Red Hat Enterprise Linux 9.0 Beta

Configuring and managing logical volumes

A guide to the configuration and management of LVM logical volumes
Red Hat Enterprise Linux 9.0 Beta Configuring and managing logical volumes

A guide to the configuration and management of LVM logical volumes
Abstract

This documentation provides instructions on how to manage LVM logical volumes on Red Hat Enterprise Linux 9.
# Table of Contents

**RHEL BETA RELEASE** ......................................................... 3

**MAKING OPEN SOURCE MORE INCLUSIVE** .................................. 4

**PROVIDING FEEDBACK ON RED HAT DOCUMENTATION** .................. 5

**CHAPTER 1. OVERVIEW OF LOGICAL VOLUME MANAGEMENT** .......... 6
  1.1. LVM ARCHITECTURE .............................................. 6
  1.2. ADVANTAGES OF LVM ............................................ 6

**CHAPTER 2. MANAGING LVM PHYSICAL VOLUMES** ....................... 8
  2.1. OVERVIEW OF PHYSICAL VOLUMES .............................. 8
  2.2. MULTIPLE PARTITIONS ON A DISK .............................. 9
  2.3. CREATING LVM PHYSICAL VOLUME ............................ 10
  2.4. REMOVING LVM PHYSICAL VOLUMES ........................... 11

**CHAPTER 3. MANAGING LVM VOLUME GROUPS** .......................... 12
  3.1. CREATING LVM VOLUME GROUP ................................ 12
  3.2. COMBINING LVM VOLUME GROUPS ............................. 13
  3.3. REMOVING PHYSICAL VOLUMES FROM A VOLUME GROUP ..... 13
  3.4. SPLITTING A LVM VOLUME GROUP ............................ 14
  3.5. RENAMING LVM VOLUME GROUPS .............................. 15

**CHAPTER 4. MANAGING LVM LOGICAL VOLUMES** ........................ 17
  4.1. OVERVIEW OF LOGICAL VOLUMES ............................. 17
  4.2. CREATING LVM LOGICAL VOLUME ............................. 18
  4.3. RENAMING LVM LOGICAL VOLUMES ............................ 19
  4.4. REMOVING A DISK FROM A LOGICAL VOLUME ............... 20
  4.5. REMOVING LVM LOGICAL VOLUMES ........................... 21
  4.6. REMOVING LVM VOLUME GROUPS .............................. 22

**CHAPTER 5. MODIFYING THE SIZE OF A LOGICAL VOLUME** ............. 23
  5.1. GROWING A LOGICAL VOLUME AND FILE SYSTEM .............. 23
  5.2. SHRINKING LOGICAL VOLUMES ............................... 25

**CHAPTER 6. TROUBLESHOOTING LVM** .................................. 27
  6.1. GATHERING DIAGNOSTIC DATA ON LVM ...................... 27
  6.2. DISPLAYING INFORMATION ON FAILED LVM DEVICES .......... 28
  6.3. REMOVING LOST LVM PHYSICAL VOLUMES FROM A VOLUME GROUP 29
  6.4. FINDING THE METADATA OF A MISSING LVM PHYSICAL VOLUME ..... 29
  6.5. RESTORING METADATA ON AN LVM PHYSICAL VOLUME .......... 30
  6.6. ROUNDELING ERRORS IN LVM OUTPUT ........................ 32
  6.7. PREVENTING THE ROUNDELING ERROR WHEN CREATING AN LVM VOLUME 32
RHEL BETA RELEASE

Red Hat provides Red Hat Enterprise Linux Beta access to all subscribed Red Hat accounts. The purpose of Beta access is to:

- Provide an opportunity to customers to test major features and capabilities prior to the general availability release and provide feedback or report issues.
- Provide Beta product documentation as a preview. Beta product documentation is under development and is subject to substantial change.

Note that Red Hat does not support the usage of RHEL Beta releases in production use cases. For more information, see What does Beta mean in Red Hat Enterprise Linux and can I upgrade a RHEL Beta installation to a General Availability (GA) release?
MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message.
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:
  1. Make sure you are viewing the documentation in the Multi-page HTML format. In addition, ensure you see the Feedback button in the upper right corner of the document.
  2. Use your mouse cursor to highlight the part of text that you want to comment on.
  3. Click the Add Feedback pop-up that appears below the highlighted text.
  4. 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. OVERVIEW OF LOGICAL VOLUME MANAGEMENT

Logical volume management (LVM) creates a layer of abstraction over physical storage, which helps you to create logical storage volumes. This provides much greater flexibility in a number of ways than using physical storage directly.

In addition, the hardware storage configuration is hidden from the software so it can be resized and moved without stopping applications or unmounting file systems. This can reduce operational costs.

1.1. LVM ARCHITECTURE

The following are the components of LVM:

Physical volume
A physical volume (PV) is a partition or whole disk designated for LVM use. For more information, see Managing LVM physical volumes.

Volume group
A volume group (VG) is a collection of physical volumes (PVs), which creates a pool of disk space out of which logical volumes can be allocated. For more information, see Managing LVM volume groups.

Logical volume
A logical volume represents a mountable storage device. For more information, see Managing LVM logical volumes.

The following Figure 1.1, "LVM logical volume components" illustrates the components of LVM:

Figure 1.1. LVM logical volume components

1.2. ADVANTAGES OF LVM

Logical volumes provide the following advantages over using physical storage directly:

Flexible capacity
When using logical volumes, you can aggregate devices and partitions into a single logical volume. With this functionality, file systems can extend across multiple devices as though they were a single, large one.

**Resizeable storage volumes**

You can extend logical volumes or reduce logical volumes in size with simple software commands, without reformatting and repartitioning the underlying devices.

**Online data relocation**

To deploy newer, faster, or more resilient storage subsystems, you can move data while your system is active. Data can be rearranged on disks while the disks are in use. For example, you can empty a hot-swappable disk before removing it.

**Convenient device naming**

Logical storage volumes can be managed with user-defined and custom names.

**Striped Volumes**

You can create a logical volume that stripes data across two or more devices. This can dramatically increase throughput.

**RAID volumes**

Logical volumes provide a convenient way to configure RAID for your data. This provides protection against device failure and improves performance.

