# LPIC-1 Study Group
## 3 Configuring Hardware

R. Scott Granneman

scott@granneman.com
www.granneman.com

This presentation
is based on
Roderick W. Smith's
*LPIC-1: Linux Professional Institute Certification Study Guide,*
2nd edition

That said,
there are many
additions, subtractions, & changes

# Introduction

# The BIOS & Core Hardware

# The Role of the BIOS

Basic Input/Output System (BIOS)
provides configuration tools
& initiates the OS booting process

Resides on motherboard in ROM

# What does the BIOS do when you turn on your computer?

1. Performs POST (power-on self test)
2. Initializes hardware
3. Loads boot loader from boot device
4. Passes control to boot loader, which loads OS

Why fiddle with BIOS?

Enable or disable hardware

# IRQs

*IRQ*
*Interrupt request*

Signal sent to the CPU
instructing it to
suspend current activity
& to handle an external event
such as keyboard input

Numbered 0-15

More modern computers provide more

Some reserved for specific purposes

Others are common, but may be reassigned

Others available to devices

ISA devices need their own IRQ

PCI devices can share IRQs

`cat /proc/interrupts`

View what IRQs are used for what

Linux doesn't use an IRQ
until the relevant driver is loaded

# I/O Addresses

# DMA Addresses

# Boot Disks
# &
# Geometry Settings

# Coldplug & Hotplug Devices

# Expansion Cards

# PCI Cards

# Kernel Modules

# Loading Kernel Modules

# Removing Kernel Modules

# USB Devices

# USB Basics

# Linux USB Drivers

# USB Manager Applications

# Hard Disks

# Common hard disk interfaces

PATA (ATA)
Parallel Advanced Technology Attachment

SATA
Serial Advanced Technology Attachment

SCSI
Small Computer System Interface

External

✓ USB

✓ IEEE-1394 (FireWire)

✓ SATA

✓ SCSI

# PATA

Once widely used

Now SATA is gaining

PATA found mostly
on older machines
or used for CD/DVD drives

*Parallel*
Advanced Technology Attachment

*Parallel* interface:
data transferred in parallel
over the cable at same time

Cables are wide,
either 40 or 80 lines

Each PATA connector can connect
up to 2 devices,
1 *master* (end) & 1 *slave* (middle)

Configured via jumpers on disks
or set to *cable select*:
drive attempts to configure itself
based on its position
on the PATA cable

For best performance,
disks should be placed
on separate controllers
rather than set as master & slave
on 1 controller

Until recently, most motherboards preferred to boot from PATA drives

Some BIOSs allowed you to use SCSI drives instead

In a mixed-drive setup, you may need to place the boot loader on a PATA drive

# PATA disks identified as /dev/hda, /dev/hdb, etc

`hda`

Master drive on 1st controller

`hdb`

Slave drive on 1st controller

`hdc`

Master drive on 2nd controller

`hdd`

Slave drive on 2nd controller

What does it mean if your system has `/dev/hda` & `/dev/hdc`, but nothing else?

Partitions add numbers

/dev/hda1 & /dev/hda2 & /dev/hda3

/dev/hdb1 & /dev/hdb2

Same naming rules for optical media

Usually don't have partitions

Most Linux distros
also create a soft link
at `/dev/cdrom` & `/dev/dvd`

Removable PATA disks
(Zip drives, for instance)
are named & numbered
like fixed PATA disks

Nowadays, some Linux distros
name PATA disks
like they’re SCSI disks

# SATA

*Serial*
Advanced Technology Attachment

Uses serial bus:
1 bit of data transferred at a time

Still faster than PATA, though*

SATA: 187–375 MB/s
PATA: 16–133 MB/s

* Theoretical maximums you will never reach in the real world

SATA is rapidly displacing PATA

SATA disks connect to controllers
on a one-to-one basis,
with a single cable for each device

No jumpers!

No worries
about positions on cables!

Thinner cables!

