redhat.

VDO - Data Reduction for Linux

January 17, 2018

Presented by:

Louis Imershein
Principal Product Manager, Data Reduction Technologies
Red Hat
limershe@redhat.com

What is Virtual Data Optimizer?

- VDO - is a Linux device mapper driver that provides deduplication and compression
- VDO has two kernel modules that work together to provide data efficiency
  - The kvdo module - controls block layer and compression
  - The uds module - maintains deduplication index and provides advice
- VDO can be used under:
  - Block storage
  - File storage
  - Object storage
- Utilities:
  - vdo - Manage VDO volumes (create, remove, modify) and view configurations
  - vdostats - Provides information on statistic and health

Open Source and Upstream Plans

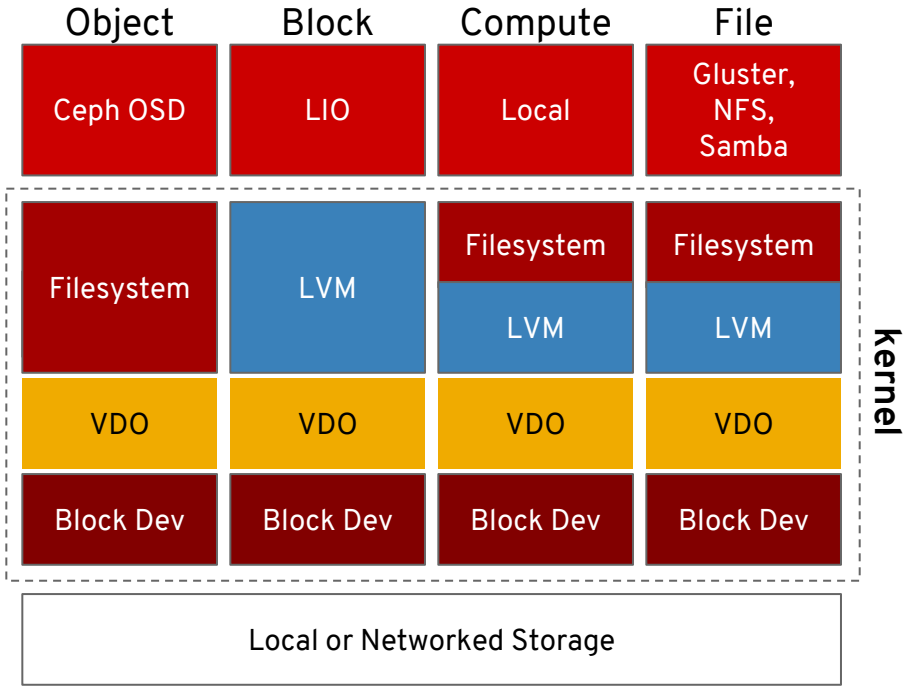
- Open Source and GPL (~165k LOC)
  https://github.com/dm-vdo
- Will ship with RHEL 7.5 as an out-of-tree kernel module
- Working with upstream community to address concerns (Tabs! Spaces! camelCase! under_scores! And some real stuff too)
- Hope to be upstream in 9-12 months

User Benefits

- VDO increases storage efficiency, reduces costs
- VDO makes high-performance storage more affordable
- Because data reduction is applied at the OS layer, VDO works anywhere the OS lives
- OS-based data reduction benefits all software layers above
- Less data to store translates to less data to move - device level replication can be vastly accelerated

Deployment

- Works directly with block devices and filesystems
- Layer other dm modules as desired (dm-crypt, dm-raid)
- In the cloud or on premises
- Addresses all storage deployment models



Install and Configure VDO

- Clone repo, build and load modules *(this should work…)*
- Create VDO volume:
  - `vdo create --name=vdo1 --device=/dev/sdb --vdoLogicalSize=2T`
- Create a file system:
  - `mkfs.xfs -K /dev/mapper/vdo1`
- Mount the file system:
  - `mkdir /mnt/vdo1`
  - `mount -o discard /dev/mapper/vdo1 /mnt/vdo1`

