Security Viewpoints

Security, operating systems and the IT industry

Search: Go

Home
About us
Security papers
Contact us
Statement of influence

Recent Entries

- Cheap supercomputing from your graphics card
- CIS releases Vmware ESX security guide
- Multiple critical vulnerabilities in all VMware products
- Dear developers: sign your code!
- CIS releases virtual machine security guide

Article topics

- Best practices
- Blind spots
- Computer industry
- Education
- Email security
- Infrastructure
- Law & enforcement
- Linux
- Little-known features
- Malware
- Myths & misconceptions
- Privacy
- Safeguarding data

- Security fun
- Virtualization
- Vulnerability assessment
- Web server security
- Windows security

Recommended links

- F-Secure blog
- Michael Geist's blog
- OS News
- OSVDB blog
- SANS Handler's Diary
- Schneier on Security
- TaoSecurity

Possibly related

- Linux software vs hardware RAID
- Sanitizing hard drives at the hardware level
- Linux high availability clustering

Archives

- October 2007
- September 2007
- August 2007
- July 2007
- May 2007
- April 2007
- March 2007
- January 2007
- December 2006
- November 2006
- October 2006
- September 2006
- August 2006

- July 2006
- May 2006
- April 2006
- August 2005

Subscribe

Get This Blog via Email:

Subscribe

Powered by Squeet.com

» View my profile

Technorati

My links on del.icio.us

211 readers BY FEEDBURNER

« Previous article — Next article »

Setting up software RAID in Ubuntu Server

April 24th, 2007 Posted by Derrick Webber

Linux has excellent software-based RAID built into the kernel. Unfortunately information on configuring and maintaining it is sparse. Back in 2003, O'Reilly published Managing RAID on Linux and that book is still mostly up to date, but finding clear instructions on the web for setting up RAID has become a chore.

Here is how to install Ubuntu Server with software RAID 1 (disk mirroring). This guide is for Ubuntu Server 6.06 LTS, but the procedure should be nearly the same for newer versions of Ubuntu and even Debian. We've also included a few other tips, like how to ensure you can still boot your server when the first drive fails and how to rebuild when replacing a failed drive.

Software RAID vs. hardware RAID

Some system administrators still sneer at the idea of software RAID. Years ago CPUs didn't have the speed to manage both a busy server and RAID activities. That's not true any more, especially when all you want to do is mirror a drive with RAID1. Linux software RAID is ideal for mirroring, and due to kernel disk caching and buffering it can actually be faster than RAID1 on lower end RAID hardware. However, for larger requirements like RAID 5, the CPU can still get bogged down with software RAID.

Software RAID is inexpensive to implement: no need for expensive controllers or identical drives. Software RAID works with ordinary EIDE, Serial ATA and SCSI drives and controllers. You can mix together whatever drive types and sizes you have on hand. When all you need are mirrored drives, software RAID is an especially good choice.

However, there are reasons you might prefer hardware RAID over software RAID:

- Hardware RAID is simpler to setup and manage.
- Depending on the server BIOS, a system using Linux software RAID probably won't be able to automatically boot if the first disk of a mirror fails (It can still be booted manually though).
- Linux will only boot when the "/boot" and "/" partitions are on RAID1. It cannot boot when those partitions are on RAID5. Other partitions, however, can be RAID5.
- With software RAID, after replacing a failed drive the administrator must login and enter commands to add the new drive to the array and re-sync the contents. Good hardware RAID controllers re-sync automatically as soon as they see a new drive, without operator intervention.

Notice I said "good hardware controllers". Low-end controllers like those integrated with consumer-grade motherboards that require software drivers are not good controllers for server use. Cheap motherboard RAID is designed for gamers who want RAID to boost disk read times, not for reliability. These "fake RAID" controllers are really no better than Linux software RAID. Good hardware RAID requires serious controllers from Adaptec, 3ware or another reputable manufacturer.

A simple RAID1 example

For this example we'll construct a simple RAID1 mirror using a server that has two 4 GB serial ATA drives. Such a configuration will keep running if either drive fails, but (obviously) not if both fail.

