page 1 (home) page 2 (back)

Use 'cat /etc/fstab' and make sure all your md devices are shown. If they are not, something went wrong. You might consider starting over from the beginning:

```
cat /etc/fstab
```

```
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    defaults        0       0
/dev/md2        /               ext3    defaults,errors=remount-ro 0       1
/dev/md0        /boot           ext3    defaults        0       2
/dev/md1        none            swap    sw              0       0
/dev/hdd        /media/cdrom0   udf,iso9660 user,noauto     0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto  0       0
```

Use 'cat /proc/mdstat' to see what is going on.

```
cat /proc/mdstat
```

Output on my machine:

```
Personalities : [raid0] [raid1] [raid5]
md0 : active raid1 sda1[0] sdb1[1]
      240832 blocks [2/2] [UU]

md1 : active raid1 sda2[0] sdb2[1]
      682688 blocks [2/2] [UU]

md2 : active raid1 sda3[0] sdb3[1]
      16747648 blocks [2/2] [UU]

unused devices: <none>
```

This shows both drives are up ([UU]) for all three md devices. At this point if drive /dev/sdb failed (pie), we would continue to function. We could shut down, replace pie with (a cleaned) spare, and we would still be able to boot up using /dev/sda (apple). Since we have a bootable system, we could get the spare drive back on line with a few well chosen commands. Sounds like everything is OK then, right? Well it is if /dev/sda is never the one that fails. So our immediate challenge is to get a GRUB MBR installed on /dev/sdb, so if /dev/sda fails, we can boot from the secondary drive. Grub sees our secondary drive as (hd1) and we have set the boot flag on the first partition of that drive. Grub starts counting from zero, so it sees that partition as 0. We will install grub on (hd1,0).

We are going to reconfigure grub's menu.lst. Start by making a backup copy:

```
cp /boot/grub/menu.lst /boot/grub/menu.lst-backup
```

Edit menu.lst:

```
vi /boot/grub/menu.lst
```

*We are going to add a new menu item that tells grub to boot from our secondary drive. Make a duplicate of your existing top menu stanza, **place the duplicate above the existing stanza**, and modify it in the same manner I have. This example shows partition 0 is the partition flagged as bootable on my system. You can run something like 'fdisk -l /dev/sda' to determine which partition is bootable on your system but your original stanza will be correct:*

```
title           Debian GNU/Linux, kernel 2.6.8-2-386 (hd1)
root            (hd1,0)
kernel          /vmlinuz-2.6.8-2-386 root=/dev/md2 ro
initrd          /initrd.img-2.6.8-2-386
savedefault
boot

title           Debian GNU/Linux, kernel 2.6.8-2-386
root            (hd0,0)
kernel          /vmlinuz-2.6.8-2-386 root=/dev/md2 ro
```

```
initrd          /initrd.img-2.6.8-2-386
savedefault
boot
```

Save and exit the file. Just a note: because we mount a /boot partition, you will not see the above entries in the form "/boot/initrd.img-2.6.8-2-386". If you do not mount a separate /boot partition, you will see the entries in that form. To install grub on both hard drives, start the grub command shell:

```
grub
```

At the 'grub>' shell prompt create new boot records for both drives:

```
root (hd0,0)
setup (hd0)
root (hd1,0)
setup (hd1)
quit
```

OK. We can test by rebooting. Do this at the console so if should fail you can simply choose another item from the menu.:

```
reboot
```

If all went well, once again we will edit menu.lst:

```
vi /boot/grub/menu.lst
```

We will add a fallback entry that will (hopefully) automatically choose the next item in the menu if the first item fails. So, just below "default 0", add this entry:

```
fallback 1
```

OK, now I'm going to simulate a failed drive. I don't recommend you try this (your system may explode), but at least you can learn from my system. I am carefully going to remove the power cable from the primary drive, apple. Once I do this, it will be "dirty" and should not be used again in this system without first being cleaned. This is what 'cat /proc/mdstat' shows as a result: sda1 and sda2 still show they are up because we have not had any read/write operations on them recently, sda3 shows it has failed.

```
md0 : active raid1 sda1[0] sdb1[1]
      240832 blocks [2/2] [UU]

md1 : active raid1 sda2[0] sdb2[1]
      682688 blocks [2/2] [UU]

md2 : active raid1 sda3[2](F) sdb3[1]
      16747648 blocks [2/1] [_U]
```

