Multi-Attach for OpenStack, Kubernetes and Lightbits: Implementation Framework

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

Abstract:

In today's dynamic cloud landscape, multi-attach capabilities are essential for enabling flexible storage access, improving application availability, and optimizing infrastructure utilization. By integrating OpenStack's robust cloud orchestration and Kubernetes' container management with Lightbits' high-performance, software-defined block storage, organizations can achieve seamless scalability, ultra-low latency, and enterprise-grade data resiliency. This white paper presents a comprehensive framework for implementing multi-attach functionality in OpenStack and Kubernetes environments using Lightbits—empowering businesses to deliver highly available, efficient, and scalable cloud-native services with full control over their storage and compute resources.

1. Introduction

This white paper provides a step-by-step guide to seamlessly integrating Lightbits storage with OpenStack and Kubernetes with multi-attach capabilities per volume from Lightbits. You'll learn how to configure OpenStack and Kubernetes to make that happen. Lightbits itself supports multi-attach to a volume for the OpenStack Cinder driver without any changes. For the Kubernetes CSI driver, specific settings must be applied in the CSI configuration.

The focus of this white paper is on OpenStack volume type, Kubernetes storage class, and Lightbits CSI driver.

2. Prerequisites

Before diving into seamless multi-attach with OpenStack, Kubernetes and Lightbits, a solid foundation is essential. You'll need a fully functional OpenStack deployment or Kubernetes cluster 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 OpenStack services. Lastly, a well-configured network is critical, enabling secure, efficient communication between OpenStack components and Lightbits for optimal performance and scalability. With these key prerequisites in place, you're set to unlock the full potential of multi-attach cloud storage.

3. Lightbits CSI driver

3.1 Lightbits CSI driver overview

The Lightbits CSI driver consists of two pods, the lb-csi-controller and the lb-csi-node. The lb-csi-controller is responsible for:

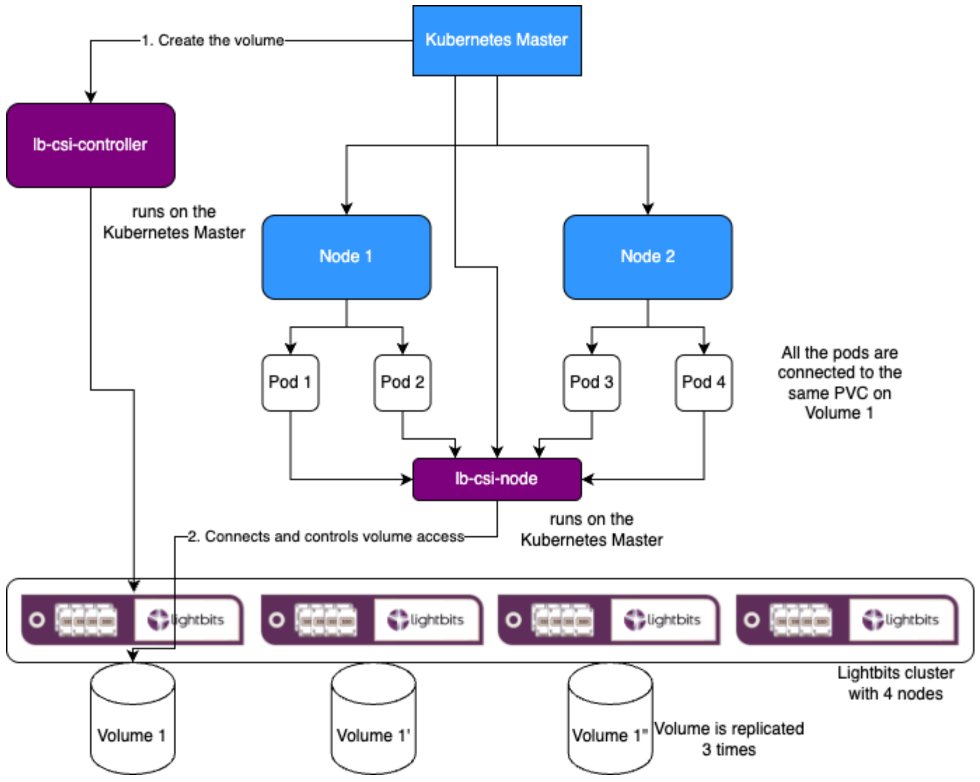
- Creation and deletion of volumes on the Lightbits cluster.
- Making these volumes accessible to the Kubernetes cluster nodes that consume the storage on an as-needed basis.

