



# Why Linux?

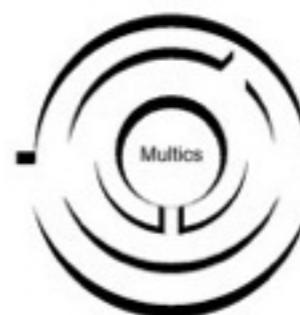
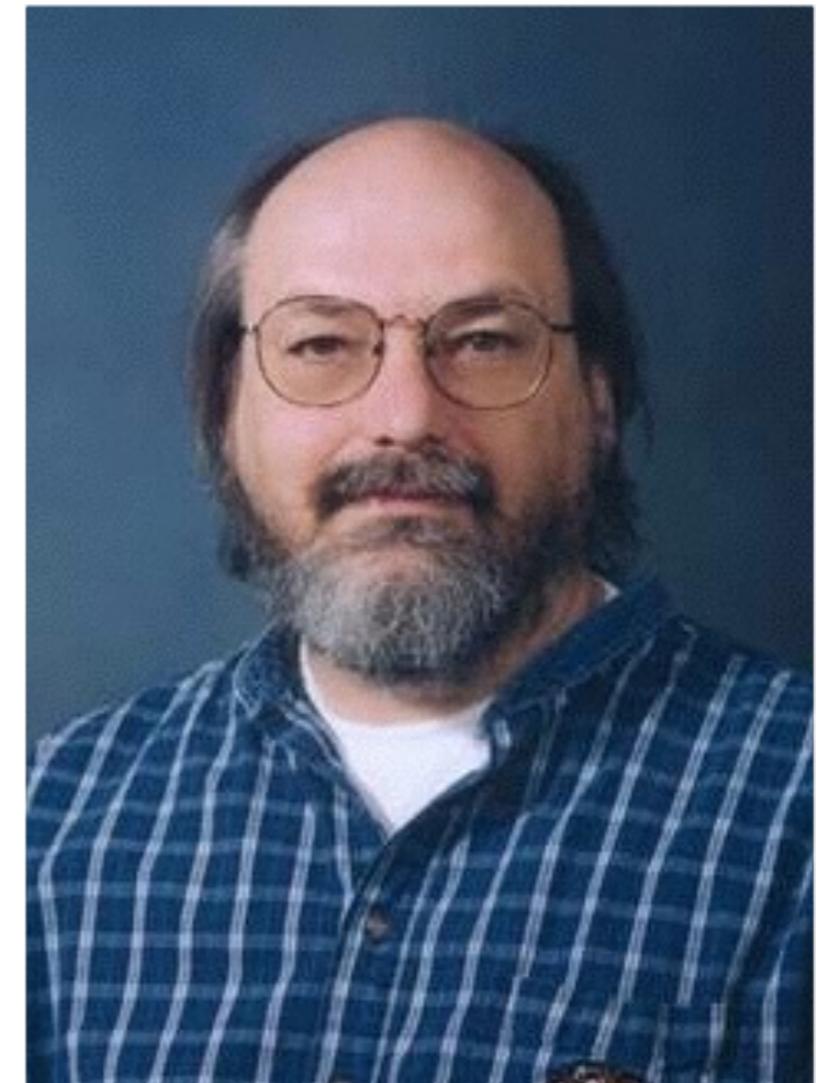
Dr. Gareth Roy (x6439)  
[gareth.roy@glasgow.ac.uk](mailto:gareth.roy@glasgow.ac.uk)

- A brief history lesson
- Why Linux?
- The parts of an Operating System
- The Command Line

- A brief history lesson
- Why Linux?
- The parts of an Operating System
- The Command Line

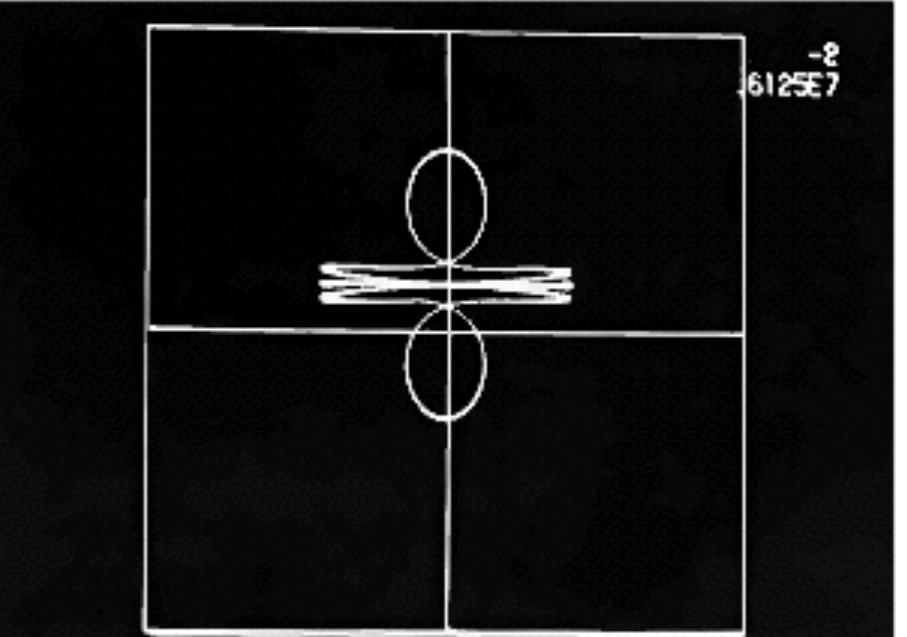
# Kenneth Lane Thomson (1943 - )

- Received his Bachelor's and Master's degrees in E&EE from the University of California at Berkley
- In 1966 was hired by Bell Labs (owned by AT&T) to work on the MULTICS project.
- MULTICS (**M**ultiplexed **I**nformation and **C**omputing **S**ervice) was a mainframe time sharing operating system.
- Worked on developing the OS until Bell Labs withdrew from the consortium in 1969.



# Space Travel

- While working on MULTICS, Ken created a game called Space Travel.
- It allowed the player to travel around a 2D solar system.
- Originally intend for MULTICS, Ken re-wrote it to run on an obsolete PDP-7 located in the lab.
- Ken wanted a better system to run his game on so in the Summer of 1969 he took one month to write:
  - Kernel
  - Shell
  - Editor
  - Assembler



# Birth of UNIX

- Bell Labs left the MULTICS consortium in 1969.
- Bell Labs purchased new PDP-11, and development continued on what would become UNIX.
- In 1970 Ken created the ‘B’ programming language which was the precursor to C.
- Joined by Dennis Ritchie in 1972, Ken re-wrote the UNIX kernel in C.
- UNIX was presented to the world in 1973.



Ken Thomson and Dennis Ritchie working on the PDP-11 used to develop UNIX

# What is UNIX

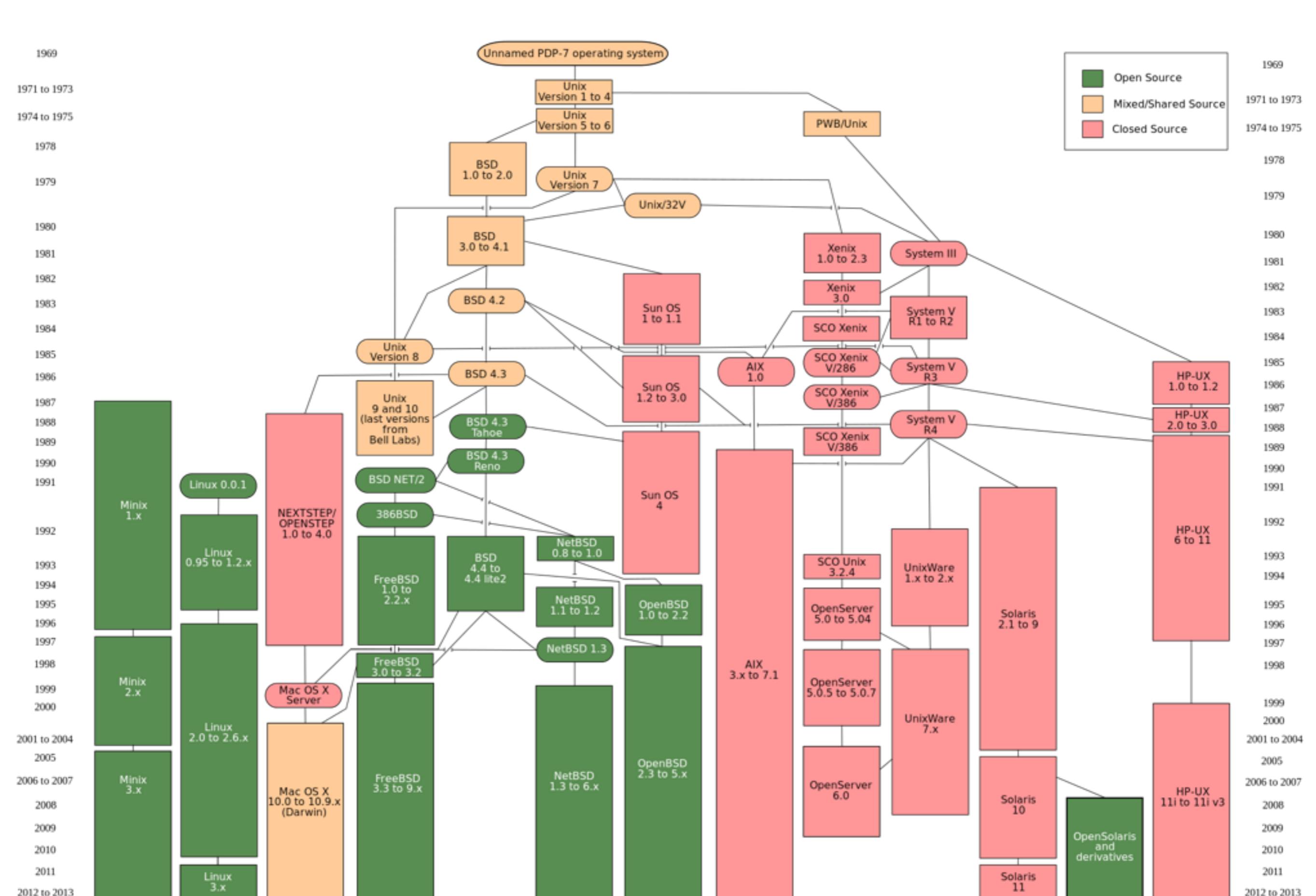
- UNIX is a Multi-user, Multi-tasking operating system (at present a family of Operating Systems).
- Any UNIX system has some key features:
  - the use of plain text for storing data
  - a hierarchical file system
  - devices and processes represented as files
  - the use of a large number of software tools, small programs that can be composed as opposed to using a single monolithic program
- In 2015 to be called a UNIX requires certification by the OpenGroup