**Volume snapshots**

You can take snapshots, which is a point-in-time copy of logical volumes for consistent backups or to test the effect of changes without affecting the real data.

**Thin volumes**

Logical volumes can be thinly provisioned. This allows you to create logical volumes that are larger than the available physical space.

**Cache volumes**

A cache logical volume uses a fast block device, such as an SSD drive, to improve the performance of a larger and slower block device.
CHAPTER 2. MANAGING LVM PHYSICAL VOLUMES

The physical volume (PV) is a partition or whole disk designated for LVM use. To use the device for an LVM logical volume, the device must be initialized as a physical volume.

If you are using a whole disk device for your physical volume, the disk must have no partition table. For DOS disk partitions, the partition id should be set to 0x8e using the `fdisk` or `cfdisk` command or an equivalent. For whole disk devices only the partition table must be erased, which will effectively destroy all data on that disk. You can remove an existing partition table by zeroing the first sector by using the `dd if=/dev/zero of=<PhysicalVolume> bs=512 count=1` command as root.

2.1. OVERVIEW OF PHYSICAL VOLUMES

Initializing a block device as a physical volume places a label near the start of the device. The following describes the LVM label:

- An LVM label provides correct identification and device ordering for a physical device. An unlabeled, non–LVM device can change names across reboots depending on the order they are discovered by the system during boot. An LVM label remains persistent across reboots and throughout a cluster.

- The LVM label identifies the device as an LVM physical volume. It contains a random unique identifier, the UUID for the physical volume. It also stores the size of the block device in bytes, and it records where the LVM metadata will be stored on the device.

- By default, the LVM label is placed in the second 512-byte sector. You can overwrite this default setting by placing the label on any of the first 4 sectors when you create the physical volume. This allows LVM volumes to co-exist with other users of these sectors, if necessary.

The following describes the LVM metadata:

- The LVM metadata contains the configuration details of the LVM volume groups on your system. By default, an identical copy of the metadata is maintained in every metadata area in every physical volume within the volume group. LVM metadata is small and stored as ASCII.

- Currently LVM allows you to store 0, 1, or 2 identical copies of its metadata on each physical volume. The default is 1 copy. Once you configure the number of metadata copies on the physical volume, you cannot change that number at a later time. The first copy is stored at the start of the device, shortly after the label. If there is a second copy, it is placed at the end of the device. If you accidentally overwrite the area at the beginning of your disk by writing to a different disk than you intend, a second copy of the metadata at the end of the device will allow you to recover the metadata.

The following Figure 2.1, “Physical volume layout” illustrates the layout of an LVM physical volume. The LVM label is on the second sector, followed by the metadata area, followed by the usable space on the device.

NOTE

In the Linux kernel and throughout this document, sectors are considered to be 512 bytes in size.
2.2. MULTIPLE PARTITIONS ON A DISK

You can create physical volumes (PV) out of disk partitions by using LVM.

Red Hat recommends that you create a single partition that covers the whole disk to label as an LVM physical volume for the following reasons:

**Administrative convenience**

It is easier to keep track of the hardware in a system if each real disk only appears once. This becomes particularly true if a disk fails.

**Striping performance**

LVM cannot tell that two physical volumes are on the same physical disk. If you create a striped logical volume when two physical volumes are on the same physical disk, the stripes could be on different partitions on the same disk. This would result in a decrease in performance rather than an increase.

**RAID redundancy**

LVM cannot determine that the two physical volumes are on the same device. If you create a RAID logical volume when two physical volumes are on the same device, performance and fault tolerance could be lost.

Although it is not recommended, there may be specific circumstances when you will need to divide a disk into separate LVM physical volumes. For example, on a system with few disks it may be necessary to move data around partitions when you are migrating an existing system to LVM volumes. Additionally, if you have a very large disk and want to have more than one volume group for administrative purposes then it is necessary to partition the disk. If you do have a disk with more than one partition and both of those partitions are in the same volume group, take care to specify which partitions are to be included in a logical volume when creating volumes.

Note that although LVM supports using a non-partitioned disk as physical volume, it is recommended to create a single, whole-disk partition because creating a PV without a partition can be problematic in a mixed operating system environment. Other operating systems may interpret the device as free, and overwrite the PV label at the beginning of the drive.
2.3. CREATING LVM PHYSICAL VOLUME

This procedure describes how to create and label LVM physical volumes (PVs).

In this procedure, replace the /dev/vdb1, /dev/vdb2, and /dev/vdb3 with the available storage devices in your system.

Prerequisites

- The lvm2 package is installed.

Procedure

1. Create multiple physical volumes by using the space-delimited device names as arguments to the `pvcreate` command:

   ```bash
   # pvcreate /dev/vdb1 /dev/vdb2 /dev/vdb3
   Physical volume "/dev/vdb1" successfully created.
   Physical volume "/dev/vdb2" successfully created.
   Physical volume "/dev/vdb3" successfully created.
   ```

   This places a label on /dev/vdb1, /dev/vdb2, and /dev/vdb3, marking them as physical volumes belonging to LVM.

2. View the created physical volumes by using any one of the following commands as per your requirement:

   a. The `pvdisplay` command, which provides a verbose multi-line output for each physical volume. It displays physical properties, such as size, extents, volume group, and other options in a fixed format:

   ```bash
   # pvdisplay
   --- NEW Physical volume ---
   PV Name  /dev/vdb1
   VG Name
   PV Size   1.00 GiB
   [...]
   --- NEW Physical volume ---
   PV Name  /dev/vdb2
   VG Name
   PV Size   1.00 GiB
   [...]
   --- NEW Physical volume ---
   PV Name  /dev/vdb3
   VG Name
   PV Size   1.00 GiB
   [...]
   ```

   b. The `pvs` command provides physical volume information in a configurable form, displaying one line per physical volume:

   ```bash
   # pvs
   PV     VG  Fmt  Attr  PSize  PFree
   /dev/vdb1 lvm2  1020.00m  0
   ```
c. The `pvscan` command scans all supported LVM block devices in the system for physical volumes. You can define a filter in the `lvm.conf` file so that this command avoids scanning specific physical volumes:

```
# pvscan
PV /dev/vdb1  lvm2 [1.00 GiB]
PV /dev/vdb2  lvm2 [1.00 GiB]
PV /dev/vdb3  lvm2 [1.00 GiB]
```

### Additional resources

- `pvcreate(8)`, `pvdisplay(8)`, `pvs(8)`, and `pvscan(8)` man pages

## 2.4. REMOVING LVM PHYSICAL VOLUMES

If a device is no longer required for use by LVM, you can remove the LVM label by using the `pvremove` command. Executing the `pvremove` command zeroes the LVM metadata on an empty physical volume.

