LVM RAID: architecture and operations

Redundancy, layout, monitoring, rebuilds and failure handling

What LVM RAID provides

LVM RAID uses the device-mapper RAID target to build redundant or striped logical volumes from underlying PVs. RAID1 mirrors data; RAID5 and RAID6 use parity; RAID10 combines striping and mirroring. RAID0 is striping without redundancy. A RAID LV is still a block device that needs a filesystem or another consumer.

Redundancy helps survive some device failures, but it does not protect against accidental deletion, corruption replicated to every member, theft or disaster. Backups remain necessary. RAID layouts also differ in write performance, usable capacity and recovery behaviour; select them based on workload and failure tolerance rather than capacity alone.

```
lvs -a -o lv_name,segtype,lv_size,devices,lv_health_status
```

Planning placement and failure domains

For a two-way RAID1 mirror, provide at least two suitable physical devices. For parity RAID, provide the required number of data and parity legs. Multiple partitions on one disk are not independent failure domains; losing that disk can lose multiple RAID legs. LVM may allocate images across PVs according to available space and allocation policy, so verify the actual device placement after creation.

Keep device size, sector size and expected rebuild duration in mind. A RAID6 configuration tolerates two member failures, but that statement assumes independent failures and an otherwise healthy array. Rebuild stress and latent read errors can expose further problems.

```
pvs -o pv_name,pv_size,pv_free,vg_name
lvs -a -o lv_name,segtype,devices
```

Creating and examining a mirrored LV

This example assumes a VG already has space on two independent disks. The -m1 option requests one additional copy, giving two data copies for RAID1. The exact allocation can be influenced by LVM configuration and existing VG layout. Inspect the internal rimage and rmeta components rather than assuming the placement is correct.

Format and mount the resulting LV as you would any other LV. Device-mapper handles the mirrored I/O beneath the filesystem.

```
sudo lvcreate --type raid1 -m1 -L 10G -n critical vgdata
lvs -a -o lv_name,segtype,devices,lv_health_status vgdata
sudo mkfs.ext4 /dev/vgdata/critical
```

Health, synchronisation and repair

Use lvs fields such as raid_sync_action, raid_mismatch_count and lv_health_status where supported by your LVM version. During initial synchronisation or recovery, an LV may be usable while not yet fully redundant. Review kernel messages for I/O errors and failed members. Avoid running repair commands blindly; first identify the failed PV and ensure the replacement device is healthy.

LVM RAID can use lvconvert --repair to replace failed RAID components when sufficient spare space exists, but exact options and automatic allocation behaviour depend on version and configuration. Confirm the documented recovery procedure for your distribution and retain console access during repair.

```
lvs -a -o lv_name,segtype,raid_sync_action,raid_mismatch_count,lv_health_status
journalctl -k --since today | grep -Ei "raid|device-mapper|I/O error"
man lvconvert
```

Operational trade-offs

A mirror consumes roughly twice the physical capacity of its logical data. Parity RAID can improve capacity efficiency but introduces parity write overhead and more complex degraded-mode behaviour. Thin provisioning over RAID is possible, but it adds another failure and monitoring layer: a healthy RAID array does not prevent a thin pool from filling.

Record which physical devices hold each leg, set up alerts, and test a recovery procedure before an actual failure. RAID state, free extents, filesystem integrity and off-system backup success are distinct health indicators.

```
lvs -a -o +devices,lv_health_status
vgs -o vg_name,vg_free
```

Reference scope: commands and defaults vary by distribution and software version. Consult the locally installed manual pages for exact syntax and behaviour.