Modern BIOSs detect SATA disks
& may allow you to boot from them

Most SATA disks
named like SCSI disks

Older drivers, though,
name them like PATA disks

# SCSI

SCSI is a many-headed beast

Different kinds of
SCSI definitions,
cables,
& speeds

Traditionally a parallel bus
but newest version is serial:
Serial Attached SCSI

Faster than PATA,
but also more expensive,
so rare except for
older or very high-end systems
(also really old Macs)

SCSI is a PITA*

Let me count the ways

* Pain in the ass

Up to 8 or 16 devices per bus
(Including
the SCSI host adapter itself)

Also have cable-length limits
(differ for each SCSI variety)

Each device has a unique ID number,
assigned via a jumper

Standard BIOS
does not detect SCSI disks

You can boot from SCSI
if your SCSI adapter
has its own BIOS for booting

Otherwise boot
from PATA or SATA disk

Unfortunately, SCSI IDs aren't used to name devices in Linux

Hard drives
/dev/sda & /dev/sdb

SCSI tapes use numbers, not letters
/dev/st0 & /dev/st1
/dev/nst0 & /dev/nst1

SCSI CD- & DVD-ROMs use numbers
/dev/scd0 & /dev/scd1

SCSI device naming
(e.g., `/dev/sda` & `/dev/scd0`)
usually assigned in increasing order
based on SCSI ID order

This can greatly complicate your life

SCSI ID 2 has /dev/sda

SCSI ID 4 has /dev/sdb

What happens when you add another SCSI disk with ID of 0 or 1?

New disk is now /dev/sda

Old /dev/sda is now /dev/sdb

Old /dev/sdb is now /dev/sdc

SCSI ID 2 has /dev/sda

SCSI ID 4 has /dev/sdb

What happens when you add another SCSI disk with ID of 3?

Old /dev/sda is still /dev/sda

New disk is now /dev/sdb

Old /dev/sdb is now /dev/sdc

To further complicate things,
some SCSI adapters
start numbering at 7
& work down to 0!

Wide SCSI
goes from 7 to 0
& then from 14 to 8!

To make your life easier,
give hard drives lowest SCSI IDs
so you can add disks later
with minimal disruption

Multiple SCSI adapters?

Linux assigns device filenames to all disks on 1st adapter, then goes down 2nd adapter

Result:
You may not be able to control which adapter takes precendence

Don’t forget that
USB & SATA devices
also get SCSI device names,
so your hard disks
may be named unexpectedly

Loving SCSI yet?

Each end of a SCSI bus must be *terminated*

Devices in the middle must *not* be terminated

Different SCSI devices terminate in different ways

If your SCSI devices are acting weird, check termination first


Most SCSI busses
are *not* hot-pluggable,
so connecting or disconnecting
a SCSI device
while the computer is running
is a very bad idea

# External

External SCSI disks
are named
just like internal SCSI disks

External USB & IEEE-1394 disks
are also named
like SCSI devices

`/dev/sdf` & `/dev/sdh`,
for instance

# Designing a Hard Disk Layout

# Why Partition?

# Types of Disk Partitions

# An Alternative to Partitions: LVM

# Mount Points

# Common Partitions & Filesystem Layouts

Some directories
are commonly split off
into their own partitions

Not required

No one does all of these

Use your intelligence

Swap
(not mounted)

~2x RAM size

Adjunct to system RAM

/home

Holds user files

Allows you to upgrade system without disturbing user files

/boot

Contains critical boot files

Putting it in a separate partition lets you circumvent limitations of older BIOSs & boot loaders on hard disks over 8 GB

/usr

Contains Linux program & data files

(Author says this is sometimes the largest partition—huh?)

`/usr/local`

Linux programs & data files
that you compiled
or installed separately
from your package manager

/opt

Linux programs & data files
that you compiled
or installed separately
from your package manager,
especially commercial

/var

Variable files
that may change size regularly
or disappear when no longer needed

Often used on servers
for web pages, databases, logs, etc.