### Procedure

1. Remove a physical volume:

   ```
   # pvremove /dev/vdb3
   Labels on physical volume "/dev/vdb3" successfully wiped.
   ```

2. View the existing physical volumes and verify if the required volume is removed:

   ```
   # pvs
   PV      VG   Fmt    Attr  PSize    PFree
   /dev/vdb1  lvm2         1020.00m  0
   /dev/vdb2  lvm2         1020.00m  0
   ```

   If the physical volume you want to remove is currently part of a volume group, you must remove it from the volume group with the `vgreduce` command. For more information, see [Removing physical volumes from a volume group](#)

### Additional resources

- `pvremove(8)` man page
A volume group (VG) is a collection of physical volumes (PVs), which creates a pool of disk space out of which logical volumes (LVs) can be allocated.

Within a volume group, the disk space available for allocation is divided into units of a fixed-size called extents. An extent is the smallest unit of space that can be allocated. Within a physical volume, extents are referred to as physical extents.

A logical volume is allocated into logical extents of the same size as the physical extents. The extent size is thus the same for all logical volumes in the volume group. The volume group maps the logical extents to physical extents.

3.1. CREATING LVM VOLUME GROUP

This procedure describes how to create an LVM volume group (VG) `myvg`, by using the `/dev/vdb1` and `/dev/vdb2` physical volumes.

**Prerequisites**

- The `lvm2` package is installed.
- One or more physical volumes are created. For more information on creating physical volumes, see Creating LVM physical volume.

**Procedure**

1. Create a volume group:

   ```bash
   # vgcreate myvg /dev/vdb1 /dev/vdb2
   Volume group "myvg" successfully created.
   ```

   This creates a VG with the name of `myvg`. The PVs `/dev/vdb1` and `/dev/vdb2` are the base storage level for the `myvg` VG.

2. View the created volume groups by using any one of the following commands as per your requirement:

   a. The `vgs` command provides volume group information in a configurable form, displaying one line per volume groups:

   ```bash
   # vgs
   VG  #PV #LV #SN Attr  VSize   VFree
   myvg  4   1   0 wz--n-   3.98g 1008.00m
   ```

   b. The `vgdisplay` command displays volume group properties such as size, extents, number of physical volumes, and other options in a fixed form. The following example shows the output of the `vgdisplay` command for the volume group `myvg`. If you do not specify a volume group, all existing volume groups are displayed:

   ```bash
   # vgdisplay myvg _ --- Volume group --- VG Name _myvg
   System ID
   Format        lvm2
   Metadata Areas  4
   ```
c. The `vgscan` command scans all supported LVM block devices in the system for volume group:

```
# vgscan
Found volume group "myvg" using metadata type lvm2
```

3. Optional: Increase a volume group’s capacity by adding one or more free physical volumes:

```
# vgextend myvg /dev/vdb3
Physical volume "/dev/vdb3" successfully created.
Volume group "myvg" successfully extended
```

**Additional resources**
- `pvcreate(8)`, `vgextend(8)`, `vgdisplay(8)`, `vgs(8)`, and `vgscan(8)` man pages

### 3.2. COMBINING LVM VOLUME GROUPS

To combine two volume groups into a single volume group, use the `vgmerge` command. You can merge an inactive “source” volume with an active or an inactive “destination” volume if the physical extent sizes of the volume are equal and the physical and logical volume summaries of both volume groups fit into the destination volume groups limits.

**Procedure**

- Merge the inactive volume group databases into the active or inactive volume group `myvg` giving verbose runtime information:

```
# vgmerge -v myvg databases
```

**Additional resources**
- `vgmerge(8)` man page

### 3.3. REMOVING PHYSICAL VOLUMES FROM A VOLUME GROUP

To remove unused physical volumes from a volume group, use the `vgreduce` command. The `vgreduce` command shrinks a volume group’s capacity by removing one or more empty physical volumes. This frees those physical volumes to be used in different volume groups or to be removed from the system.

**Procedure**

1. If the physical volume is still being used, migrate the data to another physical volume from the same volume group:

```
# pvmove /dev/vdb3
/dev/vdb3: Moved: 2.0%
... 
```
2. If there are not enough free extents on the other physical volumes in the existing volume group:
   a. Create a new physical volume from `/dev/vdb4`:
      ```
      # pvcreate /dev/vdb4
      Physical volume "/dev/vdb4" successfully created
      ```
   b. Add the newly created physical volume to the `myvg` volume group:
      ```
      # vgextend myvg /dev/vdb4
      Volume group "myvg" successfully extended
      ```
   c. Move the data from `/dev/vdb3` to `/dev/vdb4`:
      ```
      # pvmove /dev/vdb3 /dev/vdb4
      /dev/vdb3: Moved: 33.33%
      /dev/vdb3: Moved: 100.00%
      ```
3. Remove the physical volume `/dev/vdb3` from the volume group:
   ```
   # vgreduce myvg /dev/vdb3
   Removed "/dev/vdb3" from volume group "myvg"
   ```

**Verification**

- Verify if the `/dev/vdb3` physical volume is removed from the `myvg` volume group:
  ```
  # pvs
  PV       VG    Fmt  Attr  PSize   PFree   Used
  /dev/vdb1 myvg lvm2 a--  1020.00m  0   1020.00m
  /dev/vdb2 myvg lvm2 a--  1020.00m  0   1020.00m
  /dev/vdb3  lvm2 a--  1020.00m  1008.00m  12.00m
  ```

**Additional resources**

- `vgreduce(8)`, `pvmove(8)`, and `pvs(8)` man pages

**3.4. SPLITTING A LVM VOLUME GROUP**

This procedure describes how to split the existing volume group. If there is enough unused space on the physical volumes, a new volume group can be created without adding new disks.