A screenshot of a terminal window titled "2. bash". The window shows a command-line interface with the following text:

```
Last login: Tue Jan 13 13:14:55 on ttys001
Folkvangr:~ gareth$ pwd
/Users/gareth
Folkvangr:~ gareth$ ls
Applications Documents Installers Music Public
Backups Downloads Library Personal University
Desktop Dropbox Movies Pictures
Folkvangr:~ gareth$
```

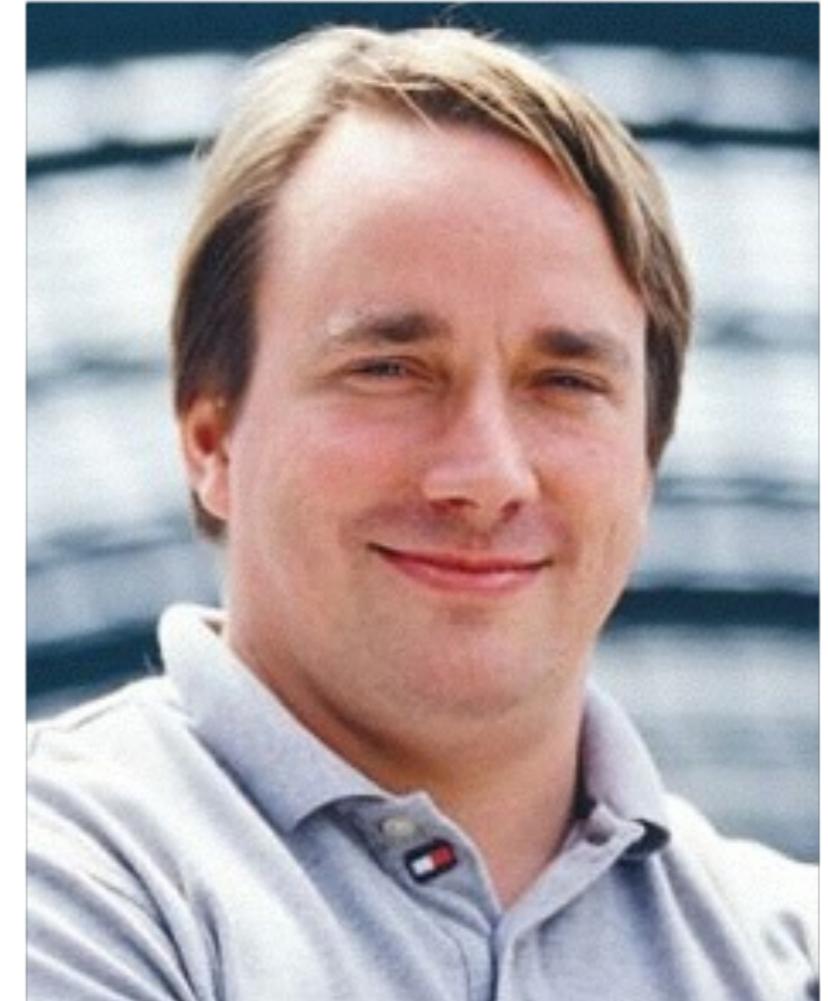
The window has a dark background and light-colored text, with standard Mac OS X window controls at the top.

<http://www.unix.org/version4/>



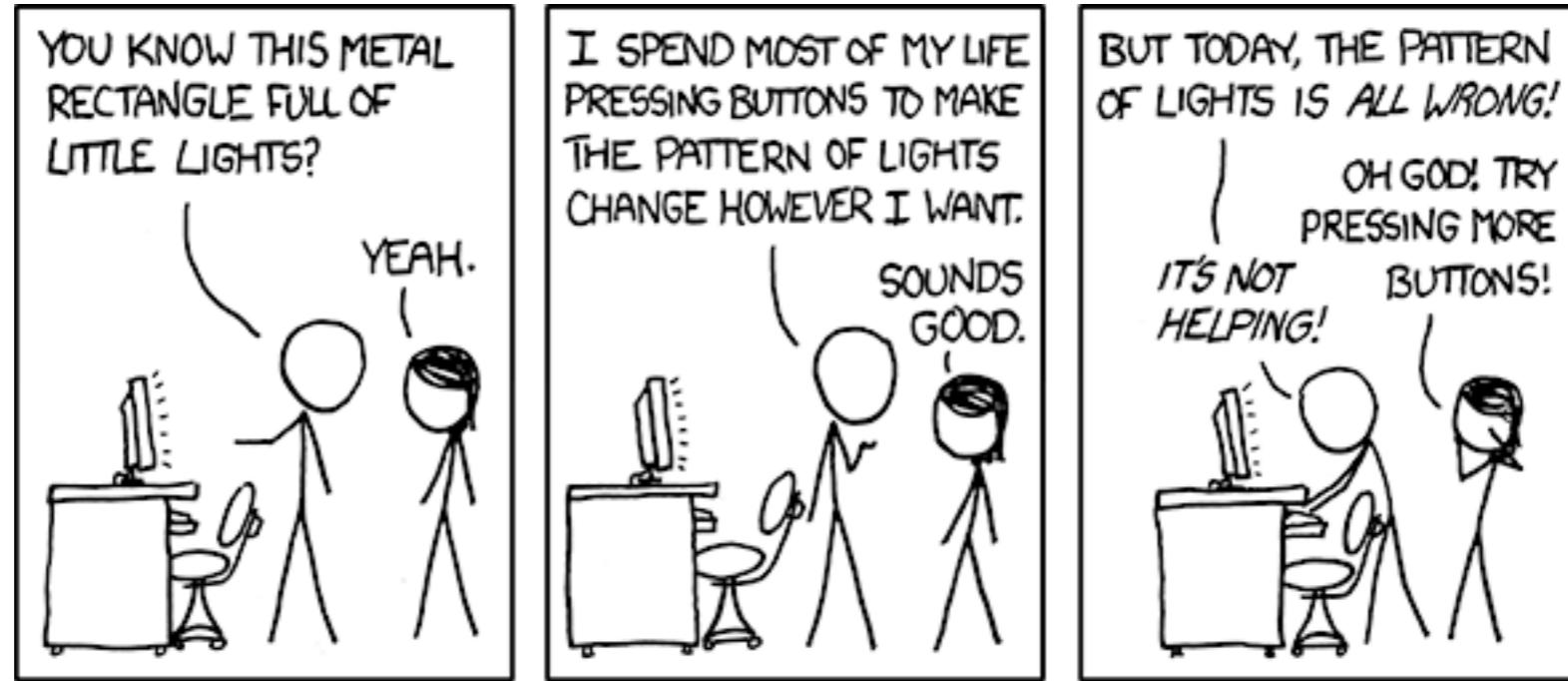
# Linus Benedict Torvalds (1969 - )

- After becoming frustrated with the License associated with Minix, Linus began work on his own operating system.
- In 1991, Linux v.0.01 was released on comp.os.minix
- Originally intended to be called Freax, renamed by a co-worker (Ari Lemmke) when it was placed on the server.
- Linux was only the OS Kernel itself and so a complete Linux OS heavily leveraged the GNU utilities and tools.
- This later led to Linux being released under a GPL license becoming open source.



*"I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones."*

- A brief history lesson
- Why Linux?
- The parts of an Operating System
- The Command Line



xkcd.com

# Why Linux?

- Linux is a free, with no cost required to obtain a complete fully featured OS.
- Linux source code is freely available and can be modified to fit user needs.
- Linux is a multi-user, multi-tasking operating system based on the tested principles of UNIX.
- Linux is ubiquitous in scientific computing with 485 of the top 500 supercomputers in the world running Linux.
- Scientific code bases for Cosmology, Particle Physics, Nuclear, Atmospheric, Fluid Dynamics all run on Linux platforms.

