Red Hat Enterprise Linux 8.0 Beta

Deduplicating and compressing storage

Using VDO to optimize storage capacity in Red Hat Enterprise Linux 8

Last Updated: 2019-04-17

Red Hat Enterprise Linux 8.0 Beta Deduplicating and compressing storage

Using VDO to optimize storage capacity in Red Hat Enterprise Linux 8

Legal Notice

Abstract

This documentation collection provides instructions on how to use the Virtual Data Optimizer (VDO) to manage deduplicated and compressed storage pools in Red Hat Enterprise Linux 8.

Table of Contents

THIS IS A BETA VERSION!

Thank you for your interest in Red Hat Enterprise Linux 8.0 Beta. Be aware that:

- Beta code should not be used with production data or on production systems.
- Beta does not include a guarantee of support.
- Feedback and bug reports are welcome. Discussions with your account representative, partner contact, and Technical Account Manager (TAM) are also welcome.
- Upgrades to or from a Beta are not supported or recommended.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Please let us know how we could make it better. To do so:

- For simple comments on specific passages, make sure you are viewing the documentation in the Multi-page HTML format. Highlight the part of text that you want to comment on. Then, click the **Add Feedback** pop-up that appears below the highlighted text, and follow the displayed instructions.

- For submitting more complex feedback, create a Bugzilla ticket:

  1. Go to the Bugzilla website.

  2. As the Component, use **Documentation**.

  3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.

  4. Click **Submit Bug**.

CHAPTER 1. GETTING STARTED WITH VDO

As a system administrator, you can use VDO to create deduplicated and compressed storage pools.

This chapter describes the following use cases of VDO deployment:

- the direct-attached use case for virtualization servers, such as those built using Red Hat Virtualization, and

- the cloud storage use case for object-based distributed storage clusters, such as those built using Ceph Storage.

This chapter provides examples for configuring VDO for use with a standard Linux file system that can be easily deployed for either use case.

1.1. INTRODUCTION TO VDO

Virtual Data Optimizer (VDO) provides inline data reduction for Linux in the form of deduplication, compression, and thin provisioning. When you set up a VDO volume, you specify a block device on which to construct your VDO volume and the amount of logical storage you plan to present.

- When hosting active VMs or containers, Red Hat recommends provisioning storage at a 10:1 logical to physical ratio: that is, if you are utilizing 1 TB of physical storage, you would present it as 10 TB of logical storage.

- For object storage, such as the type provided by Ceph, Red Hat recommends using a 3:1 logical to physical ratio: that is, 1 TB of physical storage would present as 3 TB logical storage.

In either case, you can simply put a file system on top of the logical device presented by VDO and then use it directly or as part of a distributed cloud storage architecture.

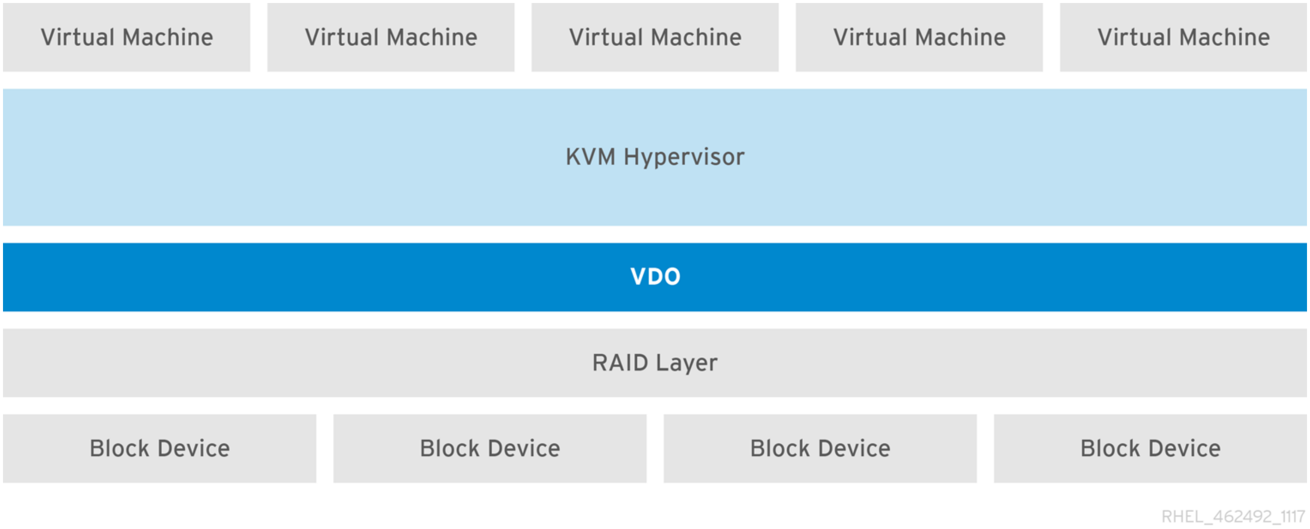
Because VDO is thin provisioned, the file system and applications only see the logical space in use and are not aware of the actual physical space available. Scripting should be used to monitor the actual available space and generate an alert if use exceeds a threshold: for example, when the VDO volume is 80% full. See Section 1.5, “Monitoring VDO” for details.