In the initial setup, the volume group `myvg` consists of `/dev/vdb1`, `/dev/vdb2`, and `/dev/vdb3`. After completing this procedure, the volume group `myvg` will consist of `/dev/vdb1` and `/dev/vdb2`, and the second volume group, `yourvg`, will consist of `/dev/vdb3`.

**Prerequisites**

- [Red Hat Enterprise Linux 9.0 Beta Configuring and managing logical volumes](https://example.com)
You have sufficient space in the volume group. Use the `vgscan` command to determine how much free space is currently available in the volume group.

Depending on the free capacity in the existing physical volume, move all the used physical extents to other physical volume using the `pvmove` command. For more information, see Removing physical volumes from a volume group.

**Procedure**

1. Split the existing volume group `myvg` to the new volume group `yourvg`:
   ```
   # vgsplit myvg yourvg /dev/vdb3
   Volume group "yourvg" successfully split from "myvg"
   ```

   **NOTE**
   
   If you have created a logical volume using the existing volume group, use the following command to deactivate the logical volume:
   ```
   # lvchange -a n /dev/myvg/mylv
   ```
   
   For more information on creating logical volumes, see Managing LVM logical volumes.

2. View the attributes of the two volume group:
   ```
   # vgs
   VG     #PV #LV #SN Attr   VSize  VFree
   myvg   2   1   0 wz--n- 34.30G 10.80G
   yourvg 1   0   0 wz--n- 17.15G 17.15G
   ```

**Verification**

- Verify if the newly created volume group `yourvg` consists of `/dev/vdb3` physical volume:
  ```
  # pvs
  PV           VG      Fmt   Attr   PSize        PFree      Used
  /dev/vdb1 myvg lvm2   a--  1020.00m    0          1020.00m
  /dev/vdb2 myvg lvm2   a--  1020.00m    0          1020.00m
  /dev/vdb3 yourvg lvm2 a--  1020.00m 1008.00m  12.00m
  ```

**Additional resources**

- `vgsplit(8)`, `vgs(8)`, and `pvs(8)` man pages

### 3.5. RENAMING LVM VOLUME GROUPS

This procedure renames an existing volume group `myvg` to `myvg1`.

**Procedure**
1. Deactivate the volume group. If it is a clustered volume group, deactivate the volume group on all nodes where it is active by using the following command on each such node:

```
# vgchange --activate n myvg
```

2. Rename an existing volume group:

```
# vgrename myvg myvg1
Volume group "myvg" successfully renamed to "myvg1"
```

You can also rename the volume group by specifying the full paths to the devices:

```
# vgrename /dev/myvg /dev/myvg1
```

**Additional resources**

- `vgrename(8)` man page
CHAPTER 4. MANAGING LVM LOGICAL VOLUMES

A logical volume is a virtual, block storage device that a file system, database, or application can use. To create an LVM logical volume, the physical volumes (PVs) are combined into a volume group (VG). This creates a pool of disk space out of which LVM logical volumes (LVs) can be allocated.

4.1. OVERVIEW OF LOGICAL VOLUMES

An administrator can grow or shrink logical volumes without destroying data, unlike standard disk partitions. If the physical volumes in a volume group are on separate drives or RAID arrays, then administrators can also spread a logical volume across the storage devices.

You can lose data if you shrink a logical volume to a smaller capacity than the data on the volume requires. Further, some file systems are not capable of shrinking. To ensure maximum flexibility, create logical volumes to meet your current needs, and leave excess storage capacity unallocated. You can safely extend logical volumes to use unallocated space, depending on your needs.

IMPORTANT

On AMD, Intel, ARM systems, and IBM Power Systems servers, the boot loader cannot read LVM volumes. You must make a standard, non-LVM disk partition for your /boot partition. On IBM Z, the zipl boot loader supports /boot on LVM logical volumes with linear mapping. By default, the installation process always creates the / and swap partitions within LVM volumes, with a separate /boot partition on a physical volume.

The following are the different types of logical volumes:

Linear volumes

A linear volume aggregates space from one or more physical volumes into one logical volume. For example, if you have two 60GB disks, you can create a 120GB logical volume. The physical storage is concatenated.

Striped logical volumes

When you write data to an LVM logical volume, the file system lays the data out across the underlying physical volumes. You can control the way the data is written to the physical volumes by creating a striped logical volume. For large sequential reads and writes, this can improve the efficiency of the data I/O.

Striping enhances performance by writing data to a predetermined number of physical volumes in round-robin fashion. With striping, I/O can be done in parallel. In some situations, this can result in near-linear performance gain for each additional physical volume in the stripe.

RAID logical volumes

LVM supports RAID levels 0, 1, 4, 5, 6, and 10. RAID logical volumes are not cluster-aware. When you create a RAID logical volume, LVM creates a metadata subvolume that is one extent in size for every data or parity subvolume in the array.

Thin-provisioned logical volumes (thin volumes)

Using thin-provisioned logical volumes, you can create logical volumes that are larger than the available physical storage. Creating a thinly provisioned set of volumes allows the system to allocate what you use instead of allocating the full amount of storage that is requested.

Snapshot volumes

The LVM snapshot feature provides the ability to create virtual images of a device at a particular instant without causing a service interruption. When a change is made to the original device (the
origin) after a snapshot is taken, the snapshot feature makes a copy of the changed data area as it was prior to the change so that it can reconstruct the state of the device.

**Thin-provisioned snapshot volumes**

Using thin-provisioned snapshot volumes, you can have more virtual devices to be stored on the same data volume. Thinly provisioned snapshots are useful because you are not copying all of the data that you are looking to capture at a given time.

**Cache volumes**

LVM supports the use of fast block devices, such as SSD drives as write-back or write-through caches for larger slower block devices. Users can create cache logical volumes to improve the performance of their existing logical volumes or create new cache logical volumes composed of a small and fast device coupled with a large and slow device.