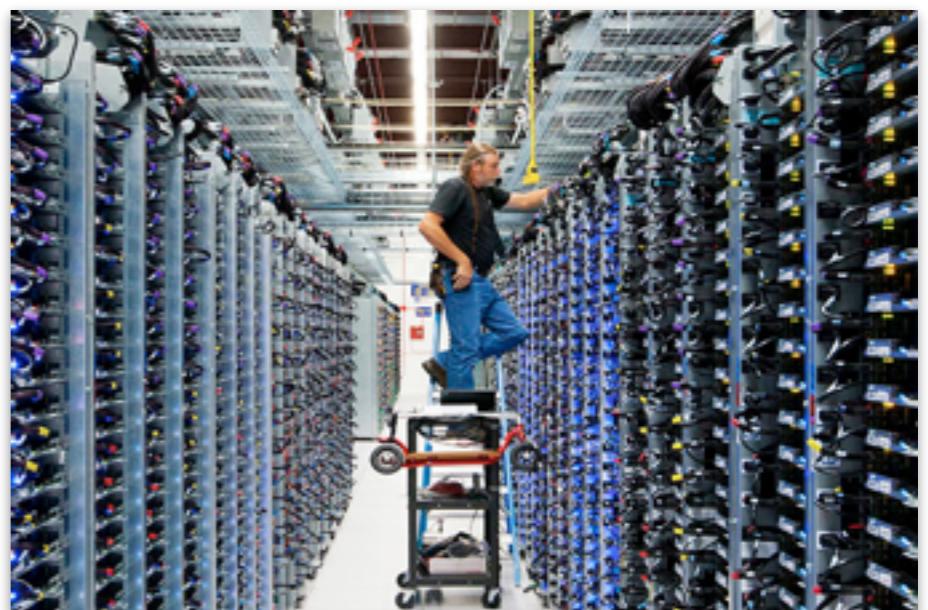
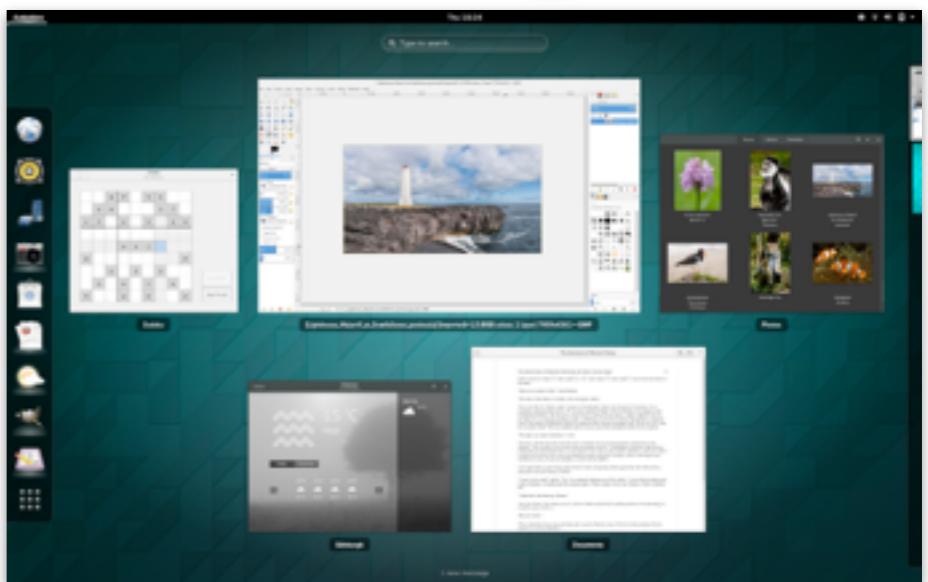
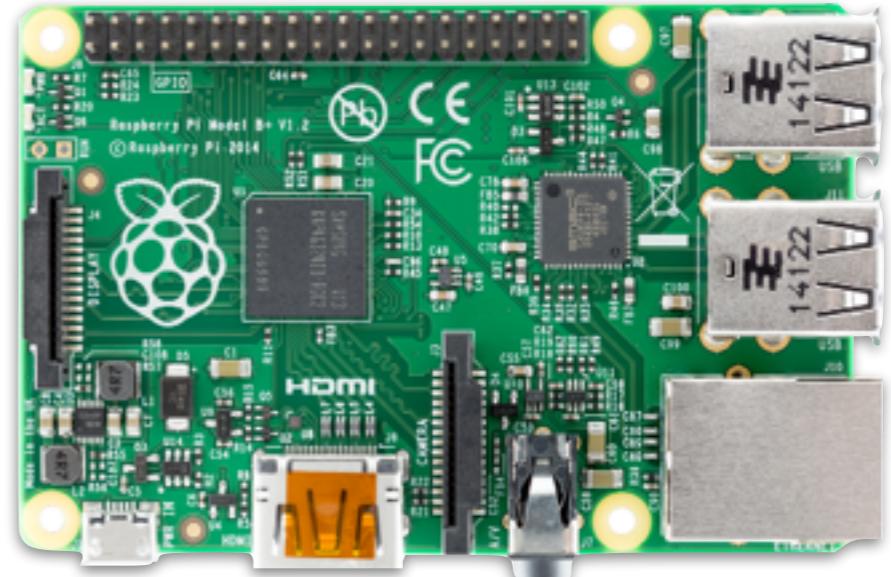


Tianhe-2 (MilkyWay-2)

- 3,120,000 CPU Cores
- 1,024,000 GB of RAM
- 33,862.7 TFlop/s
- 17,808 kW

# Why Linux?

- Due to it's open nature Linux has been ported to many platforms and architectures.
- It scales from embedded devices such as the Raspberry PI to powering companies such as Google and Facebook.
- Linux is on your phone (Android), it's on your router (dd-wrt), it's on your TV (lots), in your car (BMW), it runs the city of Munich, and your data is already stored there if you use Facebook/Google/DropBox etc...
- It runs much of the worlds internet services with, until recently, most deployments running LAMP.



# Why NOT Linux?

- Rough around the edges, oft times documentation is poor and no single source for information.
- Driver support can be weak for new hardware, and particularly for hardware that has closed data sheets (WiFi).
- Many applications target other platforms, so no Photoshop, MS Office or Assassin's Creed.
- Many problems need to be resolved on the command line rather than through more intuitive wizards.



# So I've convinced you, how do you get it?

- Because Linux is the kernel you need other pieces of software to make an OS.
- Since most/all of it is open source many people have taken Linux and created different variants, called Distributions.
- There are two main core Distributions - Debian and Fedora (created by RedHat).
- Many other distros exists but in general many of these are “Forks” of either Debian or Fedora.

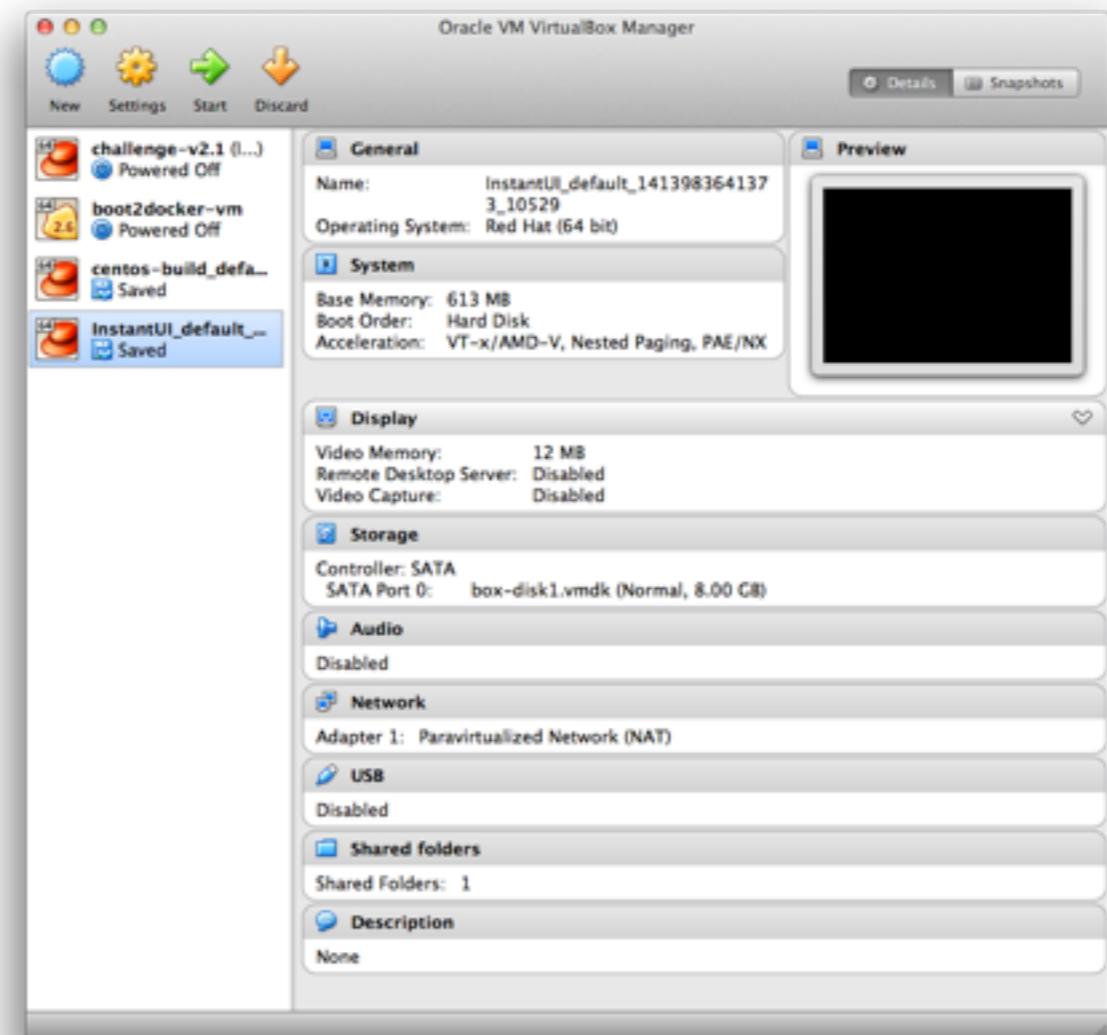


A screenshot of the DistroWatch.com website. The header includes the site's name, a search bar, and links for various languages. The main content area features news headlines, a sidebar with distribution news, and a 'Latest Distributions' section listing recent releases like OpenSUSE 13.2.1 and Mint 17.1. The footer contains links to ZK Software products and a 'Get The Usertab' button.

