CSN08601 – Computer Systems

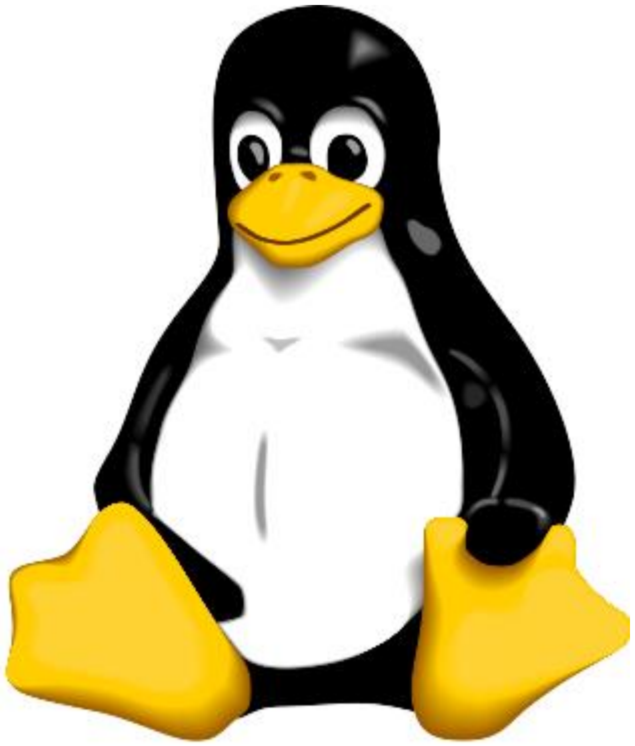
OS Extras

A brief history of UNIX

A brief history of UNIX
Timeline

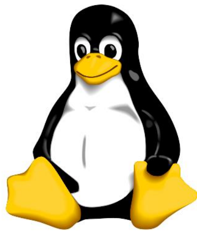
- First developed in the70s at Bell Labs, who licenced it to various vendors
  - BSD (University of California, Berkeley), AIX (IBM), Solaris (Sun Microsystems), Xenix (Microsoft)
- Sold to Novell in the 90s and then to SCO (Santa Cruz Operation)
- UNIX trademark passed to The Open Group who give it to anyone complying with the Single UNIX Specification standard
- The most widely used UNIX-compliant OS is macOS
  - *BSD* versions are not compliant but very related
  - Linux is not compliant either

Linux



What is Linux?

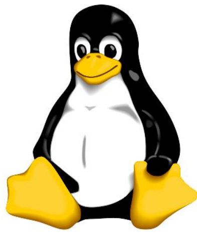
- Linux is an OS designed to emulate the functionality of UNIX (i.e. it pretends to be compliant but it’s not)
- Linus Torvalds wrote the kernel first
  - He got frustrated of licencing fees
- Now maintained by the open source community
- Linux is **only the kernel**
  - The others are **ports** or **distributions**
  - The GNU project (‘GNU is Not Unix’) adds many essential features (e.g. the GNU C Compiler: gcc)



Where is Linux?

- Linux is deployed on all sorts of hardware
  - Servers: massive market share
  - Mobiles: Android uses the Linux kernel
    - Although Android it is **not** considered a “proper” Linux
    - Massive market share
  - Desktops: not user friendly, not many games, unpopular
  - Other: network hardware, TV boxes etc

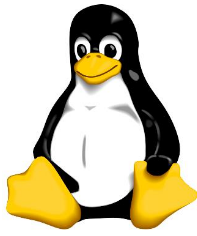
With all the above considered, Linux holds most of the OS market. It won the OS wars.



Linux

Linux is free, as in speech

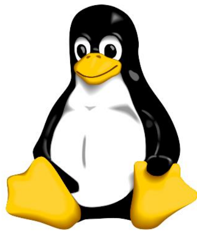
* Free to download versions are available
  * But there are maintenance costs!
* Free, not because you don't have to pay for the Kernel, but because it is not constrained/controlled
  * Free as in "free speech" – not as in "free beer"
* Many versions are available as **distributions**
  1. Pack the free Linux Kernel with UI, drivers, tools etc
  2. Ship it in a downloadable installation
  3. ???
  4. Profit!



