

## Hands On with RAID, v1.4

### Directions

In this exercise, you will have an opportunity to do some exercises with RAID devices. Do these exercises from the command line. Log in with a secure shell session, so you can copy and paste (don't use the VIC console). You should follow along line by line, at the end you'll email me output from some commands so I can see your progress. If you don't follow along pretty closely, you won't get credit.

### Part I. Adding a disk

Your VM has one disk. We will add 2 more small disks as we have done in previous labs. We'll simulate adding disks to our VM by turning the machine off, adding 2 disks in VMware, and turning the VM back on.

Turn the machine off. Remember, logging in as root directly is a bad habit. Use another account.  
[glporter@magenta ~]\$ sudo /sbin/shutdown -hy now

In VMware, edit your machine's setting and add 2 disks. Choose "Add". Choose "Hard Disk". Choose "Create a new virtual disk" (default). **We are desperately short of disk space. We will intentionally add 2 ridiculously small disk drives.** Choose 100 MB. **Make sure "MB" is selected.** For "Location", choose "Store with virtual machine" (default). On the next screen "Advanced Options" leave everything defaults (make no changes), choose "Next". Choose "Finish". Repeat this set of steps and add another (third) 100 MB hard drive. Choose "OK". You might want to look at your VM's settings again, just to make sure you have 3 drives now.

Turn your VM back on. Note that Linux will detect the new drives during power-up. It's possible with some fancy tricks to get Linux to notice a new drive without rebooting, but rebooting is the easiest way.

Log in. Look at your disks.

```
[glporter@magenta ~]$ sudo /sbin/fdisk -l  
Password:
```

```
Disk /dev/sda: 2684 MB, 2684354560 bytes  
255 heads, 63 sectors/track, 326 cylinders  
Units = cylinders of 16065 * 512 = 8225280 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	13	104391	83	Linux
/dev/sda2		14	326	2514172+	8e	Linux LVM

```
Disk /dev/sdb: 104 MB, 104857600 bytes  
64 heads, 32 sectors/track, 100 cylinders  
Units = cylinders of 2048 * 512 = 1048576 bytes
```

Disk /dev/sdb doesn't contain a valid partition table

Disk /dev/sdc: 104 MB, 104857600 bytes  
64 heads, 32 sectors/track, 100 cylinders  
Units = cylinders of 2048 \* 512 = 1048576 bytes

Disk /dev/sdc doesn't contain a valid partition table

You should have 3 disks. The VM resides on /dev/sda; /dev/sdb and /dev/sdc are the two new ones you just added.

## Part II. Using RAID

We're going to use whole disks for RAID. (Optionally, one could use just a part of the disks for RAID. You would have to hard partition the disks, and then mark the partitions for use with RAID as "Linux raid auto", type fd.)

We have 2 extra blank drives. We are going to make a software RAID device with them. They are identical, the same size, etc., so they are perfect candidates for mirroring. We will make a software RAID device /dev/md0 that's mirrored (RAID-1). The "md" terminology for a device like this means "meta-device", that's something akin to "logical volume. Some of this is inherited from old Solaris terminology where you often dealt with "metas" or "meta-devices". **Make sure you are using the correct disks!**

```
[glporter@magenta ~]$ sudo /sbin/mdadm --create --verbose /dev/md0
--level=mirror --raid-devices=2 /dev/sdb /dev/sdc
Password:
mdadm: size set to 102336K
mdadm: array /dev/md0 started.
```

That's it! No muss, fuss.

Take a look at the new software RAID device.

```
[glporter@magenta ~]$ cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdc[1] sdb[0]
      102336 blocks [2/2] [UU]
```

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

So /dev/md0 is your new "drive". It can be treated like a real physical drive. You could chose to (be lame and) use it as a whole disk with no partition, to partition it and make file system(s) on the partition(s), or to use it for LVM. For purposes of example here, we'll go with LVM.

Just like a physical disk, you have to prep your new "disk" for use with LVM with pvcreate.

```
[glporter@magenta ~]$ sudo /usr/sbin/pvcreate -v /dev/md0
```

```
Set up physical volume for "/dev/md0" with 204672 available sectors
Zeroing start of device /dev/md0
Physical volume "/dev/md0" successfully created
```

Now we'll build a new volume group. A volume group acts like an imaginary disk.

