

Setting up bootable RAID 1 in a safe way

By Harakiri - July 06 2005 - PDF version

I 've set up this document because there is no recent documentation on how to set up a bootable RAID 1 system (mirror). Or at least it 's hard to find. Using bits and pieces from other documents and a preinstalled Mandrake RAID 1 system on a VMWare, I 've setup this Mini-Howto, which is nothing more than an exemplary procedure for myself, which I 'm glad to share with the rest of the world.

Unfortunately, RAID is not supported yet in the current versions of TRK (stable currently 1.1), but will be in future versions and maybe I 'll throw in some scripts that will automate this procedure of having your disk mirrored.

The systems I used were a Mandrake 10.0 with a custom built 2.6.6 kernel and a Mandriva 10.2 with a recompiled 2.6.11mdk kernel. This setup should work for any recent Linux distro.

1. Before you begin

Compile a kernel with all your necessary device drivers compiled in so you don 't need an initrd. Initrds only complicate life

So include at least in the kernel:

- * raid support + at least support for raid1 (just select all, you 're not running on a 386 with 4 megs of ram right?)
- * your disk controller(s)
- * ext3 filesystem support (default in normally)
- * if using lilo, use a version equal or higher than 22.5.9. If not, you might encounter the error "Lilo - Timestamp Mismatch" at boot time. This is a bug in Lilo and happens because this procedure mirrors everything from your disks, including partition table and volume ID. Apparently duplicate volume IDs causes lilo to go in error and prevent you from booting. At that time, you need to boot with a disconnected disk, break your mirror if already made, upgrade lilo, re-add and rebuild the removed disk. Better to check beforehand with "lilo -V"

2. Layout of our disk to mirror

```
Disk /dev/sda: 203.9 GB, 203928109056 bytes
255 heads, 63 sectors/track, 24792 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

```
Device Boot Start End Blocks Id System
/dev/sda1 1 62 497983+ 83 Linux
/dev/sda2 63 24792 198643725 5 Extended
/dev/sda5 63 187 1004031 82 Linux swap
/dev/sda6 188 1172 7911981 83 Linux
/dev/sda7 1173 1671 4008186 83 Linux
```

```
/dev/sda8 1672 24700 184980411 83 Linux
```

This is our /etc/fstab

```
/dev/sda6 / ext2 defaults 1 1
/dev/sda1 /boot ext2 defaults 1 2
none /dev/pts devpts mode=0620 0 0
/dev/sda8 /home ext2 defaults 1 2
/dev/hdb /mnt/cdrom auto umask=0,user,codepage=850,ioccharset=iso8859-1,noauto,ro
,exec 0 0
none /mnt/floppy supermount dev=/dev/fd0,fs=ext2:vfat,--,umask=0,ioccharset=utf8,
sync 0 0
none /proc proc defaults 0 0
/dev/sda7 /var ext2 defaults 1 2
/dev/sda5 swap swap defaults 0 0
```

-We 're going to mirror /, /boot and /var.

/home will be backed up with a cronjob in a tar.gz file every day to its counterpart partition on the other disk. Since we have a big disk, this will allow us to have at least 2 different backups as long as the disks aren 't half filled. This contains only user data, no system files are on this disk.

The safest way to mirror is to setup a broken mirror and first copy the data to it, boot from it and add the original disk to the mirror, overwriting its data with its own copy. Still with me? Ok, practically this means copying /, /boot and /var to the other disk, sdb, mounted somewhere as a half mirror. Do this in runlevel 1 so no data changes meanwhile. Next boot from it, sfdisk /dev/sdb to /dev/sda and add the raid partitions from sda to the mirrors. This will copy back the data from sdb paritions to the sda partitions. More on that later on.

3. Get on with mdadm

-Zero out any superblock still existing from previous mirror attempts

```
mdadm --zero-superblock /dev/sdb1 mdadm --zero-superblock /dev/sdb6 mdadm --zero-
superblock /dev/sdb7
```

-Copy partition information from sda to sdb using sfdisk

```
sfdisk -d /dev/sda > sda.out
```

Dump layout of sda to a file edit file with vi and replace sda entries by sdb

```
sfdisk /dev/sdb < sda.out
```

-Change partition types of partitions to mirror with fdisk. **This is very important**, otherwise Linux won 't boot from it!