EIDE or SCSI drives can also be used with Linux RAID, but right now serial ATA provides the best combination of low cost, performance and flexibility.

This example was done on Ubuntu Server 6.06 LTS, but the procedure should be nearly the same for newer versions of Ubuntu. The concepts involved also apply to any version of Linux running a 2.6 kernel, though the exact setup procedure differs for other flavors of Linux.

For this example, partitioning will be done as simply as possible:

Drive	Partition	Type	Mounted on	Size
Drive0	/dev/sda1	Primary	/	4.1 GB
	/dev/sda2	Primary	(swap area)	(remainder of disk)
Drive1	/dev/sdb1	Primary	/	4.1 GB
	/dev/sdb2	Primary	(swap area)	(remainder of disk)

In Linux software RAID each mount point is usually configured as a separate RAID device. It's possible for entire drives to be RAID members rather than each partition (e.g. combine /dev/sda and /dev/sdb) but the resulting device will not be bootable.

In this example partitions sda1 and sdb1 will be made members of a RAID1 device named /dev/md0. Partitions sda2 and sdb2 will be members of a RAID1 device named /dev/md1.

RAID device	Type	Mounted on	Size	Members
/dev/md0	RAID1 mirror	/	4.1 GB	/dev/sda1
				/dev/sdb1
/dev/md1	RAID1 mirror	(swap)	(remainder of disk)	/dev/sda2
				/dev/sdb2

On a real world server it's a good idea to have at least /var and /home on their own partitions, but the above scheme is good enough for this example. We are also purposely avoiding complications like logical volume management (LVM), just to keep things simple.

In Linux RAID, corresponding partitions on each drive in a RAID device should be the same size. If they aren't, software RAID will still function but each RAID device will only be as large as the smallest member partition (e.g. if you add a 10GB partition and a 20GB partition into a RAID1 array, the resulting array will only be 10GB in size).

Installing Ubuntu server with RAID1

To install a fresh Ubuntu System with RAID, boot from the CD-ROM as usual. Follow the prompts until you get at the "Partition Disks" dialog.

- From the "Partitions Disks" dialog box, select "Manually edit the partition table".
- Select the first disk ("sda")
- Say yes to "Create a new empty partition table on this device?"

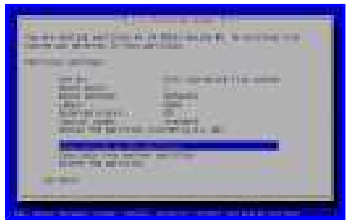
- Use the dialog boxes to create one primary partition large enough to hold the root filesystem (4.1 GB in this example)
- For "How to use this partition" select "physical volume for RAID", *not* the default "Ext3 journaling file system"
- Make the partition *bootable*.
- Use the dialogs to create one other primary partition taking up the remaining disk space (197.4 MB in this example). Later this will be used for swap.
- For "How to use this partition" select "physical volume for RAID", *not* the default "Ext3 journaling file system" and *not* "swap area"
- Repeat the above steps to create identical partitions on the second drive. Remember to mark partition one on both drives as "bootable". The final result should look similar to the following:

(click for full size)

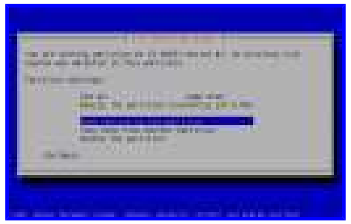
- Once the partitions are configured, at the top of the "Partition Disks" main dialog select "Configure Software RAID"
- When asked "Write the changes to the storage devices and configure RAID" select "Yes".
- For "Multidisk configuration actions" select "Create MD device"
- For "Multidisk device type" select "RAID1″
- For "Number of active devices for the RAID1 array" enter "2″
- For Number of spare devices for the RAID1 array" enter "0″ (zero)
- When asked to select "Active devices for the RAID1 multidisk device" select both /dev/sda1 and /dev/sdb1
- From the next dialog select "create MD device"
- Repeat the above steps to create an MD device that contains /dev/sda2 and /dev/sdb2
- Finally, from the dialog "Mulidisk configuration actions" select "Finish"

Next configure device md0 to be mounted as the "/" filesystem and device md1 to be mounted as swap:

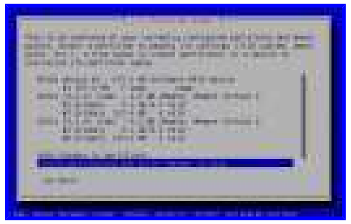
- From the "Partition Disks" dialog, move the cursor bar under "RAID device #0″ and select "#1 4.1 GB"
- Configure the device as an Ext3 filesystem mounted on /, as shown:


(click image for full size)

- From the Partition Disks dialog under “RAID device #1″ select “#1 197.3 MB”
- Configure the device as “swap area”, as shown:

 (click image for full size)

- The final partitioning screen should resemble the following:

 (click image for full size

- Select “Finish partitioning and write changes to disk”.The RAID1 mirrors are created and made active, the the filesystem is formatted and installation of Ubuntu proceeds as usual.
- Allow the installation to complete then reboot when requested.

Making every drive bootable

The Ubuntu installation program installs the GRUB boot loader into the master boot record (MBR) of the first disk so if you made the first partitions of each hard drive bootable as described above, the system should have no problem booting from the RAID1 array.

However, the bootloader is not installed on other drives. This leaves you with an unbootable system when drive 0 fails. Also, depending on your server it’s possible that GRUB will not be installed on the first drive at all, also leaving you with an unbootable system.

To fix a non-bootable system and ensure GRUB is installed on both drives, manually install GRUB as follows:

- Reboot the server from the original Ubuntu Server CDROM
- From the Ubuntu boot menu, select “Rescue a broken system”
- Continue through the prompts until the screen “Device to use as a root file system” appears
- Press Alt-F2 to switch to a second console screen then press Enter to activate it.
- Mount the md0 RAID device and use chroot and grub to install the bootloader onto both sda and sdb using the following commands

```
mount /dev/md0 /mnt
chroot /mnt
grub
device (hd0) /dev/sda
root (hd0,0)
setup (hd0)
device (hd1) /dev/sdb
root (hd1,0)
setup (hd1)
quit
```

(many thanks to BigDiver for the above)

- Reboot the system with command "shutdown -r now", remove the CDROM and allow the system to boot from the hard drive.

Why RAID swap?

You might be wondering why we put swap on a RAID device, causing system swap activity to suffer the additional overhead of RAID.

Though Linux is capable of handling multiple independent swap partitions on multiple drives, if a drive containing an active swap partition dies it may take the system down with it. That defeats the point of having RAID in the first place, so to avoid that possibility we put the swap in RAID.

This creates more overhead, but swap is only meant as temporary substitute memory during rare moments of high load. If the system is regularly using swap, performance is already being severely impacted and it's time to add more physical memory.

Care and feeding

Having two drives configured in a RAID1 mirror allows the server to continue to function when either drive fails. When a drive fails completely, the kernel RAID driver automatically removes it from the array.

However, a drive may start having seek errors without failing completely. In that situation the RAID driver may not remove it from service and performance will degrade. Luckily you can manually remove a failing drive using the "mdadm" command. For example, to manually mark both of the RAID devices on drive sda as failed:

```
mdadm /dev/md0 –fail /dev/sda1
mdadm /dev/md1 –fail /dev/sda2
```

The above removes both RAID devices on drive sda from service, leaving only the partitions on drive sdb active.

Removing a failed drive

When a drive fails it is vital to act immediately. RAID drives have an eerie habit of all failing around the same time, especially when they are identical models purchased together and put into service at the same time. Even drives from different manufacturers sometimes fail at nearly the same time… probably because they all experience the same environmental factors (power events, same number of power downs, the same janitor banging the vacuum into the server every night, etc.)

When Ubuntu sees that RAID has been configured, it automatically runs the mdadm command in "monitor mode" to watch each device and send email to root when a problem is noticed. You can also manually inspect RAID status using commands like the following:

```
cat /proc/mdstat
mdadm –query –detail /dev/md0
mdadm –query –detail /dev/md1
```

It's also wise to use "smartctl" to monitor each drive's internal failure stats. However as noted in a recent analysis by Google (PDF link), drives are perfectly capable to dying without any warning showing in their SMART monitors.