[distrowatch.com](http://distrowatch.com)

# Installing Linux in a VM

- Simplest way to check out Linux.
- VirtualBox is free and available for most platforms (Linux, Windows and OS X).
- Download an ISO of your favourite Distribution and point VirtualBox at it.
- Be sure to give the VM enough memory and enable graphic acceleration for best performance.
- You can also install “guest additions” to allow you to mount directories from your host to the VM.

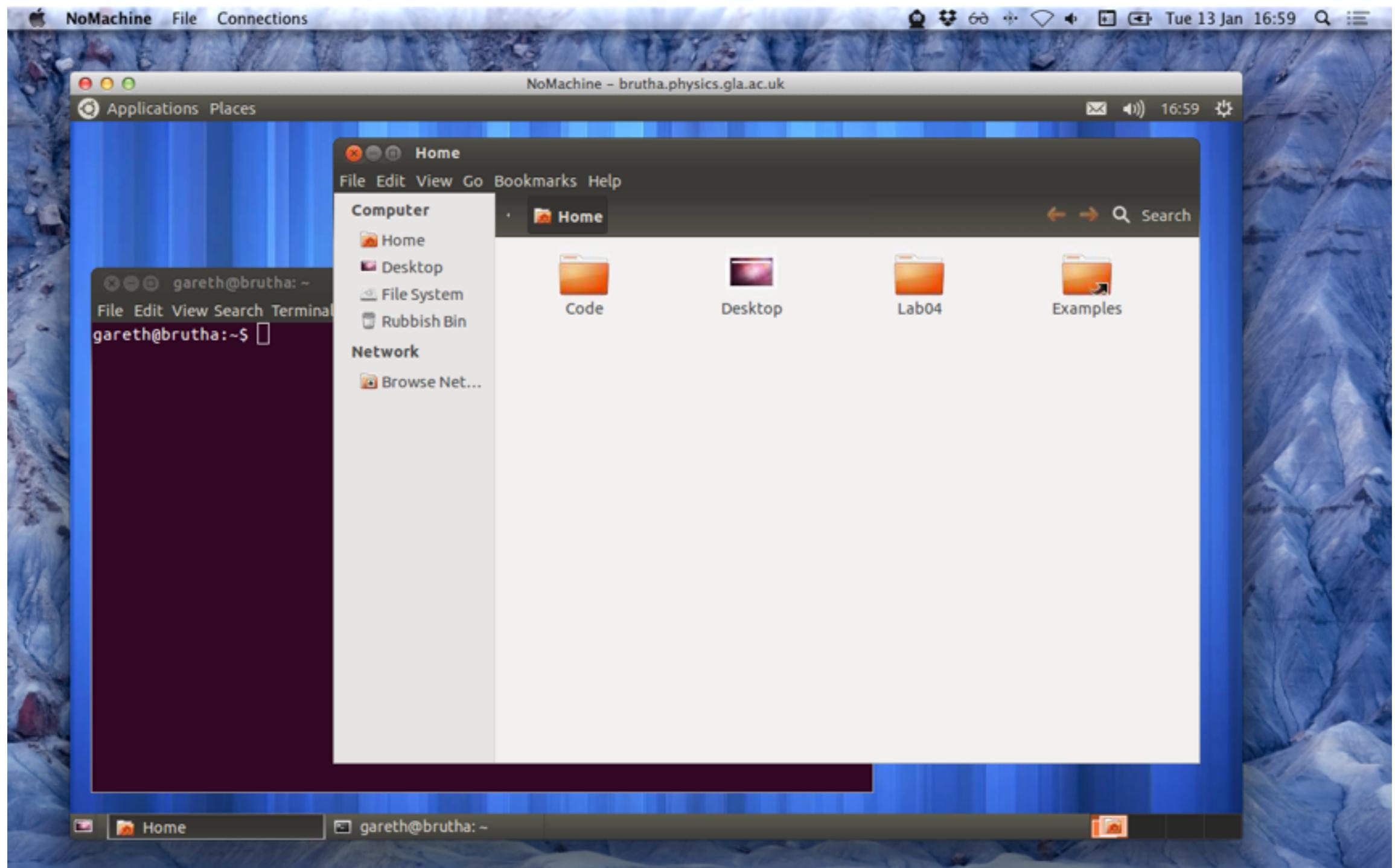


# What you'll be using for this course

- For this course you'll be given access to remote linux server.
- You'll be using a distribution known as Ubuntu.
- This is a user friendly fork of Debian that is extremely popular.
- You'll be able to log into the remote server and get a desktop via a piece of software called openNX
- Instructions are up on Moodle2

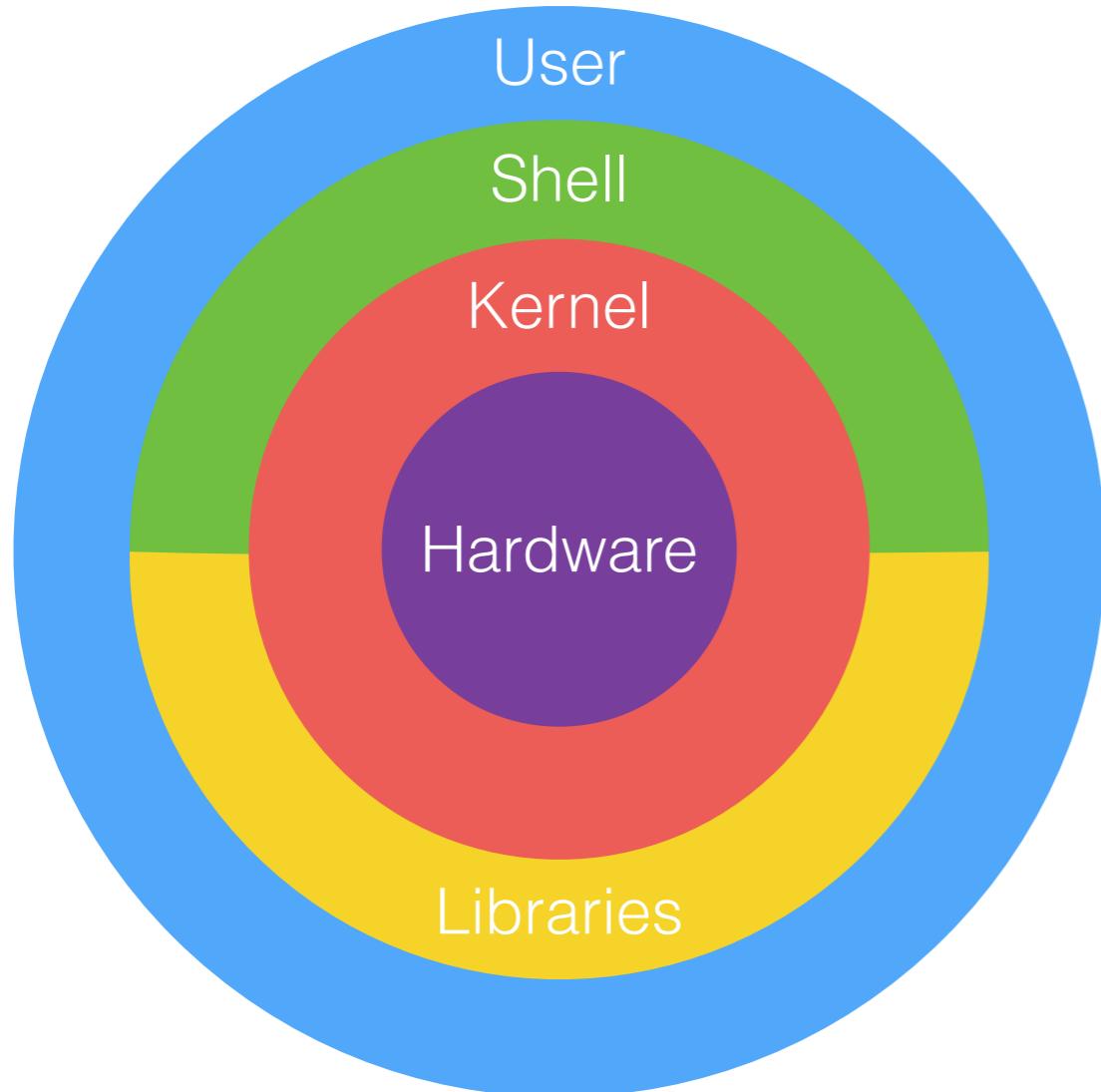


# Accessing brutha.physics.gla.ac.uk



- A brief history lesson
- Why Linux?
- The parts of an Operating System
- The Command Line