### 4.2. CREATING LVM LOGICAL VOLUME

This procedure describes how to create `mylv` LVM logical volume (LV) from the `myvg` volume group, which is created by using the `/dev/vdb1`, `/dev/vdb2`, and `/dev/vdb3` physical volumes.

**Prerequisites**

- The `lvm2` package is installed.
- The volume group is created. For more information, see Creating LVM volume group.

**Procedure**

1. Create a logical volume:

   ```bash
   # lvcreate -n mylv -L 500M myvg
   ```

   Use the `-n` option to set the LV name to `mylv`, and the `-L` option to set the size of LV in units of Mb, but it is possible to use any other units. The LV type is linear by default, but the user can specify the desired type by using the `--type` option.

   **IMPORTANT**

   The command fails if the VG does not have a sufficient number of free physical extents for the requested size and type.

2. View the created logical volumes by using any one of the following commands as per your requirement:

   a. The `lvs` command provides logical volume information in a configurable form, displaying one line per logical volume:

      ```bash
      # lvs
      LV  VG  Attr  LSize   Pool Origin Data%  Meta% Move Log Cpy%Sync Convert
      mylv myvg -wi-ao----  500.00m
      ```

   b. The `lvdisplay` command displays logical volume properties, such as size, layout, and mapping in a fixed format:

      ```bash
      # lvdisplay -v /dev/myvg/mylv
      ```
--- Logical volume ---
LV Path    /dev/myvg/mylv
LV Name    mylv
VG Name    myvg
LV UUID    YTnAk6-kMlT-c4pG-HBFZ-Bx7t-ePMk-7YjhaM
LV Write Access    read/write

3. Create a file system on the logical volume. The following command creates an xfs file system on the logical volume:

```
# mkfs.xfs /dev/myvg/mylv
```

```
meta-data=/dev/myvg/mylv isize=512 agcount=4, agsize=32000 blks
  =        sectsz=512 attr=2, projid32bit=1
  =        crc=1     finobt=1, sparse=1, rmapbt=0
  =        reflink=1
data     =        bsize=4096 blocks=128000, imaxpct=25
  =        sunit=0  swidth=0 blks
naming   =version 2 bsize=4096 ascii-ci=0, ftype=1
log      =internal log bsize=4096 blocks=1368, version=2
  =        sectsz=512 sunit=0 blks, lazy-count=1
realtime =none extsz=4096 blocks=0, rtextents=0
Discarding blocks...Done.
```

4. Mount the logical volume and report the file system disk space usage:

```
# mount /dev/myvg/mylv /mnt

# df -h
Filesystem  1K-blocks  Used  Available  Use%  Mounted on
/dev/mapper/myvg-mylv  506528  29388  477140    6%    /mnt
```

Additional resources

- lvcreate(8), lvdisplay(8), lvs(8), lvscan(8), and mkfs.xfs(8) man pages

### 4.3. RENAMING LVM LOGICAL VOLUMES

This procedure describes how to rename an existing logical volume mylv to mylv1.

**Procedure**

1. If the logical volume is currently mounted, unmount the volume:

```
# umount /mnt
```

Replace /mnt with the mount point.
2. If the logical volume exists in a clustered environment, deactivate the logical volume on all
nodes where it is active. Use the following command on each such node:

```bash
# lvchange --activate n myvg/mylv
```

3. Rename an existing logical volume:

```bash
# lvrename myvg mylv mylv1
Logical volume "mylv" successfully renamed to "mylv1"
```

You can also rename the logical volume by specifying the full paths to the devices:

```bash
# lvrename /dev/myvg/mylv /dev/myvg/mylv1
```

Additional resources

- lvrename(8) man page

# 4.4. REMOVING A DISK FROM A LOGICAL VOLUME

This procedure describes how to remove a disk from an existing logical volume, either to replace the disk
or to use the disk as part of a different volume.

In order to remove a disk, you must first move the extents on the LVM physical volume to a different
disk or set of disks.

Procedure

1. View the used and free space of physical volumes when using the LV:

```bash
# pvs -o+pv_used
PV       VG    Fmt  Attr  PSize   PFree   Used
/dev/vdb1 myvg lvm2  a--  1020.00m  0     1020.00m
/dev/vdb2 myvg lvm2  a--  1020.00m  0     1020.00m
/dev/vdb3 myvg lvm2  a--  1020.00m  1008.00m  12.00m
```

2. Move the data to other physical volume:

   a. If there are enough free extents on the other physical volumes in the existing volume group,
      use the following command to move the data:

      ```bash
      # pvmove /dev/vdb3
      /dev/vdb3: Moved: 2.0%
      ...
      /dev/vdb3: Moved: 79.2%
      ...
      /dev/vdb3: Moved: 100.0%
      ```

   b. If there are no enough free extents on the other physical volumes in the existing volume
group, use the following commands to add a new physical volume, extend the volume group
using the newly created physical volume, and move the data to this physical volume:

      ```bash
      # pvcreate /dev/vdb4
      ```
Physical volume "/dev/vdb4" successfully created

```
# vgextend myvg /dev/vdb4
  Volume group "myvg" successfully extended
```

```
# pvmove /dev/vdb3 /dev/vdb4
/dev/vdb3: Moved: 33.33%
/dev/vdb3: Moved: 100.00%
```

3. Remove the physical volume:

```
# vgreduce myvg /dev/vdb3
Removed "/dev/vdb3" from volume group "myvg"
```

If a logical volume contains a physical volume that fails, you cannot use that logical volume. To remove missing physical volumes from a volume group, you can use the `--removemissing` parameter of the `vgreduce` command, if there are no logical volumes that are allocated on the missing physical volumes:

```
# vgreduce --removemissing myvg
```

Additional resources

- `pvmove(8)`, `vgextend(8)`, `vereduce(8)`, and `pvs(8)` man pages

### 4.5. REMOVING LVM LOGICAL VOLUMES

This procedure describes how to remove an existing logical volume `/dev/myvg/mylv1` from the volume group `myvg`.

