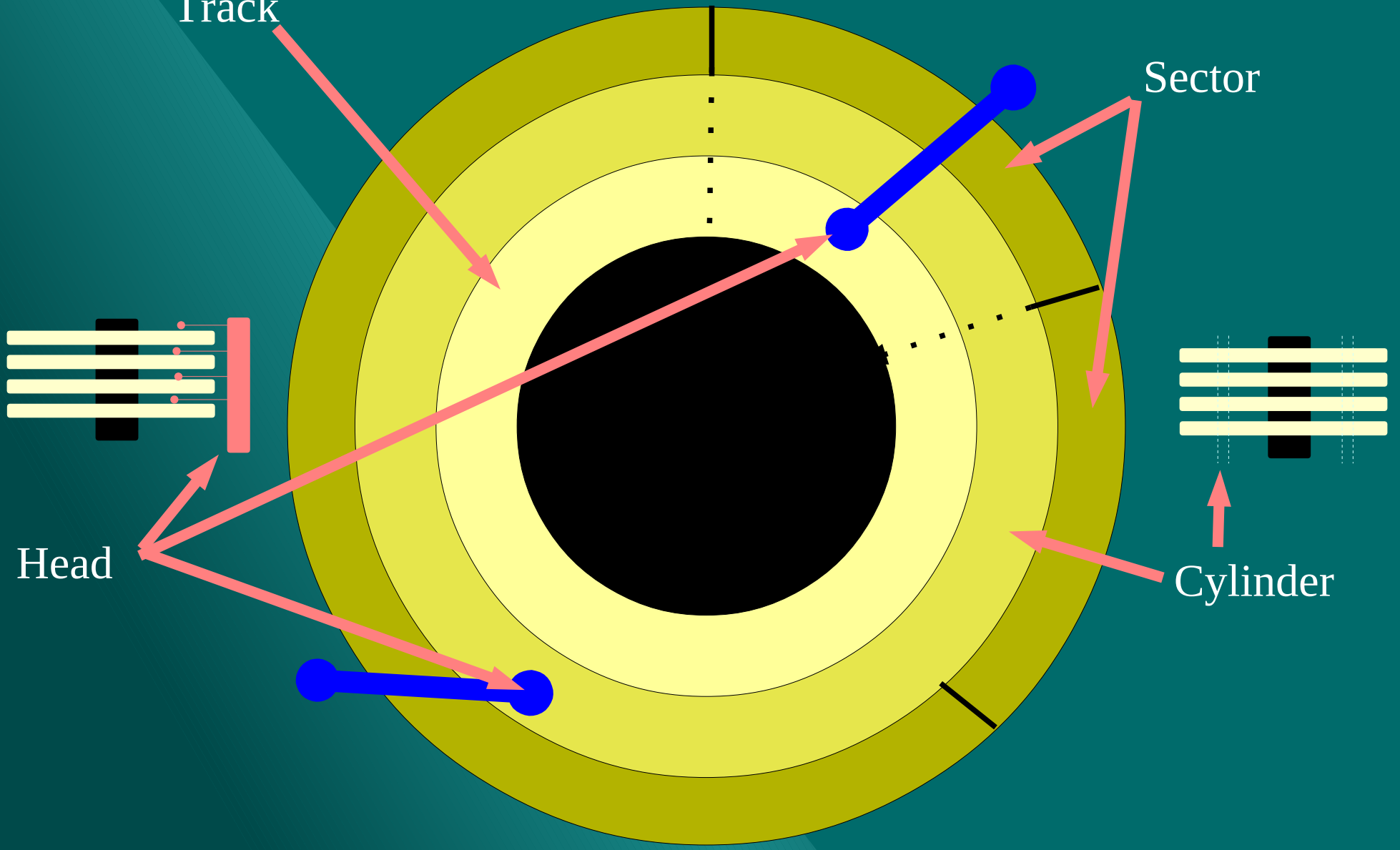
Linux Disk Management

Copyright

If you find this document useful or further it's distribution we would appreciate you letting us know.

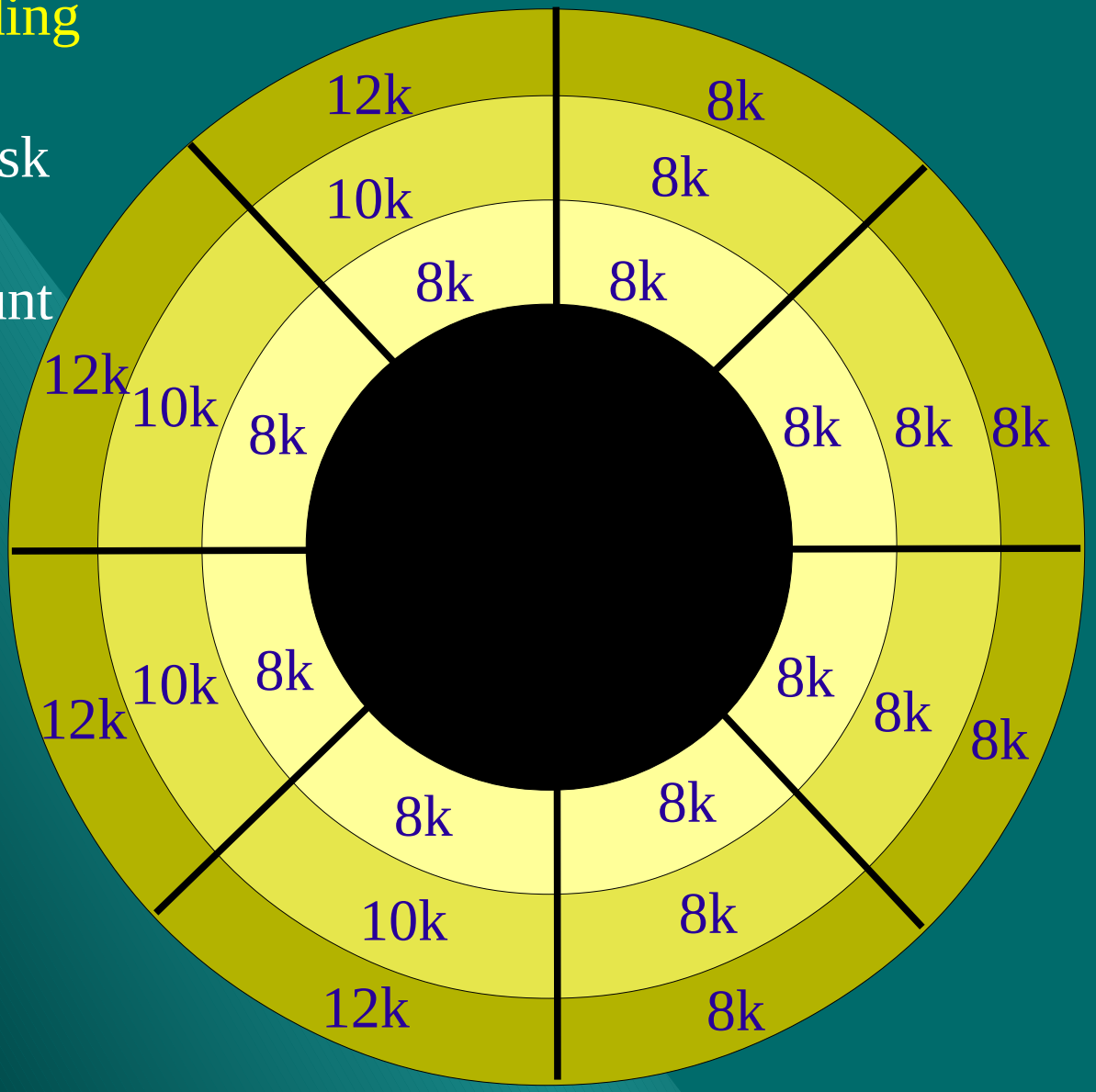
The Disk Itself

Geometry



Zone Bit Recording

Zone Bit Recording (ZBR) permits sectors on the disk to contain a "*variable*" amount of information and is one of the reasons for the sudden explosion in disk capacity.



With ZBR

120k

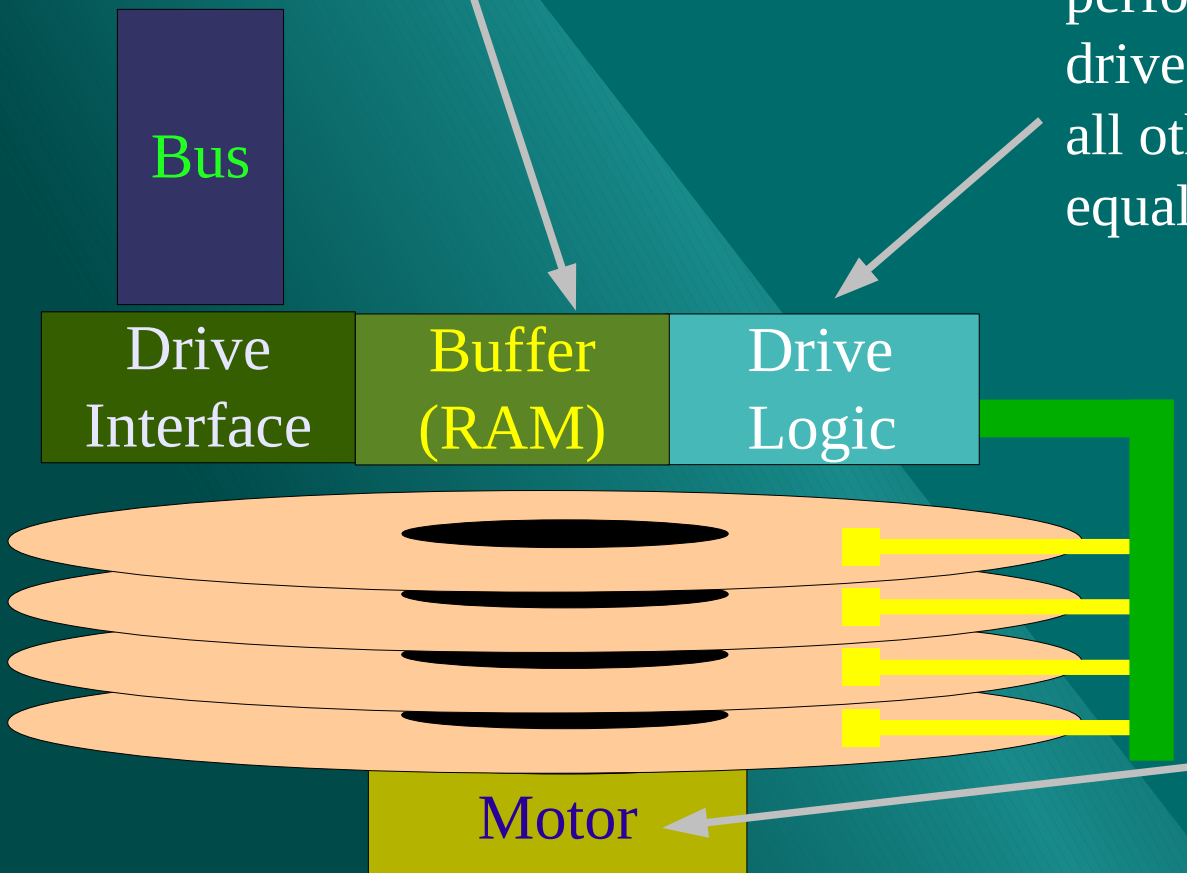
Without ZBR

96k

Beyond the platter...

