CSCI 2461, Computer Networking 3 - Linux

Week 04 (Rev. 0)

Wednesday, January 31, 2018

Intro

Linux Wipes Windows in Seconds

Figure 1: Linux Wipes Windows in Seconds

Due Homework

How was your experience with Chapter 3, Devices?

- HLW Ch 3 Devices, put command and operations into scripts
  - Using Linux Command.org "Writing Shell Scripts" as a reference
- Readings:
  - How Cache Works - So, what are page tables?,
  - Git Handbook,
  - Linux Filesystem Hierarchy Standard,
  - Linux Standard Base
- Familiarize yourself with moving your script submissions from D2L to Git
  - GitHub Cheat Sheet

Opening Lab

Git Setup

We're going to open up class today with Git and getting your personal repositories setup, and then you'll follow your classmates repositories.

After that, you'll submit your Week 3 scripts and check out your peers repository and review their scripts.

- Launch your Debian VM
- Launch Terminal in the Debian VM, you should see a *$* prompt
- Open GitHub.com in a browser and Login
- Open Guides.GitHub.com "Be Social"
- Find your peers on GitHub and follow them.
  - Post your username to D2L Discussions if you haven't already.

Clone your repo

Now that you're following your peers, it's time to push out something for them to comment on.

- In your GitHub.com logged in browser
  - Create a new repository
  - Select "Initialize with a README",
  - Apache License 2.0, MIT or BSD recommended
- Open your newly created repository
- Click "Clone or Download", then "Copy to Clipboard"
- Paste into the terminal a command similar to:
  - `git clone https://github.com/exampleuser/repo-name.git`

cd repo; nano README; git {add *, commit, push}

- Now *c*hange *d*irectory to the repo-name and *l*i*s*t the contents
- You should see a README file, let's edit it with nano.
  - If you decide to use vi/vim do the OpenVIM tutorial
- Next, copy your Chapter 3 script (ch3.sh) to the folder you just created.
- Add your script and the README
  - `git add *`
- commit your changes
  - `git commit -m "README update and Chapter 3 script"`
- push your changes
  - `git push`

Devices (HLW Ch 3)

How Linux Works, Chapter 3 (Device Files, pg 46)

- Open Debian Linux Devices
- `echo blah blah > /dev/null`
- `cd /dev; ls -l`

How Linux Works, Chapter 3 (sysfs device path)

- `blkid; lsblk`
- `find /sys | grep sd | less`
- `ls -l /dev /dev/mapper | grep '^b'`
- `ls /proc/partitions`

How Linux Works, Chapter 3 (dd)

- `cd ~; pwd; dd if=/dev/zero of=./empty.file bs=1M count=1`
- `hexdump empty.file`

How Linux Works, Chapter 3 (Device Names)

- `dmesg | grep sd`
- Read TLDP HowTo Partition

Disks & Filesystems (HLW Ch 4)

Linux Disk Schematic (pg 66)