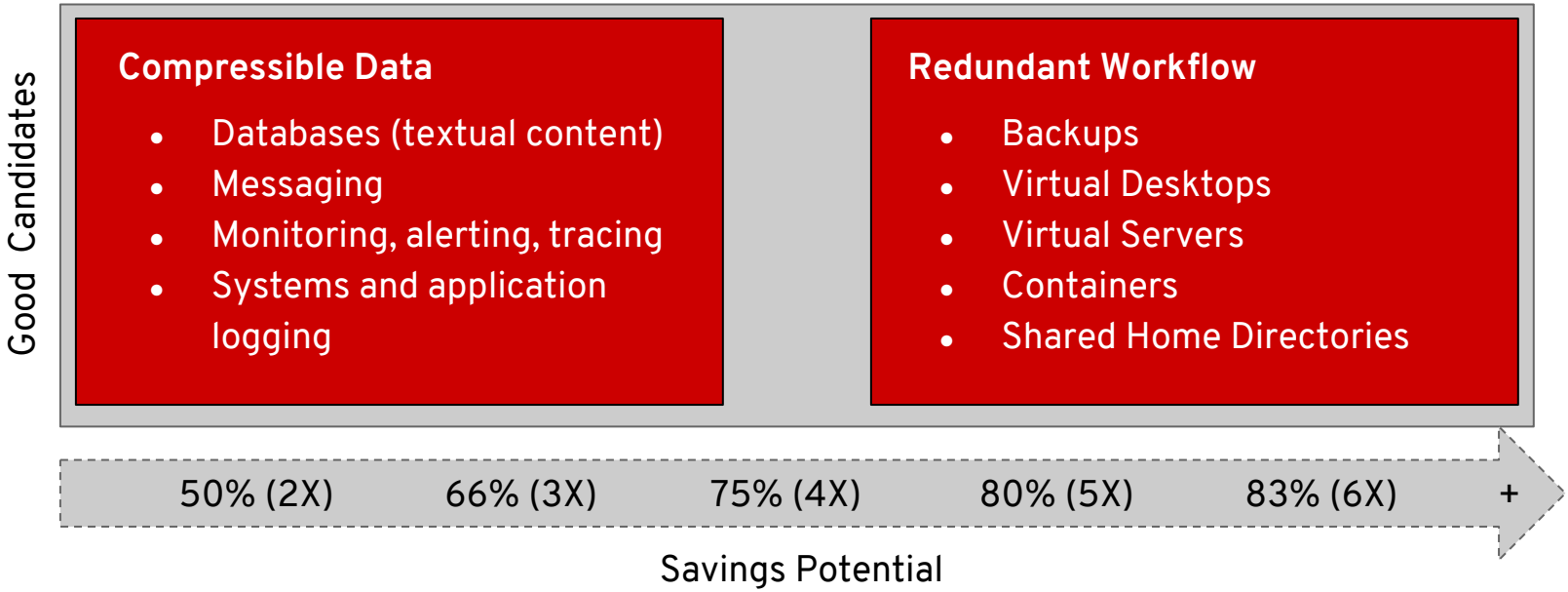
Install and Configure VDO

- Example `vdostats` and `df` results:

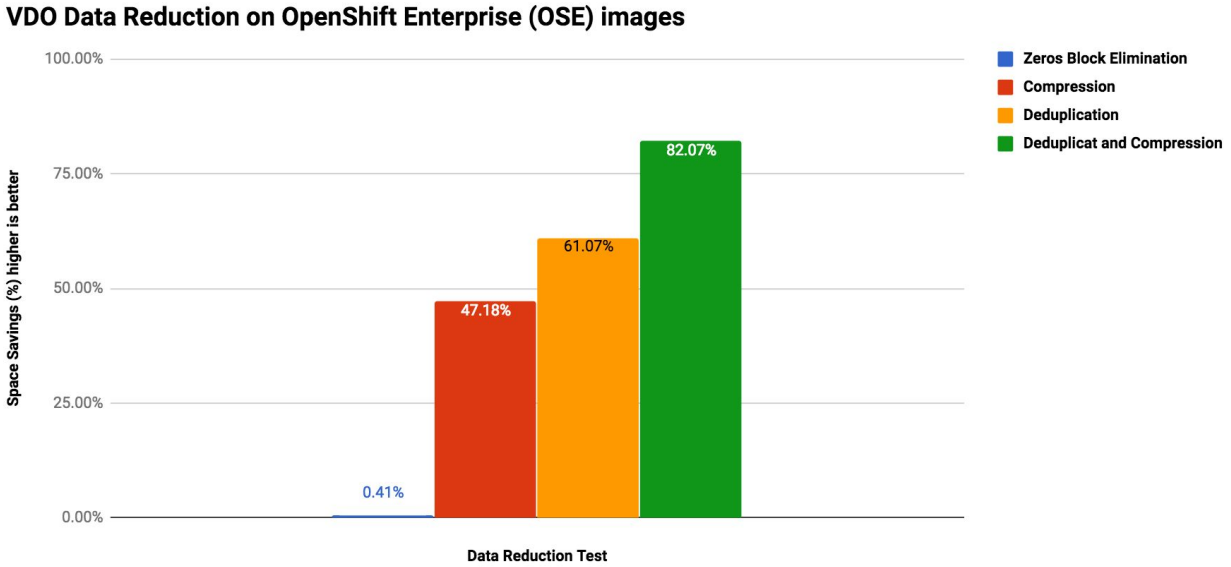
```
    # vdostats
    Device                 1K-blocks      Used  Available Use% Space saving%
    /dev/mapper/vdo1       860463104   1214924 859248180   0%
99%

    # df /mnt/vdo
    Filesystem             1K-blocks      Used   Available Use% Mounted on
    /dev/mapper/vdo1    10735331288     33256 10735298032   1% /mnt/vdo1
```