Data waiting to be sent to the host or written to the drive platter is stored in RAM on the drive itself. More is better.

The drive firmware controls how the drive deals with error conditions, performs read-ahead, etc. High end drives often perform better even though all other characteristics are roughly equal.



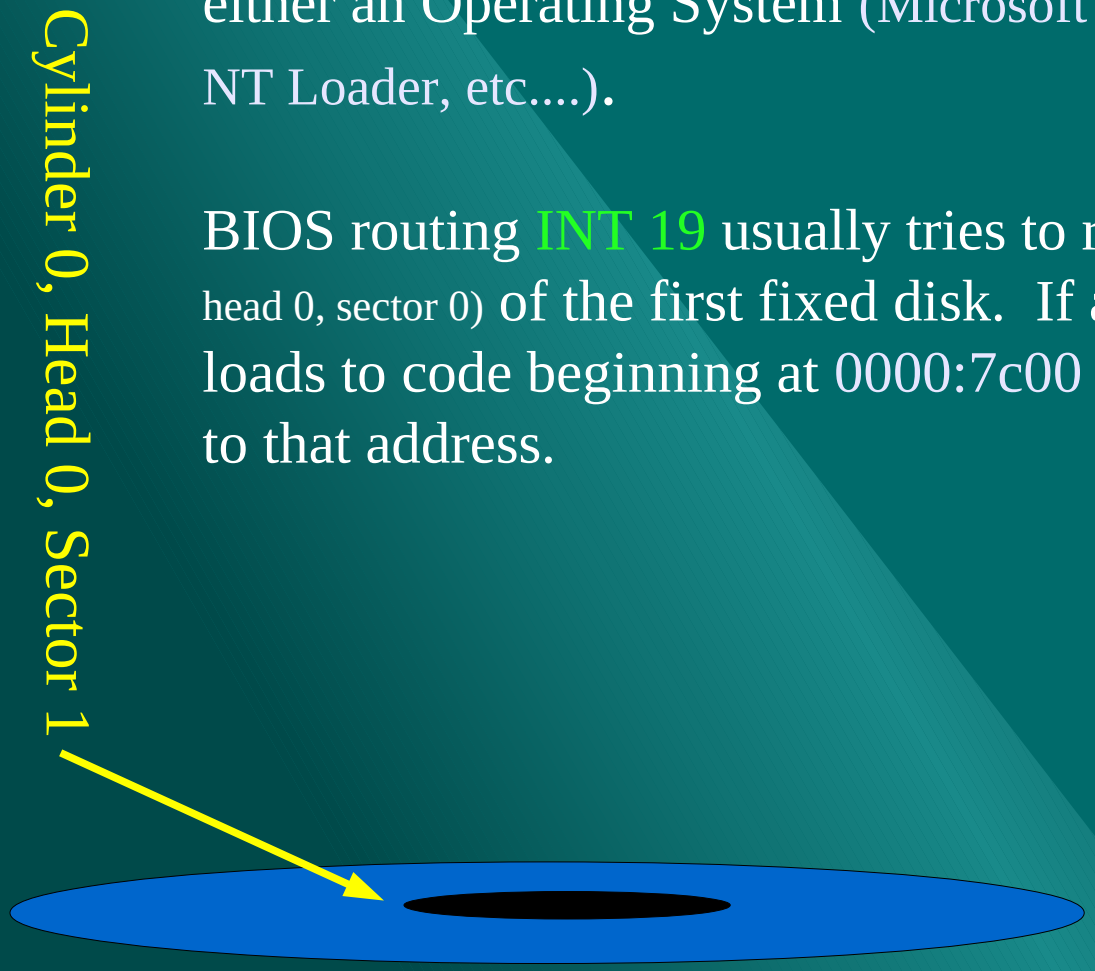
The rate at which the platters spin is rated in RPM. Typical speeds are 5,400, 7,200, and 10,000. Drives that spin faster are *generally* better performers as a given area of the platter passes under the head sooner / more often.

The Master Boot Record

These boots are made for walking... -Nancy Sinatra

After your Intel computer completes POST (Power On Self Test), it attempts to find a 16 bit real-mode program in order to commence operation; this is either an Operating System (Microsoft DOS) or a boot loader (LILO, GRUB, NT Loader, etc....).

BIOS routing INT 19 usually tries to read the Master Boot Record (cylinder 0, head 0, sector 0) of the first fixed disk. If a valid boot record is found, INT19 loads to code beginning at 0000:7c00 and moves the CPUs instruction pointer to that address.

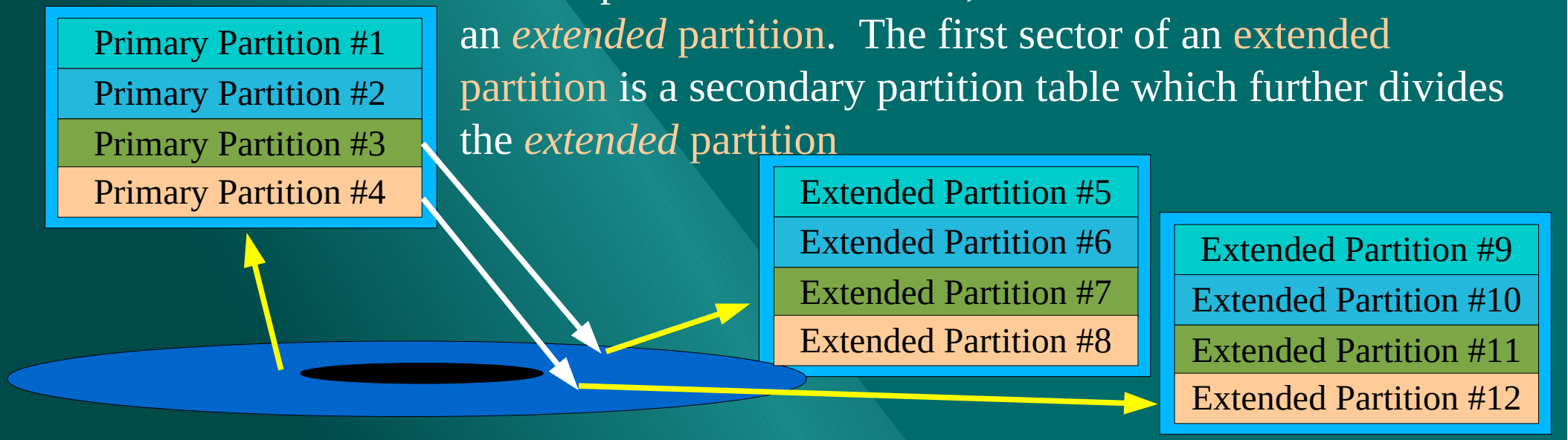


The Partition Table

To stand upon time's table, until the petals fall – Billy M. Smallwoord

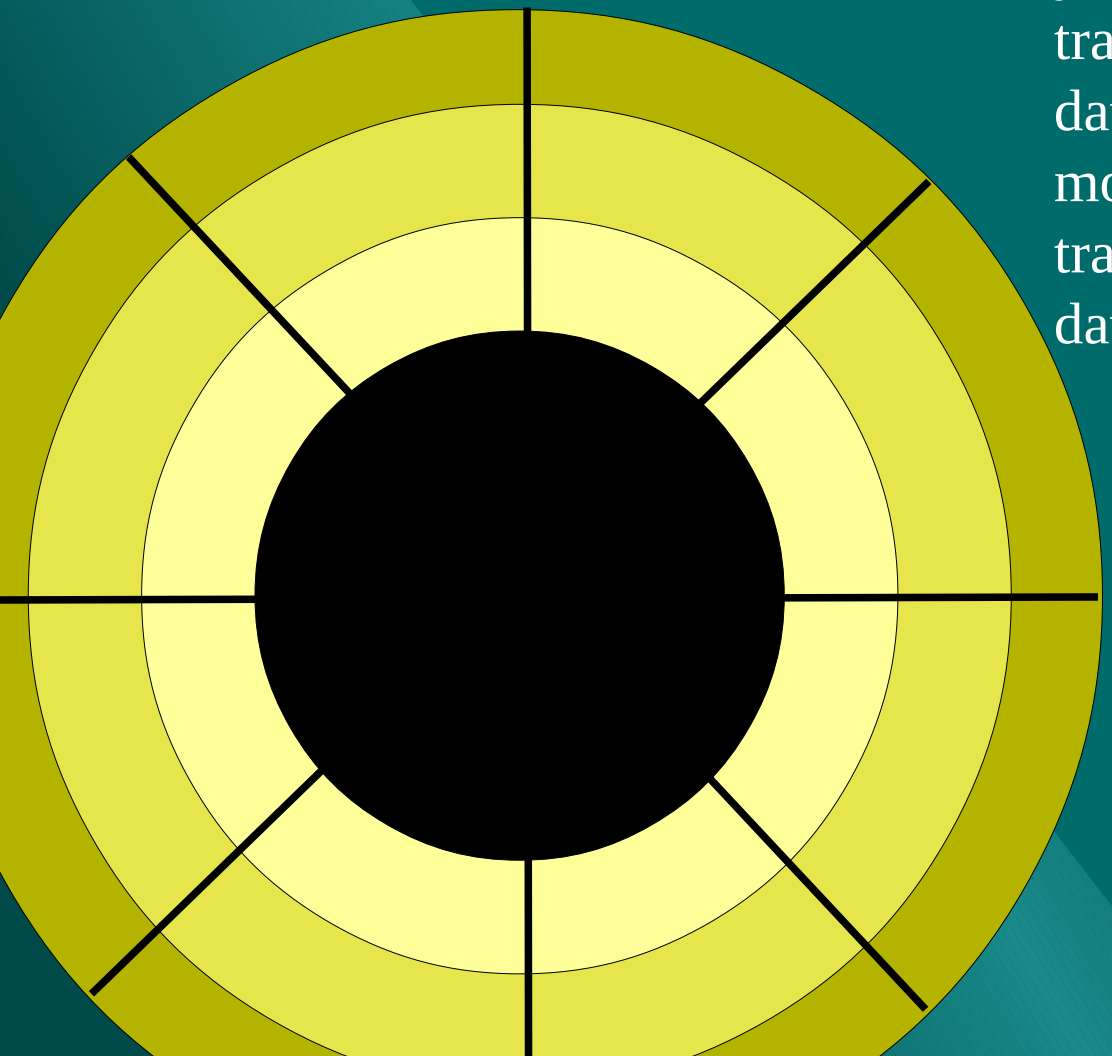
A disk is divided up into ranges of sectors referred to as partitions. Each partition is labeled with a type (FAT-12, FAT-16, Linux swap, OS/2) byte which advises the operating system how to deal with the contents of the partition (which may involve ignoring the partition entirely).