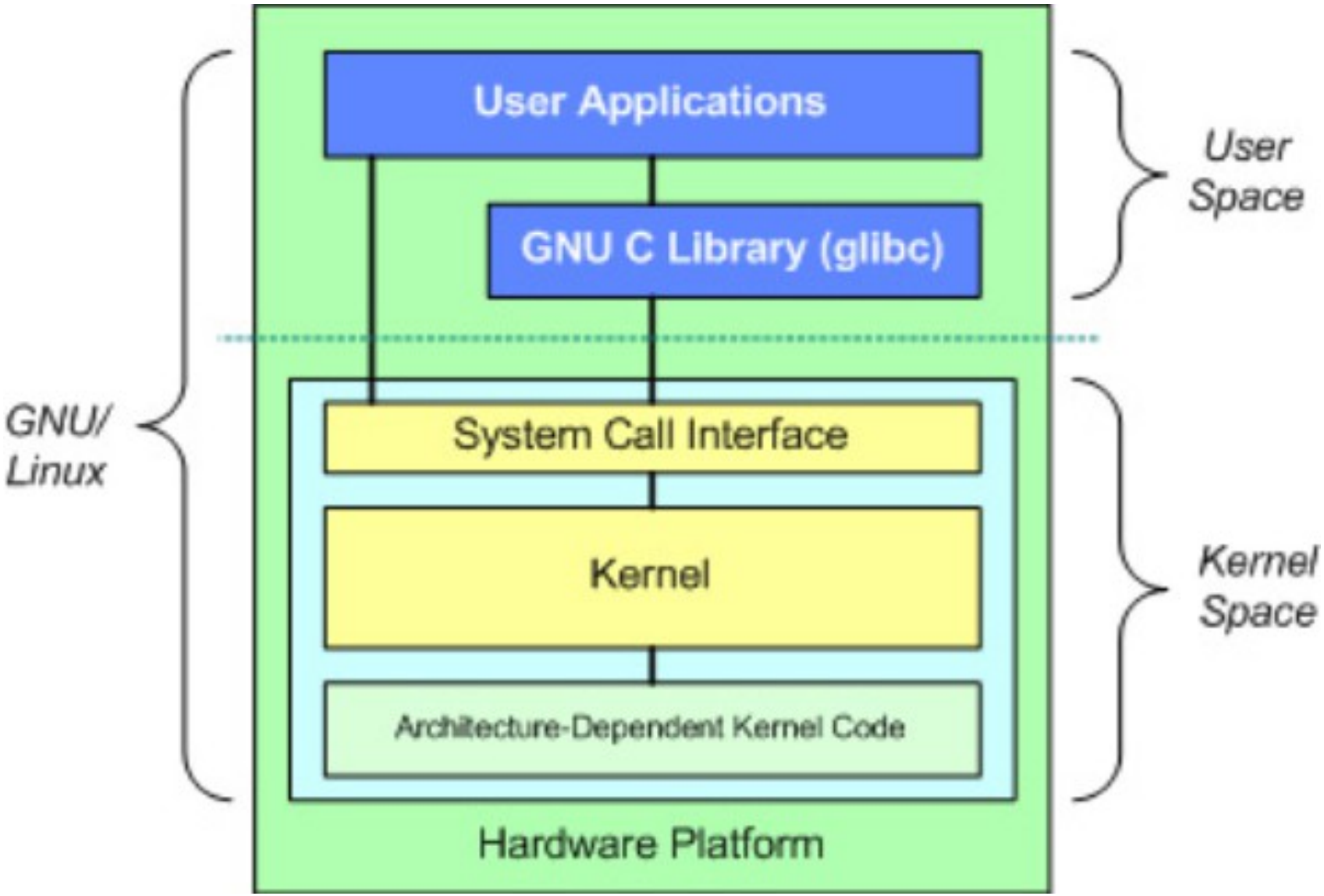
- Partition Table
- `man udevadm`
- Individual Partitions
  - `fdisk -l /dev/sda`

```
[mjh@mascon 2461]$ sudo fdisk -l /dev/sda
Disk /dev/sda: 465.8 GiB, 500107862016 bytes, 976773168 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 0662F73A-9362-41F4-95D7-0D231F672E78

Device       Start       End   Sectors   Size Type
/dev/sda1     2048    411647    409600   200M EFI System
/dev/sda2   411648   1460223   1048576   512M Linux filesystem
/dev/sda3  1460224 976773134 975312911 465.1G Solaris root
```