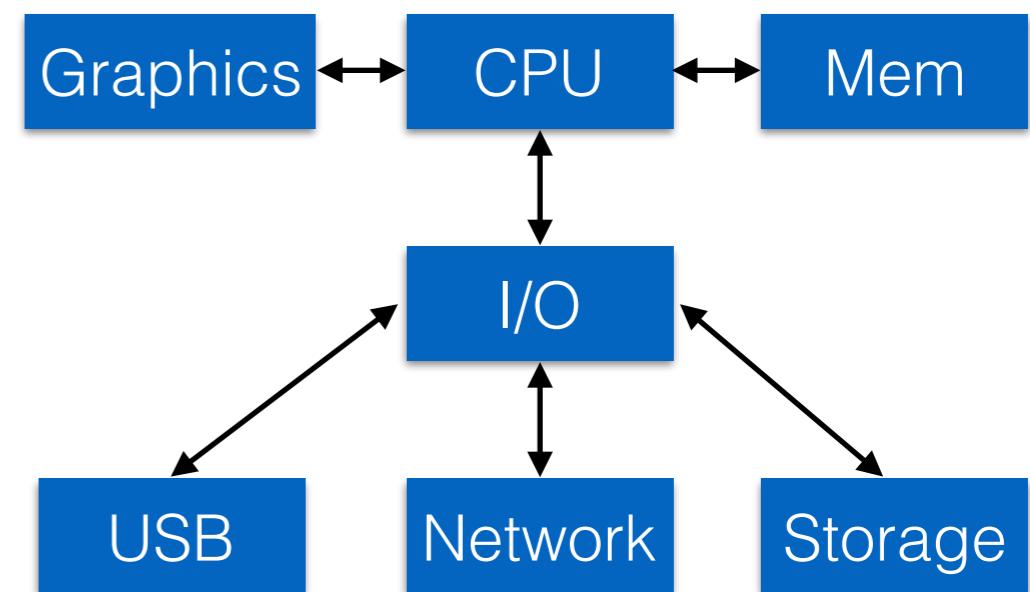
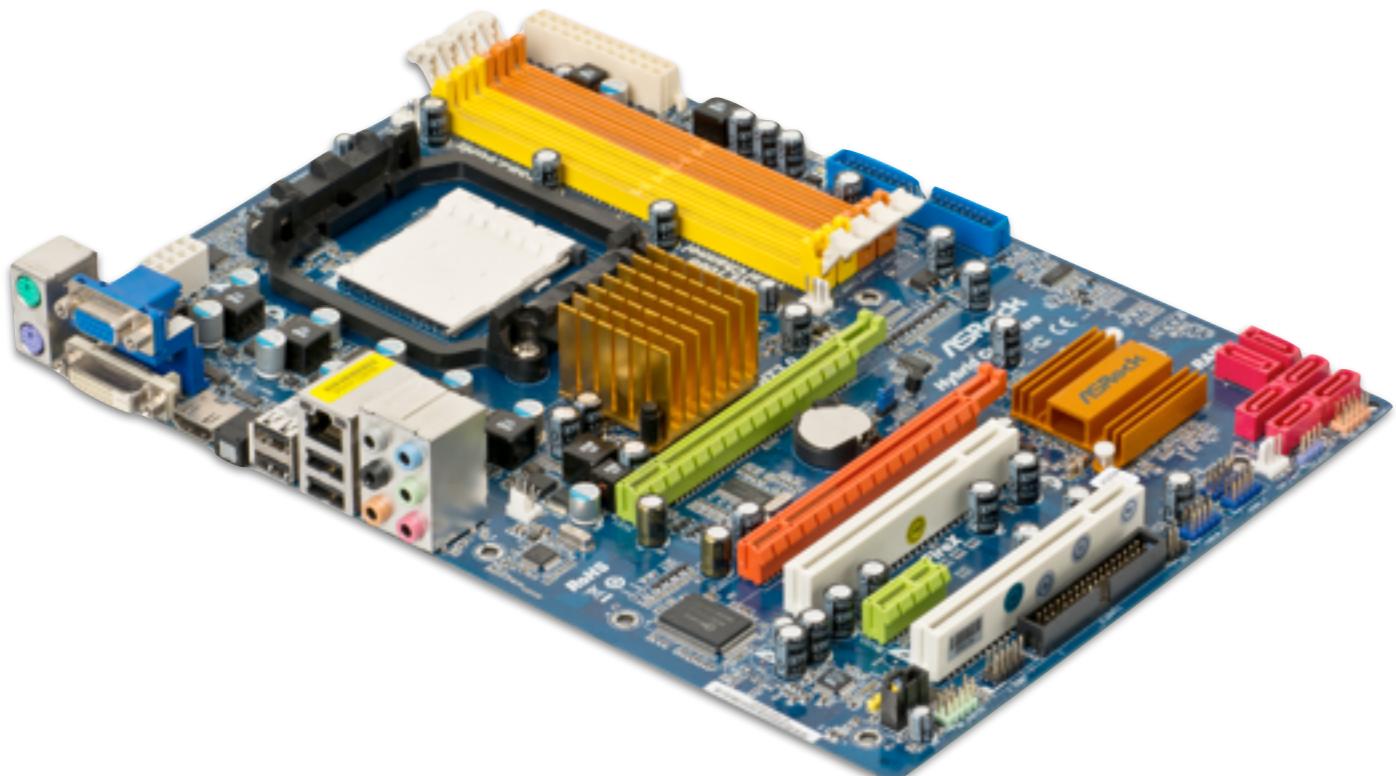
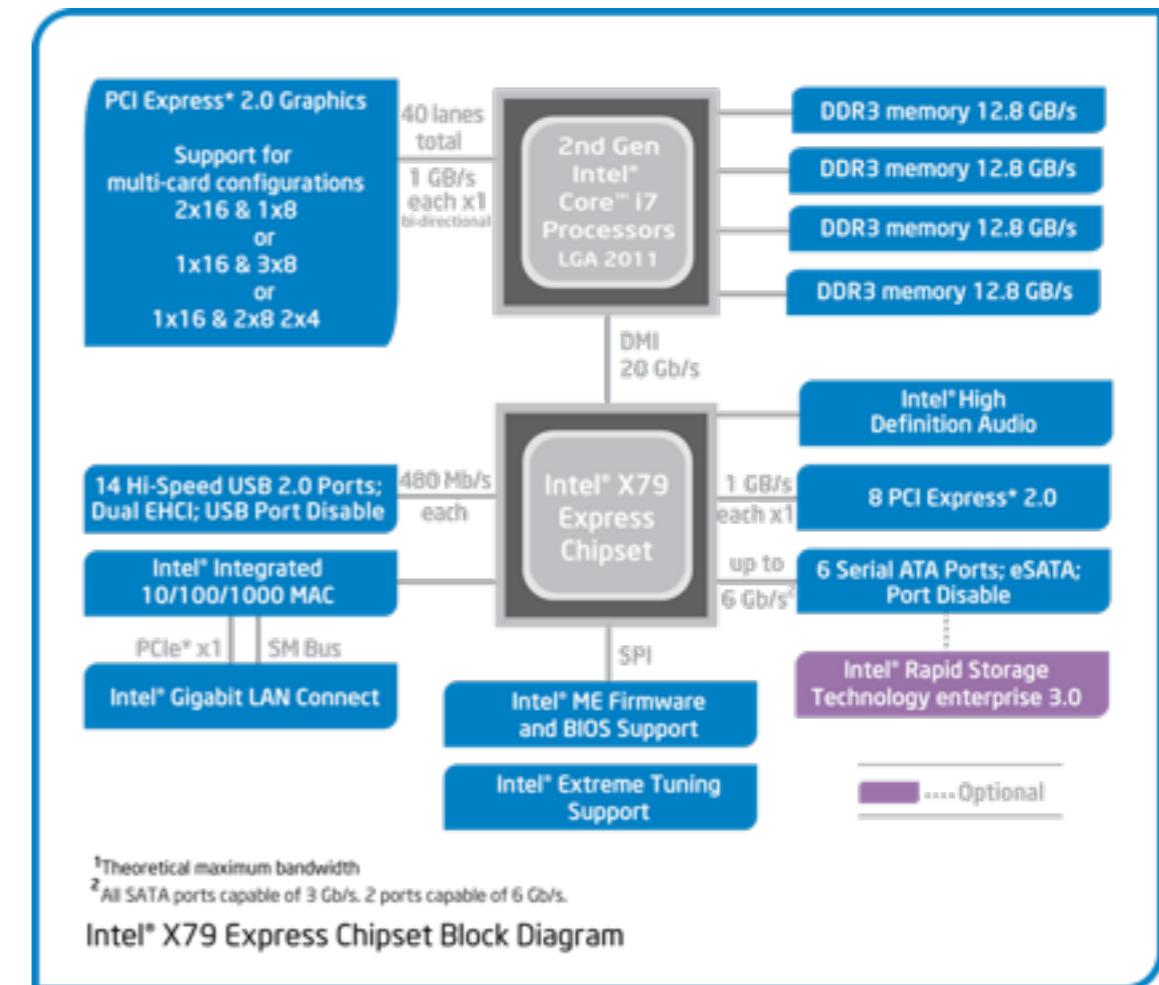
# Linux Architecture