The lb-csi-node is responsible for:

- Making the storage volumes exported by the Lightbits clusters accessible to the Kubernetes nodes.
- Formatting and checking the file system integrity of the volumes, if necessary.

- Making the volumes accessible to the specific workload pods scheduled to the cluster node in question.

The system is working as follows:


Kubernetes Master
1. Create the volume
lb-csi-controller
runs on the Kubernetes Master
Node 1
Node 2
Pod 1
Pod 2
Pod 3
Pod 4
All the pods are connected to the same PVC on Volume 1
lb-csi-node
2. Connects and controls volume access
runs on the Kubernetes Master
lightbits
Lightbits cluster with 4 nodes
Volume 1
Volume 1'
Volume 1"
Volume is replicated 3 times

Step 1: The CSI driver creates the volume (Volume 1), which is controlled by the lb-csi-controller
Step 2: The CSI driver mounts Volume 1, known in Kubernetes as a PVC
Step 3: The pods claim the same PVC to use

The Lightbits lb-csi-node controls the access to the volume and acts as a middleman between the pods and the volume.

3.2 Configuring the Lightbits CSI driver*

The first step we take is to install the Lightbits CSI driver. The Lightbits CSI driver is open-sourced. To install the driver on the Kubernetes master, do the following:

```
Unset
curl -l -O
```

```
'https://dl.lightbitslabs.com/public/lightos-csi/raw/files/lb-csi-bu
ndle-1.18.0.14010542861.tar.gz'

sudo tar -xvf lb-csi-bundle-1.18.0.14010542861.tar.gz
```

The tar file creates the following directories:
- helm
- k8s
- examples

The file, which needs to be adjusted to make the CSI driver use the RWX (ReadWriteMany) for the volume, is in the directory k8s and is called lb-csi-plugin-k8s-v1.30-dc.yaml. In this file, we need to create two extra entries to enable the RWX for the CSI driver. Please edit the file and add in the containers section the following two lines, (line number 313):

```
- name: LB_CSI_RWX
       value: "true"
```

The service account lb-csi-node-sa section looks like this:

```
serviceAccount: lb-csi-node-sa

containers:
    - name: lb-csi-plugin
     # if hosting the plugin in a different registry, e.g. a local private
     # Docker registry, modify the image identifier below accordingly:
     image: docker.lightbitslabs.com/lightos-csi/lb-csi-plugin:1.17.0
     args :
       - "-P"
     env:
       - name: CSI_ENDPOINT
         value: unix:///csi/csi.sock
       - name: KUBE_NODE_NAME
         valueFrom:
           fieldRef:
             fieldPath: spec.nodeName
       - name: LB_CSI_NODE_ID
         value: $(KUBE_NODE_NAME).node
       - name: LB_CSI_LOG_LEVEL
         value: debug
       - name: LB_CSI_LOG_ROLE
         value: node
       - name: LB_CSI_LOG_FMT
         value: text
       - name: LB_CSI_LOG_TIME
         value: "true"
       - name: LB_CSI_RWX
         value: "true"
```

