

# RAID

Redundant array of inexpensive disk  
Redundant array of independent disk

Marzieh Babeianjelodar



# What is RAID?

Redundant Array of Independent Disks

Raid allows you to take a collection of disks & configure them in such way that your computer recognizes them as one logical disk unit.

Another way to look at it:

A data storage virtualization technology that combines multiple physical disk drive components into one or more logical units for the purposes of data redundancy, performance improvement, or both.



# RAID 0

I want speed and efficient use of drive space!

RAID 0 is the answer.

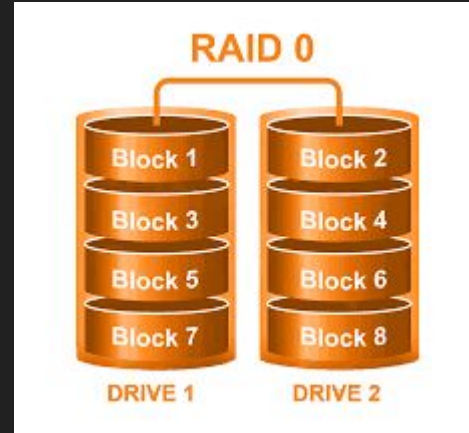
provides no redundancy.

stripes data across multiple drives.

You need at least two storage devices.

Downside: If you lose one drive you will no longer have the full dataset.

Not ideal for a mission critical system.



# RAID 1

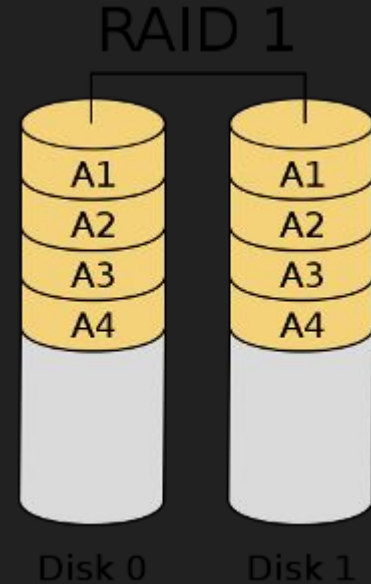
Why? Redundancy!

With RAID 1 you mirror your drives.

Any drive that you save on a drive, you save on another.

If you end up having a drive failure, you have a spare.

Downsides? We are using 2 TB of storage for only 1 usable TB of storage.



# RAID 10

RAID 10, is known as RAID 1+0.

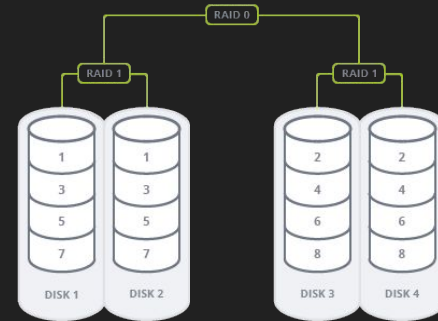
Combines RAID 0 and RAID 1.

Combines disk mirroring & disk striping to protect data.

It requires a minimum of four disks.

Stripes data across mirrored pairs.

RAID 10 provides redundancy and performance.



# Creating the VM

Open Virtual Box (VB) -> Add an existing image to VB

Click on New -> Name -> otest (any name), Type -> Linux, Version -> Ubuntu 64

Ram: 4096, Hard disk: Use an existing virtual hard disk file (RAIDlab.vdi) -> create

Right click -> settings -> storage -> Controller -> SATA -> right click -> Add hard disk -> choose existing disk -> add the four disks one by one

Note: for creating from scratch: Create a new disk -> next (go with the default settings) -> create

Boot the VM

# Identify the hard drives, adding and formatting

Check the hard drives information:

`lsblk -o NAME,SIZE,FSTYPE,TYPE,MOUNTPOINT`

Partition the hard drives:

`fdisk name-of-drive : fdisk /dev/sdb (b/d/c/e)`

Command (m for help) : m

Add a new partition: n

P (primary) -> 1 (default) ->

enter (first sector) -> enter (last sector) -> w (write)

```
root@rd10:~  
[root@rd10 ~]# fdisk /dev/sdb  
Device contains neither a valid DOS partition table, nor Sun, SGI or OS  
F disklabel  
Building a new DOS disklabel with disk identifier 0xce2ac1fe.  
Changes will remain in memory only, until you decide to write them.  
After that, of course, the previous content won't be recoverable.  
  
Warning: invalid flag 0x0000 of partition table 4 will be corrected by  
w(rite)  
  
WARNING: DOS-compatible mode is deprecated. It's strongly recommended t  
o  
switch off the mode (command 'c') and change display units to  
sectors (command 'u').  
  
Command (m for help): n  
Command action  
e extended  
p primary partition (1-4)  
p  
Partition number (1-4): 1  
First cylinder (1-2349, default 1):  
Using default value 1  
Last cylinder, +cylinders or +size{K,M,G} (1-2349, default 2349):  
Using default value 2349  
  
Command (m for help): t  
Selected partition 1  
Hex code (type L to list codes): fd  
Changed system type of partition 1 to fd (Linux raid autodetect)  
  
Command (m for help): p  
  
Disk /dev/sdb: 19.3 GB, 19327352832 bytes  
255 heads, 63 sectors/track, 2349 cylinders  
Units = cylinders of 16065 * 512 = 8225280 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disk identifier: 0xce2ac1fe  
  
   Device Boot      Start         End      Blocks   Id  System  
/dev/sdb1            1         2349     18868311    fd  Linux raid auto  
detect  
  
Command (m for help): w  
The partition table has been altered!  
  
Calling ioctl() to re-read partition table.  
Syncing disks.  
http://www.tecmint.com
```