#### Procedure

1. If the logical volume is currently mounted, unmount the volume:

   ```
   # umount /mnt
   ```

2. If the logical volume exists in a clustered environment, deactivate the logical volume on all nodes where it is active. Use the following command on each such node:

   ```
   # lvchange --activate n vg-name/lv-name
   ```

3. Remove the logical volume using the `lvremove` utility:

   ```
   # lvremove /dev/myvg/mylv1
   Do you really want to remove active logical volume "mylv1"? [y/n]: y
   Logical volume "mylv1" successfully removed
   ```
**NOTE**

In this case, the logical volume has not been deactivated. If you explicitly deactivated the logical volume before removing it, you would not see the prompt verifying whether you want to remove an active logical volume.

**Additional resources**

- lvremove(8) man page

---

**4.6. REMOVING LVM VOLUME GROUPS**

This procedure describes how to remove an existing volume group.

**Prerequisites**

- The volume group contains no logical volumes. To remove logical volumes from a volume group, see Removing LVM logical volumes.

**Procedure**

1. If the volume group exists in a clustered environment, stop the **lockspace** of the volume group on all other nodes. Use the following command on all nodes except the node where you are performing the removing:

   ```
   # vgchange --lockstop vg-name
   ```

   Wait for the lock to stop.

2. Remove the volume group:

   ```
   # vgremove vg-name
   ```

   Volume group "vg-name" successfully removed

**Additional resources**

- vgremove(8) man page
CHAPTER 5. MODIFYING THE SIZE OF A LOGICAL VOLUME

After you have created a logical volume, you can modify the size of the volume.

5.1. GROWING A LOGICAL VOLUME AND FILE SYSTEM

This procedure describes how to extend the logical volume and grow a file system on the same logical volume.

To increase the size of a logical volume, use the `lvextend` command. When you extend the logical volume, you can indicate how much you want to extend the volume, or how large you want it to be after you extend it.

**Prerequisites**

1. You have an existing logical volume (LV) with a file system on it. Determine the file system type by using the `df -Th` command.
   For more information on creating LV and a file system, see Creating LVM logical volume.

2. You have sufficient space in the volume group to grow your LV and file system. Use the `vgs -o name,vgfree` command to determine the available space.

**Procedure**

1. Optional: If the volume group has insufficient space to grow your LV, then add a new physical volume to the volume group by using the following command:

   ```
   # vgextend myvg /dev/vdb3
   Physical volume "/dev/vdb3" successfully created.
   Volume group "myvg" successfully extended
   
   For more information, see Creating LVM volume group.
   ```

2. Now that the volume group is large enough, execute any one of the following steps as per your requirement:

   a. To extend the LV with the provided size, use the following command:

   ```
   # lvextend -L 3G /dev/myvg/mylv
   Size of logical volume myvg/mylv changed from 2.00 GiB (512 extents) to 3.00 GiB (768 extents).
   Logical volume myvg/mylv successfully resized.
   ```

   **NOTE**

   You can use the `-r` option of the `lvextend` command to extend the logical volume and resize the underlying file system with a single command:

   ```
   # lvextend -r -L 3G /dev/myvg/mylv
   ```
b. To extend the mylv logical volume to fill all of the unallocated space in the myvg volume group, use the following command:

```
# lvextend -l +100%FREE /dev/myvg/mylv
```

Size of logical volume myvg/mylv changed from 10.00 GiB (2560 extents) to 6.35 TiB (1665465 extents).

Logical volume myvg/mylv successfully resized.

As with the `lvcreate` command, you can use the `-l` argument of the `lvextend` command to specify the number of extents by which to increase the size of the logical volume. You can also use this argument to specify a percentage of the volume group, or a percentage of the remaining free space in the volume group.

3. If you are not using the `r` option with the `lvextend` command to extend the LV and resize the file system with a single command, then resize the file system on the logical volume by using the following command:

```
xfs_growfs /mnt/mnt1/
```

```
meta-data=/dev/mapper/myvg-mylv isize=512  agcount=4, agsize=65536 blks
  =  sectsz=512  attr=2, projid32bit=1
  =  crc=1  finobt=1, sparse=1, rmapbt=0
  =  reflink=1
data  =  bsize=4096  blocks=262144, imaxpct=25
  =  sunit=0  swidth=0 blks
naming =version 2  bsize=4096  ascii-ci=0, ftype=1
log  =internal log  bsize=4096  blocks=2560, version=2
  =  sectsz=512  sunit=0 blks, lazy-count=1
realtime =none  extsz=4096 blocks=0, rtextents=0
data blocks changed from 262144 to 524288
```

**NOTE**

Without the `-D` option, `xfs_growfs` grows the file system to the maximum size supported by the underlying device. For more information, see *Increasing the size of an XFS file system*.

For resizing an `ext4` file system, see *Resizing an ext4 file system*.

**Verification**

- Verify if the file system is growing by using the following command:

```
# df -Th
```
<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Type</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>devtmpfs</td>
<td>devtmpfs</td>
<td>1.9G</td>
<td>0</td>
<td>1.9G</td>
<td>0%</td>
<td>/dev</td>
</tr>
<tr>
<td>tmpfs</td>
<td>tmpfs</td>
<td>1.9G</td>
<td>0</td>
<td>1.9G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>tmpfs</td>
<td>tmpfs</td>
<td>1.9G</td>
<td>8.6M</td>
<td>1.9G</td>
<td>1%</td>
<td>/run</td>
</tr>
<tr>
<td>tmpfs</td>
<td>tmpfs</td>
<td>1.9G</td>
<td>0</td>
<td>1.9G</td>
<td>0%</td>
<td>/sys/fs/cgroup</td>
</tr>
<tr>
<td>/dev/mapper/rhel-root</td>
<td>xfs</td>
<td>45G</td>
<td>3.7G</td>
<td>42G</td>
<td>9%</td>
<td>/</td>
</tr>
<tr>
<td>/dev/vda1</td>
<td>xfs</td>
<td>1014M</td>
<td>369M</td>
<td>646M</td>
<td>37%</td>
<td>/boot</td>
</tr>
<tr>
<td>tmpfs</td>
<td>tmpfs</td>
<td>374M</td>
<td>0</td>
<td>374M</td>
<td>0%</td>
<td>/run/user/0</td>
</tr>
<tr>
<td>/dev/mapper/myvg-mylv</td>
<td>xfs</td>
<td>2.0G</td>
<td>47M</td>
<td>2.0G</td>
<td>3%</td>
<td>/mnt/mnt1</td>
</tr>
</tbody>
</table>

Additional resources

- `vgextend(8)`, `lvextend(8)`, and `xfs_growfs(8)` man pages

### 5.2. SHRINKING LOGICAL VOLUMES

You can reduce the size of a logical volume with the `lvreduce` command.