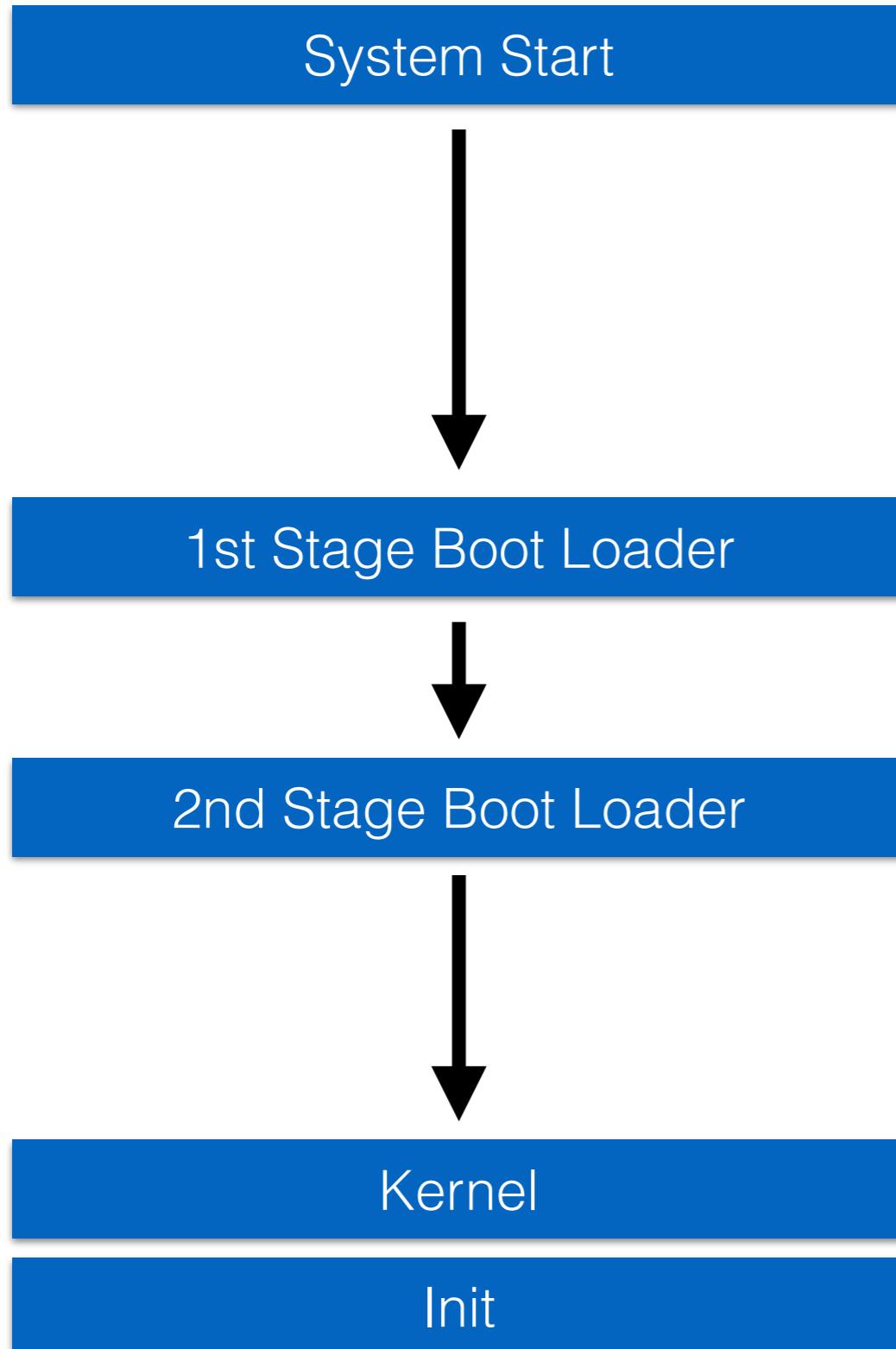
- The Kernel is the core part of Linux, it includes all the drivers need to interact with the underlying hardware.
- The Shell allows interactive access to system resource through a set of builtin commands or by running commands found on the filesystem
- Additionally access to system resources can be via system libraries that expose access to the underlying Kernel (c.f. glibc)
- User applications will usually be started by the shell, or by the init process at system startup.

# Hardware

Component	Size	Purpose
CPU	GHz	Numeric / Logic / Shift / Vector Operations
Memory	GB	Running Software and Data storage.
Network	Gbit	Data transmission, TCP/IP packet decoding.
Storage	TB	Data storage preserved without power requirements
Graphics	MOps	VGA Display, 2D/3D Acceleration, Floating point ops.



# Boot Process



## Power On Self Test (POST)

- Verify BIOS code
- Verify CPU and Registers
- Find and Verify Memory
- Initialise the BIOS
- Discover and Initialise Hardware

## Master Boot Record

- 512 bytes located on first sector
- Small program that loads 2nd Stage

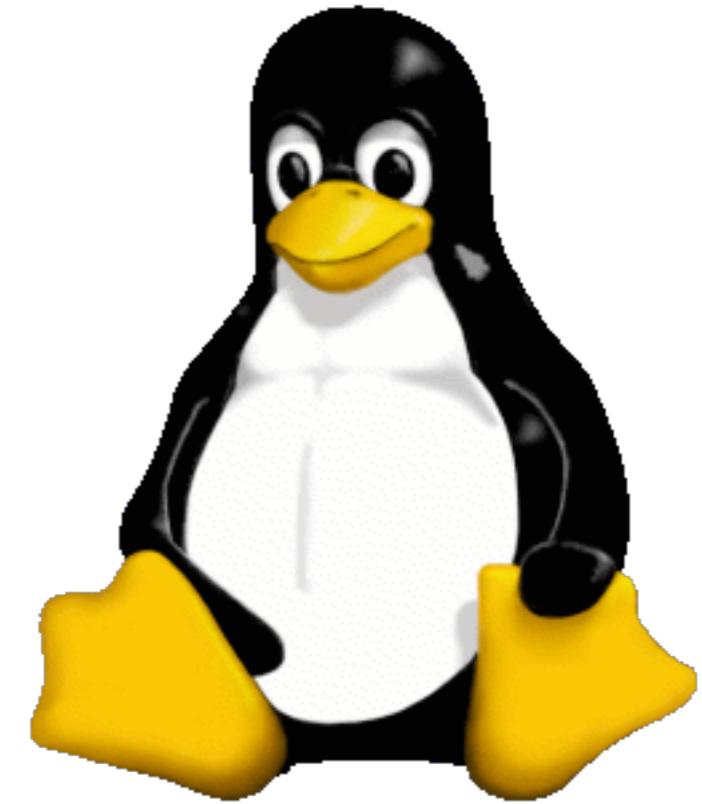
## Loads the Kernel

- Loads required components into Mem.
- Enters Protected mode
- Transfers control to the Kernel

Kernel starts and brings up system services. Once devices have been initialised and important subsystems started, starts init services.

# Kernel

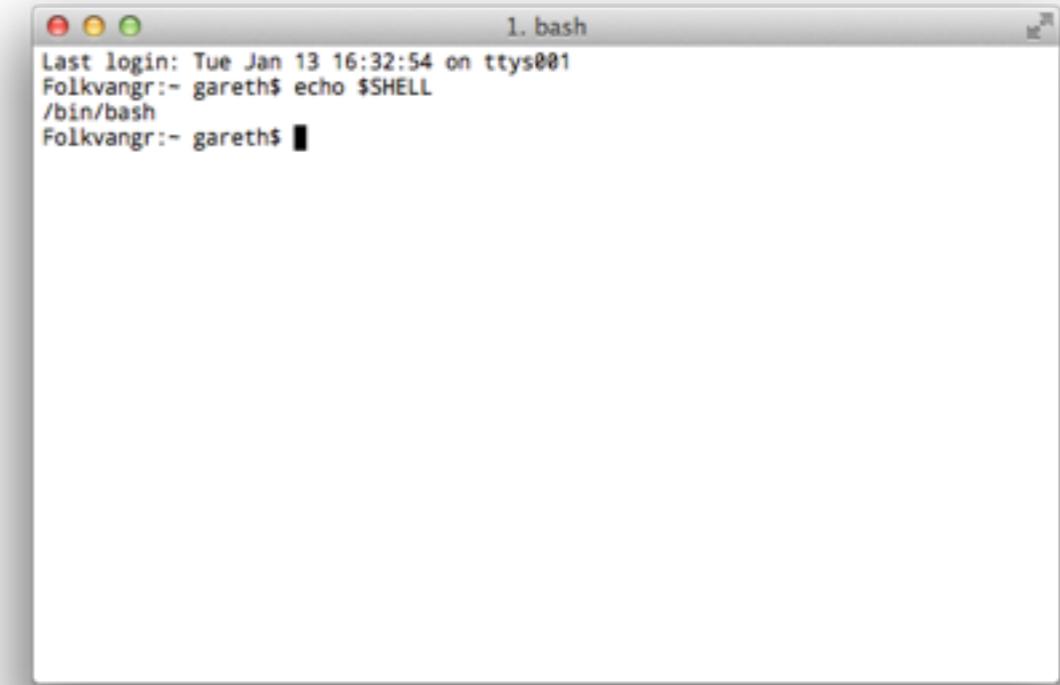
- The Kernel looks after most interactions with Hardware and all of the process running on the system
- It provides:
  - Process scheduler
  - Inter-Process Communication
  - Memory Management
  - A Virtual Filesystem
  - Networking
  - Device Mapper
  - Sound Subsystem
  - etc...