# Create the array (RAID 0 or 1, 10)

Pass them in to the mdadm --create command. Stands for: Multiple Disk and Device Administration

Specify the device name you wish to create (/dev/md0), the RAID level, and the number of devices:

```
mdadm --create --verbose /dev/md10 --level=10 --raid-devices=4 /dev/sdb /dev/sdc ...
```

Make sure the RAID was successfully created:

```
cat /proc/mdstat
```

Output: Personalities : [linear] [multipath] [raid0] [raid1] [raid6] [raid5] [raid4] [raid10]

```
md0 : active raid0 sdc[1] sdb[0]
```

```
209584128 blocks super 1.2 512k chunks
```

```
unused devices: <none>
```



# Create & mount the file system

Create a file system on the array:

```
mkfs.ext4 -F /dev/md10
```

Create a mount point to attach the new file system:

```
mkdir -p /mnt/md10
```

You can mount the filesystem by typing:

```
mount /dev/md10 /mnt/md10
```

Check whether the new space is available by typing:

```
df -h -x devtmpfs -x tmpfs
```

```
root@stor:~  
[root@stor ~]#  
[root@stor ~]# mkfs.ext4 /dev/md0  
mke2fs 1.41.12 (17-May-2010)  
Discarding device blocks: done  
Filesystem label=  
OS type: Linux  
Block size=4096 (log=2)  
Fragment size=4096 (log=2)  
Stride=128 blocks, Stripe width=256 blocks  
2359296 inodes, 9434112 blocks  
471705 blocks (5.00%) reserved for the super user  
First data block=0  
Maximum filesystem blocks=4294967296  
288 block groups  
32768 blocks per group, 32768 fragments per group  
8192 inodes per group  
Superblock backups stored on blocks:  
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,  
    4096000, 7962624  
  
Writing inode tables: done  
Creating journal (32768 blocks): done  
Writing superblocks and filesystem accounting information: done  
  
This filesystem will be automatically checked every 27 mounts or  
180 days, whichever comes first.  Use tune2fs -c or -i to override.  
[root@stor ~]#
```

# Save the array layout:

To make sure that the array is reassembled automatically at boot, we will have to adjust the `/etc/mdadm/mdadm.conf` file. You can automatically scan the active array and append the file by typing:

```
mdadm --detail --scan >> /etc/mdadm.conf
```

To check the file content:

```
vim /etc/mdadm.conf
```

Add the new filesystem mount options to the `/etc/fstab` file for automatic mounting at boot:

```
echo '/dev/md0 /mnt/md0 ext4 defaults,nofail,discard 0 0' >> /etc/fstab
```

# Failing & Adding devices to RAID

Check the status of a RAID device:

```
mdadm --detail /dev/md10
```

Mark a device as faulty:

```
mdadm /dev/md10 -f /dev/sdb
```

Remove the device that is Faulty:

```
mdadm /dev/md10 -r /dev/sdb
```

Re-add back the device:

```
mdadm /dev/md10 -a /dev/sdb
```

# Remove an mdadm Raid array

The first thing is to unmount the array:

```
umount /dev/md0 (check with df -h = no longer see /dev/md0)
```

Stop the array using:

```
mdadm --stop /dev/md0
```

Zero the superblock for each partition of the array:

```
sudo mdadm --zero-superblock /dev/sdb  
sudo mdadm --zero-superblock /dev/sdc...
```

# Remove an mdadm Raid array from Config files:

Remove from the mdadm configuration file:

```
Vim /etc/mdadm.conf
```

Remove from the file system table:

```
Vim /etc/fstab
```

# What you have to do:

Create RAID 10 array according to the steps in the slides.

Write your command in a sheet with a print screen of your output in each of the following stages (You can use the resources at the end of these slides):

- 1) Fail a device from the raid array.
- 2) Remove the failed device from the raid array.
- 3) Add back the device.

Write two paragraphs of benefits & shortcomings of using any RAID.

Have a good break!

# Reference:

<https://www.digitalocean.com/community/tutorials/how-to-create-raid-arrays-with-mdadm-on-ubuntu-16-04>

<https://www.digitalocean.com/community/tutorials/how-to-manage-raid-arrays-with-mdadm-on-ubuntu-16-04>

<http://www.ducea.com/2009/03/08/mdadm-cheat-sheet/>

<https://bencane.com/2011/07/06/mdadm-manually-fail-a-drive/>