If your hardware supports hot swappable drives I think you should mark the remaining two devices faulty, then use mdadm (**m**ulti-**d**isk **adm**inistrator) to remove all three faulty devices from our array before inserting the new drive. You cannot use "mdadm --remove" on drives that are in use, so they need to be set as faulty first. **You do not need to do this** if you are going to *power down* the system and replace the drive with a clean drive. Make doubly sure you are failing the partitions on the drive that has failed!

```
mdadm --set-faulty /dev/md0 /dev/sda1
mdadm --set-faulty /dev/md1 /dev/sda2
mdadm --remove /dev/md0 /dev/sda1
mdadm --remove /dev/md1 /dev/sda2
mdadm --remove /dev/md2 /dev/sda3
```

Shut it down:

```
shutdown -h now
```

To clean a drive, I install it in a computer by itself, boot up using a DBAN disk and change the Method to Quick.

If you would like a tomsrtbt disk, insert a blank floppy, then:

```
cd /usr/local/src
wget http://ftp.sunsite.utk.edu/ftp/pub/mini-linux/tomsrtbt/tomsrtbt-
2.0.103.tar.gz
tar xzvf tomsrtbt-2.0.103.tar.gz
rm tomsrtbt-2.0.103.tar.gz
cd tomsrtbt-2.0.103
./install.s
```

For consistency (to keep my sanity) I always move the good drive to the primary position (if it is not already there) and place the new clean drive in the secondary position. We have shut down, so disconnect the good drive, clean apple, move pie (the good drive) into the primary position, place the cleaned apple in the secondary position and bring the system back up. On my system all I have to do to swap the two SCSI drives is to move the jumper from one drive to the other. OK, my system did boot up.
First we see what's going on (cat /proc/mdstat). As you can see, sdb1, sdb2 and sdb3 are missing:

```
md0 : active raid1 sda1[1]
      240832 blocks [2/1] [_U]

md1 : active raid1 sda2[1]
      682688 blocks [2/1] [_U]

md2 : active raid1 sda3[1]
      16747648 blocks [2/1] [_U]
```

We start by copying the partition structure from /dev/sda to /dev/sdb. We do this for what should now be an obvious reason: the secondary drive is empty, but it needs to have the same structure as the primary drive. If the disk was first cleaned, and is large enough, you should have no errors:

```
sfdisk -d /dev/sda | sfdisk /dev/sdb
```

To further clean the drive we make sure the superblocks are zeroed out on the new drive (be careful you do this to the correct drive). Superblocks are used by RAID for identification purposes. Edit as needed:

```
mdadm --zero-superblock /dev/sdb1
mdadm --zero-superblock /dev/sdb2
mdadm --zero-superblock /dev/sdb3
```

Now we add our three sdb partitions to the corresponding md's.
Understand what you are doing here before you do it, edit as needed:

```
mdadm --add /dev/md0 /dev/sdb1
mdadm --add /dev/md1 /dev/sdb2
mdadm --add /dev/md2 /dev/sdb3
```

They should start to synchronize. To watch them sync:
`watch -n 6 cat /proc/mdstat`

Of course it's [Ctrl]+c to cancel 'watch'. NEVER reboot during synchronization. Once the recovery is complete (and not until then), create a new boot record on our secondary (replacement) drive:
`grub`

From the grub> prompt (edit partition number if needed):

```
root (hd1,0)
setup (hd1)
quit
```

We are working again.

You might want to reboot from the console to make sure you actually boot from the secondary drive.

In A RAID system it is a good idea to avoid kernel version upgrades (security upgrades should be performed of course). If you are currently running Linux kernel 2.6.8 or older, installing a kernel newer than 2.6.8 may replace devfs with udev. If this happens you could be in big trouble. I have not been able to repair a system once it migrates from devfs to udev. If you are already using udev, upgrading the kernel to a newer kernel that also uses udev should not be an issue. Some of the new 2.6 kernels (from 2.6.12) no longer use mkinitrd to create the initrd.img. The set of programs now used to create the ramdisk image (initramfs-tools - run 'man mkinitramfs') for some reason may not create an initrd.img that is able to boot into our md devices. As a result, after an upgrade to one of the newer kernels, your system may not boot to the new kernel. This bug may be resolved at some point in the future (and may be resolved when you read this) but to work around the problem I was able to use mkinitrd to create the ramdisk image. The use of mkinitrd is deprecated when used with kernels 2.6.12 or newer. I don't know why this worked for me. I would not be surprised if it does not work for you. I am using Sarge but as a test I will install a kernel from 'testing' (Etch) to illustrate.