```
[glporter@magenta ~]$ sudo /usr/sbin/vgcreate -v my_volume_group /dev/md0
Password:
Wiping cache of LVM-capable devices
Adding physical volume '/dev/md0' to volume group 'my_volume_group'
Archiving volume group "my_volume_group" metadata (seqno 0).
Creating volume group backup "/etc/lvm/backup/my_volume_group" (seqno
1).
Volume group "my_volume_group" successfully created
```

Take a look at the volume group.

```
[glporter@magenta ~]$ sudo /usr/sbin/vgdisplay my_volume_group
--- Volume group ---
VG Name                my_volume_group
System ID
Format                 lvm2
Metadata Areas        1
Metadata Sequence No  1
VG Access              read/write
VG Status              resizable
MAX LV                 0
Cur LV                0
Open LV                0
Max PV                 0
Cur PV                1
Act PV                1
VG Size                96.00 MB
PE Size                4.00 MB
Total PE              24
Alloc PE / Size       0 / 0
Free PE / Size        24 / 96.00 MB
VG UUID                yZBJQk-81g0-L12I-W16z-3DbI-H7xd-lMsp1E
```

Note that as far as LVM is concerned you are currently using one physical volume (Cur PV 1). You know better, the "PV" named /dev/md0 is actually a software RAID device with 2 real drives in it.

Now add a logical volume. We'll make a 80 MB logical volume.

```
[glporter@magenta ~]$ sudo /usr/sbin/lvcreate -v -L80 -ntest_log_vol
my_volume_group
Setting logging type to disk
Finding volume group "my_volume_group"
```

```
/dev/cdrom: open failed: Read-only file system
Archiving volume group "my_volume_group" metadata (seqno 1).
Creating logical volume test_log_vol
Creating volume group backup "/etc/lvm/backup/my_volume_group"
Found volume group "my_volume_group"
Creating my_volume_group-test_log_vol
Loading my_volume_group-test_log_vol table
Resuming my_volume_group-test_log_vol (253:2)
Clearing start of logical volume "test_log_vol"
Creating volume group backup "/etc/lvm/backup/my_volume_group"
Logical volume "test_log_vol" created
```

(You can ignore the "open failed" error, if you get one.) Take a look at the logical volume you just made.

```
[glporter@magenta ~]$ sudo /usr/sbin/lvdisplay -v
```

```
/dev/my_volume_group/test_log_vol
```

```
Using logical volume(s) on command line
```

```
--- Logical volume ---
```

```
LV Name                /dev/my_volume_group/test_log_vol
VG Name                my_volume_group
LV UUID                pYl2in-I20z-IcXU-C5ZQ-UaLz-yYEM-i3RSSW
LV Write Access        read/write
LV Status              available
# open                 0
LV Size                80.00 MB
Current LE             20
Segments              1
Allocation             inherit
Read ahead sectors    auto
- currently set to    256
Block device           253:2
```

Make a file system on the new logical volume. Make a ext3 journaling file system.

```
[glporter@magenta ~]$ sudo /sbin/mkfs -t ext3 -v
```

```
/dev/my_volume_group/test_log_vol
```

```
mkfs 1.39 (29-May-2006)
```

```
Filesystem label=
```

```
OS type: Linux
```

```
Block size=1024 (log=0)
```

```
Fragment size=1024 (log=0)
```

```
20480 inodes, 81920 blocks
```

```
4096 blocks (5.00%) reserved for the super user
```

```
First data block=1
```

```
Maximum filesystem blocks=67371008
```

```
10 block groups
```

```
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729
```

```
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

This filesystem will be automatically checked every 25 mounts or 180 days, whichever comes first. Use `tune2fs -c` or `-i` to override.