To replace a drive that has been marked as failed (either automatically or by using "mdadm –fail"), first remove all partitions on that drive from the array. For example to remove all partitions from drive sda:

```
mdadm –remove /dev/md0 /dev/sda1
mdadm –remove /dev/md1 /dev/sda2
```

Once removed it is safe to power down the server and replace the failed drive.

Boot problems

If it was the first drive that failed, after replacing it with a new unformatted drive the system may no longer boot: some BIOS only attempt to boot from the lowest numbered hard drive (e.g. sda or hda) and if it is blank the system will hang. In that case you'll need a rescue CD capable of running a GRUB boot prompt so you can manually boot from the second physical drive.

There are many free Linux-based rescue CDs available (e.g. SystemRescueCD) but for quick access to GRUB try the Super Grub Disk. This small download can be written to bootable floppy or CDROM and give quick access to system boot tools, especially the GRUB command line.

Whatever rescue tool you use, use it to boot to a GRUB command prompt and force the system to boot from the second installed hard drive using commands similar to the following:

```
root (hd1,0)
kernel /boot/vmlinuz-whatever root=/dev/md0 ro
```

```
initrd /boot/initrd.img-whatever
boot
```

To find the correct file names for the “kernel” and “initrd” parameters, GRUB has bash-style command-line completion… type just enough of the path then press TAB to auto-complete or see a list of available choices.

Preparing the new drive

Once system as been rebooted with the new unformatted replacement drive in place, some manual intervention is required to partition the drive and add it to the RAID array.

The new drive must have an identical (or nearly identical) partition table to the other. You can use fdisk to manually create a partition table on the new drive identical to the table of the other, or if both drives are identical you can use the “sfdisk” command to duplicate the partition. For example, to copy the partition table from the second drive “sdb” onto the first drive “sda”, the sfdisk command is as follows:

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

Warning: be careful to specify the right source and destinations drives when using sfdisk or your could blank out the partition table on your good drive.

Once the partitions have been created, you can add them to the corresponding RAID devices using “mdadm –add” commands. For example:

```
mdadm –add /dev/md0 /dev/sda1
mdadm –add /dev/md1 /dev/sda2
```

Once added, the Linux kernel immediately starts re-syncing contents of the arrays onto the new drive. You can monitor progress via “cat /proc/mdstat”. Syncing uses idle CPU cycles to avoid overloading a production system, so performance should not be affected too badly. The busier the server (and larger the partitions), the longer the re-sync will take.

Note that you don’t have to wait until all partitions are re-synced… servers can be on-line and in production while syncing is in progress: no data will be lost and eventually all drives will become synchronized.

Summary

Linux software RAID is far more cost effective and flexible than hardware RAID, though it is more complex and requires manual intervention when replacing drives. In most situations, software RAID performance is as good (and often better) than an equivalent hardware RAID solution, all at a lower cost and with greater flexibility. When all you need are mirrored drives, software RAID is often the best choice.

More information on Linux RAID:

- Managing RAID on Linux (O'Reilly Media, 2003)
- Software RAID HOWTO (Linux Documentation project)
- Linux: Why software RAID? (Jeff Garzik)

Tags: Linux, RAID, software RAID, system administration
Posted in **Linux**, **Safeguarding data** | Trackback

16 Responses to "Setting up software RAID in Ubuntu Server"

1. *Nathan* Says:
   April 25th, 2007 at 10:51 am

   An excellent article, I've used software RAID-1 in the past with exceptional results, however, if using EIDE I would recommend placing each drive as master on each IDE channel instead of a master-slave configuration on the primary IDE channel. This will maximize disk throughput and bandwidth across the drives.

2. *D Webber* Says:
   April 25th, 2007 at 11:21 am

   Yes, good point. Since EIDE can only write to one drive on a channel at a time, each EIDE drive in a RAID mirror should be the master on separate channels.

   Another consideration is that the EIDE channels used for hard drives should not be shared with slower devices like CDROM and tape drives since the channel is a shared bus that only communicates at the speed of the slowest attached device.