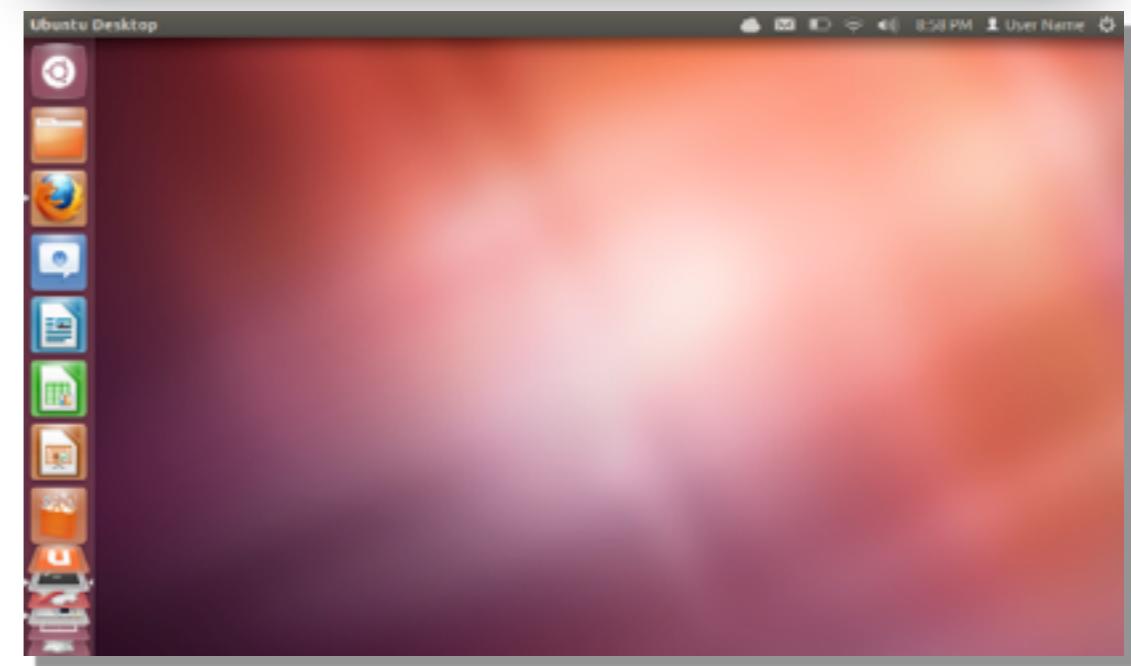
```
Isapnp: No Plug & Play device found
Linux NET4.0 for Linux 2.4
Based upon Swansea University Computer Society NET3.029
Initializing RT netlink socket
Starting kswapd
MSG: Disk quotas vqfquot_6.5.1
vesafb: framebuffer at 0xe0000000, mapped to 0xc2000000, size 51200k
vesafb: mode is 000x600x0, linelength=000, pages=0
vesafb: protected mode interface info at a5f3:1f5f
vesafb: scrolling: redraw
Console: switching to colour frame buffer device 100x37
fb0: VGA VGA frame buffer device
ptp: 256 Unix98 ptys configured
Uniform Multi-Platform E-IDE driver Revision: 0.31
ide: Assuming 50MHz system bus speed for PIO modes; override with idebus=xx
hda: Generic I2C4, ATAPI CD/DVD-ROM drive
ide0 at 0x1f0-0x1f7:0x3f6 on irq 14
hda: ATAPI 4X CD-ROM drive, 512kB Cache
Uniform CD-ROM driver Revision: 3.12
FDC 0 is an 0272A
NMDISK driver initialized: 16 IBM disks of 4096K size 1024 blocksize
Cronyx Ltd. Synchronous PPP and CISCO HDLC (c) 1994
Linux port (c) 1996 Building Number Three Ltd & Jan "Yenga" Kasprzak.
SCSI subsystem driver Revision: 1.00
scsi0 : SCSI host adapter emulation for IDE ATAPI devices
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP, IGMP
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP: Hash tables configured (established 2048 bind 2048)
NET4: Unix domain sockets 1.0/SMP for Linux NET4.0.
NMDISK: Compressed Image found at block 0
```

# Shell

- In Linux the shell handles interaction between Users and the system.
- It can be via a text interface known as the command line as well as through a graphical interface, for instance the “gnome-shell”
- It allows other processes/applications to be started such as editors, compilers, office suites.
- We'll cover the command line in more detail later in this set of slides.

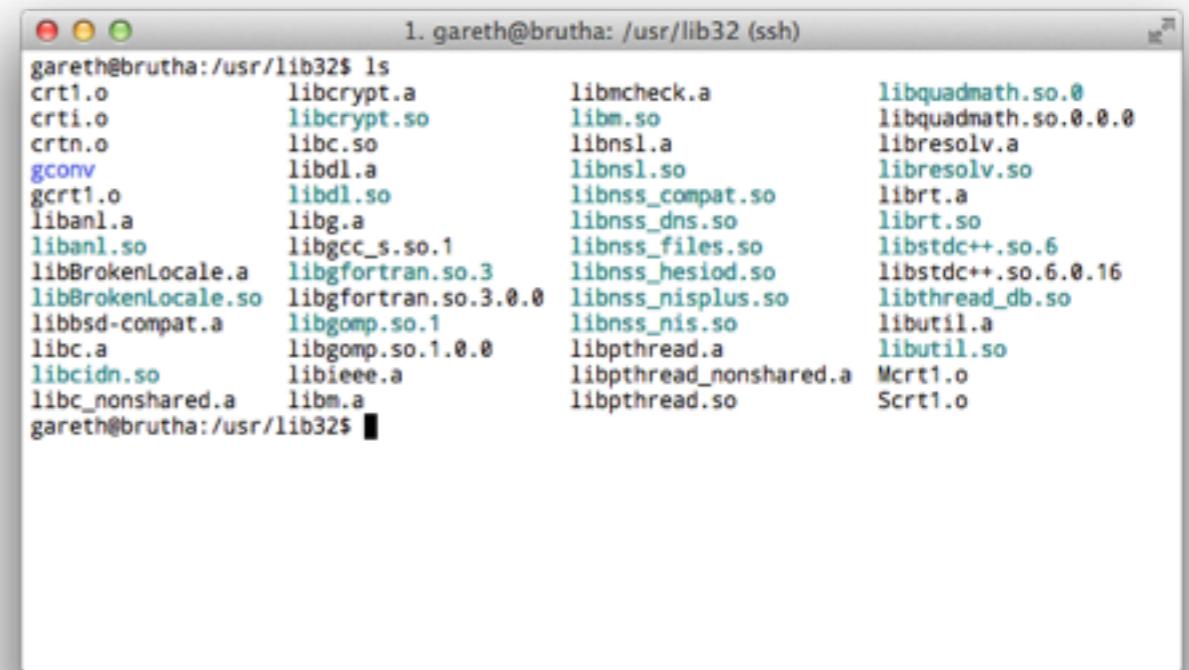


```
Last login: Tue Jan 13 16:32:54 on ttys001
Folkvangr:- gareth$ echo $SHELL
/bin/bash
Folkvangr:- gareth$
```



# System Libraries

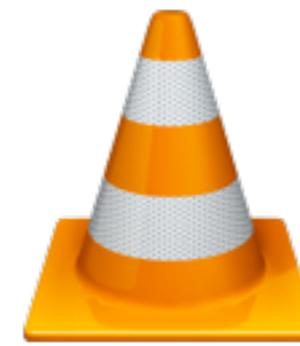
- Running programs also need to interact with system resources.
- This is carried out via system libraries.
- There are a wide range of these most of which are beyond the scope of this course.
- You will however use the standard C libraries as part of the C programming component of this course.



```
1. gareth@brutha: /usr/lib32 (ssh)
gareth@brutha:/usr/lib32$ ls
crt1.o          libcrypt.a      libmcheck.a      libquadmath.so.0
crti.o          libcrypt.so     libm.so          libquadmath.so.0.0.0
crtn.o          libc.so         libnsl.a        libresolv.a
gconv           libdl.a         libnsl.so       libresolv.so
gcrt1.o         libdl.so        libnss_compat.so librt.a
libanl.a         libgcc.a       libnss_dns.so   librt.so
libanl.so        libgcc_s.so.1  libnss_files.so libstdc++.so.6
libBrokenLocale.a libgfortran.so.3 libnss_hesiod.so libstdc++.so.6.0.16
libBrokenLocale.so libgfortran.so.3.0.0 libnss_nisplus.so libthread_db.so
libbsd-compat.a libgomp.so.1    libnss_nis.so    libutil.a
libc.a          libgcc_s.so.1  libieee.a      libutil.so
libcidn.so       libgomp.so.1.0.0 libm.a          Mlibcrt1.o
libc_nonshared.a libmcheck.a    libpthread.a    Scrt1.o
gareth@brutha:/usr/lib32$
```

# Userland

- User processes are the final level of the Operating System.
- This is likely the most familiar and includes programs such as web browsers, office suites, editors etc.
- Most user orientated distributions ship with software that allows you to do most common tasks.



- A brief history lesson
- Why Linux?
- The parts of an Operating System
- The Command Line

# The Command Line

- There are a variety of different shells available: Sh, BASH, CSh, TCSH, ZSh...
- C-style shells: C-like syntax for scripting
- Bash-style shells: slightly more consistent, and more widely adopted in the research-computing community.
- We'll use Bash in this course, but feel free to experiment!
- To open the terminal, click the ubuntu button and type "terminal"!