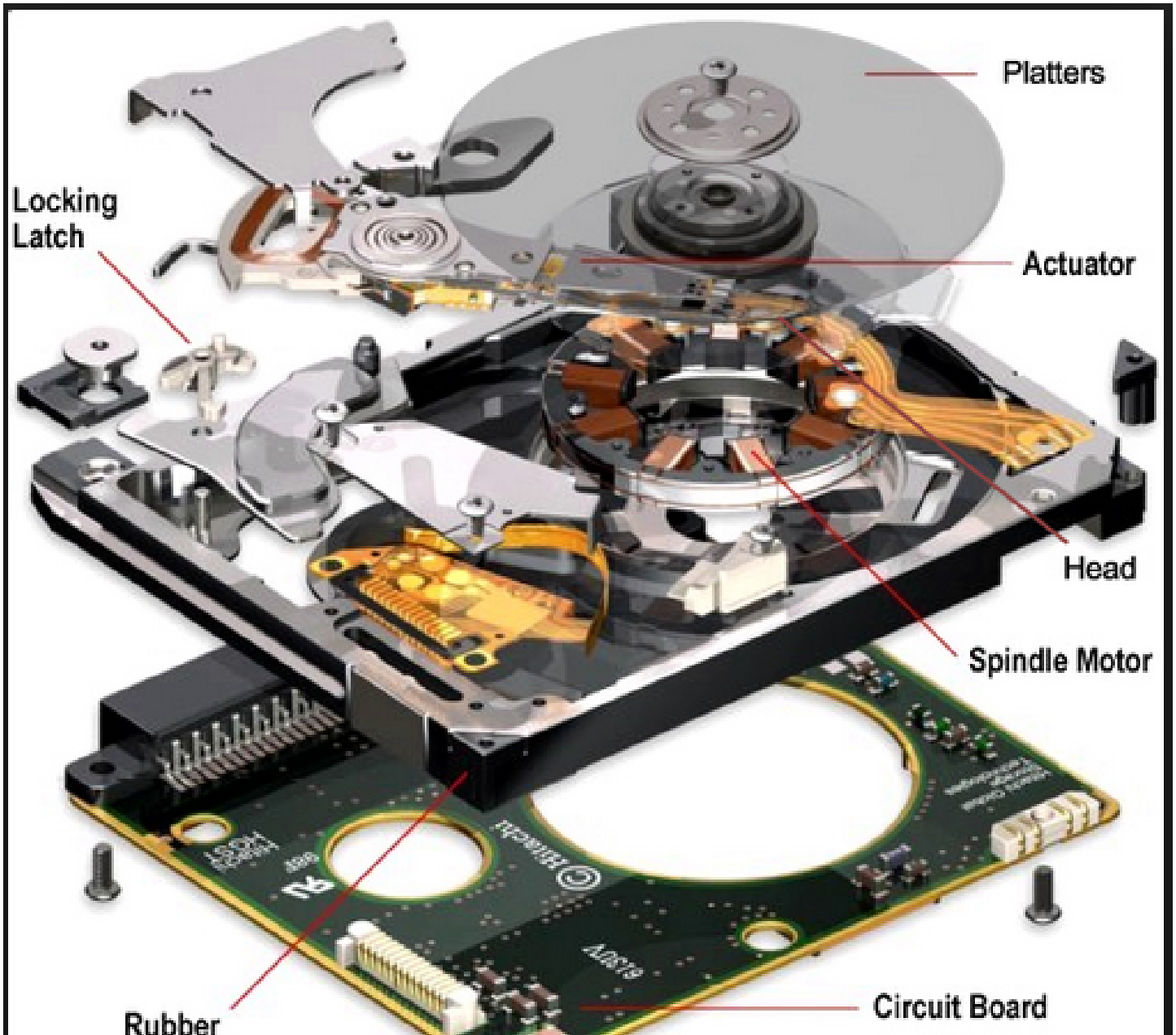
Figure 2: fdisk -l /dev/sda

Kernel Schematic for Disk Access (pg 67)



IBM Developer Works Properties of the Linux Kernel

Disks (Spinning)



Disks (Spinning: Components)