Make a directory for a mount point. Mount the new file system.

```
[glporter@magenta ~]$ sudo mkdir /my_mount_point
[glporter@magenta ~]$ sudo mount /dev/my_volume_group/test_log_vol
/my_mount_point
[glporter@magenta ~]$ mount
/dev/mapper/VolGroup00-LogVol100 on / type ext3 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sda1 on /boot type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
/dev/mapper/my_volume_group-test_log_vol on /my_mount_point type ext3 (rw)
```

Look at the new file system.

```
[glporter@magenta ~]$ df -h /my_mount_point/
Filesystem                Size  Used Avail Use% Mounted on
/dev/mapper/my_volume_group-test_log_vol
    78M  5.6M  68M   8% /my_mount_point
[glporter@magenta ~]$ ls -al /my_mount_point/
total 21
drwxr-xr-x  3 root root  1024 Feb 17 21:35 .
drwxr-xr-x 26 root root  4096 Feb 17 20:26 ..
drwx-----  2 root root 12288 Feb 17 21:35 lost+found
```

Looks pretty much like we've seen before. Put some items in it.

```
[glporter@magenta ~]$ cd /my_mount_point/
[glporter@magenta my_mount_point]$ sudo touch i_love_raid
[glporter@magenta my_mount_point]$ sudo touch raid_is_cool
[glporter@magenta my_mount_point]$ sudo mkdir mirrors_are_hard_to_break
[glporter@magenta my_mount_point]$ ls -al
total 25
```

```

drwxr-xr-x  4 root root  1024 Feb 17 21:41 .
drwxr-xr-x 26 root root  4096 Feb 17 20:26 ..
-rw-r--r--  1 root root     0 Feb 17 21:41 i_love_raid
drwx----- 2 root root 12288 Feb 17 21:35 lost+found
drwxr-xr-x  2 root root  1024 Feb 17 21:41 mirrors_are_hard_to_break
-rw-r--r--  1 root root     0 Feb 17 21:41 raid_is_cool

```

In the previous LVM lab, we saw how LVM allowed us to easily increase the size of logical volumes (and the file systems in them). We also saw that we could easily make a volume group larger as well. That applies here as well. If we wanted, we could make our logical volume larger. If we added more drives (RAID or not), we could extend the volume group.

For this lab, we're going to simulate the failure of one of our physical drives that compose the software RAID device. In a plain LVM volume group, loss of one of the physical drives that comprise the volume group would be fatal. The volume group would not be usable. Linux would not be able to "activate" the volume group at boot. Since Linux wouldn't be able to use the volume group, none of the logical volumes that dwell in that volume group would be available. None of the file systems that "live" in the logical volumes would be present or mountable.

So dead drive equals dead volume group; dead volume group equals no logical volumes; no logical volumes equals no file systems. Bad deal. You'd probably have to buy a new drive, build a replacement volume group, and recover your file systems from tape.

But in this case, the death of a physical drive is not a big deal. It's one of two identical copies, they are mirrored. Not good, but not fatal.

Uh, oh. The cheapo drive we bought at NewEgg died.

```

We're going to simulate the death of the drive like so:
[glporter@magenta my_mount_point]$ sudo /sbin/mdadm --manage --set-
faulty /dev/md0 /dev/sdc
mdadm: set /dev/sdc faulty in /dev/md0

```

```

Let's see what a "failed" drive looks like. Check out /proc/mdstat.
[glporter@magenta my_mount_point]$ sudo cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdc[2](F) sdb[0]
      102336 blocks [2/1] [U_]

```

```

unused devices: <none>

```

The status looks different than it did earlier. It's a bit cryptic, but the "F" indicates a failed drive. Let's try a few more things. Look at the end of the system log.

```

[glporter@magenta my_mount_point]$ sudo tail /var/log/messages
Feb 17 22:20:07 magenta kernel: EXT3-fs: mounted filesystem with ordered

```

data mode.

```
Feb 17 22:22:43 magenta kernel: raid1: Disk failure on sdc, disabling device.
```

```
Feb 17 22:22:43 magenta kernel: Operation continuing on 1 devices
```

```
Feb 17 22:22:43 magenta kernel: RAID1 conf printout:
```

```
Feb 17 22:22:43 magenta kernel: --- wd:1 rd:2
```

```
Feb 17 22:22:43 magenta kernel: disk 0, wo:0, o:1, dev:sdb
```

```
Feb 17 22:22:43 magenta kernel: disk 1, wo:1, o:0, dev:sd
```

```
Feb 17 22:22:43 magenta kernel: RAID1 conf printout:
```

```
Feb 17 22:22:43 magenta kernel: --- wd:1 rd:2
```

```
Feb 17 22:22:43 magenta kernel: disk 0, wo:0, o:1, dev:sdb
```