Linux

Open Source/GNU License

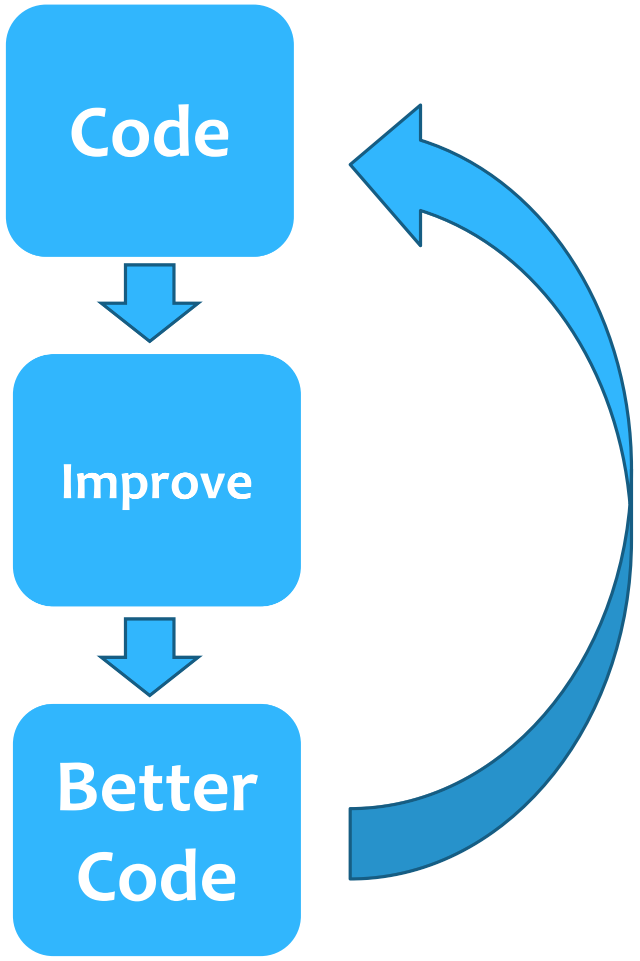
- Linux is Licensed under the GNU License, the basic principles of which are:
  - You can get the code for free
  - You can change it in any way you want
  - You can even sell it for profit
- But
  - You must give credits to previous code contributors
  - You must open-source any improvements/changes



Note: There are other open source licences which differ from GNU

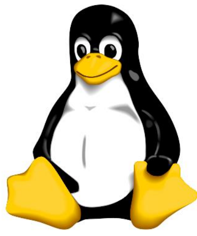
Linux

The value of open-source



1. Get free code
2. Improve it
3. Produce better code
4. Give it back

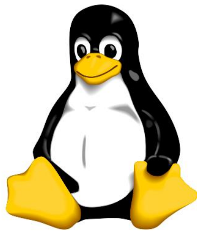
Code is "paid" with effort
This leads to improvements
Everyone gets better code
Everyone contributes in
  making the code better
Bonus: free auditing!



Linux

Why run Linux?

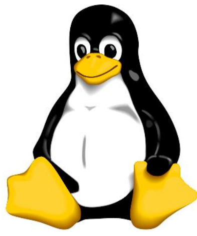
- No need to pay licences
- High performance
- Server-orientated philosophy
  - Easy to run email/web servers with minimal cost
- Amazon's Cloud and Google use Linux
- Security & reliability
  - Open source philosophy assist in quality
- Specialist applications are available



Linux

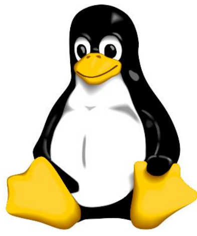
Requirements

- Low hardware requirements
  - Runs well on older machines
  - Can be run on a VM in your main machine
- Needs less disk space
- Can also be run from a ‘live’ USB stick
  - Nothing installed on the hard disk
- Lots of your time to do simple things



Linux
Setup

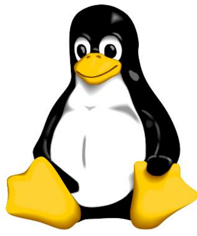
* Backup your existing system and files
* Download a distribution (e.g. Ubuntu or Mint)
* Start the installation (follow the instructions). It will:
  * Re-organise (partition) your hard-disk to fit both Linux and Windows
  * Install the Linux Distribution
  * Configure your hardware
  * Set-up your system to give you a choice which OS to load (dual-boot)



Linux

What's included?

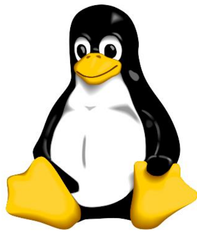
* Details vary, but typically they will include:
  * Linux kernel
  * A packaging system – helps you locate, download and install application packages easily
  * GCC (GNU C Compiler) & utilities
  * Graphical UI (you can also change this)
  * Web browser/email (e.g. Firefox/Chrome)
  * Office suite (usually Openoffice)
  * Multimedia (e.g. mp3/mp4 playback)



Linux

Which Version should I get?