Bytes 446 through 509 (0x1be - 0x1fd) of the master boot record contain the drives primary partition table. All sectors between the master boot record and the first partition are unused. The primary partition table can only define four partitions. If more partitions are desired, one of the slots can indicate an *extended* partition. The first sector of an extended partition is a secondary partition table which further divides the *extended* partition



Physical Performance Effects

Until you get it on the red line overload – Kenny Loggins



One result of ZBR is that outer tracks yield a much higher "*sustained*" data transfer rate since more contiguous data can be read without head movement. On some drives the outer tracks will yield almost TWICE the data transfer rate as inner tracks.

"Two heads are better than one." Since the heads actually perform all the work, placing potentially high demand data under as many heads as possible will result in superior performance. This typically means "*stripe-ing*" the data across multiple/many disk devices.

The Bus
(Disk Connectivity)

The Bus / Controller

The wheels on the bus go round and round – Tweenies

A fixed disk attaches to the system via some type of Bus (and corresponding type of controller or adapter). The most common two classes of peripheral buses in "i386" class machines are IDE and SCSI.

Most USB attached fixed disks or removable media storage use SCSI-over-USB emulation.

See: http://www.linux-usb.org

Most parallel port attached fixed disks or removable media storage use IDE emulation, with each device containing its own IDE controller and bus.

See: http://www.torque.net/linux-pp.html

Linux also supports several "obsolete" drive topologies such as MFM, RLL, ESDI, and "primitive IDE". There are rumors that support for some of these old technologies will drop out somewhere in the 2.4.*x* series.

IDE

AT Attachment Peripheral Interface (ATAPI)

Two devices may be setup on an IDE bus, one set as Master and the other as Slave. If one IDE device dies or hangs, it takes the entire bus with it.

Traditional IDE transfer rates are defined by PIO (Programmed I/O) modes:

PIO Mode	Speed
1	3.3
2	5.2
3	8.3
4	16.6

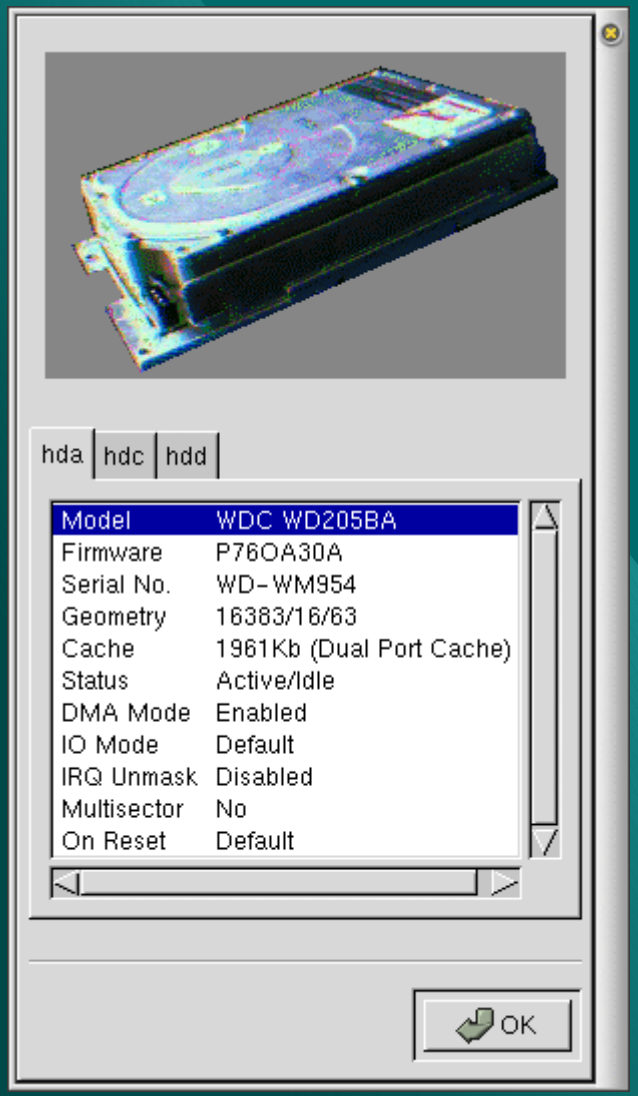
Bus (ide0)	Master (hda)
	Slave (hdb)
Bus (ide1)	Master (hdc)
	Slave (hdd)
Bus (ide2)	Master (hde)
	Slave (hdf)

Two devices on the same IDE bus may be limited to the PIO mode of the slowest device. Some IDE chipsets support "Independent Device Timing" in an attempt to overcome this limitation.

Ultra-IDE defines a standard for Bus Mastering IDE controllers that can transfer data to/from main memory via DMA as well as a parity checksum for data transferred on the bus. Unfortunately, very few drives currently support parity.

The IDE device is "blocking" when one device is active. The bus and other

IDE Tuning & Ximian GNOME



Because the IDE/ATAPI specification is quite vague about certain issues, various IDE chipsets handle advanced features, such as DMA, slightly differently. To avoid possible catastrophic problems, the Linux kernel typically assumes conservative defaults when establishing the IDE driver.

Ximian GNOME provides a GUI tool, called "idetool", for enabling the various features of your IDE controller and devices.

There is also a command line tool called "hdparm".

For more information on tuning your IDE devices see: ftp://kalamazoolinux.org/pub/pdf/PerfTune2001.pdf

SCSI

- SCSI is a more robust/advanced I/O bus than IDE, and until recently cost significantly more.
  - SCSI supports an entire range of devices, not just fixed disks and CD-ROM drives.
    - SCSI is commonly used for scanners, tape drives, H.A. solutions and high performance printers.
  - Each device on a SCSI chain is assigned a unique ID, with the controller itself usually set to ID "7".
    - Devices with lower ID's have precedence when the bus is under contention.
  - Typically a controller will attempt to boot from device "0" if its BIOS is enabled.
    - This can be adjusted at power-on with most controllers.
  - The end of a SCSI chain must be "terminated". Or the last device must support self-termination. Lack of termination or poor termination will result in bus errors and/or erratic behavior.
  - Different SCSI controllers vary widely in features/performance. Caveat Emptor.
  - There is no hard limit to the number of SCSI controllers that can be installed in a system.
  - A SCSI bus can handle device failures in most cases.

The Bus Showdown

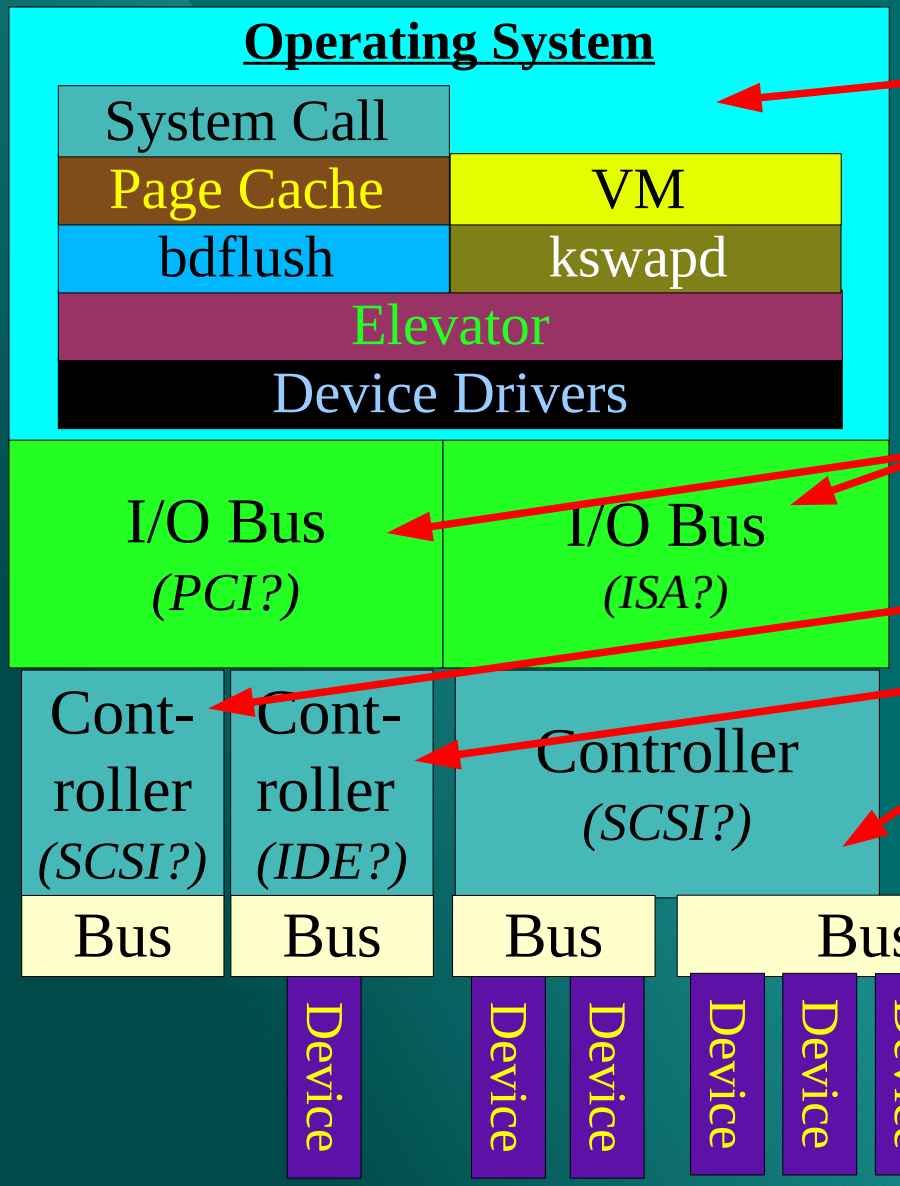
I'm going down in a blaze of glory – Jon Bon Jovi