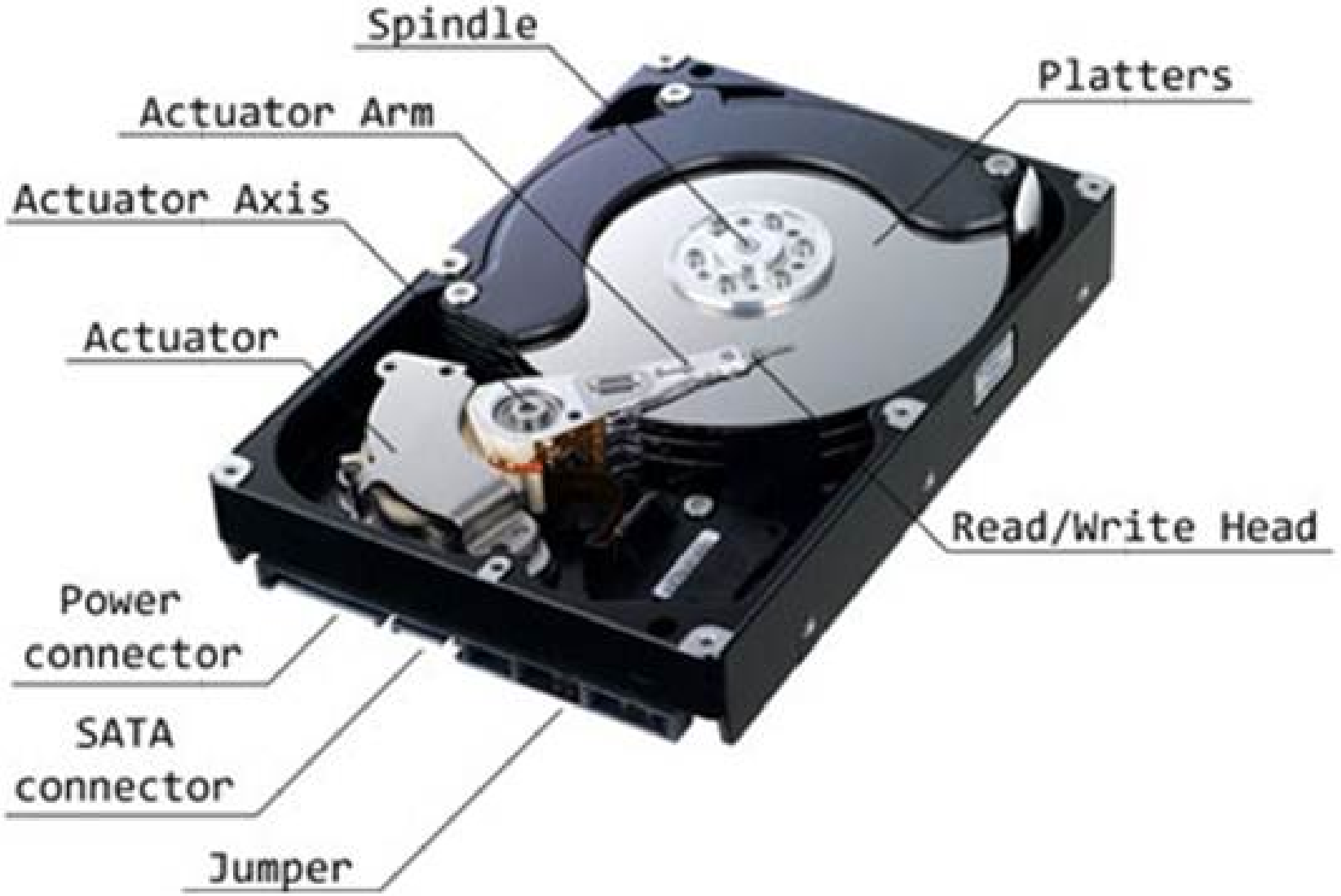