```
fdisk /dev/sdb
```

"t" for toggle partition type

change raidable partitions (check your fstab) to "fd" = Linux Raid Autodetect

"w" to write and quit fdisk

-Set up a broken mirror on /dev/sdb partitions

```
mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb1 missing
mdadm --create /dev/md1 --level=1 --raid-devices=2 /dev/sdb6 missing
mdadm --create /dev/md2 --level=1 --raid-devices=2 /dev/sdb7 missing
```

This will start the raid devices. Dump this information to mdadm.conf for later use

```
mdadm -Esb /dev/sdb1 >> /etc/mdadm.conf
mdadm -Esb /dev/sdb6 >> /etc/mdadm.conf
mdadm -Esb /dev/sdb7 >> /etc/mdadm.conf
```

I like editing this file manually afterwards.

=> remove the lines where it says "devices="... and add a line above ARRAY that says

```
"DEVICE /dev/sda1 /dev/sdb1 /dev/sda6 /dev/sdb6 /dev/sda7 /dev/sdb7"
```

This will make your kernel complain about missing devices when you restart from your broken mirror, but this way this file is already what it should be

-Next, create filesystems on it. I 'm gonna take the opportunity to make them ext3 instead of the original ext2

```
mkfs.ext3 /dev/md0
mkfs.ext3 /dev/md1
mkfs.ext3 /dev/md2
```

4. Offline operations

-Better than switching to runlevel 1, boot from a bootable Linux CD that supports raid and your harddisk controller.

I used "System Rescue CD-Rom"

Next versions of TRK will contain the necessary tools too.

Why? Because filesystems like /dev will not be mounted over your existing /dev, which contains the static /dev special file entries in case udev doesn 't work.

Make dirs to mount your stuff on

```
mkdir /sda1
mkdir /sda6
mkdir /sda7
mkdir /md0
mkdir /md1
mkdir /md2
```

-Mount /dev/sda partitions in read-only and copy mdadm.conf from /sda6/etc/mdadm.conf to your current /etc from your bootcd

```
mount -o ro /dev/sda1 /sda1
mount -o ro /dev/sda6 /sda6
```

```
mount -o ro /dev/sda1 /sda7
cp -f /sda6/etc/mdadm.conf /etc
```

-Start your RAID devices

```
mdadm -As
```

-Mount your RAID devices

```
mount /dev/md0 /md0
mount /dev/md0 /md1
mount /dev/md0 /md2
```

-Copy all your data

```
cp -a /sda1/* /md0
cp -a /sda6/* /md1
cp -a /sda7/* /md2
```

You can check the progress of this copy in another console with "df"

-Once the data is copied, we 're gonna modify the fstab, next lilo

The fstab on /md1/etc/ should become something like this:

```
/dev/md1 / ext3 defaults 1 1
/dev/md0 /boot ext3 defaults 1 2
none /dev/pts devpts mode=0620 0 0
/dev/sda8 /home ext2 defaults 1 2
/dev/hdb /mnt/cdrom auto umask=0,user,codepage=850,iocharset=iso8859-1,noauto,ro
,exec 0 0
none /mnt/floppy supermount dev=/dev/fd0,fs=ext2:vfat,--,umask=0,iocharset=utf8,
sync 0 0
none /proc proc defaults 0 0
/dev/md2 /var ext3 defaults 1 2
/dev/sda5 swap swap defaults 0 0
/dev/sdb5 swap swap defaults 0 0
```

5. Lilo

The easiest way for lilo not to complain is to chroot to the mounted raid root, but first submount its /boot partition to it

```
umount /md0
mount /dev/md0 /md1/boot
```

-System Rescue CD specific: copy zsh to your soon to be chrooted /bin

