + L2Advertisement
- 3.openstack_netconfig.yaml — RHOSO NetConfig including LightbitsMgmt network
- 4.openstack_control_plane_glance_using_pvc.yaml — Full OpenStackControlPlane manifest with Lightbits Cinder backend and Glance PVC storage

10. 2 Data Plane Setup (data-plane-setup.tgz)

- 00-edpm-ansible-vars-configmap.yaml — Per-node network variable ConfigMap
- 01-nova-lightbits-configmap.yaml — nova.conf privsep fragment for nova_compute
- 02-nova-lightbits-service.yaml — OpenStackDataPlaneService with discovery client mount playbook
- 03-nova-lightbits-nodeset.yaml — OpenStackDataPlaneNodeSet with Lightbits network and service list
- 04-edmp-deployment.yaml — OpenStackDataPlaneDeployment trigger resource

About Lightbits Labs™

Lightbits Labs (Lightbits) is leading the digital data center transformation by making high-performance elastic block storage available to any cloud. Creators of the NVMe® over TCP (NVMe/TCP) protocol, Lightbits software-defined storage is easy to deploy at scale and delivers performance equivalent to local flash to accelerate cloud-native applications in bare metal, virtual, or containerized environments. Backed by leading enterprise investors including Cisco Investments, Dell Technologies Capital, Intel Capital, JP Morgan Chase, Lenovo, and Micron, Lightbits is on a mission to make high-performance elastic block storage simple, scalable and cost-efficient for any cloud.

🌐 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

LBWP25/2026/5