**NOTE**

Shrinking is not supported on a GFS2 or XFS file system, so you cannot reduce the size of a logical volume that contains a GFS2 or XFS file system.

If the logical volume you are reducing contains a file system, to prevent data loss you must ensure that the file system is not using the space in the logical volume that is being reduced. For this reason, it is recommended that you use the `--resizefs` option of the `lvreduce` command when the logical volume contains a file system.

When you use this option, the `lvreduce` command attempts to reduce the file system before shrinking the logical volume. If shrinking the file system fails, as can occur if the file system is full or the file system does not support shrinking, then the `lvreduce` command will fail and not attempt to shrink the logical volume.

**WARNING**

In most cases, the `lvreduce` command warns about possible data loss and asks for a confirmation. However, you should not rely on these confirmation prompts to prevent data loss because in some cases you will not see these prompts, such as when the logical volume is inactive or the `--resizefs` option is not used.

Note that using the `--test` option of the `lvreduce` command does not indicate where the operation is safe, as this option does not check the file system or test the file system resize.

**Procedure**

- To shrink the `mylv` logical volume in `myvg` volume group to 64 megabytes, use the following command:
# lvreduce --resizefs -L 64M myvg/mylv

fsck from util-linux 2.37.2

/dev/mapper/myvg-mylv: clean, 11/25688 files, 4800/102400 blocks

resize2fs 1.46.2 (28-Feb-2021)

Resizing the filesystem on /dev/mapper/myvg-mylv to 65536 (1k) blocks.
The filesystem on /dev/mapper/myvg-mylv is now 65536 (1k) blocks long.

Size of logical volume myvg/mylv changed from 100.00 MiB (25 extents) to 64.00 MiB (16 extents).

Logical volume myvg/mylv successfully resized.

In this example, mylv contains a file system, which this command resizes together with the logical volume.

- Specifying the - sign before the resize value indicates that the value will be subtracted from the logical volume’s actual size. To shrink a logical volume to an absolute size of 64 megabytes, use the following command:

  # lvreduce --resizefs -L -64M myvg/mylv

Additional resources

- lvreduce(8) man page
CHAPTER 6. TROUBLESHOOTING LVM

You can use LVM tools to troubleshoot a variety of issues in LVM volumes and groups.

6.1. GATHERING DIAGNOSTIC DATA ON LVM

If an LVM command is not working as expected, you can gather diagnostics in the following ways.

Procedure

- Use the following methods to gather different kinds of diagnostic data:
  - Add the -v argument to any LVM command to increase the verbosity level of the command output. Verbosity can be further increased by adding additional v’s. A maximum of four such v’s is allowed, for example, -vvvv.
  - In the log section of the /etc/lvm/lvm.conf configuration file, increase the value of the level option. This causes LVM to provide more details in the system log.
  - If the problem is related to the logical volume activation, enable LVM to log messages during the activation:
    i. Set the activation = 1 option in the log section of the /etc/lvm/lvm.conf configuration file.
    ii. Execute the LVM command with the -vvvv option.
    iii. Examine the command output.
    iv. Reset the activation option to 0. If you do not reset the option to 0, the system might become unresponsive during low memory situations.
  - Display an information dump for diagnostic purposes:
    # lvmdump
  - Display additional system information:
    # lvs -v
    # pvs --all
    # dmsetup info --columns
  - Examine the last backup of the LVM metadata in the /etc/lvm/backup/ directory and archived versions in the /etc/lvm/archive/ directory.
  - Check the current configuration information:
    # lvmconfig
  - Check the /run/lvm/hints cache file for a record of which devices have physical volumes on them.
6.2. DISPLAYING INFORMATION ON FAILED LVM DEVICES

You can display information about a failed LVM volume that can help you determine why the volume failed.

**Procedure**

- Display the failed volumes using the `vgs` or `lvs` utility.

**Example 6.1. Failed volume groups**

In this example, one of the devices that made up the volume group `myvg` failed. The volume group is unusable but you can see information about the failed device.

```
# vgs --options +devices
/dev/vdb1: open failed: No such device or address
/dev/vdb1: open failed: No such device or address
  WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
  WARNING: VG myvg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last written to /dev/sdb1).
  WARNING: Couldn't find all devices for LV myvg/mylv while checking used and assumed devices.
```

```
VG   #PV #LV #SN Attr  VSize  VFree  Devices
myvg  2   2   0 wz-pn- <3.64t <3.60t [unknown](0)
myvg  2   2   0 wz-pn- <3.64t <3.60t [unknown](5120),/dev/vdb1(0)
```

**Example 6.2. Failed logical volume**

In this example, one of the devices failed due to which the logical volume in the volume group failed. The command output shows the failed logical volumes.

```
# lvs --all --options +devices
/dev/vdb1: open failed: No such device or address
/dev/vdb1: open failed: No such device or address
  WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
  WARNING: VG myvg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last written to /dev/sdb1).
  WARNING: Couldn't find all devices for LV myvg/mylv while checking used and assumed devices.
```

```
LV  VG  Attr   LSize  Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert Devices
  mylv myvg -wi-a---p- 20.00g                                                     [unknown](0)
  [unknown](5120),/dev/sdc1(0)
```
6.3. REMOVING LOST LVM PHYSICAL VOLUMES FROM A VOLUME GROUP

If a physical volume fails, you can activate the remaining physical volumes in the volume group and remove all the logical volumes that used that physical volume from the volume group.