`/tmp`

Temporary files

/mnt
/media

Not a separate partition

Subdirectories within
are used as mount points
for removable media

**Never** place these directories
on separate partitions,
as they hold critical system files

`/etc`
`/bin`
`/sbin`
`/lib`
`/dev`

# Partitions & Filesystems

Partitioning involves 2 tasks

1. Creating the partition(s)

2. Preparing the partition(s) to be used

Don’t go partition crazy

Take your time

Learn from how your system is used

Think ahead

# Partitioning a Disk

`fdisk`

(*f*ixed *disk*)

Traditional tool for disk partitioning

`parted`

Newer tool that can both
partition & prepare,
& resize without losing data

# fdisk

Good idea to start
by viewing current partition table

```
fdisk -l /dev/hda
```

# fdisk /dev/hda

| | |
|---|---|
| m or ? | Show help |
| p | *P*rint current partition table |
| n | Create *n*ew partition |
| d | *D*elete partition |
| t | Change partition *t*ype |
| l | *L*ist partition types |
| a | Make partition bootable |
| q | *Q*uit without saving changes |
| w | *W*rite changes to disk & exit |

parted

Cross-platform:
works with x86 & non-x86
partition tables

More features
& easier to use
than `fdisk`

```
parted /dev/hda
```

Not covered on the LPI exam

# Preparing a Partition for Use

After you create a partition,
you must prepare it for use
by formatting the partition
(AKA creating a filesystem)

You write low-level data structures
to disk that tell Linux
how to access & store files

# Common Linux filesystem types

ext2

ext3

ext4

ReiserFS

JFS

XFS

Btrfs

`ext2`

Second Extended File System

Created for Linux
& dominant through 1990s

Good for small `/boot` partitions
where you don't need journaling
(although there's now
a non-journaled version of ext4)

ext3

Third Extended File System

Merged with Linux kernel in 2001

ext2 + journaling

(recovers from power outages
& system crashes
more quickly & reliably)

ext4

Fourth Extended File System

Merged with Linux kernel in 2008

Adds ability to work with large disks
(over 32 TB!)
or very large files
(over 2 TB!)
& better performance

Backward compatible with ext2/3

Standard filesystem now

JFS

Journaled File System

Developed by IBM for AIX & OS/2

Sophisticated journaling filesystem

Fast & reliable,
with good performance
under different kinds of load

XFS

Extents File System

Developed by Silicon Graphics (SGI)
for IRIX

Good at
robustness, speed, & flexibility

Very good at parallel I/O
(used at NASA
Advanced Supercomputing Division
with 2 300+ TB XFS filesystems
on 2 SGI Altix archival storage servers)

Btrfs

B-tree file system

Developed by Oracle
starting in 2007

Still considered unstable

Goals center around
scaling, reliability, & management

The future default Linux filesystem?

# What about ZFS?

Developed by Sun in 2001

128-bit file system:
can store up to
256 quadrillion zettabytes!

ZFS distributed under the Sun CDDL
(Common Development
& Distribution License)
which is incompatible with the GPL
that governs the Linux kernel

# Other non-native Linux filesystems

FAT

NTFS

HFS

HFS+

ISO-9660

Joliet

UDF

FAT

File Allocation Table

Old, primitive, ubiquitous

Limited to 8.3 filenames

Linux filesystem type code: `msdos`

FAT32

File Allocation Table

Includes 32-bit pointers

Supports long filenames

Linux filesystem type code: `vfat`

NTFS

New Technology File System

Windows NT/200x/XP/Vista/7/8

Linux 2.6 & above
can read & overwrite existing files

To write new files,
use the NTFS-3G driver,

included in most Linux distros,
which runs in user, not kernel, space

# HFS

Hierarchical File System

Introduced in 1985 by Apple for Macs

Superseded in 1998 by HFS+

Crude & limited

Full read & write support in Linux

# HFS+

Hierarchical File System Plus

Introduced in 1998 by Apple
for Macs

Supports larger files, Unicode,
compression, & encryption

Linux includes `hfsplus` module
to read & write to HFS+ disks,
but corrupts data
on drives over 2 TB!

# ISO-9660

Standard filesystem for CD-ROMs

*Rock Ridge* extensions to ISO-9660 support long filenames, permissions, soft links, etc.