```
cp /bin/zsh /md1/bin
```

-Chroot, run bash and edit lilo.conf

```
chroot /md1
bash
```

vi /etc/lilo.conf

-Here 's what the original lilo.conf looks like:

```
# File generated by DrakX/drakboot
# WARNING: do not forget to run lilo after modifying this file

default="linux"
boot=/dev/sda
map=/boot/map
keytable=/boot/be2-latin1.klt
prompt
nowarn
timeout=100
message=/boot/message
menu-scheme=wb:bw:wb:bw
image=/boot/vmlinuz
label="linux"
root=/dev/sda1
initrd=/boot/initrd.img
append="resume=/dev/sda5 splash=silent"
vga=788
read-only
image=/boot/vmlinuz
label="linux-nonfb"
root=/dev/sda1
initrd=/boot/initrd.img
append="resume=/dev/sda5"
read-only
image=/boot/vmlinuz
label="failsafe"
root=/dev/sda1
initrd=/boot/initrd.img
append="failsafe resume=/dev/sda5 devfs=nomount"
read-only
```

-Here 's the new one:

```
default="linux-md"
boot=/dev/md0
raid-extra-boot=mbr
map=/boot/map
keytable=/boot/be2-latin1.klt
prompt
nowarn
timeout=100
message=/boot/message
menu-scheme=wb:bw:wb:bw

image=/boot/vmlinuz-2.6.11custom
label="linux-md"
root=/dev/md1
vga=1
read-only

image=/boot/vmlinuz
label="linux-nonfb"
root=/dev/sda1
initrd=/boot/initrd.img
append="resume=/dev/sda5"
read-only
```

```
image=/boot/vmlinuz
label="failsafe"
root=/dev/sda1
initrd=/boot/initrd.img
append="failsafe resume=/dev/sda5 devfs=nomount"
read-only
```

My newly compiled kernel is called /boot/vmlinuz-2.6.11custom, my boot device is /dev/md0.

See to it that the underlying partitions have their "active" bit set (in fdisk, "a", sdb1) The line "raid-extra-boot=mbr" will write the bootsector to the currently active raid disks, which will be /dev/sdb.

Now run lilo

If your system is unable to boot from a secondary drive from BIOS, copy lilo.conf to lilo.conf.md, remodify lilo.conf, comment out the "raid-extra-boot=mbr" and change boot back to "boot=/dev/sda"

Run lilo again.

This will write the same boot information to the bootsector of /dev/sda.

You will only need this once, just to be able to start from the "linux-md"

I left an old entry in case the new kernel won 't boot from the mirror.

I 've also thrown out framebuffer support. It 's a personal preference, I don 't like graphics on a server.

"vga=1" will give me a 80x50 screen (you 'll need to remove the SYSFONT entries from /etc/rc.sysinit).

6. Running from a broken mirror and hotadd devices

-Restart and boot "linux-md"

Your system should be running on a broken raid 1.

See that all runs well and your data is all there, processes are running normally.

Check that the output of "mount" gives you md devices

-Once you 've made sure of that, sfdisk your /dev/sdb information to /dev/sda

```
sfdisk -d /dev/sdb > sdb.out
```

Dump layout of sdb to a file edit file with vi and replace sdb entries by sda

```
sfdisk /dev/sda < sdb.out
```

Check with fdisk that /dev/sda contains exactly the same partitions as /dev/sdb

-Once you 've done all that, you can add your inactive sda partitions to your running raid1 devices, making your system redundant.

```
mdadm /dev/md0 --add /dev/sda1
mdadm /dev/md1 --add /dev/sda6
```

```
mdadm /dev/md2 --add /dev/sda7
```

Your disks should be synching now. Check the progress in /proc/mdstat

-Copy /etc/lilo.conf.md back to /etc/lilo.conf and run lilo again

Once your disks are synchronised, you should be able to boot from any 2 disks in your system.

-You might need to initiate the second swap on /dev/sdb5

```
swapon /dev/sdb5
```

That 's it, you 're running Linux from RAID 1