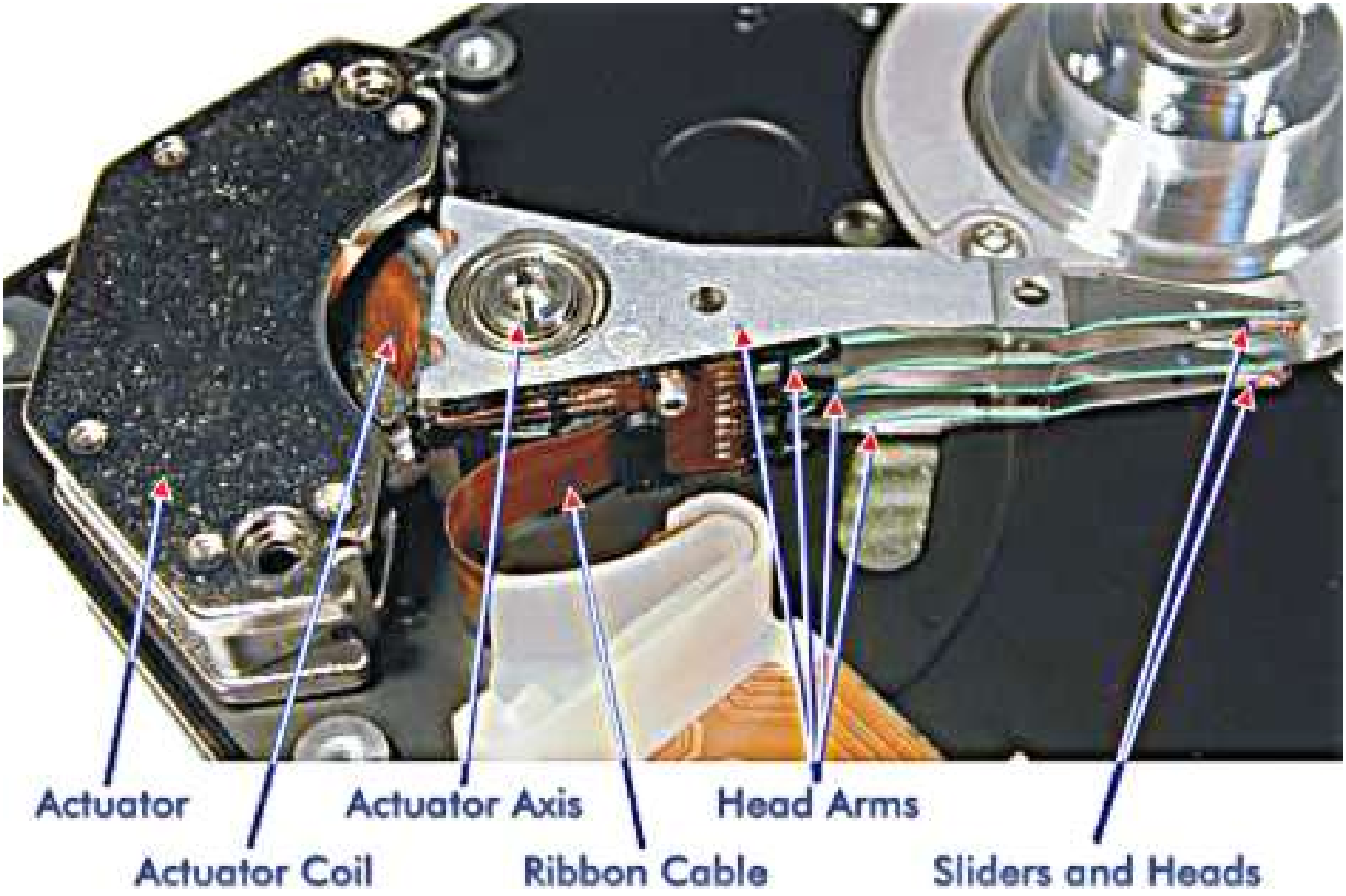