How Much Can Be Saved?

Dependent on data type and workflow



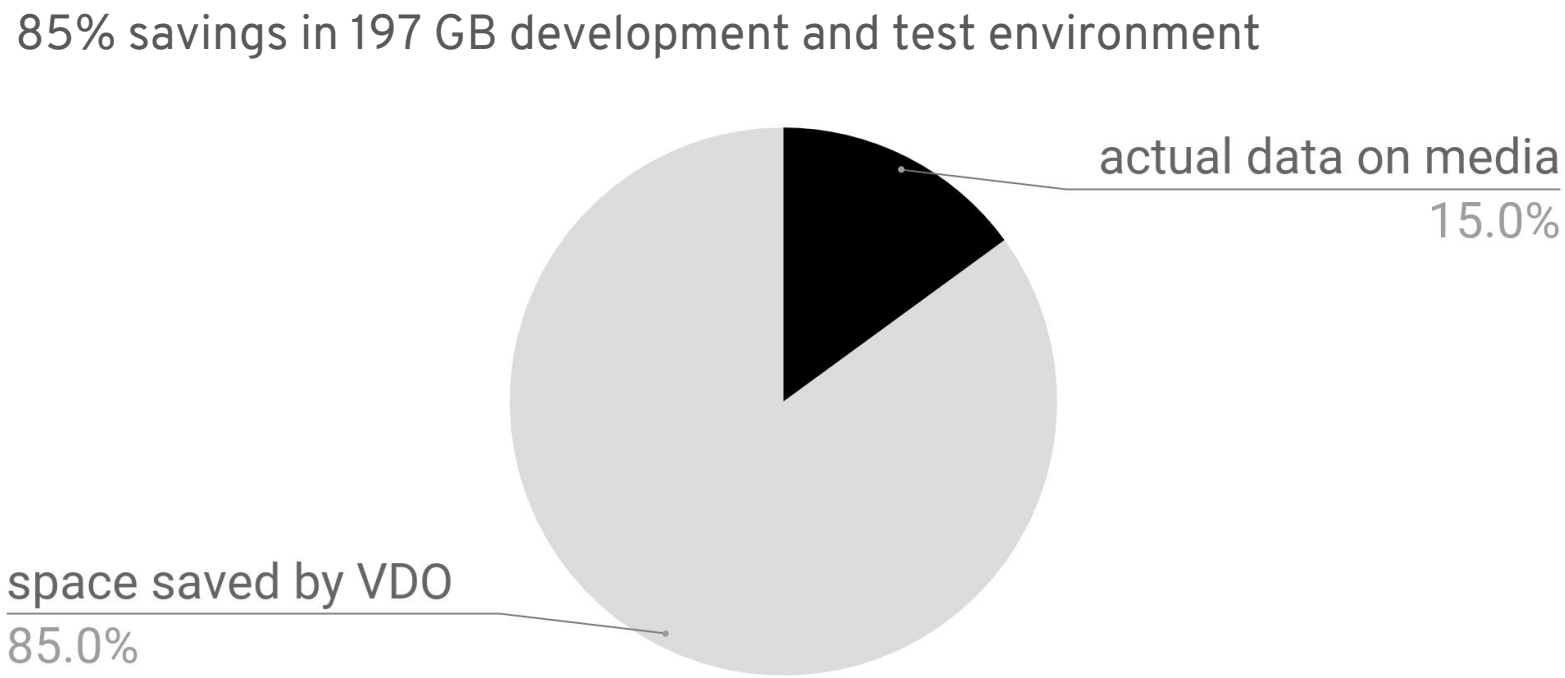
Real-world Results



- Results from 900 container images: 259 GB reduced to 49 GB

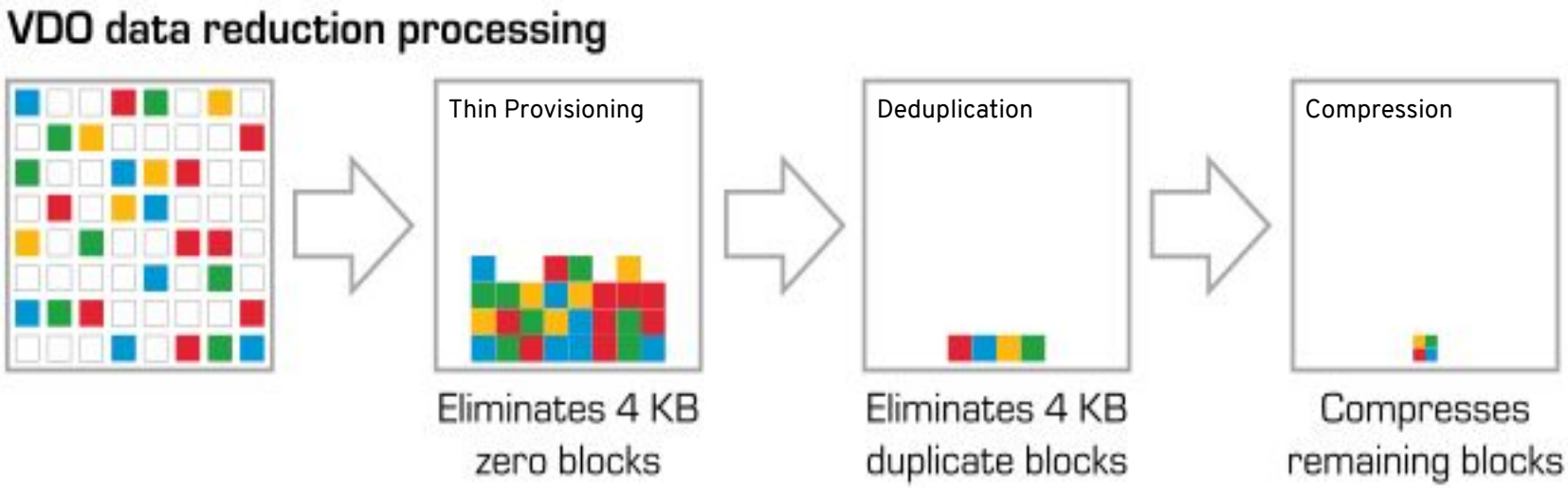
Source: Red Hat testing by Huamin Chen

More Real-world Results



Source: Red Hat testing by Huamin Chen

How It Works



Key Features

- Inline data deduplication, 4 KB granularity
- Inline compression
- Zero-block elimination
- Thin provisioning
- Extend physical storage up to 256 TB (online)
- Extend logical storage up to 4 PB (online)
- Sync and async modes

VDO Requirements

- OS Requirements
  - RHEL 7 (and surely others)
- RAM
  - ~500 MB plus 268 MB / TB physical storage
    - 4 TB physical requires ~1.5 GB RAM
    - 16 TB physical requires ~4.5 GB RAM
- CPU
  - x86_64
  - Other arch coming soon
- Software Requirements
  - Python 2.7+
  - libc 2.7+, libz, zlib1g, PyYAML

Next Steps, Q&A

- Coming soon to Red Hat Enterprise Linux, Fedora Copr and CentOS
- Upstream dm-vdo in the Linux kernel - just getting started
- Integration with other dm management tools e.g. dmeventd
- Integration with stratis - https://github.com/stratis-storage/

- Questions? Complaints? I'm Louis Imershein <limershe@redhat.com>

- Join us at vdo-devel@redhat.com: **https://www.redhat.com/mailman/listinfo/vdo-devel**