   That creates a problem since most motherboards only provide two EIDE channels. If both of those are dedicated to the two mirrored RAID hard drives, nothing is left for the CD.

   Ideally you'd install a EIDE controller card on the PCI bus and use that for the slower EIDE devices like CDROM.

   Fortunately Serial ATA is pretty much the standard now and doesn't have the shared bus limitation.

3. *ZXpower* Says:
   May 14th, 2007 at 5:33 pm

   Nice job done!

   But I also have a question - how to setup RAID1 on already running Ubuntu Server without reinstall? On FreeBSD it can be done somehow easily using geom, but what about Ubuntu? Any ideas?

4. *D Webber* Says:
   May 15th, 2007 at 10:07 am

   @ZXpower:

   The process of adding RAID to a running Linux server is not as easy as typing a single command. I'm not going to write another step-by-step HOWTO on how to do it (do you have any idea how long it takes to research, test and write these procedures? ;-), but you can find some guidance on how to do it by searching the net.

   It's also covered on Page 81 of the Managing RAID on Linux book. Everyone working with Linux RAID should own a copy.

   If you plan to try it, make sure you have a solid restorable backup of the target system… chances are it will not boot first time and one mistyped device name could wipe out a production partition. A virtualization product (like the free VMware Server) is an excellent way of testing and developing procedures like this before trying them in production.

5. *Otto* Says:
   June 16th, 2007 at 4:40 pm

   I believe there is a typo
   "In this example partitions sda1 and sda2 will be made members of a RAID1 device named /dev/md0. "
   should read"In this example partitions sda1 and sdb1 will be made members of a RAID1 device named /dev/md0. "

6. *D Webber* Says:
   June 16th, 2007 at 7:39 pm

   @otto:

   You're right. Thanks very much for pointing out the mistake! Fixed.

7. *Mike H* Says:
   July 14th, 2007 at 11:21 am

   I'm trying to do what ZPower was asking about. I've been trying to setup RAID-1 on Ubuntu 7.04 Feisty Desktop for several days. I'm not doing a clean install - I want to effectively use my existing drive as half of the RAID without having to do a rebuild. I've created copies of the partitions on a new drive, set the partition types to fd, set-up the arrays (with one drive missing) and then copying the data from the non-RAID drive onto the 'half a RAID drive', if you see what I mean. So far so good. However, I've been trying to base the layout on the original Ubuntu default install, i.e. just a / and swap partition - NO separate /boot partition.

   Your excellent article led me to believe that this was possible - you use the same layout in your examples, but I cannot do the mount, chroot, grub bit that installs the boot loader to make both disks bootable. I’m completely unable to configure Grub on the half-RAID drive.

Grub seems to refuse to be ‘raid aware’ - something that I have read elsewhere, which is why I was interested when I first found your article.

Attempting to do anything to the array device using grub fails with error 15. I initially found this when trying to use find to locate the stage1 files on the two disks, but the same problem arises irrespective of the file being found.

For instance if I do the following (as root):-

```
mount /dev/md0 /mnt/md0
find /mnt/md0/etc/fstab
```

… it works as expected and reports the presence of fstab

However, if I run find from within grub…

```
(/dev/md0 is still mounted)
grub
find /mnt/md0/etc/fstab
```

… it reports error 15 file not found

Is this caused by a difference in the version of Grub used in the server and desktop Ubuntu images?

Or is it something basic I’ve missed - quite possible as I’m new to Linux/Ubuntu…

Any suggestions, anyone?

Thanks,

Mike

8. *D Webber* Says:
   July 15th, 2007 at 3:36 pm

   Grub is “RAID aware” to the extent it can boot from a RAID 1 mirror (but apparently no other RAID level).

   First, make sure the /etc/fstab file on the RAID drive is pointing to the correct md devices (e.g. /dev/md0 not /dev/sda1).

   Second, the file /boot/grub/menu.lst should also point to the md device for booting. Here’s an example straight from a production Ubuntu 6.06 server booting from RAID1:

```
title Ubuntu, kernel 2.6.15-28-server
root (hd0,0)
kernel /vmlinuz-2.6.15-28-server root=/dev/md0 ro quiet splash
initrd /initrd.img-2.6.15-28-server
savedefault
boot
```

Note the “root=/dev/md0″ in the above.