Figure 4: Hard Disk Components

Disks (Spinning: Zoom In)

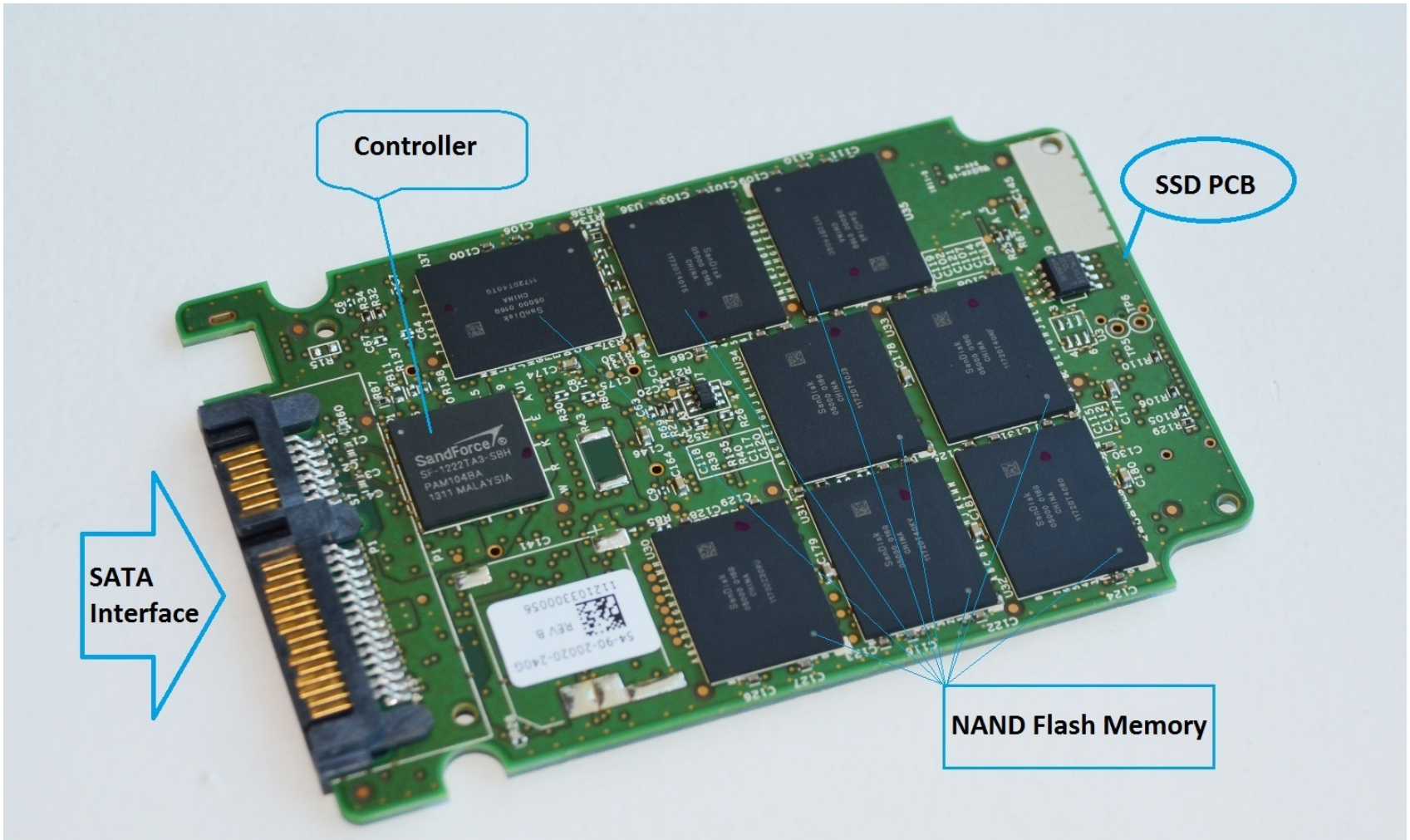


Disks (HDD vs SSD)

A quick comparison of an HDD (spinning magnetic disk) vs a SSD (solid state disk)

Disks (SSD: Chips on a printed circuit board)

Here we see a solid state disk, chips on a PCB. This is very similar to what is in your USB Flash Drive.



Disks (HDD & SSD Performance Considerations)

Most existing filesystems were designed with spinning hard disks in mind, and on the enterprise side file servers with large amounts of RAM. As a result, file systems wrote data wherever it could based on where the disk was in its rotation.

With solid state, data is typically read in 4096-byte chunks and need to align the beginning of the partition with one of those chunks so only one chunk needs to be read, not two. This is a significant performance consideration when writing data to "USB flash drives" or SSDs.

Filesystems & Types (pg 72)

Filesystems exist to allow users a structured way to interact with the kernel to access the hardware.