Bus Type	Raw Speed	Device Count	Bus Width	Error Correction	Multiple Active
Traditional IDE	16.6	2	16	N	N
Ultra-33 IDE	33.3	2	16	?	N
Ultra-66 IDE	66.6	2	16	?	N
SCSI-1	5	7	8	Y	Y
Wide SCSI-2	10	15	16	Y	Y
Fast SCSI-2	10	7	8	Y	Y
F/W SCSI-2	20	7	16	Y	Y
Ultra SCSI-3	20	7	8	Y	Y
W/U SCSI-3	40	15	16	Y	Y
S-SCSI (Firewire)	12~50	?	1	Y	?

- Some newer SCSI buses in high end machines support up to 160Mbps bus speed.
- ? = The spec. defines an error-correction procedure that is not used by all drives.
- Bus speed bears no relation to the throughput of a given drive or device.

The I/O Stack

The I/O Stack



The OS manages all access to fixed disk devices.

Some bus architecture exists between the CPU(s)/core memory and the I/O device controllers. A single I/O bus may manage multiple I/O device controllers.

The controller integrates the system's I/O bus to the device I/O bus. A single controller may manage multiple device buses.

The device bus (cable) links the controller and devices of the appropriate type.

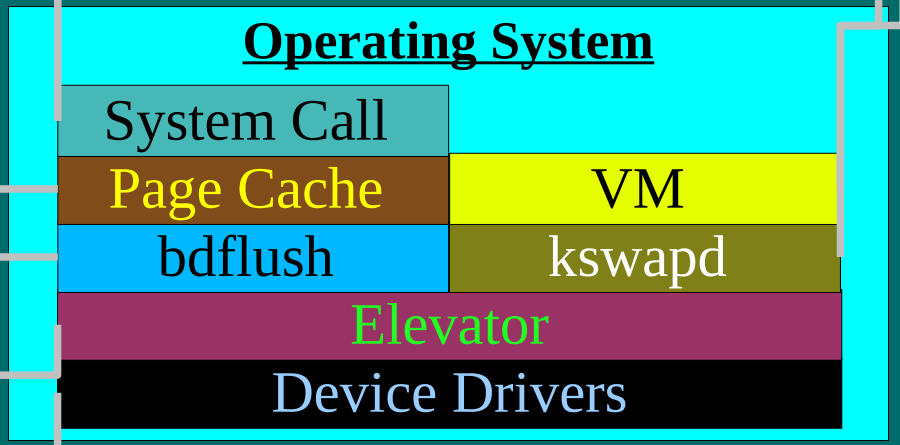
The I/O Stack

API used by user-space processes to interact with the kernel, in this case to modify or access to contents of the page cache.

Manages writing and allocating of virtual memory pages into and out of swap.

Provides the virtual memory subsystem required for multitasking many processes.

Stores disk pages in memory for improved performance. Modifications to pages flag them as “dirty”.



Periodically selects dirty (modified) pages from the page cache to be written to disk.

Submits I/O requests to the individual devices, determines the order in which the operations occur.

Processes the actual communication between the OS and the device. Several device drivers may actual be involved in a request. For example: PCI Bus, SCSI Controller, and SCSI disk.

The exact operational parameters of the various subsystems can be optimized for your environment, see: ftp://kalamazoolinux.org/pub/pdf/PerfTune2001.pdf

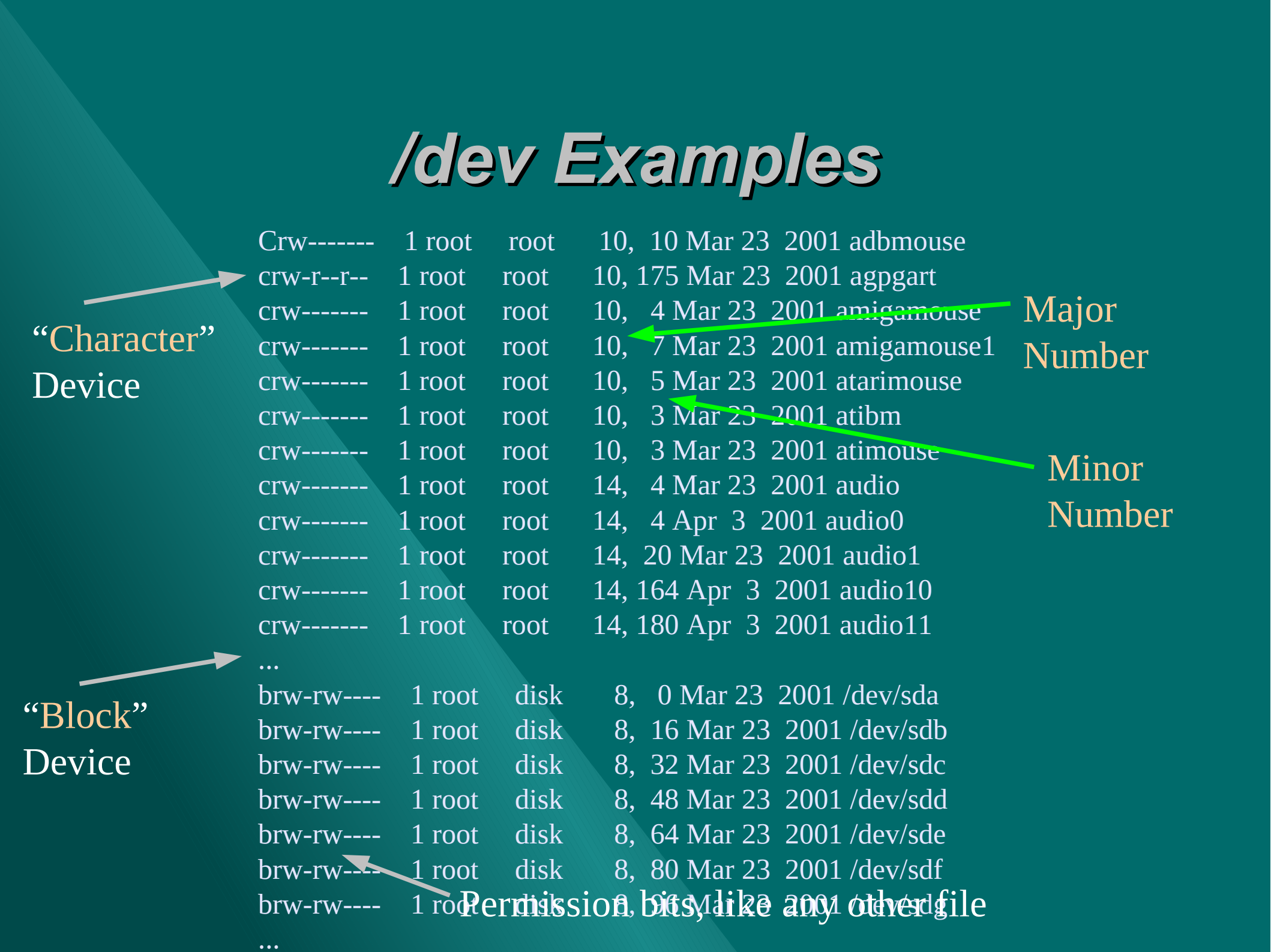
/dev

- According to the UN*X model, everything is a file*. This includes devices such as fixed disk, scanners, tape drives, CD-ROMs, Cameras, etc....
- The files that represent devices are located in or beneath /dev.
- Processes that need to access a device open the device file and perform operations on the file referred to as ioctl() calls.
- Each device file has a major and minor device number**.
  - The major number indicates a category of device, and in effect a specific driver.
    - Example: 8 = SCSI fixed disk, 3 = IDE fixed disk, 4= RS-232 port
  - The minor number indicates a specific device of the major number type.
    - Example: For major number 8 (SCSI Fixed Disk) a minor number of 0 means the first enumerated SCSI disk, and 16 means the second enumerated SCSI disk.
- On “older” systems /dev typically resides on disk in the / filesystem.
- On “newer” systems /dev is a virtual filesystem mounted at the /dev point and maintained dynamically by the kernel.
  - This allows easier support for hot-plug devices such as USB, Firewire, and LVD SCSI.

* Some devices under Linux are not represented by a file; network interfaces being the most noticeable example.

** There is some movement away from the major/minor number scheme due to the truly enormous number and range of devices which may be connected to a modern system.