Finally, follow the steps in the above article for copying the grub bootloader… for example:

```
mount /dev/md0 /mnt
chroot /mnt
grub
device (hd0) /dev/sda
root (hd0,0)
setup (hd0)
```

(Substitute /dev/sda for the appropriate physical drive device on your system)

9. *Anthony* Says:
   August 19th, 2007 at 6:42 pm

   Hi all,

   Can you boot from only one of the drives? I’m thinking of the situation where it turns out that after an outage, one of the drives is dead and the system tries to reboot. I currently have a “mdadm: no devices listed in config file where found” error. Is this normal?

   Thanks in adavance for your ideas 😀
   Anthony

10. *eRIZ* Says:
    September 3rd, 2007 at 1:04 pm

    Cool!

    I’ve been looking for any manual, but - as you mentioned - it’s difficult, to find anything that describes the installation process on RAID.

I’m going to write a Polish version of the solution and I’ll drop you trackback. ;]

Thanks for a good article! You saved my life. ;P

11. *eRIZ’s weblog » Instalacja Ubuntu na RAID-1* Says:
    September 6th, 2007 at 10:23 am

    […] ojczystym języku informatyki, jakim jest angielski, znalazłem już coś ciekawszego. Trafiłem na artykuł, który okazał się remedium na mój […]

12. *Matt* Says:
    October 9th, 2007 at 7:54 am

    Many Thanks,

    Excellent guide - worked flawlessly with Ubuntu server 7.04

13. *fentex* Says:
    October 25th, 2007 at 7:32 pm

    I’ve followed your thankfully clear and understandable directions but have hit a roadblock while trying to make both disks bootable.

    With Ubuntu Server 7.10 (AMD64) when I attempt to execute:
    chroot /mnt

    I get the response:
    chroot: cannot execute /bin/sh: No such file or directory

    It has stalled me. Advice on what to do here, from what I’ve found on the web, stymies me as I’ve only a superficial understanding of Linux and haven’t found any help I understand.

    Could someone please tell me what I need do to proceed?

14. *Derrick Webber* Says:
    October 28th, 2007 at 2:12 pm

    @fentex:

    That sounds like the partition didn’t really get mounted before you did the chroot (or the wrong device got mounted). Perhaps check there are files in the /mnt directory before doing the chroot… in particular, there should be a /mnt/bin/bash file present.

15. *Kieran* Says:
    October 30th, 2007 at 3:04 pm

    I am curious about using software raid and hotswap enclosures for offsite storage. Currently I have a hardware raid 1 setup with windows 2000server and I can remove one of the drives and plug one back in and the system rebuilds automatically thus giving me 100% offsite back up of the entire system.
    Is something like this possible with ubuntu software raid1? or what kind of commands would be needed to rebuild the drive after a hotswap?
    Or I guess alternatly could just do a shut down and then swap the drive and have it rebuild upon reboot. This has saved our company twice now! Once during a theft and the other due to a massive power surge that took out pretty much the entire server. Only lost a day or two of work each time.

    Basically I want to be able to take a full working backup offsite every few days with minimal hassle and swap between a few drives each time.

    Any input would be GREATLY appreciated!
    ps Fantastic tutorial!!!!

16. *Derrick Webber* Says:
    October 30th, 2007 at 7:33 pm

    @Kieran:

    Linux does support hot swap for many types of devices, including SATA and SCSI drives (e.g. see http://linux-ata.org/software-status.html#hotplug). So plugging / unplugging drives is supported at the hardware and OS level.

    However, Linux software raid doesn’t do an automatic rebuild / resync (same as Windows software RAID). You have to issue a few commands manually as root to add the drive and rebuild the mirror. I suppose it would be possible to write a detect / rebuild script that fires when a hotplug action is detected though.

    To be able to plug an unformatted drive into a RAID array and have the system automatically rebuild the mirror on it, you really need a hardware RAID solution. Some folks have had good experience with 3ware controllers under Linux but I can’t vouch for any particular vendor personally.