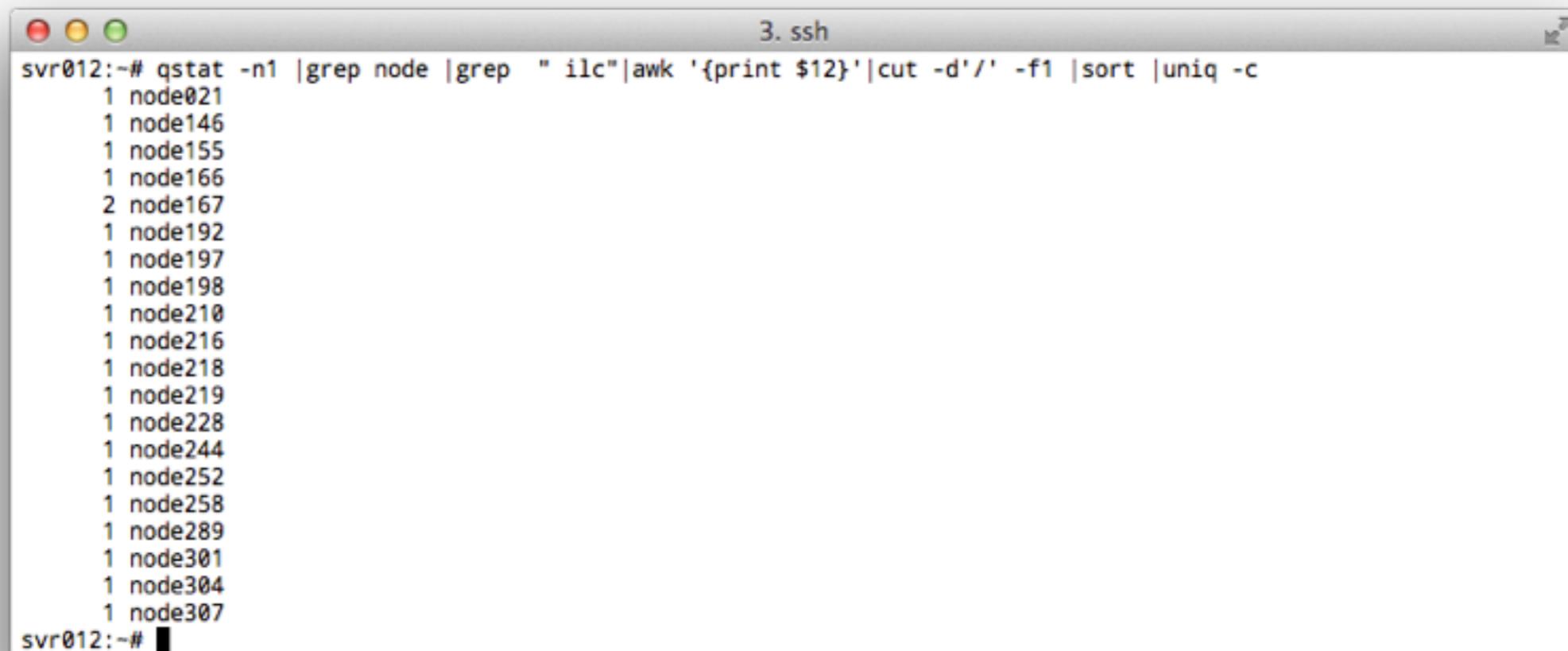


# The Command Line

- The command line lets us input command at the shell prompt and receive output in a text form.
- Commands are of the form:

```
prompt# <command> <flags> <arguments>
```

```
prompt# du -h --max-depth=1
```
- Commands can be chained together to build up complex interactions and obtain the results we want.



A screenshot of a terminal window titled "3. ssh". The window contains a command-line session on a Linux system named "svr012". The user has run a pipeline of commands to count the occurrences of each node name. The commands entered were:

```
svr012:~# qstat -n1 |grep node |grep " ilc" |awk '{print $12}' |cut -d'/' -f1 |sort |uniq -c
```

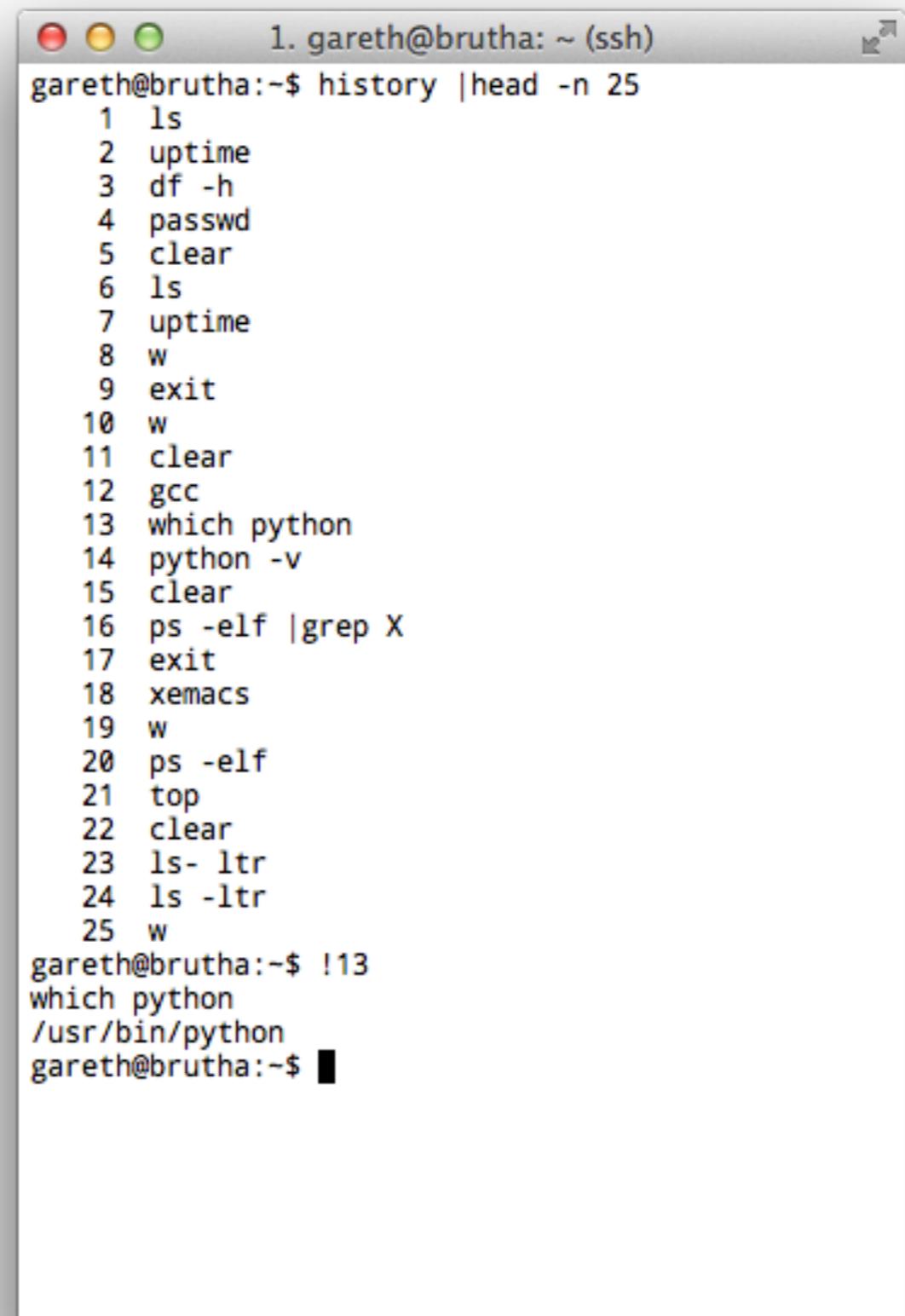
The output shows the count of each node name:

Node Name	Count
node021	1
node146	1
node155	1
node166	1
node167	2
node192	1
node197	1
node198	1
node210	1
node216	1
node218	1
node219	1
node228	1
node244	1
node252	1
node258	1
node289	1
node301	1
node304	1
node307	1

The terminal prompt "svr012:~#" is visible at the bottom.

# Useful Commands

- **history** - gives all commands you've entered
- **!!** - runs the last command
- **!<number>** runs the command at <number> i.e. !484 runs command 484
- **up/down arrow** cycles through previous commands
- **tab** auto completes names



A screenshot of a terminal window titled "1. gareth@brutha: ~ (ssh)". The window displays the output of the command "history | head -n 25", which shows a list of 25 commands run by the user. Below this, the user types "!13" and presses Enter, which executes the command "which python" and shows its path "/usr/bin/python". The terminal window has a standard OS X style with red, yellow, and green close buttons.

```
gareth@brutha:~$ history | head -n 25
1  ls
2  uptime
3  df -h
4  passwd
5  clear
6  ls
7  uptime
8  w
9  exit
10 w
11 clear
12 gcc
13 which python
14 python -v
15 clear
16 ps -elf |grep X
17 exit
18 xemacs
19 w
20 ps -elf
21 top
22 clear
23 ls -ltr
24 ls -ltr
25 w
gareth@brutha:~$ !13
which python
/usr/bin/python
gareth@brutha:~$ █
```

# Useful Commands (2)

- **man** will give you the manual page for a particular command.
  - hitting the **spacebar** takes you to the next page and **q** quits
- **ctrl-r** allows you to search through previous commands
- **clear** empties the screen (useful for clearing your head sometimes).

The screenshot shows a terminal window with the title "l.sh". The content of the window is the man page for the "ls" command. It includes sections for NAME, SYNOPSIS, DESCRIPTION, and a list of options:

**NAME**  
ls -- list directory contents

**SYNOPSIS**  
ls [-ABCFGHLOPRSTUW@abcdefghijklmnopqrstuvwxyz] [file ...]

**DESCRIPTION**  
For each operand that names a file of a type other than directory, ls displays its name as well as any requested, associated information. For each operand that names a file of type directory, ls displays the names of files contained within that directory, as well as any requested, associated information.

If no operands are given, the contents of the current directory are displayed. If more than one operand is given, non-directory operands are displayed first; directory and non-directory operands are sorted separately and in lexicographical order.

The following options are available:

- @ Display extended attribute keys and sizes in long (-l) output.
- 1 (The numeric digit "'one'"). Force output to be one entry per line. This is the default when output is not to a terminal.
- A List all entries except for .. and ... Always set for the super-user.
- a Include directory entries whose names begin with a dot (.).
- B Force printing of non-printable characters (as defined by ctype(3) and current locale settings) in file names as \xxx, where xxx is the numeric value of the character in octal.
- b As -B, but use C escape codes whenever possible.
- C Force multi-column output; this is the default when output is to a terminal.
- c Use time when file status was last changed for sorting (-t) or long printing (-l).
- d Directories are listed as plain files (not searched recursively).
- e Print the Access Control List (ACL) associated with the file, if present, in long (-l) output.
- F Display a slash ('/') immediately after each pathname that is a directory, an asterisk ('\*') after each that is executable, an at

- A brief history lesson
- Why Linux?
- The parts of an Operating System
- The Command Line