/dev Examples




```
Crw-------   1 root     root      10,  10 Mar 23  2001 adbmouse
crw-r--r--   1 root     root      10, 175 Mar 23  2001 agpgart
crw-------   1 root     root      10,   4 Mar 23  2001 amigamouse
crw-------   1 root     root      10,   7 Mar 23  2001 amigamouse1
crw-------   1 root     root      10,   5 Mar 23  2001 atarimouse
crw-------   1 root     root      10,   3 Mar 23  2001 atibm
crw-------   1 root     root      10,   3 Mar 23  2001 atimouse
crw-------   1 root     root      14,   4 Mar 23  2001 audio
crw-------   1 root     root      14,   4 Apr  3  2001 audio0
crw-------   1 root     root      14,  20 Mar 23  2001 audio1
crw-------   1 root     root      14, 164 Apr  3  2001 audio10
crw-------   1 root     root      14, 180 Apr  3  2001 audio11
...
brw-rw----   1 root     disk       8,   0 Mar 23  2001 /dev/sda
brw-rw----   1 root     disk       8,  16 Mar 23  2001 /dev/sdb
brw-rw----   1 root     disk       8,  32 Mar 23  2001 /dev/sdc
brw-rw----   1 root     disk       8,  48 Mar 23  2001 /dev/sdd
brw-rw----   1 root     disk       8,  64 Mar 23  2001 /dev/sde
brw-rw----   1 root     disk       8,  80 Mar 23  2001 /dev/sdf
brw-rw----   1 root     disk       8,  96 Mar 23  2001 /dev/sdg
...
```

Device Enumeration

First name the four free peoples – Treebeard

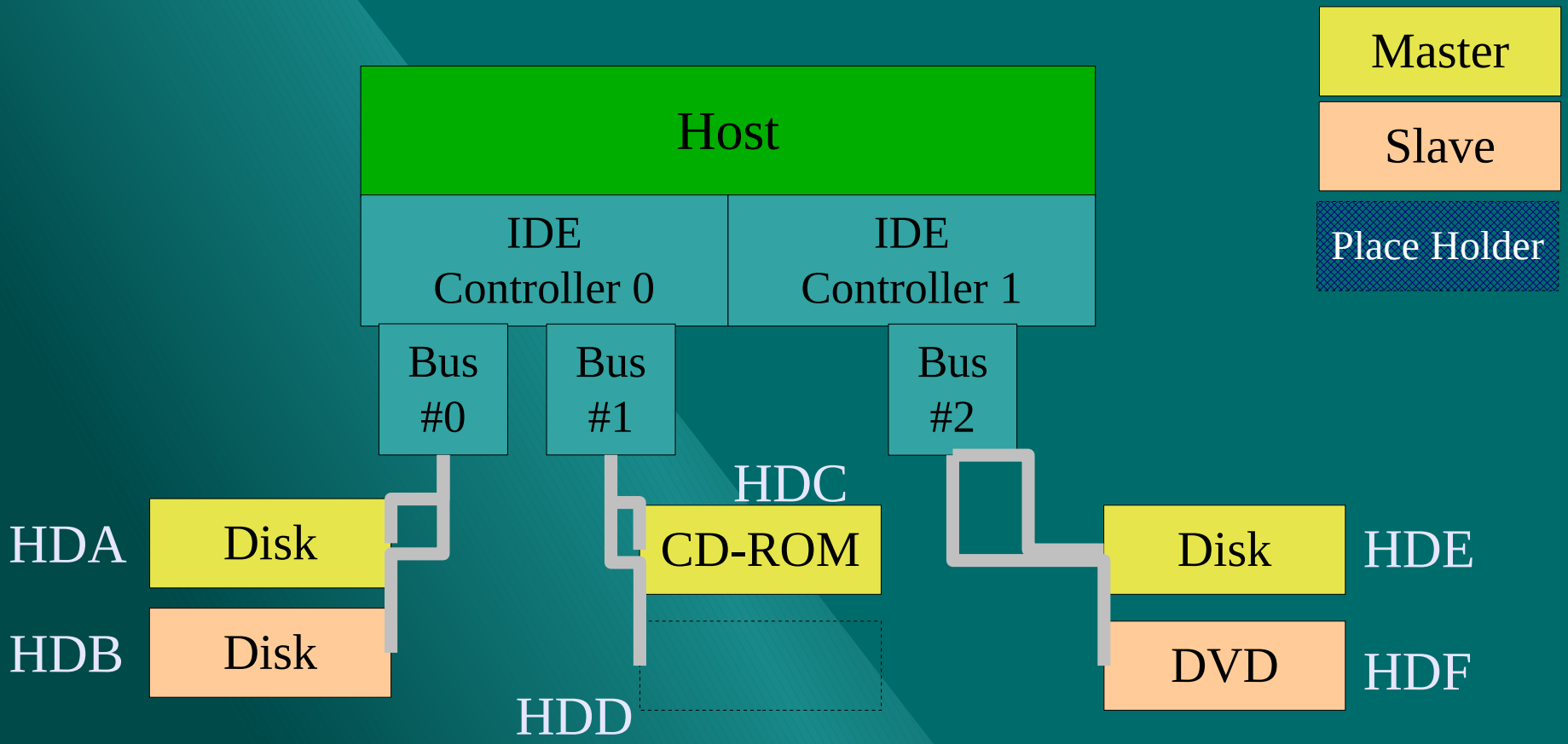
- All devices must be associated with a device file in order to be used by the system, this process is called enumeration. This can simply be thought of as the system naming each device it discovers.
- Disk devices are recognized when their controller is initialized.
  - Subsequent changes to hardware may not be recognized.
    - Only SCSI-3 SCA/LVD supports hot swapping.
- If required, additional device files can be created with the mknod command.
  - man mknod
- Disk devices do NOT contain data/filesystems and DO NOT correspond to a DOS C:/D:.

```
scsi0 : IBM PCI ServeRAID 4.80.26  <ServeRAID>
  Vendor: IBM       Model: SERVERAID        Rev: 1.00
  Type:   Direct-Access                     ANSI SCSI revision: 02
  Vendor: IBM       Model: SERVERAID        Rev: 1.00
  Type:   Direct-Access                     ANSI SCSI revision: 02
  Vendor: IBM       Model: SERVERAID        Rev: 1.00
  Type:   Processor                         ANSI SCSI revision: 02
Attached scsi disk sda at scsi0, channel 0, id 0, lun 0
Attached scsi disk sdb at scsi0, channel 0, id 1, lun 0
```

An example dmesg section where we can see devices being discovered and enumerated.

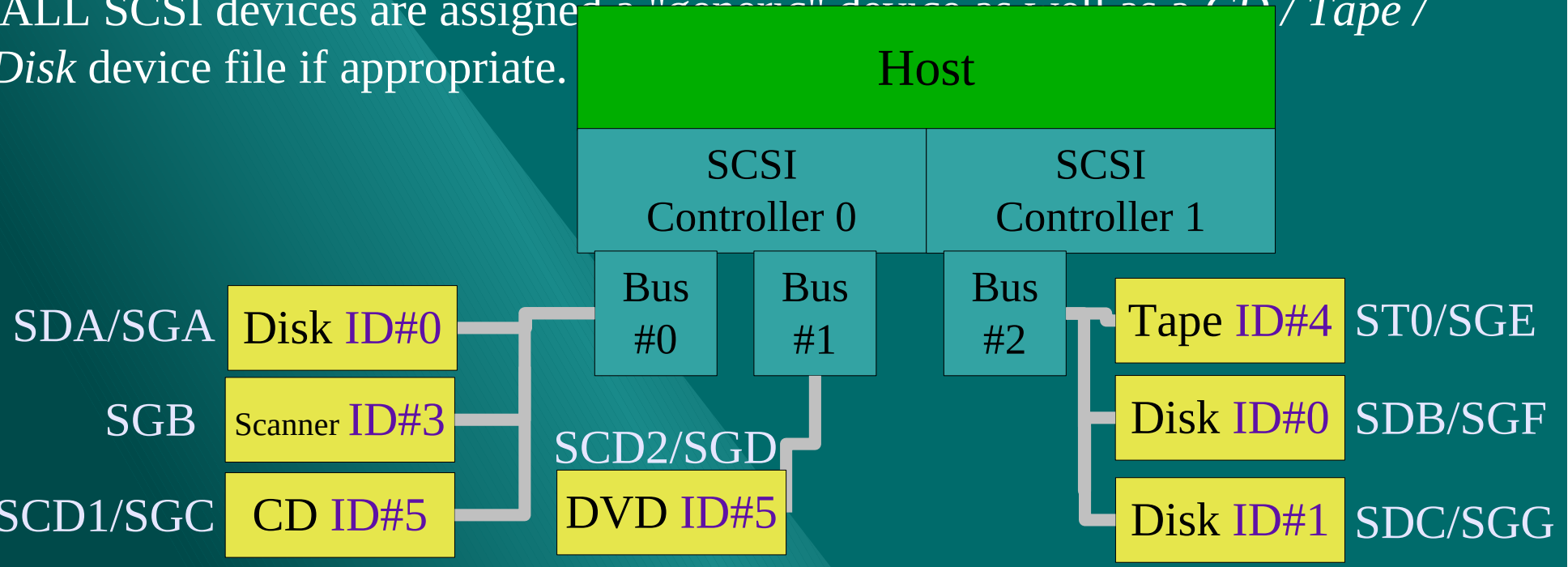
IDE Device Enumeration

- IDE disks are assigned device files in the order they are found:
  - hda, hdb, hdc, hdd, hde,.... (max default = 8)
  - IDE tape drives, CD-ROM drives, etc... fall into the same sequence.
    - Example: hda is a fixed disk, while hdb is a CD-ROM.



SCSI Device Enumeration

- SCSI disks are assigned device files in the order they are found.
  - sda, sdb, sdc, sde, sdf, ... (max default = 16)
  - SCSI tape drives, CD-ROMS drives, etc... have their own device file names and do NOT fall into the "sda?" sequence.
    - The SCSI CD-ROM sequence is scd1, scd2, scd3, etc... (max default = 8)
    - The SCSI tape drive sequence is st0, st1, st2, etc.... (max default = 8)
    - The SCSI "generic" sequence is sga, sgb, sgc, sgd, etc... (max default = 8)
  - ALL SCSI devices are assigned a "generic" device as well as a *CD / Tape / Disk* device file if appropriate.



Partitions

I hear younger voices call Far beyond your big stone wall -Josef Marais

- Partitions are enumerated per device: *{device}{partition number}*
  - The first partition on device sda (first SCSI disk) is /dev/sda1.
  - 1 - 4 are the four primary partition numbers defined in the partition table of the master boot record.
  - 5 + represent “extended” partitions.
  - All partitions, regardless of type, are enumerated.
    - This is different behavior than is exhibited by most operating systems.
  - IDE drives have a maximum of 16 partitions per device.
  - SCSI drives have a maximum of 15 partitions per device.