This also needs to be added to the section serviceAccount: lb-csi-ctrl-sa (line number 448) and looks like the following:

```
    serviceAccount: lb-csi-ctrl-sa
    containers:
     - name: lb-csi-plugin
       # if hosting the plugin in a different registry, e.g., a local private
       # Docker registry, modify the image identifier below accordingly:
       image: docker.lightbitslabs.com/lightos-csi/lb-csi-plugin:1.17.0
       args :
         - "-P"
       env:
       - name: CSI_ENDPOINT
         value: unix:///var/lib/csi/sockets/pluginproxy/csi.sock
       - name: KUBE_NODE_NAME
         valueFrom:
           fieldRef:
             fieldPath: spec.nodeName
       - name: LB_CSI_NODE_ID
         value: $(KUBE_NODE_NAME).ctrl
       - name: LB_CSI_LOG_LEVEL
         value: debug
       - name: LB_CSI_LOG_ROLE
         value: controller
       - name: LB_CSI_LOG_FMT
         value: text
       - name: LB_CSI_LOG_TIME
         value: "true"
       - name: LB_CSI_RWX
```

The next thing is to run the yaml with the kubectl command:

```
Unset
kubectl create -f lb-csi-plugin-k8s-v1.30-dc.yaml
```

The output will be like the following:

```
serviceaccount/lb-csi-ctrl-sa created
serviceaccount/lb-csi-node-sa created
clusterrole.rbac.authorization.k8s.io/external-attacher-runner created
clusterrole.rbac.authorization.k8s.io/lb-csi-external-resizer-runner-role created
clusterrole.rbac.authorization.k8s.io/external-provisioner-runner created
clusterrole.rbac.authorization.k8s.io/external-snapshotter-runner created
clusterrolebinding.rbac.authorization.k8s.io/csi-attacher-role created
clusterrolebinding.rbac.authorization.k8s.io/lb-csi-external-resizer-runner-binding created
clusterrolebinding.rbac.authorization.k8s.io/csi-provisioner-role created
clusterrolebinding.rbac.authorization.k8s.io/csi-snapshotter-role created
role.rbac.authorization.k8s.io/external-provisioner-cfg created
role.rbac.authorization.k8s.io/external-snapshotter-leaderelection created
rolebinding.rbac.authorization.k8s.io/csi-provisioner-role-cfg created
rolebinding.rbac.authorization.k8s.io/external-snapshotter-leaderelection created
daemonset.apps/lb-csi-node created
statefulset.apps/lb-csi-controller created
csidriver.storage.k8s.io/csi.lightbitslabs.com created
```

To verify that the Lightbits pods are running:

```
Unset
kubectl get pod -n kube-system | grep lb
```

The output will be similar to the following:

```
lb-csi-controller-0          5/5   Running   0        7m57s
lb-csi-node-f8tkb            3/3   Running   0        7m57s
```

The CSI driver is now configured to work with RWX.

*For more information on the CSI driver, reference these documents:
https://documentation.lightbitslabs.com/lightbits-plug-ins/lightbits-kubernetes-configuration
https://documentation.lightbitslabs.com/lightbits-plug-ins/static-manifests#rwx-support
https://documentation.lightbitslabs.com/lightbits-plug-ins/multi-attach

4. Configure CSI to use RWX with Lightbits

4.1 Create the storage class to use RWX

The first thing to create is a storage class. The storage class used in this example is called storage-class-block-rwx.yaml. Look at the example below:

```
# Source: lb-csi-workload-examples/charts/storageclass/templates/storageclass.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: lb-sc-block-rwx
provisioner: csi.lightbitslabs.com
allowVolumeExpansion: true
parameters:
  mgmt-endpoint: 192.168.1.41:443,192.168.1.42:443,193.168.1.43:443
  replica-count: "3"
  compression: enabled
  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
```

Create the storage class

```
Unset
kubectl create -f storage-class-block-rwx.yaml
```

The output is as follows

```
persistentvolumeclaim/lightbits-test-pvc-rwx created
```

4.2 Create a Physical Volume Claim

A configuration file is needed to create a Physical Volume Claim (PVC), which sets the volumes to be able to use RWX in block mode. In the yaml file the parameter to set is:

```
spec:
  accessModes:
    - ReadWriteMany
  volumeMode: Block
```