UDF

Universal Disc Format

Next-gen filesystem for optical discs

Used on DVD-ROMs
& recordable optical discs

Reading supported on Linux;
writing supported to a point

# Creating a filesystem

mkfs.*fstype*

*fstype* is a filesystem type code

```
mkfs.ext4 /dev/hda1
mkfs.vfat /dev/hda4
```

Same thing as

```
mkfs -t ext4 /dev/hda1
```

`-c`

Perform bad-block *c*heck
on every disk sector

Takes a while to run,
but better safe than sorry

`-m` *`percent`*

Sets reserved space percentage

Default is 5%

Causes Linux to report
disk is full
before it really is,
so root can log in to fix things

Set larger (`-m 8`) to get more room,
or set smaller (`-m 2`) to take up less

Similar command structure
for non-Linux filesystems

`mkdosfs`
`mkfs.msdos`
`mkfs.vfat`

Create FAT filesystems

Creating swap space

2 kinds

1. swap partition

2. swap file

`mkswap /dev/hda7`

Create swap space

`swapon /dev/hda7`

Activate swap space

To permanently activate swap space, create an entry in `/etc/fstab` (next week!)

# Maintaining Filesystem Health

Problems

Overloaded with too much data

Tuned inappropriately

Corrupted due to
buggy drivers,
buggy utilities,
or hardware errors

Pay attention
if a maintenance tool
needs to be run
when the filesystem is *not* mounted

# Tuning Filesystems

Set filesystem options
that affect performance

dumpe2fs

tune2fs

debugfs

`dumpe2fs`

Provide info about
current configuration
of ext2/ext3/ext4 filesystem

OK to run on mounted filesystem

`dumpe2fs -h /dev/hda1`

Only display superblock info without details

```
root@gilgamesh:~# dumpe2fs -h /dev/xvdf
dumpe2fs 1.42 (29-Nov-2011)
Filesystem volume name:   <none>
Last mounted on:          /var
Filesystem UUID:          e27147f1-a484-4d7d-b156-eef356319d46
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype needs_recovery extent
 flex_bg sparse_super large_file huge_file uninit_bg dir_nlink extra_isize
Filesystem flags:         signed_directory_hash
Default mount options:    user_xattr acl
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              6553600
Block count:              26214400
Reserved block count:     1310496
Free blocks:              14848177
Free inodes:              6218979
First block:              0
Block size:               4096
Fragment size:            4096
Reserved GDT blocks:      1017
Blocks per group:         32768
Fragments per group:      32768
Inodes per group:         8192
Inode blocks per group:   512
Flex block group size:    16
Filesystem created:       Wed May  2 13:15:18 2012
Last mount time:          Fri Aug 17 00:53:27 2012
Last write time:          Fri Aug 17 00:53:27 2012
Mount count:              12
Maximum mount count:      -1
Last checked:             Wed May  2 13:15:18 2012
Check interval:           0 (<none>)
Lifetime writes:          444 GB
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:               256
Required extra isize:     28
Desired extra isize:      28
Journal inode:            8
```

If you're using XFS

```
xfs_info /dev/hda1
xfs_info /var
```

Requires filesystem is mounted

```
xfs_metadump /dev/hda1 /tmp/dump.txt
```

Copies metadata to file

`tune2fs options device`

Change ext2/3/4 filesystem parameters

Should *not* use
on a mounted filesystem!

`tune2fs -c 50 /dev/hda1`

Change maximum number of times disk can be mounted before mandatory `fsck` check

`tune2fs -C 50 /dev/hda1`

Set mount counter to 50 so `fsck` is put off or runs immediately

`tune2fs -i 30d /dev/hda1`

Run `fsck` every 30 days

`tune2fs -i 4w /dev/hda1`

Run `fsck` every 4 weeks

`tune2fs -i 6m /dev/hda1`

Run `fsck` every 6 months

`tune2fs -j /dev/hda1`

Adds journal to filesystem

Converts ext2 to ext3

`tune2fs -m 2 /dev/hda1`

Set percentage of disk space reserved for use by root

Default of 5%
is silly on huge multi-GB disks

Set to `0` on removable disks

`tune2fs -r 500 /dev/hda1`

Set number of blocks of disk space reserved for use by root

Just like `-m`,

but it uses blocks instead of a percentage

Much easier to use `-m`!