Common Partition Types

0	Empty	12	Compaq Diagnostic	63	GNU HURD	9f	BSD/OS
1	FAT 12 (MS-DOS)	14	Hidden FAT16 < 32Mb	64	Netware	a5	BSD/386
4	FAT16 < 32Mb	16	Hidden FAT16	65	Netware	a6	OpenBSD
5	Extended	17	Hidden HPFS / NTFS	81	Minix	b7	BSD fs
6	FAT16	1b	Hidden Win95 FAT32	82	Linux Swap	b8	BSD swap
7	HPFS / NTFS	1c	Hidden Win95 FAT32 (LBA)	83	Linux	da	Non-FS data
b	Win95 FAT32	1e	Hidden Win95 FAT16 (LBA)	85	Linux Extended	de	Dell Utility
c	Win95 FAT32 (LBA)	3c	Paritition Magic	86	NTFS Volume Set	eb	BeOS
e	Win95 FAT32 (LBA)	31	PPC Prep Boot	87	NTFS Volume Set	f2	DOS secondary
f	Win95 Extended (LBA)	52	CP/M	8e	Linux LVM	ef	Linux raid auto
11	Hidden FAT12	55	EZ-Drive	a0	IBM Thinkpad		

File-systems

Filesystems

A filesystem is simple a structure stored on disk (or other media) that facilitates the storage and organization of arbitrary blocks of data (files).

Filesystems	Journalized	Description
ext2	N	The standard Linux filesystem
ext3	Y	A journalized version of ext2
Highly stable		
Very slow on directories containing many (hundreds or thousands) of files		
Integreates with underlying RAID or MD systems		
rieser	M	A btree structued UNIX filesystem
Stable, but still under very active development		
Recent versions have had interoperability issues with NFS		
Very fast and efficient with small files and large directories		
fat??	N	The MS-DOS filesystem
iso9660	N	Read-Only filesystem used on Cds
streamfs	N	Multimedia oriented filesystem
Gaurantees no fragmentation, used for few very large files		
May be currently unmaintained		
xfs	Y	A UNIX filesystem ported from SGI IRIX
High performance filesystem with support for ACLs and very large files		
jfs	Y	A UNIX filesystem ported from IBM AIX
High performance filesystem, currently under development		

Making A File System

- A filesystem is created with the "mkfs {options} {partition-device}" command.
  - Common Options:
    - -t {filesystem type}
      - If no filesystem type is specified, an "ext2" filesystem is created.
    - -c Check device for bad blocks before creating filesystem.
    - -m %%% Percentage of blocks to reserve for use by the super-user only.
    - -b {block-size} Valid sizes are 1024, 2048, and 4096 bytes.
- Different filesystems have different options which can be given as parameters to the "mkfs" command. For specific filesystem options, do a "man mk{filesystem type}"
  - Example: man mke2fs, man mkreiserfs, man mkmsdos, mkswap

Common options specific to the ext2 filesystem:

-m %%% Percentage of blocks to reserve for use by the super-user only.

-b {block-size} Valid sizes are 1024, 2048, and 4096 bytes. For any filesystem larger than a few hundred megabytes, the block size should be 4096 bytes or performance will suffer. Smaller block sizes result in more efficient use of space.

-N {###} Number of inodes (directories and files each take an inode) to create in the filesystem. If not specified, the number of inodes is calculated based upon file-system size.

/etc/fstab

The filesystems "*known*" to the system are recorded in the file /etc/fstab.

A sample section of an /etc/fstab file

Device	Mount Point	Filesystem Type	Mount Options	Dump	fsck Pass Number
/dev/sda1	/	ext2	exec,dev,suid,rw	1	1
none	/proc	proc	exec,dev,suid,rw	1	1
none	/dev/pts	devpts	gid=5,mode=620	0	0
/dev/sda2	none	swap	pri=1,rw,sw	1	1
/dev/sdb1	none	swap	pri=1,rw,sw	1	1
/dev/sysvg0/usr	/usr	ext3	exec,suid,rw	1	2
/dev/sysvg0/var	/var	ext3	noexec,rw	1	2
/dev/sysvg0/tmp	/tmp	ext3	noexec,rw	1	2
/dev/mirror0/home	/home	xfs	rw	1	2
/dev/mirror0/ldap	/var/ldap	ext3	noexec,noatime,rw	1	2

Order in which filesystem should be checked at system start by the fsck utility.

There are some virtual filesystems (maintained by the kernel) that are mounted on all Linux systems.

Should the dump utility include this filesystem in the dump image of the system?

mount options

- When a filesystem is mounted (attached to the filesystem hierarchy) various options may be specified.
  - If a filesystem is defined in /etc/fstab, the mount options there are the default.
  - Additional or overriding mount options can be specified if the filesystem is mounted manually with the mount command.
  - Filesystem options can usually be adjusted on a live filesystem by re-mounting the file filesystem with the mount command.
    - remount is a mount option.
  - There are several generic mount options available on all filesystems.
    - Examples: exec, suid, noexec, ro
  - Filesystem types may have native mount options not available on other filesystem types.
    - Examples: noatime, data=journal, dev

Common Mount Options

	Generic Mount Options
dev	Enable device files functionality
nodev	Disable device file functionality
suid	Enable set-uid functionality
nosuid	Disable set-uid functionality
exec	Enable execution of programs
noexec	Disable execution of programs
user	Allow anyone to mount and unmount
	This option implies nosuid, nodev, and noexec. These defaults can be overriden by specifying their counterpart enable options.
nouser	Only super-user can mount and unmount
ro	Mount filesystem read-only
rw	Mount filesystem read-write
auto	Mount filesystem automatically
	This option is typically assumed unless noauto is specifically stated.
noauto	Do not mount filesystem automatically

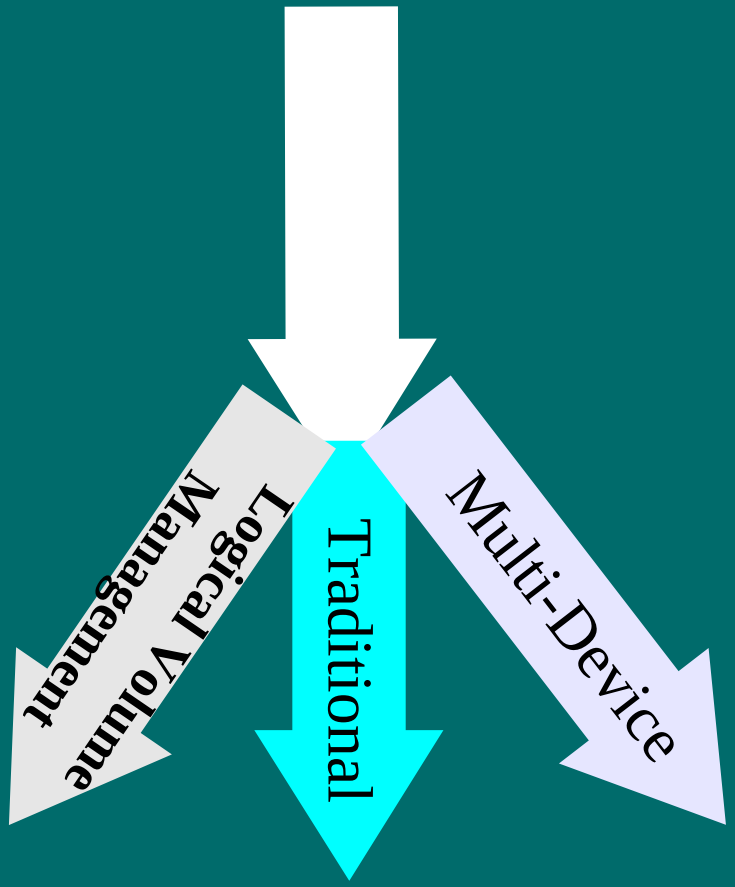
	XFS Mount Options
logbufs	Set number of in-memory log buffers (2-8)
logbsize	Set size of in-memory log buffers
	Valid sizes are 16,384 - 32768
logdev	Set the log device fo external logging
quota	Enable user quotas
grpquota	Enable group quotas

	Generic UNIX Mount Options
atime	Disable maintenance of last access times
uid	Set the user owner of files
gid	Set the group owner of files
mode	Set the mode of files

	ext3 Mount Options
data	How data is handled in regard to journalling
	A value of journal causes full data journaling, ordered causes data to be written before meta-data, and writeback (the default) writes meta data before data.
noload	Do not load journal on mount
	ext2 Mount Options
nouid32	Use 16 bit UIDs and GIDs
resgid	GID permitted to use reserved space
resuid	UID permitted to use reserved space
errors	Enable user quotas
	A value of continue indicates that errors should be noted, remount-ro indicates that the filesystem should automatically remount in read-only mode, panic will halt the system upon a filesystem error.
grpdid	?

All ext2 mount options also are available for ext3 filesystems.

Three Paths



Linux Storage Solutions

You know so many ways to be wicked – Tom Petty

Method	Kernel Level	Description
Traditional	1.0	One filesystems = One partition. There is a tight correlation between data storage and physical configuration.
MD (Multi-Device)	2.2	Software RAID. Physical partitions can be grouped to provide additional capacity, higher availability, or better performance.
LVM (Logical Volume Manaement)	2.4	Partitions are grouped into storage volumes. Storage capacity can be allocated dynamically and configuration changed on the fly.

Multi-Device

RAID Levels

RAID- Redundant Array of Independent Disks, is a technology used to improve availability and performance.

Level	Number Of Disks	Common Name	Availability Effect	Performance Effect On Read Operations	Performance Effect On Write Operations
linear	any	Appending	Additional Points Of Failure	None or Negligable	None or Negligable
0	even	Striping	Additional Points Of Failure	Improved, additional disk heads available.	Improved, additional disk heads available.
1	even	Mirroring	Withstands Single Point Failure	Improved, additional disk heads available.	Reduced, all operations must be performed twice.
5	minimum of 2, usually odd	Parity Array	Withstands Single Point Failure	None or Negligable	Reduced. Parity must be recalculated and stored for each operation.

!!!RAID-5 NOTES!!!

She's just a little bit dangerous – Roxette

ONLY USE RAID 5 ON DRIVES THAT PERFORM ERROR CHECKSUM:
SCSI and "some" ULTRA-DMA ATAPI drives.

RAID-5 IS NOT SAFE ON TRADITIONAL IDE DRIVES, AS SINGLE BIT ERRORS CAN GO UNNOTICED AND BE PROPOGATED INTO THE PARITY INFORMATION RESULTING IN CORRUPTED DATA UPON RE-CONSTRUCTION FROM THE INEVITABLE DRIVE FAILURE.

DRIVES OFTEN DO **NOT** DIE CATOSTROPHICALLY, BUT OVER TIME, ONLY MANIFESTING THE PROBLEM WHEN THE PROBLEM HAS REACHED A HIGH LEVEL OF SEVERITY.