Procedure

1. Activate the remaining physical volumes in the volume group:
   ```
   # vgchange --activate y --partial myvg
   ```

2. Check which logical volumes will be removed:
   ```
   # vgreduce --removemissing --test myvg
   ```

3. Remove all the logical volumes that used the lost physical volume from the volume group:
   ```
   # vgreduce --removemissing --force myvg
   ```

4. Optional: If you accidentally removed logical volumes that you wanted to keep, you can reverse the `vgreduce` operation:
   ```
   # vgcfgrestore myvg
   ```

**WARNING**
If you remove a thin pool, LVM cannot reverse the operation.

6.4. FINDING THE METADATA OF A MISSING LVM PHYSICAL VOLUME

If the volume group’s metadata area of a physical volume is accidentally overwritten or otherwise destroyed, you get an error message indicating that the metadata area is incorrect, or that the system was unable to find a physical volume with a particular UUID.

This procedure finds the latest archived metadata of a physical volume that is missing or corrupted.

Procedure

1. Find the archived metadata file of the volume group that contains the physical volume. The archived metadata files are located at the `/etc/lvm/archive/volume-group-name_backup-number.vg` path:
   ```
   # cat /etc/lvm/archive/myvg_00000-1248998876.vg
   ```

   Replace `00000-1248998876` with the backup-number. Select the last known valid metadata file, which has the highest number for the volume group.

29
2. Find the UUID of the physical volume. Use one of the following methods.
   - List the logical volumes:
     ```bash
     # lvs --all --options +devices
     Couldn't find device with uuid 'FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk'.
     ```
   - Examine the archived metadata file. Find the UUID as the value labeled `id =` in the `physical_volumes` section of the volume group configuration.
   - Deactivate the volume group using the `--partial` option:
     ```bash
     # vgchange --activate n --partial myvg
     PARTIAL MODE. Incomplete logical volumes will be processed.
     WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
     WARNING: VG myvg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last written to /dev/vdb1).
     0 logical volume(s) in volume group "myvg" now active
     ```

6.5. RESTORING METADATA ON AN LVM PHYSICAL VOLUME

This procedure restores metadata on a physical volume that is either corrupted or replaced with a new device. You might be able to recover the data from the physical volume by rewriting the metadata area on the physical volume.

**WARNING**

Do not attempt this procedure on a working LVM logical volume. You will lose your data if you specify the incorrect UUID.

**Prerequisites**

- You have identified the metadata of the missing physical volume. For details, see Finding the metadata of a missing LVM physical volume.

**Procedure**

1. Restore the metadata on the physical volume:
   ```bash
   # pvcreate --uuid physical-volume-uuid \
     --restorefile /etc/lvm/archive/volume-group-name_backup-number.vg \
     block-device
   ```
The command overwrites only the LVM metadata areas and does not affect the existing data areas.

Example 6.3. Restoring a physical volume on /dev/vdb1

The following example labels the /dev/vdb1 device as a physical volume with the following properties:

- The UUID of FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk
- The metadata information contained in VG_00050.vg, which is the most recent good archived metadata for the volume group

```bash
# pvcreate --uuid "FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk" \
--restorefile /etc/lvm/archive/VG_00050.vg \
/dev/vdb1
```

Physical volume "/dev/vdb1" successfully created

2. Restore the metadata of the volume group:

```bash
# vgcfgrestore myvg
```

Restored volume group myvg

3. Display the logical volumes on the volume group:

```bash
# lvs --all --options +devices myvg
```

The logical volumes are currently inactive. For example:

```
LV     VG   Attr   LSize   Origin Snap%  Move Log Copy%  Devices
mylv myvg -wi--- 300.00G                               /dev/vdb1 (0),/dev/vdb1(0)
mylv myvg -wi--- 300.00G                               /dev/vdb1 (34728),/dev/vdb1(0)
```

4. If the segment type of the logical volumes is RAID, resynchronize the logical volumes:

```bash
# lvchange --resync myvg/mylv
```

5. Activate the logical volumes:

```bash
# lvchange --activate y myvg/mylv
```

6. If the on-disk LVM metadata takes at least as much space as what overrode it, this procedure can recover the physical volume. If what overrode the metadata went past the metadata area, the data on the volume may have been affected. You might be able to use the fsck command to recover that data.

Verification steps
6.6. Rounding Errors in LVM Output

LVM commands that report the space usage in volume groups round the reported number to 2 decimal places to provide human-readable output. This includes the `vgdisplay` and `vgs` utilities.

As a result of the rounding, the reported value of free space might be larger than what the physical extents on the volume group provide. If you attempt to create a logical volume the size of the reported free space, you might get the following error:

```
Insufficient free extents
```

To work around the error, you must examine the number of free physical extents on the volume group, which is the accurate value of free space. You can then use the number of extents to create the logical volume successfully.

6.7. Preventing the Rounding Error When Creating an LVM Volume

When creating an LVM logical volume, you can specify the number of logical extents of the logical volume to avoid rounding error.

Procedure

1. Find the number of free physical extents in the volume group:

```
# vgdisplay myvg
```

**Example 6.4. Free extents in a volume group**

For example, the following volume group has 8780 free physical extents:

```
--- Volume group ---
VG Name     myvg
System ID
Format      lvm2
Metadata Areas  4
Metadata Sequence No  6
VG Access     read/write
[...]        
Free PE / Size   8780 / 34.30 GB
```

2. Create the logical volume. Enter the volume size in extents rather than bytes.
Example 6.5. Creating a logical volume by specifying the number of extents

```
# lvcreate --extents 8780 --name mylv myvg
```

Example 6.6. Creating a logical volume to occupy all the remaining space

Alternatively, you can extend the logical volume to use a percentage of the remaining free space in the volume group. For example:

```
# lvcreate --extents 100%FREE --name mylv myvg
```

Verification steps

- Check the number of extents that the volume group now uses:

```
# vgs --options +vg_free_count,vg_extent_count
```

```
<table>
<thead>
<tr>
<th>VG</th>
<th>#PV</th>
<th>#LV</th>
<th>#SN</th>
<th>Attr</th>
<th>VSize</th>
<th>VFree</th>
<th>Free</th>
<th>#Ext</th>
</tr>
</thead>
<tbody>
<tr>
<td>myvg</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>wz--n-</td>
<td>34.30G</td>
<td>0</td>
<td>0</td>
<td>8780</td>
</tr>
</tbody>
</table>
```