If you use XFS

`xfs_admin`

`xfs_admin -j /dev/hda1`

Use version 2 journaling

`xfs_admin -l /dev/hda1`

Get filesystem label (name)

e.g., `home`

`xfs_admin -u /dev/hda1`

Get filesystem UUID (Universally Unique ID)

e.g., `e40486c6-84d5-4f2f-b99c-032281799c9d`

`xfs_admin -L label`

Change filesystem's label

`xfs_admin -U uuid`

Change filesystem's UUID

`xfs_admin -U generate`

Generate new filesystem UUID

# debugfs

Interactively modify
a filesystem's features

Do *not* use on mounted filesystem

Combines
dumpe2fs, tune2fs, & other tools

```
$ debugfs /dev/hda1
debugfs: [enter commands]
```

Can `cd`, `ln`, `rm`, etc.
but also *much* more

`show_super_stats`

OR

`stats`

Display superblock info
(like `dumpe2fs`)

`stat` *file*

OR

`stat` *directory*

Display inode data

`undelete` *`inode name`*

*inode* = inode number of deleted file

*name* = new filename

Not very useful
since you need inode number

`lsdel`

OR

`list_deleted_inodes`

Display list of deleted inodes,
but may not help

`write` *`internal-file external-file`*

Extract file
without mounting filesystem

*internal-file* = name of file
in filesystem

*external-file* = filename
on main Linux system

```
list_requests
```
OR
```
lr
```
OR
```
help
```
OR
```
?
```

List available commands

`quit`

Exit debugfs

If you use XFS

`xfs_db`

Really intended for XFS experts

# Maintaining a Journal

A *journaling filesystem*
maintains a *journal,*
a data structure
describing pending operations

If a crash or power failure occurs,
system examines journal
& fixes problems

Vastly faster `fsck` on boot

Common journaling filesystems

ext3s

ext4s

ReiserFS

XFS

JFS

To use a journal,
you must mount the filesystem
with the correct type code
(e.g., `ext3` instead of `ext2`)

`tune2fs -J size=100`

Set journal to 100 MB

`tune2fs -J device=/dev/hda3`

Set device on which journal is stored

# Checking Filesystems

Common to check a filesystem for errors

`fsck` *`options filesystem`*

Verify filesystem integrity & correct problems (Actually a front end for `exfsck`, `xfs_check`, & others)

`fsck -A`

Check all filesystems marked to be checked in `/etc/fstab`

`fsck -C`

Display progress indicator

`fsck -V`

Be verbose

`fsck -N`

Dry run

`fsck -t ext3 /dev/hda1`

Instead of letting `fsck` determine filesystem type automatically, force the type

`fsck -A -t ext3`

Check all ext3 filesystems

# Monitoring Disk Use

Prevent filling your disk up

80% full? Clean it!

`df`

Summarize disk use for partition

`du`

Summarize disk use for directory

`df`

Shows how much
of the *d*isk is *f*ull

```
# df
Filesystem 1K-blocks      Used Available Use% Mounted on
/dev/xvda1  20743348   1945828  17749100  10% /
udev         8736940         4   8736936   1% /dev
tmpfs        3497968       180   3497788   1% /run
none            5120         0      5120   0% /run/lock
none         8744916         0   8744916   0% /run/shm
/dev/xvdf  103613900  45797752  52574164  47% /var


# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/xvda1       20G  1.9G   17G  10% /
udev            8.4G  4.0K  8.4G   1% /dev
tmpfs           3.4G  180K  3.4G   1% /run
none            5.0M     0  5.0M   0% /run/lock
none            8.4G     0  8.4G   0% /run/shm
/dev/xvdf        99G   44G   51G  47% /var
```

df -h
df --human-readable
Use 1024s

df -H
df -si
Also human readable,
but use 1000s

```
df -k
df --kilobytes
```

```
df -m
df --megabytes
```