- Good performance with software RAID-5 requires an MMX or "equivalent" enabled processor.
- Due to write characteristics software, RAID-5 is probably NOT suitable for RDBMS storage. In fact, hardware RAID-5 is probably not suitable either, unless accompanied by a **LARGE** **ECC** memory RAID controller cache.

Multi-Device Intro

The raidtools RPM add the following binaries to /sbin:

mkraid	Initialize an MD device defined in the config file.
raidstop	Stop an MD device, or all MD devices.
raidstart	Start an MD device, or all MD devices.
raid0run	Obsolete, do not use.
raidhotadd	Re-insert a previously failed partition into an array
raidhotremove	Remove a “good” partition from an array.

In order to perform a raidhotremove, operation requires the presence of a spare disk in the affected array.

Note: If any of the drives in an MD configuration are jumpered to Read-Only, it will crash the system.

The MD Config File
(/etc/raidtab)

```
raiddev /dev/md0
  raid-level      5
  nr-raid-disks       3
  nr-spare-disks      1
  chunk-size          32
  parity-algorithm          left-symmetric
  persistent-superblock   1
  device          /dev/sdf1
  raid-disk       0
  device          /dev/sde1
  raid-disk       1
  device          /dev/sdd1
  raid-disk       2
  device          /dev/sdc1
  spare-disk          0

raiddev /dev/md6
  raid-level          0
  nr-raid-disks       2
  nr-spare-disks      0
  chunk-size          4
  persistent-superblock   1
  device                /dev/sdb5
  raid-disk           0
```

MD device file to assign to this virtual device.

RAID level of this virtual device.

If RAID-5, the parity algorithm.

Partition device to include in virtual device.

Enumerate the devices to include.

Specify a drive as a "spare" disk.

Number of devices in this virtual device.

Number of "spare" devices in this virtual device, not included in "nr-raid-disks".

Size of block in virtual device.

chunk-size

if you want but I lose chunks with the bomb -DMX

chunk-size 32

A ^2 number between 4 and 128 that specifies the block size of the virtual device.

This setting makes little difference for RAID levels other than 0 and 5.

- Raid Level 0
  - A chunk size matching the "average" read/write operation is best, suggest 4.
  - A SCSI controller supporting "tagged command queue" will provide superior RAID 0 (sriping) performance.
- Raid Level 5
  - A large chunk size is recommended to ease the compute load of parity.

RAID 0 Performance Chart

Chunk	Block	Read	Write
4	1	19712	18035
4	4	34048	27061
8	1	19301	18091
8	4	33920	27118
16	1	19330	18179
16	2	28161	23682
16	4	33990	27229
32	1	19251	18194
32	4	34071	26976

RAID 5 Performance Chart

Chunk	Block	Read	Write
8	1	11190	6879
8	4	13474	12229
32	1	11442	8291
32	2	16089	10926
32	4	18724	12627

The configuration of the chunk size relative to the filesystem block size can have a significant effect on performance.

Making a RAID Array

Once you have a virtual device described in /etc/raidtab you simply initialize the device: mkraid /dev/md3

"mkraid" is a very smart utility. If a partition is in use, looks like it contains a filesystem, or another raid-device, it will warn you. And don't EVER ignore its warning.

You can immediately create a filesystem and copy data to the virtual device, any setup or regeneration is handled in the background and can be monitored by watching /proc/mdstat.

cat /proc/mdstat

If you create an ext2 filesystem on a RAID level 4 or 5 device you can use the ext2 "stride" option to increase performance:

mke2fs -b 4096 -R stride=8 /dev/md3

*With a chunk size of 32k, 8 4k blocks will fit in one block of the RAID device. This causes ext2 to perform writes as efficiently as possible. The effect of stride on RAID levels other than 4 or 5 has not been documented.

/proc/mdstat

The file /proc/mdstat contains all the information about the current MD config.

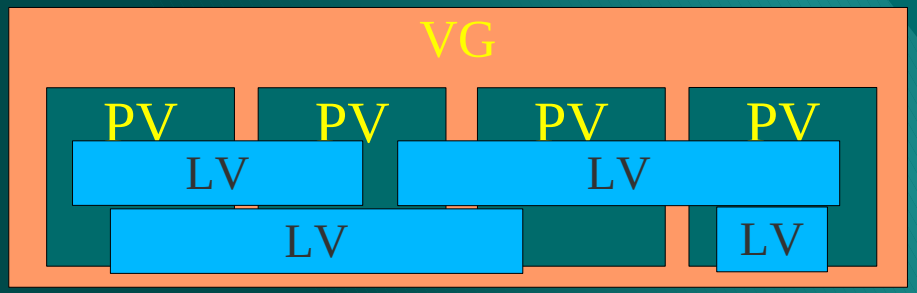
```
Personalities : [linear] [raid0] [raid1] [raid5]
read_ahead 1024 sectors
md7 : active raid0 sdb8[1] sda9[0] 3646464 blocks 4k chunks
md6 : active raid0 sda7[1] sdb5[0] 626304 blocks 4k chunks
md5 : active raid0 sdc6[3] sdd6[2] sde6[1] sdf6[0] 9358848 blocks 4k chunks
md4 : active raid0 sde5[1] sdd5[0] 3688192 blocks 16k chunks
md3 : active raid1 sdf3[1] sde3[0] 1024960 blocks [2/2] [UU]
md2 : active raid0 sdf2[1] sde2[0] 2049920 blocks 4k chunks
md1 : active raid1 sdd2[1] sdc2[0] 2048960 blocks [2/2] [UU] resync=1%
finish=18.7min
md0 : active raid5 sdc1[3] sdd1[2] sde1[1] sdf1[0] 5121792 blocks level 5, 32k
chunk,
algorithm 2 [3/3] [UUU]
unused devices: <none>
```

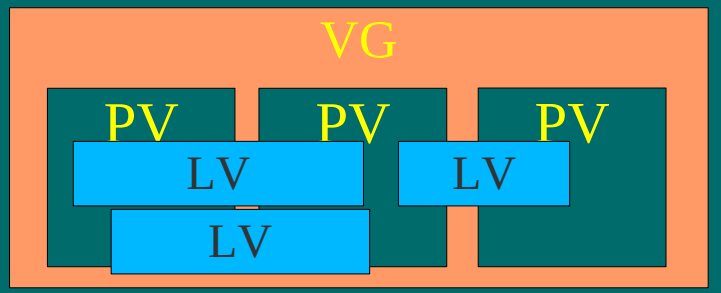
It is safe to shutdown the system during a "resync" (regeneration). It will simply

Logical Volume Management

LVM (The Future)

- With the 2.4.x kernel series comes the Linux Volume Manager (LVM).
  - *http://www.sistina.com*
  - LVM completely abstracts disk management. A disk can simply be partitioned into one large chunk, and LVM manages the usage of the storage capacity.
    - A partition is a "physical volume".
    - "Physical volumes" are grouped into "Volume groups".
    - Space is allocated from a "volume group" to a "logical volume" for use by a filesystem.
    - If supported by the filesystem (as it is with ext2), the logical volume can be resized on the fly.
    - Striping is simply a parameter passed to the volume manager when creating a logical volume.
      - RAID, at this point, is still handled by the Multi-Device (md) modules.
    - Logical volumes can be forcibly moved among physical volumes.





The LVM Fork

LVM1 - The default 2.4.*x* LVM subsystem. The original LVM code included functionality provided by later 2.4.*x* kernels. This results in duplication of effort, greater bug potential, and issues with maintainability. LVM1 is being maintained, but development effort is focused on.....

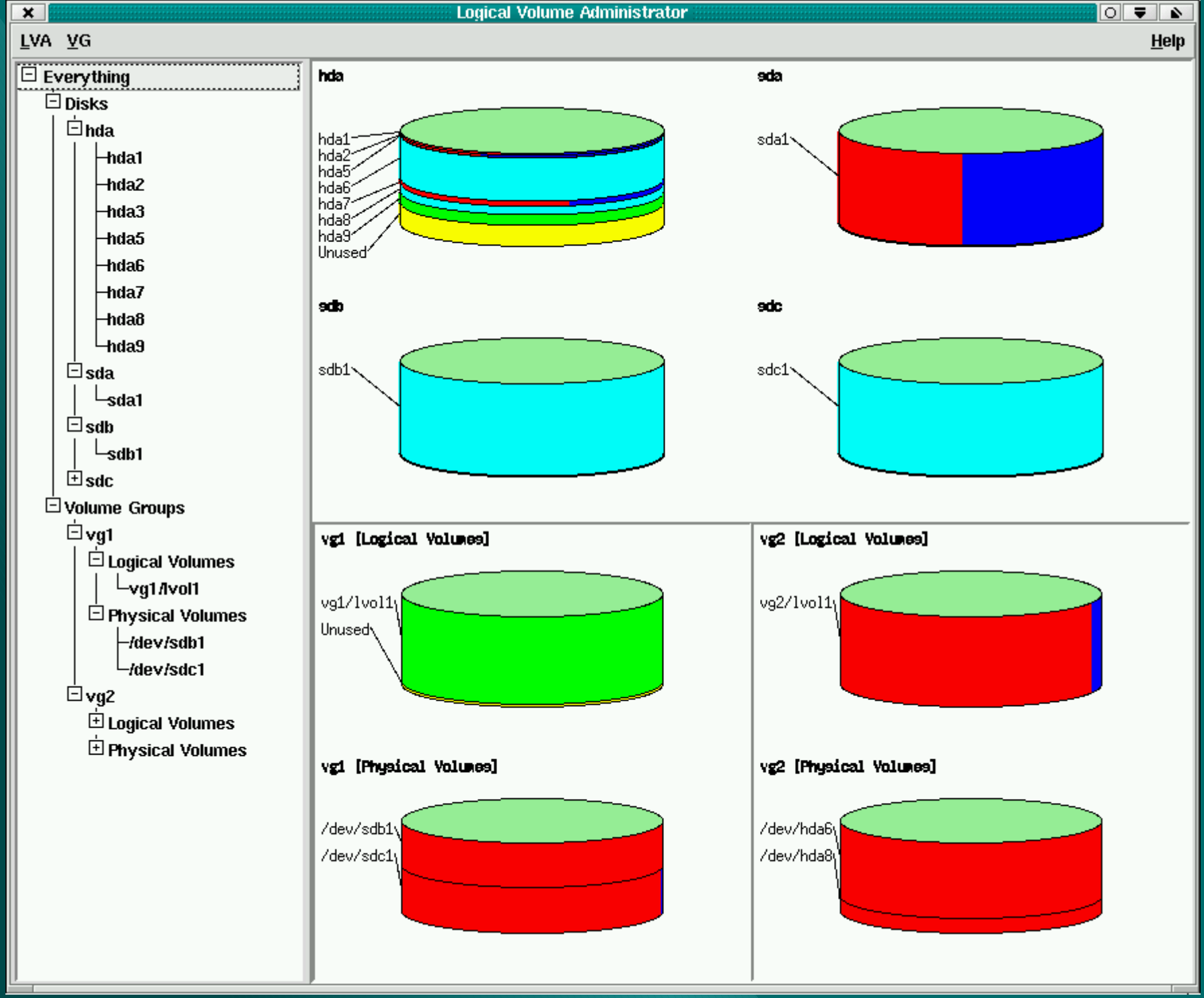
LVM2 - The new & shiny 2.4.*x* LVM subsystem. LVM2 relies on the facilities of the later 2.4.*x* kernels (most importantly device_mapper). By using native kernel services, LVM2 will be more robust and enable to developers to focus on features (such as *transaction disk support*) rather than re-tooling infrastructure.