Windows primarily uses file systems like (ex)FAT and NTFS

Linux general purpose filesystem is called the "extended filesystem" (ext2-4) and has four major versions currently all of which are mostly compatible between each other. Additionally, Linux supports the ISO 9660 CD-ROM standard and Apple's HFS+

Filesystem Creation (make a local image)

Let's start with making some local partitions in files before working on hardware.

Create a 32 MB file with the *dd* command:

- `dd if=/dev/zero of=./32MB.img bs=1M count=32`

Explained:

- `if=/dev/zero` reads /dev/zero as an input file
- `of=./32MB.img` writes to 32MB.img in the current directory (./)
- `bs=1M` means to write with a 1 MegaByte block size
- `count=32` means to write 32 blocks.

Filesystem Creation (look at the image)

Now, let's look at what we just created and then put a filesystem onto it

- `hexdump ./32MB.img | less`

What is in there?

Filesystem Creation (make a filesystem)

Now, write the ext4 filesystem using the mkfs command:

- `mkfs -t ext4 ./32MB.img`

Explained:

- `mkfs` is the "make filesystem" series of commands
- `-t ext4` is the flag to specify the ext4 type of filesystem
- `./32MB.img` is a reference to your file that you just created with *dd*

There are many other filesystems and command shortcuts:

- `ls -l /sbin/mkfs.*`

Filesystem Mounting (mount our image)

We now have a filesystem in an image. Make a mount point, and mount it.

- `mkdir /mnt/tmp`
- `sudo mount ./32MB.img /mnt/tmp`
- `mount; df -h`
- `lsblk`

Now let's put a little "Hello World" file in there

- `cd /tmp/tmp; ls`
- `echo "Hello World" > ./hello.txt`

Now unmount the image

- `sudo umount /mnt/tmp`

Filesystem Storage (look for hello.txt)

Let's look for our file again with hexdump, this time with the –canonical option which will show us hex and the ASCII characters we wrote.

Make sure to pipe it to *less* so you can page through the output.

- `hexdump --canonical ./32MB.img | less`

Use "/" within less to search, e.g. type /ello and / again to go to the next result.

Special Filesystems

- /dev (devfs)
- /proc (proc filesystem)
- /sys (sysfs)
- /run (tmpfs)

Swap Sapce

Swap Space is like Windows pagefiles, it is disk space allocated as RAM type memory.

- `free`
- `dd if=/dev/zero of=./16MB.swap bs=1M count=16`
- `mkswap ./16MB.swap`
- `sudo chmod 0600 16MB.swap; sudo chown root 16MB.swap`
- `swapon ./16MB.swap`
- `swapon -s`

Inside a Filesystem (page 87)

Inode table connects to the data pool (of inodes)

Make some directories and files within those directories and make a *link*.

- `mkdir dir_1 dir_2`
- `echo "a" > dir_1/file_1`
- `echo "b" > dir_1/file_2`
- `echo "c" > dir_1/file_3`
- `echo "d" > dir_2/file_4`
- `ln dir_1/file_3 dir_2/file_5`

Inside a Filesystem (inodes)

Now take a look at the inode IDs

- `ls -iR dir_*`

Compare the far left column of dir_1/file_3 and dir_2/file_5

Lab

Ending Lab

Use the rest of the class time to:

- Ask questions.
- Start your homework.
- Finish your Chapter 4 shell scripts and push to your git repo.
- Choose your Debian/Linux distribution for for Chapter 5
- Talk about your script, talk to a peer about their script.

Homework

Reading

- Read “How Linux Works” Chapter 4 “Disks & Filesystems”, make applicable scripts for all commands.
- Review “How Linux Works” Chapter 5 “Linux Kernel Booting”
- Read, review, and run peer scripts from GitHub in your VM.
- Catch up on any other readings you haven’t done yet. :-)

Lab

- Setup a bootable Debian based image on your USB drive, use *persistence*.
- Setup git on your bootable Debian drive.
- git clone and comment on peers scripts

Ciao!