I installed a 'testing' source in /etc/apt/sources.list and ran 'apt-get update'. We need a newer version of initrd-tools that can be used with the newer kernel:

```
apt-get -t testing install initrd-tools
```

Now I determine the correct kernel for my architecture:

```
apt-cache search linux-image
```

I am going to install:

```
apt-get install linux-image-686
```

Running this command installed version 2.6.15-1-686 on my system. At this point the kernel may not boot into our md devices. You can try if you like to see if the bug has been fixed. If not then you will have to boot up using the old kernel. Go ahead and reboot. If it fails to boot using the new kernel then we will again compile a new /boot/initrd.img using mkinitrd. Here I create a backup copy of the current initrd.img, compile a new image to a temp file using the new kernel modules (/lib/modules/2.6.15-1-686/), then copy the temp file on top of the current initrd.img as we did before:

```
cp /boot/initrd.img-2.6.15-1-686 /boot/initrd.img-2.6.15-1-686-backup

mkinitrd -o /boot/initrd.img-2.6.15-1-686-temp /lib/modules/2.6.15-1-686/
cp /boot/initrd.img-2.6.15-1-686-temp /boot/initrd.img-2.6.15-1-686
```

We are now going to reconfigure grub once again.

```
vi /boot/grub/menu.lst
```

Make a duplicate of your existing top menu stanza, place the duplicate above the existing stanza, and modify it in the same manner I have (so we boot off of hd1):

```
title           Debian GNU/Linux, kernel 2.6.15-1-686 (hd1)
root            (hd1,0)
kernel          /vmlinuz-2.6.15-1-686 root=/dev/md6 ro
initrd          /initrd.img-2.6.15-1-686
savedefault
boot

title           Debian GNU/Linux, kernel 2.6.15-1-686 (hd0)
root            (hd0,0)
kernel          /vmlinuz-2.6.15-1-686 root=/dev/md6 ro
initrd          /initrd.img-2.6.15-1-686
savedefault
boot
```

Save and exit the file, then start up the grub command prompt:

```
grub
```

at the grub> prompt enter these commands to reinstall grub on the secondary drive (edit partition number if

needed):

```
root (hd1,0)
setup (hd1)
quit
```

Reboot to make sure it works.

The Debian installer did a lot of work for us when we created our RAID devices. To appreciate the difficulty of building a RAID array on a system that is currently running and is not configured as one please read http://www200.pair.com/mecham/raid/raid1-degraded.html. The introduction substantially duplicates this document but the remainder explains important additional aspects. Start reading at the paragraph just above the illustration.

References (alphabetical order). Not all of these are good, but all were interesting to me in one way or another. Trust me, there are a lot more documents similar to these out there:

http://alioth.debian.org/project/showfiles.php?group_id=30283&release_id=288
http://deb.riseup.net/storage/software-raid/
http://forums.whirlpool.net.au/forum-replies-archive.cfm/471585.html
http://nepotismia.com/debian/raidinstall/
http://nst.sourceforge.net/nst/docs/user/ch14.html
http://piirakka.com/misc_help/Linux/raid_starts_degraded.txt
http://thegoldenear.org/toolbox/unices/server-setup-debian.html
http://togami.com/~warren/guides/remoteraidcrazies/
http://www.debian-administration.org/articles/238
http://www.debian-administration.org/users/philcore/weblog/4
http://www.doorbot.com/guides/linux/x86/grubraid/
http://www.epimetrics.com/topics/one-page?page_id=421&topic=Bit-head%20Stuff&page_topic_id=120
http://www.james.rcpt.to/programs/debian/raid1/
http://www.linuxjournal.com/article/5898
http://www.linuxsa.org.au/mailing-list/2003-07/1270.html
http://www.linux-sxs.org/storage/raid_setup.html
http://www.parisc-linux.org/faq/raidboot-howto.html
http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html
http://trinityhome.org/misc/bootable-raid1.html
http://www.vermittlungsprovision.net/367.html
http://xtronics.com/reference/SATA-RAID-debian-for-2.6.html

page 1 (home) page 2 (back)

Gary V
mr88talent at yahoo dot com
Last edited 30 APR 2006