`df -a`

Include all filesystems,
includes pseudo ones
like `/proc`, `/sys`, & `/proc/bus/usb`

`df -i`

`df --inodes`

Get report
on available & used inodes

Too many small files
can deplete available inodes
before disk space is depleted

Doesn’t work on filesystems
that create inodes dynamically,
like ReiserFS

`df -l`

`df --local`

Omit network filesystems

`df -T`

`df --print-type`

Show filesystem types

`df -t fstype`

`df --type=fstype`

Display only information
for specified filesystem types

`df -x fstype`

`df --exclude-type=fstype`

Exclude specified filesystem types

du

Show how disk space
a *d*irectory is *u*sing

Recursive,
so totals subdirectories too

```
# du
44  ./vim/scripts
8 ./vim/ftdetect
136 ./vim/plugin
8 ./vim/otlbin/outlinerconf/todo
32  ./vim/otlbin/outlinerconf
116 ./vim/otlbin
24  ./vim/syntax
…
20  ./vim/ftplugin
88  ./vim/doc
612 ./vim
656 .
```

```
# du -h
44K ./vim/scripts
8.0K   ./vim/ftdetect
136K   ./vim/plugin
8.0K   ./vim/otlbin/outlinerconf/todo
32K ./vim/otlbin/outlinerconf
116K   ./vim/otlbin
24K ./vim/syntax
…
20K ./vim/ftplugin
88K ./vim/doc
612K   ./vim
656K
```

`du -h`

`du --human-readable`

Use 1024s

`du -H`

`du --si`

Also human readable,
but uses 1000s

```
du -k
du --kilobytes
```

```
du -m
du --megabytes
```

`du -a`

`du --all`

Report on files
as well as directories

`du -l`

`du --count-links`

Instead of counting hard links only once,
count each hard link independently

`du --max-depth=3`

Limit report to 3 levels deep

Note that deeper subdirectories are still counted, just not reported

`du -x`

`du --one-file-system`

Limit report to current filesystem

```
# du -hs /Users/*
200K  /Users/Shared
214G  /Users/scott
```

-s

Summarize

du -c

du --total

Give grand total at end

# Mounting & Unmounting

To access a filesystem,
you *mount* it
by associating it
with a directory

`mount`

Temporarily mount

`/etc/fstab`

Edit to persistently mount across reboots

# Temporarily Mounting or Unmounting

mount *options device mountpoint*

```
mount /dev/hda4 /mnt/temp
```

Auto-detects filesystem type
& uses default options

```
mount -w -t ext4 /dev/hda4 /mnt/temp
```

*Mount point*
Directory to which device's contents are attached

`mount -v`

Verbose

`mount -a`

Mount all filesystems in `/etc/fstab`

`mount -r`

Mount read-only,
even if read-write

`mount -w`

`mount -o rw`

Mount read-write

`mount -t fstype`

Specify filesystem type

(`ext3`, `ext4`, `reiserfs`, `jfs`, `vfat`, etc.)

`mount -L label`

Mount by label

`mount -U uuid`

Mount by UUID

Normally, only root can run `mount`

If `/etc/fstab`

specifies users or owners,
then non-root user
can mount by specifying
device or mount point,
but not both

```
mount /mnt/cdrom
```

OR

```
mount /dev/hda1
```

When filesystems are mounted, recorded in `/etc/mtab`

Don’t edit this file manually!