Try using mdadm --detail.

```
[glporter@magenta my_mount_point]$ sudo /sbin/mdadm --detail /dev/md0 /dev/md0:
```

```
Version : 00.90.03
Creation Time : Tue Feb 17 22:17:52 2009
Raid Level : raid1
Array Size : 102336 (99.95 MiB 104.79 MB)
Used Dev Size : 102336 (99.95 MiB 104.79 MB)
Raid Devices : 2
Total Devices : 2
Preferred Minor : 0
Persistence : Superblock is persistent
```

```
Update Time : Tue Feb 17 22:22:43 2009
State : clean, degraded
Active Devices : 1
Working Devices : 1
Failed Devices : 1
Spare Devices : 0
```

```
UUID : 7f2c5fe8:706bfa63:5b253d73:06e2ef13
Events : 0.4
```

Number	Major	Minor	RaidDevice	State	
0	8	16	0	active sync	/dev/sdb
1	0	0	1	removed	
2	8	32	-	faulty spare	/dev/sdc

So, yeah. The drive is failed. The RAID device is operating in degraded mode. So what does that mean to the end user? Is the file system still usable? Can you put stuff in it?

```
[glporter@magenta my_mount_point]$ sudo touch sdc_is_dead
[glporter@magenta my_mount_point]$ ls -al
```

```
total 26
drwxr-xr-x  4 root root  1024 Feb 17 22:33 .
drwxr-xr-x 26 root root  4096 Feb 17 22:16 ..
-rw-r--r--  1 root root     0 Feb 17 22:21 i_love_raid
drwx-----  2 root root 12288 Feb 17 22:19 lost+found
drwxr-xr-x  2 root root  1024 Feb 17 22:21 mirrors_are_hard_to_break
-rw-r--r--  1 root root     0 Feb 17 22:21 raid_is_cool
-rw-r--r--  1 root root     0 Feb 17 22:33 sdc_is_dead
```

Hey! Cool! Even though we lost a drive, the mirror did its job. The device stayed up. The file system stayed mounted. (You were standing in it the whole time, and didn't notice anything).

We just saw that the RAID device was operating in “degraded” mode. What does that mean? It means that the device cannot survive another failure. It's perfectly possible to make a device with multiple mirrors, or that has a “hot spare” standing by to be used in the event of failure.

One of the biggest “problems” with a failure with a RAID device is that the RAID device handles it so well. The failure occurs “silently”. If you are not vigilant, you would not realize a failure had occurred. You'd find out when the second failure occurred, the RAID device finally died, and you had big problems on your hands.

The admin tool for RAID, mdadm, can be configured to monitor your devices. Check out the '--monitor' option to mdadm (read the man page, or go online). You can get it to send alerts to an email address.

“Fixing” the failed drive is easy. Tell mdadm to remove the failed drive from the device.  
 [glporter@magenta my\_mount\_point]\$ sudo /sbin/mdadm /dev/md0 -r /dev/sdc  
 Password:  
 mdadm: hot removed /dev/sdc

(If this were a real failure of a real drive, you'd turn the server off at this point, replace the drive with a good one, and turn it back on.)

Now (in theory anyway) you'd have a new fresh replacement drive. Tell mdadm to use it.  
 [glporter@magenta my\_mount\_point]\$ sudo /sbin/mdadm /dev/md0 -a /dev/sdc  
 mdadm: re-added /dev/sdc

Look at the status of the device.  
 [glporter@magenta my\_mount\_point]\$ cat /proc/mdstat  
 Personalities : [raid1]  
 md0 : active raid1 sdc[1] sdb[0]  
 102336 blocks [2/2] [UU]

unused devices: <none>

It really does work. You replaced a failed drive **WHILE** you were standing in the directory of a file system that lived on it.

#### Part IV. Get some points

Run the following commands as below. Copy the output into the body of an email and email it to me with the subject "Hands On with RAID". **This is due by next class period.**

If you are in a Cal Poly class, email it to me at [glporter@csc.calpoly.edu](mailto:glporter@csc.calpoly.edu).

If you are in a Cuesta class, email it to me at [gregory\\_porter3@cuesta.edu](mailto:gregory_porter3@cuesta.edu).

I need to know who you are to give you credit, so if you email it from a non-campus account, make sure you mention your full name.