In the following example, we are creating a PVC on Lightbits with RWX capabilities and a size of 20GB. Look at the following example:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: lightbits-test-pvc-rwx
  namespace: default
spec:
  accessModes:
    - ReadWriteMany
  volumeMode: Block
  storageClassName: lb-sc-block-rwx
  resources:
    requests:
      storage: 20Gi
```

In this example, the file is called rwx-pvc.yaml.

Create the PVC

```
Unset
kubeclt create -f rwx-pvc.yaml
```

The output will be:

```
persistentvolumeclaim/lightbits-test-pvc-rwx created
```

To verify the creation of the PVC

```
Unset
kubectl get pvc
```

The output will be:

```
NAME             STATUS   VOLUME                      CAPACITY   ACCESS MODES   STORAGECLASS
VOLUMEATTRIBUTESCLASS   AGE
lightbits-test-pvc-rwx   Bound    pvc-6efb79f7-e94c-4267-8fb0-38fcfdedcafd   20Gi       RWX
lb-sc-block-rwx   <unset>           117s
```

4.3 Create the first Pod and attach it to the PVC

Create the pod with the PVC. In this example, we start with the first Pod. The file is called rwx-pod1.yaml and is configured as follows:

```
kind: Pod
apiVersion: v1
metadata:
  name: example-block-pod-1
spec:
  containers:
  - name: busybox
    image: busybox
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    args:
      - sleep
      - "1000000"
    imagePullPolicy: Always
    volumeDevices:
      - name: lb-csi-mount
        devicePath: /dev/lbcsiblkdev
  restartPolicy: "Never"
  volumes:
    - name: lb-csi-mount
      persistentVolumeClaim:
        claimName: lightbits-test-pvc-rwx
```

Create the pod on the PVC

```
Unset
kubectl create -f rwx-pod1.yaml
```

The output will be as follows:

```
pod/example-block-pod-1 created
```

To verify that the pod is working:

```
Unset
kubectl get pod
```

The output will be

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

To verify that the pod is running on the PVC

```
Unset
kubectl describe pod example-block-pod-1
```

The output will be

```
Volumes:
 lb-csi-mount:
  Type:       PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
  ClaimName:  lightbits-test-pvc-rwx
  ReadOnly:   false
```

4.3 Create the second Pod and attach it to the same PVC

Create the pod with the PVC. In this example, we start with the first Pod. The file is called rwx-pod2.yaml and is configured as follows:

```
kind: Pod
apiVersion: v1
metadata:
  name: example-block-pod-2
spec:
  containers:
  - name: busybox
    image: busybox
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
    args:
      - sleep
      - "1000000"
    imagePullPolicy: Always
    volumeDevices:
      - name: lb-csi-mount
        devicePath: /dev/lbcsiblkdev
  restartPolicy: "Never"
  volumes:
    - name: lb-csi-mount
      persistentVolumeClaim:
        claimName: lightbits-test-pvc-rwx
```

Create the pod on the PVC

```
Unset
kubectl create -f rwx-pod2.yaml
```

The output will be as follows:

```
pod/example-block-pod-2 created
```

To verify that the pod is working

```
Unset
kubectl get pod
```

The output will be

```
NAME              READY  STATUS   RESTARTS  AGE
example-block-pod-1  1/1   Running  0        3m59s
example-block-pod-2  1/1   Running  0        11s
```

To verify that the pod is running on the PVC

```
Unset
kubectl describe pod example-block-pod-2
```

The output will be

```
Volumes:
 lb-csi-mount:
  Type:       PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
  ClaimName:  lightbits-test-pvc-rwx
  ReadOnly:   false
```

To verify that both pods are running on the same PVC

```
Unset
kubectl describe pvc lightbits-test-pvc-rwx
```

The output will be:

```
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      20Gi
Access Modes:  RWX
VolumeMode:    Block
Used By:       example-block-pod-1
               example-block-pod-2
```

5. OpenStack Multi-Attach Configuration

In this chapter, the guidelines will take you through the cli from OpenStack. The multi-attach needs to be configured from the OpenStack environment. By default, the Cinder driver for Lightbits (upstream for OpenStack) supports the multi-attach option. There is no configuration required for Lightbits.

5.1 Create a volume type

Go to the OpenShift Cluster Master and log in as an administrator on the cli to manage the OpenStack environment. To create a new volume type with multi-attach capabilities, with compression enabled, 3 replicas, with the volume type name lb-multi-attach, please type the following command:

Unset

```
openstack volume type create --property multiattach='<is> True' --property
compression='<is> True' --property lightos:num_replicas=3 --property
volume_backend_name=lb-cluster lb-multi-attach
```

Output

```
+---------------+------------------------------------------------------------------------------------------------+
| Field         | Value                                                                                          |
+---------------+------------------------------------------------------------------------------------------------+
| description   | None                                                                                           |
| id            | 659c827c-5d87-4c7a-bd12-a98086f60901                                                           |
| is_public     | True                                                                                           |
| name          | lb-multi-attach                                                                                |
| properties    | compression='<is> True', lightos:num_replicas='3', multiattach='<is> True',
volume_backend_name='lb-cluster'                                                                               |
+-------------+--------------------------------------------------------------------------------------------------+
```

5.2 Create a new volume with volume type lb-multi-attach

The command to create a new volume is as follows:

```
Unset
openstack volume create --size 20 --type lb-multi-attach Vol1
```

Output

```
+-------------------------+------------------------------------------------------+
| Field                   | Value                                                |
+-------------------------+------------------------------------------------------+
| attachments             | []                                                   |
| availability_zone       | nova                                                 |
| bootable                | false                                                |
| consistencygroup_id     | None                                                 |
| created_at              | 2025-04-09T11:50:29.002637                           |
| description             | None                                                 |
| encrypted               | False                                                |
| id                      | 6e5711ab-2a91-4d5f-b17f-5bd499bee09a                 |
| migration_status        | None                                                 |
| multiattach             | True                                                 |
| name                    | Vol1                                                 |
| properties              |                                                      |
| replication_status      | None                                                 |
| size                    | 20                                                   |
| snapshot_id             | None                                                 |
| source_volid            | None                                                 |
| status                  | creating                                             |
| type                    | lb-multi-attach                                      |
| updated_at              | None                                                 |
| user_id                 | b2969eb5499346df88aecf4869040a99                     |
+-------------------------+------------------------------------------------------+
```

5.3 Attach the Volume Vol1 to Instance Demo

The command to create a new volume is as follows:

```
Unset
openstack server add volume Demo Vol1
```

Output

```
+----------------------------+------------------------------------------------------+
| Field                      | Value                                                |
+----------------------------+------------------------------------------------------+
| ID                         | 82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc                 |
| Server ID                  | 7843f583-1fc7-4d5c-91c1-3d9de9e1023a                 |
| Volume ID                  | 82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc                 |
| Device                     | /dev/vdb                                             |
| Tag                        | None                                                 |
| Delete On Termination      | False                                                |
+----------------------------+------------------------------------------------------+
```

5.4 Attach the Volume Vol1 to Instance Backup

The command to create a new volume is as follows:

```
Unset
openstack server add volume Backup Vol1
```

Output

```
+----------------------------+----------------------------------------------------+
| Field                      | Value                                              |
+----------------------------+----------------------------------------------------+
| ID                         | 82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc               |
| Server ID                  | 7cefd02b-97e4-482e-9c91-217ca6e65b2a               |
| Volume ID                  | 82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc               |
| Device                     | /dev/vdb                                           |
| Tag                        | None                                               |
| Delete On Termination      | False                                              |
+----------------------------+----------------------------------------------------+
```

To verify that both instances are running on the same volume