`mount`

See what’s currently mounted

```
# mount
/dev/xvda1 on / type ext4 (rw)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
tmpfs on /run type tmpfs (rw,noexec,nosuid,size=10%,mode=0755)
none on /run/shm type tmpfs (rw,nosuid,nodev)
/dev/xvdf on /var type ext4 (rw)
```

mount -o *options*

loop

remount

ro

rw

uid=*userid*

gid=*groupid*

umask=*value*

dmask=*value*

fmask=*value*

## `mount -o remount`

Change mount options
without unmounting 1st
by issuing `mount` command
on already-mounted filesystem
along with `remount`
& any changed options

`mount -o loop`

Mount a file (.img or .iso)
as if it were a disk partition

`mount -o ro`

Read-only mount

`mount -o rw`

Read-write mount

`mount -o uid=`*`userid`*

Sets owner for all files
(look in `/etc/passwd` for user IDs)

`mount -o gid=`*`groupid`*

Sets group for all files
(look in `/etc/group` for group IDs)

`mount -o umask=`*`value`*

Sets permissions on files, based on bits removed from file permissions

`umask=027`

Gives permissions of `750` (`rwxr-x---`)

To calculate:
`777-umask=permissions`

`mount -o dmask=`*value*

Sets permissions on directories only

`mount -o fmask=`*value*

Sets permissions on files only

`umount`
`Unmount filesystem`

Not uNmount!

Specify either device or mount point, not both

Non-root users can't use `umount` unless listed in `/etc/fstab`

Only user who mounted can unmount

`umount -a`

Unmount all filesystems listed in `/etc/mtab`, but will not unmount `/` & other key filesystems

`umount -f`

Force unmount

# Permanently Mounting

/etc/fstab

(*f*ile*s*ystem *tab*le)

Controls how Linux
provides access
to disk partitions
& removable media devices

```
# cat /etc/fstab
# 1                   2    3    4                          5 6
LABEL=cloudimg-rootfs /    ext4 defaults                   0 0
/dev/xvdb            /mnt auto defaults,nobootwait,comment=cloudconfig
                                                          0 2
/dev/xvda3           none swap sw,comment=cloudconfig 0 0
/dev/xvdf            /var ext4 defaults                   0 0
```

1 Device

2 Mount point

3 Filesystem

4 Options

5 dump

6 fsck

```
# cat /etc/fstab
# 1                 2    3    4                    5 6
/dev/xvdf           /var ext4 defaults             0 0
```

1 Device

Can use
device filename (/dev/hda),
label (LABEL=/home),
UUID
(UUID=e27147f1-a393-4d7d-b156-eef356319d23)
network drive (server:/home),
or Samba drive (//winsrv/share)

```
# cat /etc/fstab
# 1                 2    3    4                    5 6
/dev/xvdf           /var ext4 defaults             0 0
```

2 Mount point

Should be an empty directory

```
# cat /etc/fstab
# 1                 2    3    4                    5 6
/dev/xvdf           /var ext4 defaults             0 0
```

3 Filesystem

Type code for the filesystem
(`ext3`, `ext4`, `vfat`)

or `auto`

to let kernel auto-detect filesystem type

```
# cat /etc/fstab
# 1                 2    3    4                    5 6
/dev/xvdf           /var ext4 defaults             0 0
```

# 4 Options

# Separate options by commas

## uid=500,umask=000,rw

`defaults`

Use default options

`auto`

Mount at boot

`noauto`

Do not mount at boot

`user`

Allows ordinary users to mount

`nouser`

Do not allow ordinary users to mount

credentials=/etc/creds

```
$ cat /etc/creds
username=sucker
password=ilovebillg
$ chown root:root /etc/creds
```

Needed for SMC/CIFS shares

```
# cat /etc/fstab
# 1                 2    3    4                    5 6
/dev/xvdf           /var ext4 defaults             0 0
```

5 dump

1 = dump utility should back up partition

0 = dump should not back up partition

Pretty meaningless now,
as dump is deprecated

```
# cat /etc/fstab
# 1                 2    3    4                    5 6
/dev/xvdf           /var ext4 defaults             0 0
```

# 6 fsck

Order in which fsck checks filesystem at boot

0 = no check

/ should always have 1

All others should have 2 or 0

# Review

# Thank you!

Email: scott@granneman.com

Web: www.granneman.com

Publications: www.granneman.com/pubs

Blog: ChainSawOnATireSwing.com

Twitter: scottgranneman

# LPIC-1 Study Group

## 1 Command Line Tools

R. Scott Granneman

scott@granneman.com
www.granneman.com

# Licensing of this work



Questions? Email scott@granneman.com