It's always possible that emails get lost, intercepted, misdirected, etc., so email it from a place that keeps track of the emails you've sent (say in a "Sent Items" folder). If I don't get it, you can send it again, or print a copy and hand it to me.

```
[glporter@magenta ~]$ hostname;sudo /sbin/fdisk -l;sudo /sbin/mdadm
--detail /dev/md0;cat /proc/mdstat;sudo /usr/sbin/pvdisplay;sudo
/usr/sbin/vgdisplay;sudo /usr/sbin/lvdisplay;df -h;mount; ls -al
/my_mount_point
```

This has embedded sudo commands that may prompt for a password. If so, run it again. Send me the output from the second run with no password prompt.

#### Part V. Get some more points (ideas for extra credit)

If you try any of these, email me and tell me about it.

If you're curious, you could add multiple drives and make a RAID-5 device, or make a device with multiple mirrors, or try making a device that has a stand-by "hot spare". It's always fun to then "do terrible things to it with a fork", that is, induce failures and see what happens. Try `mdadm -- monitor`.

Supposedly hardware RAID makes things faster. Supposedly. It's probably a good bet that doing stuff with software RAID slows things down. You can test this with a benchmarking tool. A well regarded tool that's easy to get and use is `bonnie++`. Packages are widely available, I searched [rpm.pbone.net](http://rpm.pbone.net) and found [http://rpm.pbone.net/index.php3/stat/4/idpl/10503959/com/bonnie++-1.03c-1.fc10.x86\\_64.rpm.html](http://rpm.pbone.net/index.php3/stat/4/idpl/10503959/com/bonnie++-1.03c-1.fc10.x86_64.rpm.html).

Install `bonnie++`. Google and find out a little on how to use it. The default set of disk tests it'll try to do take a while. You could specify fewer shorter tests, so you don't die of boredom. One thing that isn't obvious is that small disk I/O operations don't actually touch the disk (!?!), they function out of cache in RAM. You have to make the file size that is read or written bigger than RAM to force the actual direct use of disk. Another (stupider, but valid) way of benchmarking a raw disk write is by using the device `/dev/zero`. `/dev/zero` spits out zeros as fast as the CPU can manage. If you do something like `'dd if=/dev/zero of=bigfile bs=8192 count=1000'`, you'll write a big file named 'bigfile'. The data dump (`dd`) command will tell you how long was elapsed. Get the man page on `dd` to see what the options to `dd` are.

Try some benchmarks with bonnie++ or dd like:

Make a plain file system on a plain vm disk of a particular size, benchmark it.

Make a RAID 1 disk set with 2 of the same size disks, put the same file system on it, benchmark it.

Add a couple more of the same size disks (4? 5?), make a RAID 5 set, put the same file system on it, benchmark it.

With a RAID 5 set, "kill" one of the disks. Force the set to use parity. (You could turn the machine off and actually remove one of the disks. Alternately, you could mark it "dead" like we did above).

Benchmark the degraded set. (While you have a degraded set, fiddle with it. Note what errors you see. Add a disk back and "fix" it).

I'd expect:

The plain disk with plain file system might be fastest overall. (Software RAID has overhead).

RAID-1 should read faster than it writes.

RAID-5 is probably a little slower than RAID-1, particularly for reads.

A degraded RAID-5 set is a LOT slower.

## Part VI. Clean up

If you are sure that you emailed me, and you have a copy of the email in a safe place, then put your vm back the way it was. Umount the file systems. Remove your mount point directories. To get rid of all the LVM stuff you'll have to kill the lvols, and then the volume group like so:

```
[glporter@magenta ~]$ sudo /usr/sbin/lvremove /dev/my_volume_group/test_log_vol
```

```
Do you really want to remove active logical volume "test_log_vol"? [y/n]: y
```

```
Logical volume "test_log_vol" successfully removed
```

```
[glporter@magenta ~]$ sudo /usr/sbin/vgchange -a n my_volume_group
```

```
0 logical volume(s) in volume group "my_volume_group" now active
```

```
[glporter@magenta ~]$ sudo /usr/sbin/vgremove my_volume_group
```

```
Volume group "my_volume_group" successfully removed
```

Then turn off all the RAID stuff.

```
[glporter@magenta /]$ sudo /sbin/mdadm --stop /dev/md0
```

```
mdadm: stopped /dev/md0
```

Turn your machine off. Edit the settings of your VM in VIC and remove the second and third drives we used. Turn your machine back on.