```
openstack volume show Vol1
```

The output will be

```
+---------------------------------------+------------------------------------------------------------------------------------------+
| Field                                  | Value                                                                                    |
+---------------------------------------+------------------------------------------------------------------------------------------+
| attachments              | [{'id': '82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc', 'attachment_id':                          |
|                          | '0b13b195-5ebc-42bd-9070-ce10659829bc', 'volume_id':
'82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc',  |
|                      | 'server_id': '7cefd02b-97e4-482e-9c91-217ca6e65b2a', 'host_name': 'controller', 'device':   |
|                      | '/dev/vdb', 'attached_at': '2025-04-09T12:11:45.000000'}, {'id':                       |
|                      | '82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc', 'attachment_id':                               |
|                      | 'be5bf360-a764-4cec-8216-80ce8c062ef2', 'volume_id':
'82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc',  |
|                      | 'server_id': '7843f583-1fc7-4d5c-91c1-3d9de9e1023a', 'host_name': 'controller', 'device':   |
|                      | '/dev/vdb', 'attached_at': '2025-04-09T12:10:53.000000'}]                                |
| availability_zone                  | nova                                                                                  |
| bootable                           | false                                                                                 |
| consistencygroup_id                | None                                                                                  |
| created_at                         | 2025-04-09T12:09:55.000000                                                            |
| description                        | None                                                                                  |
| encrypted                          | False                                                                                 |
| id                                 | 82b41b69-aa4f-4cc1-a71a-cdc4cb6bf2fc                                                  |
| migration_status                   | None                                                                                  |
| multiattach                        | True                                                                                  |
| name                               | Vol1                                                                                  |
| os-vol-host-attr:host              | controller@lb-cluster#lb-cluster                                                      |
| os-vol-mig-status-attr:migstat     | None                                                                                  |
| os-vol-mig-status-attr:name_id     | None                                                                                  |
| os-vol-tenant-attr:tenant_id       | 0a3da75464774fadb01506bf579c03ed                                                      |
| properties                         |                                                                                       |
| replication_status                 | None                                                                                  |
| size                               | 20                                                                                    |
| snapshot_id                        | None                                                                                  |
| source_volid                       | None                                                                                  |
| status                             | in-use                                                                                |
| type                               | lb-multi-attach                                                                       |
| updated_at                         | 2025-04-09T12:11:48.000000                                                            |
| user_id                            | b2969eb5499346df88aecf4869040a99                                                      |
+---------------------------------------+------------------------------------------------------------------------------------------+
```

6. Conclusion

The integration of Lightbits with CSI for Kubernetes and Cinder for OpenStack, both supporting multi-attachment for block storage, marks a significant leap forward in cloud-native and enterprise cloud infrastructure. This evolution empowers DevOps teams to provision and manage high-performance storage seamlessly, directly within their containerized and virtualized workflows, without needing to dive into traditional storage management. It's about speed, scale, and simplicity—brought together with Lightbits' industry-leading disaggregated, software-defined storage.

With multi-attach capabilities, users can now share a single volume across multiple pods or virtual machines, enabling advanced use cases like clustered applications, shared file systems, and highly available deployments. Whether you're orchestrating Kubernetes workloads or managing OpenStack VMs, Lightbits delivers the flexibility to meet your evolving application needs without compromising on performance or resilience.

What truly sets this integration apart is the native compatibility with the Lightbits API, streamlining the storage provisioning process into a DevOps-friendly, infrastructure-as-code approach. By eliminating the complexity of direct storage administration, DevOps teams gain full control and agility, while Lightbits handles the underlying data services with speed, efficiency, and enterprise-grade reliability.

Ultimately, these integrations are more than technical enhancements—they're a reflection of Lightbits' vision: to enable fast, simple, and scalable storage infrastructure that aligns with the pace of modern development. With this unified, API-driven approach, enterprises can confidently accelerate digital innovation while letting DevOps teams focus on what they do best: building and shipping great software.

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

About Lightbits Labs