LVM2 supports on the LVM1 meta-data format.

LVM Features

- Disks belonging to a volume group can be "exported" unattached from one machine, and then "imported" when attached to another machine, which then has access to all the logical volumes in that disk pack.
- Physical extents in a volume group can be moved from one physical volume to another physical volume without effecting the availability of the logical volume to which they belong.
- Volume groups can be "merged" and "split".
- "Instantaneous" copies of logical volumes, called snapshots, can be created.
- LVM has its own SAR(System Activity Recorded) to monitor the usage and performance of Logical Volumes.
- Logical Volumes can have "meaningful" names.
  - Such as "usr", "lib", "tmp", ... vs. "md0", "sda1", "hda5", ...
  - People argue if this is a feature or a short-sighted stupid idea.

Simply put, with LVM you don't store data on disks, you use the disks as a form of storage. (O*n disk data is no longer tightly associated with a disk partition.*)



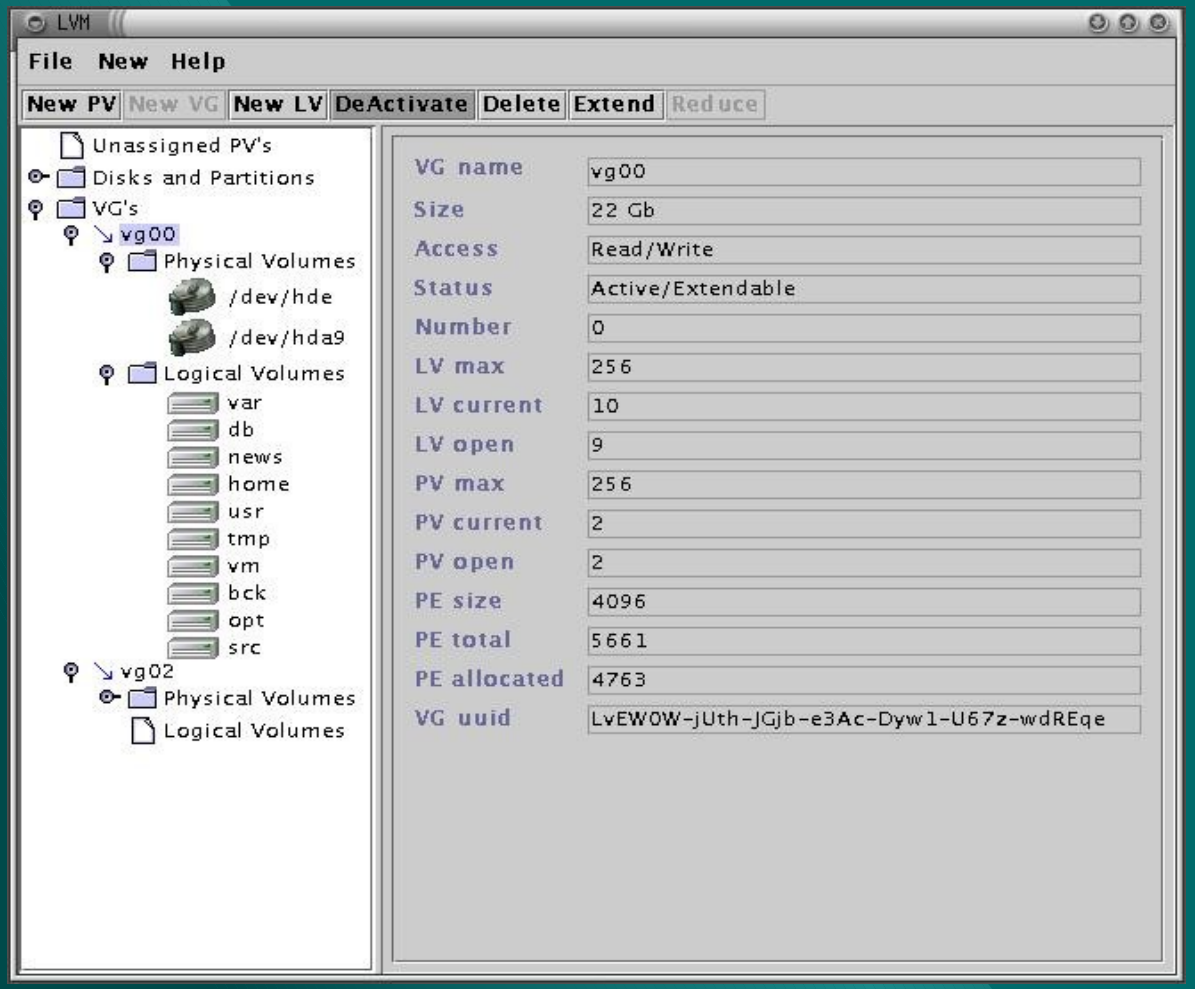
LVM Administrator

(http://www.widd.de/bernd/lva/)

This is a Perl/Tk application.

LVM Gui

(http://www.xs4all.nl/~mmj/lvm/)



This is a java application.

Setting Up LVM

- Mark a partition as a physical volume.
  - pvcreate /dev/sdf5
- List defines physical volumes.
  - pvscan
- Create a volume group to contain the physical volume.
  - vgcreate sysvg /dev/sdf5
- List defined volume groups.
  - vgdisplay -v
- Add the physical volume to the volume group.
  - vgextend sysvg /dev/sdf5
- Create a logical volume in the volume group.
  - lvcreate testvg -v --size 100M
  - Now /dev/testvg/lvol1 exists, and you can create a filesystem within it.
    - $ mkreiserfs /dev/testvg/lv01; $ mount /dev/testvg/lv01 /mnt/tmp

- vgscan;vgchange -ay must be run at boot time to load the LVM configuration.
  - This process creates files in /etc/lvmtab.d for each volume group discovered.
  - Volume groups know about themselves; they essentially have no "config" file

Snapshots

This world has only one sweet moment set aside for us - Queen

- An LVM snapshot is a "running patch" that allows you to freeze the state of a logical volume, at least for awhile.
  - lvcreate -L 100 -s --name snap /dev/vg00/lvol1
  - The above command will create a snapshot of the logical volume "lvol1" in the volume group "vg00", called "snap" until "lvol1" has changed more than 100Mb.
  - If a snapshot exists for a logical volume, any blocks written in that logical volume will first be copied to the snapshot.
    - A translation table is maintained for the snapshot. Once space in the snapshot for blocks from the original logical volume is consumed the snapshot is disabled.
  - This feature is primarily intended to ease the backup of filesystem data such as "/var/spool/mail" which changes constantly.
    - Simply create a snapshot and backup the snapshot while activity continues in the live logical volume.
    - Applications do not have to be taken down in order to back them up, or only taken down momentarily.

vgdisplay

I Can See Clearly Now The Rain is Gone – Johnny Nash

```
$ vgdisplay -v

--- Volume group ---
VG Name                sysvg
VG Access              read/write
VG Status              available/resizable
VG #                   0
MAX LV                 256
Cur LV                 1
Open LV                0
MAX LV Size            255.99 GB
Max PV                 256
Cur PV                 2
Act PV                 2
VG Size                3.52 GB
PE Size                4 MB
Total PE               900
Alloc PE / Size        25 / 100 MB
Free  PE / Size        875 / 3.42 GB

--- Logical volume ---
LV Name                /dev/sysvg/lvol1
VG Name                sysvg
LV Write Access        read/write
LV Status              available
LV #                   1
# open                 0
LV Size                100 MB
Current LE             25
Allocated LE           25
Allocation             next free
Read ahead sectors     120
Block device           58:0

--- Physical volumes ---
PV Name (#)            /dev/sdc5 (2)
PV Status              available / allocatable
Total PE / Free PE     450 / 450

PV Name (#)            /dev/sdf5 (1)
PV Status              available / allocatable
Total PE / Free PE     450 / 425
```

PE (Physical Extent) - The smallest allocatable unit of storage.

lvcreate options

Parameter	Explanation
-i *stripes*	Attempt to scatter the logical volume over the specified number of physical volumes.
-C*y|n*	Attempt to make the logical volume contiguous.
-I #	Stripe size in kilobytes, must be a power of two.
-L #	Size of the logical volume in megabyes.
-n *name*	Name of the logical volume within the volume group. Example: /dev/*volumegroup*/*name*
-s	Create a snapshot of another logical volume.
Snapshot Example:	lvcreate -L 100 -s --name snap /dev/vg00/lvol1
Standard Example:	lvcreate -i 3 -I 8 -L 100 vg00

```
/dev/sysvg0/usr       2015824  1857616    55808  98% /usr
/dev/sysvg0/var        806288    84584   680744  12% /var
/dev/sysvg0/tmp        806288   377324   388004  50% /tmp
/dev/mirror0/home     4031680  1891592  1935288  50% /home
/dev/mirror0/ldap      604736      364   573652   1% /var/ldap
/dev/mirror0/sql      1209472    16832  1131200   2% /var/lib/pgsql
/dev/mirror0/mail      757920     9908   709512   2% /var/spool/mail
```

A sample of df output from an LVM based configuration.