Linux Basics
Everyone installed Linux on their computer

Everyone launched the command line ("terminal") and ran a few commands

What problems were encountered?

- Virtualization support in processor not enabled (BIOS)
- VMWare Player (current version) only runs on Windows 64
- 3D graphics virtualization incompatible with specific hardware
- Old virtual machine software
- Others?

Tip: If you have problems maximizing your VM to full screen, or doing copy-and-paste between Linux and Windows, make sure you installed the VM tools
Person of the Day: Linus Torvalds

- Creator of **Linux Kernel**
  - Started in 1991
  - First developer – hobby project (for fun!)
  - Modern kernel is product of work by thousands of programmers
  - Currently “final authority” on what is included in the kernel

- Creator of **Git version control system**
  - Initially for Linux kernel dev
What does the OS need to do?
- Schedule processes to run
- Memory management
- Interrupt handling (manage hardware in general)
- Security (between processes)
- Network access
- Storage management (filesystem)
- Graphical user interface
  - May be a middleware layer on top of the OS
Process management is a key operating system task

OS must initially create processes when you run your program

OS can allow processes to access resources
  Must schedule access to shared resources (e.g., CPU)

OS can allow processes to communicate with each other

OS must clean up after process finishes
  Deallocate resources (e.g. memory, network sockets, file descriptors, etc...) that were created during process execution
The operating system schedules process execution

- What processes are allowed to run at all?
- What processes are allowed to run right now?

Context switches occur when the CPU is taken from one process and given to another process

- CPU state (registers, current PC, etc...) is preserved during a context switch.
Preemptive Scheduling

- Each process is allocated a timeslice.
- When the timeslice expires, a context switch occurs
- A context switch can also occur when a higher-priority process needs the CPU
Process A is forbidden from reading/modifying/writing the memory of Process B
- **Virtual memory** is a huge help here!
- Each process has a separate *virtual* address space that maps to different regions of *physical* memory

Process A has other limits besides which memory pages it can access
- **What are some other limits?**
- Amount of memory consumed
- Number of open files on disk
- Which files on disk can be read/written
OS is responsible for managing data on persistent storage

Job of the **filesystem**!
- What files exist? (i.e. names)
- How are they organized? (i.e. paths/folders)
- Who owns and can access them? (i.e. usernames, permissions)
- Where are individual file blocks stored on the disk?
  - *i.e. filename “database.dat” is really composed of 15823 blocks, of which block 1 is located at logical block address #... on the hard drive.*
Manage devices
- How do we send data to the NIC for transmission?
- How do we render an image for display on screen?
- How do we read a block of data from our RAID disk controller?

Operating systems can be extended through device drivers to manage new hardware
- Hardware vendors write software to manage their devices
- OS provides a fixed interface (API) that driver must follow

Common task for a device driver is responding to interrupts (from that device)
Who does all this essential work in the operating system? (besides the GUI)
- The kernel (i.e. the heart or core of the OS)

Kernel performs:
- Scheduling
- Synchronization
- Memory management
- Interrupt handling
- Security and protection
Operating systems with **graphical user interfaces** (GUI) were first brought to market in the 1980s.

Apple Mac OS 1.0 (released 1984)

Microsoft Windows 1.0 (released 1986)

Captures from [http://www.guidebookgallery.org/screenshots](http://www.guidebookgallery.org/screenshots)
Significant evolution in GUI design in subsequent decades
Technical perspective:

- The GUI is one of the least important parts of the operating system

- A GUI does not even have to be part of the true OS at all
  - Windows 1.0 was just a program that ran on top of MS-DOS, the true operating system (of that era)

- But to a user, the GUI is one of the most important parts of the OS!
Advantages of Command Line

Advantages of Windows / GUI
What is the shell? (e.g. BASH, CSH, SH)

- Program between user and the kernel
- Command-line interpreter
  - Parses user input and carries out commands
Shell Shortcuts

- <TAB> key to auto-complete commands
- <UP ARROW> key to cycle through previous commands

These two tips make your life so much easier!
Linux: Sudo Command

- `sudo <<command>>`
- Command is run as root user
- `root = “Administrator”`

http://xkcd.com/149/
What is a package manager?
- Where did these apps come from?

```
apt-get <<mode>> <<options>>
```

```
apt-get install gedit
```
- Mode = install a package
- Option = Gedit (name of package)

**Must run as ROOT to use!**
- `sudo apt-get ...`
Linux: Directory Tree

- **Absolute path:**
  - `/home/hpotter/thesis/intro.txt`

- **Relative path:**
  - If I am already in `/home/potter/
  - `addresses.html`

http://osl.iu.edu/~pgottsch/swc2/lec/shell01.html
Labs have (at most) two graded elements:

1. **Pre-Lab “checkpoint”** – quick verification that pre-lab *appears* to be done
   1. Due at start of first day of lab

2. **Lab Report**
   1. Submit all source code used with lab report
   2. Due by posted date after lab
Lab Reports

- Not really “reports”, more like “worksheets”
- Create in LibreOffice (aka *OpenOffice*) using example template on website
- Export in **PDF format**
- Submit
  - Via Canvas *Assignments* section for Lab 1 only!
  - Via Version control for Lab 2 and beyond
Upcoming Schedule

Today
- Lab 1 – Linux Basics

Thursday
- Lab 2 – Version Control

Deadlines
- Lab 2 pre-lab checkpoint – Start of class Thursday
- Lab 1 Report – Jan 28th, 2017 by 5am
  - Submit via Canvas
- Lab 2 Report – Jan 30th, 2017 by 5am