1.2. VDO DEPLOYMENT EXAMPLES

The following examples illustrate how VDO can be used in KVM deployments.

VDO deployment with KVM

VDO can be deployed successfully on a KVM server configured with Direct Attached Storage:



1.3. INSTALLING VDO

This procedure describes how to install software necessary to create, mount, and manage VDO volumes.

Procedure

- Install the **vdo** and **kmod-kvdo** packages:

```
# yum install vdo kmod-kvdo
```

1.4. CREATING A VDO VOLUME

This procedure describes how to create a VDO volume on a block device.

Note that multiple VDO volumes can be created for separate devices on the same machine. If you choose this approach, you must supply a different name and device for each instance of VDO on the system.

When a VDO volume is created, VDO adds an entry to the **/etc/vdoconf.yml** configuration file. The **vdo.service** systemd unit then uses the entry to start the volume by default.

Prerequisites

- Install the VDO software. See Section 1.3, “Installing VDO”.

Procedure

In all the following steps, replace *vdo-name* with the identifier you want to use for your VDO volume; for example, **vdo1**.

1. Create the VDO volume using the VDO Manager:

```
# vdo create \
      --name=vdo-name \
      --device=block-device \
      --vdoLogicalSize=logical-size
```

- Replace *block-device* with the persistent name of the block device where you want to create the VDO volume. For example, **`/dev/disk/by-id/scsi-3600508b1001c264ad2af21e903ad031f`**.

- Replace *logical-size* with the amount of logical storage that the VDO volume should present:

    - For active VMs or container storage, use logical size that is **ten** times the physical size of your block device. For example, if your block device is 1 TB in size, use **`10T`** here.

    - For object storage, use logical size that is **three** times the physical size of your block device. For example, if your block device is 1 TB in size, use **`3T`** here.

Example 1.1. Creating VDO for container storage

For example, to create a VDO volume for container storage on a 1 TB block device, you might use:

```
# vdo create \
      --name=vdo1 \
      --device=/dev/disk/by-id/scsi-
3600508b1001c264ad2af21e903ad031f \
      --vdoLogicalSize=10T
```

WARNING

If a failure occurs when creating the VDO volume, remove the volume to clean up.

2. Create a file system on top of the VDO volume:

    - For the XFS file system:

    ```
    # mkfs.xfs -K /dev/mapper/vdo-name
    ```

    - For the ext4 file system:

    ```
    # mkfs.ext4 -E nodiscard /dev/mapper/vdo-name
    ```

3. Use the following command to wait for the system to register the new device node:

    ```
    # udevadm settle
    ```

4. Mount the file system:

    ```
    # mkdir -m 1777 /mnt/vdo-name
    # mount /dev/mapper/vdo-name /mnt/vdo-name
    ```

5. To configure the file system to mount automatically at boot, add a line to the **/etc/fstab** file:

    - For the XFS file system:

    ```
    /dev/mapper/vdo-name /mnt/vdo-name xfs defaults,_netdev,x-
    systemd.device-timeout=0,x-systemd.requires=vdo.service 0 0
    ```

    - For the ext4 file system:

    ```
    /dev/mapper/vdo-name /mnt/vdo-name ext4 defaults,_netdev,x-
    systemd.device-timeout=0,x-systemd.requires=vdo.service 0 0
    ```

6. Enable the **discard** feature for the file system on your VDO device. Both batch and online operations work with VDO.

Additional resources

- The **vdo(8)** man page.

1.5. MONITORING VDO

This procedure describes how to obtain usage and efficiency information from a VDO volume.

Prerequisites

- Install the VDO software. See Section 1.3, “Installing VDO”.

Procedure

- Use the **vdostats** utility to get information about a VDO volume:

```
# vdostats --human-readable

Device                   1K-blocks    Used     Available    Use%
Space saving%
/dev/mapper/node1osd1    926.5G       21.0G    905.5G       2%
73%
/dev/mapper/node1osd2    926.5G       28.2G    898.3G       3%
64%
```

Additional resources

- The **vdostats(8)** man page.