- Visit distrowatch.com to see the most popular version
  - Mint: the most popular, very user friendly, made in IR
  - Debian: second most-popular, not for novice users
  - Ubuntu: third most popular, very user friendly
  - Don’t forget to read the reviews
- Distrowatch keep a 'top 100' chart: an indication of how fragmented the Linux scene has become



Mobile OS

Mobile OS

Introduction

- Operating systems designed to run on mobile devices
  - **Big focus on energy efficiency**
- Access to fewer hardware resources
- Support a limited range of applications software
- Communications oriented
- Single-user and single-application
  - Mobile OS are actually multitasking but only show one App at the time. The other Apps go on a sleeping state
- Offer specific features like geolocation

Mobile OS

Android

- Open-source Linux-based mobile OS
- Uses the Linux Kernel but it is **not compatible** with Linux - **not considered a "Linux"**
- Initially developed by Android Inc, now Google
- Used on mobile Phones, Tablets, TVs etc
- Apps are made in Java
  - Using a freely available and easy-to-use SDK
    - Does not support all Java libraries

Mobile OS
iOS

- Closed-source mobile OS by Apple
  - Based on Darwin, which is based on BSD and others
- Works only on Apple hardware (iPhones, iPads etc)
  - Optimised for Apple hardware **only**
- Runs Apps made in Objective-C or Swift
  - Languages primarily maintained by Apple
  - Code runs natively on the device, i.e. no JVM
    - That's more energy efficient and faster

Memory Management

More Detail

More Detail

Memory Protection

- Memory protection is an important feature of multitasking OS, especially so in modern times
  - Memory of tasks is to be protected from malicious tasks
- Page table entries include some protection bits
  - Pages can be marked as being read-only or writable
  - A U/S bit shows if the page is at User or Supervisor level
  - Further protection may be provided by separating pages for different tasks
  - Windows keeps separate page directories & tables for each task, ensuring they can’t interfere with each other

More Detail

Hits and Misses

- A ***hit*** is when a page is already in memory
- A ***miss*** (or ***page fault***) is when the page is not in memory and needs to be swapped in
  - If RAM is not completely full, the page is read from the disk and loaded into RAM
  - If RAM is full (typical), a page replacement is needed
  - The algorithm for choosing which page to replace is important for the system performance
  - Ideally, you would want to get rid of a page that is not going to be needed again

More Detail

Page Replacement

- It is not possible to predict exactly what will happen
- There are a number of page replacement algorithms
  - Different ones are used by different operating systems
  - Mostly based on taking note of whether a page has been used recently
  - The Principle of Locality shows that recently used pages are likely to be used again in the near future

More Detail

Page Replacement Algorithms

- First-In-First Out (FIFO)
  - Pros – Simple to implement
  - Cons – Pages loaded first may be heavily used.
- Least Recently Used & Least Frequently Used
  - Pros – Recent or frequent use is a good predictor of future use
  - Cons – Complexity, as each table entry requires a time field
- Second Chance
  - A page can dodge replacement once if it has been accessed.
  - Pros – Recently used pages take precedence over others.
    - Simple implementation
    - Combines FIFO and LRU features.

More Detail

Compression

- It is possible to compress the pages
  - Either in RAM or the Disk or both
- Another way to extend the memory
  - Sacrifice processor time for virtually larger memory
  - Works well for tasks that are memory-intensive but not processor intensive (e.g. running a browser)
- There are many algorithms
  - Linux and macOS offer this

More Detail

Notes

- In Windows, the file is called Pagefile.sys. Its size can be adjusted using **Control Panel.. System.. Advanced System Settings.. … … Virtual Memory**
  - Tip: select the disk for the page file wisely
- In Linux the swap file is on a separate partition

Windows Multitasking

Windows Multitasking

Introduction

- Multiple processes can run on multiple processors
  - Typically, each application is run as a process in its own memory space – which should be sandboxed
- Each process can consist of a number of threads
  - Word may have threads for file I/O, user interface, spell-checking, and so on – one thread for every small task
- Threads run concurrently

Windows Multitasking

Priority levels

- Windows processes have 32 priority levels
  - (0 is the lowest priority, and 31 is the highest)
  - They are divided into **real-time** and **variable**
- Priorities of 16-31 are used for real-time processes
  - Not swappable, fixed priority, scheduled ahead others
    - For critical parts like The Kernel
- Priorities of 0-15 are used for dynamic applications
  - Swappable, dynamically variable priorities
    - For non-critical bits of the OS

Windows Multitasking

Dynamic priorities

- Threads of a task start with the base priority of the parent process but this can become lower or higher
  - Ensure processes are not starved of processing power
  - Even when waiting for I/O data, service them immediately when they arrive
  - This is good because data can’t wait for long before they are lost and have to be resent

Windows Multitasking

User control

- Windows users can set priority of tasks
  - Via Task Manager or cmd parameter
  - Roughly: low = 4, normal = 8, high = 12
- Can also set CPU affinity
  - The right setting depends on the CPU type
- There is no typical need to change these settings
  - Most likely you'll break something
  - Unless if you've done your reading and you're looking for best performance optimisation (e.g. for a server)

Process vs Thread

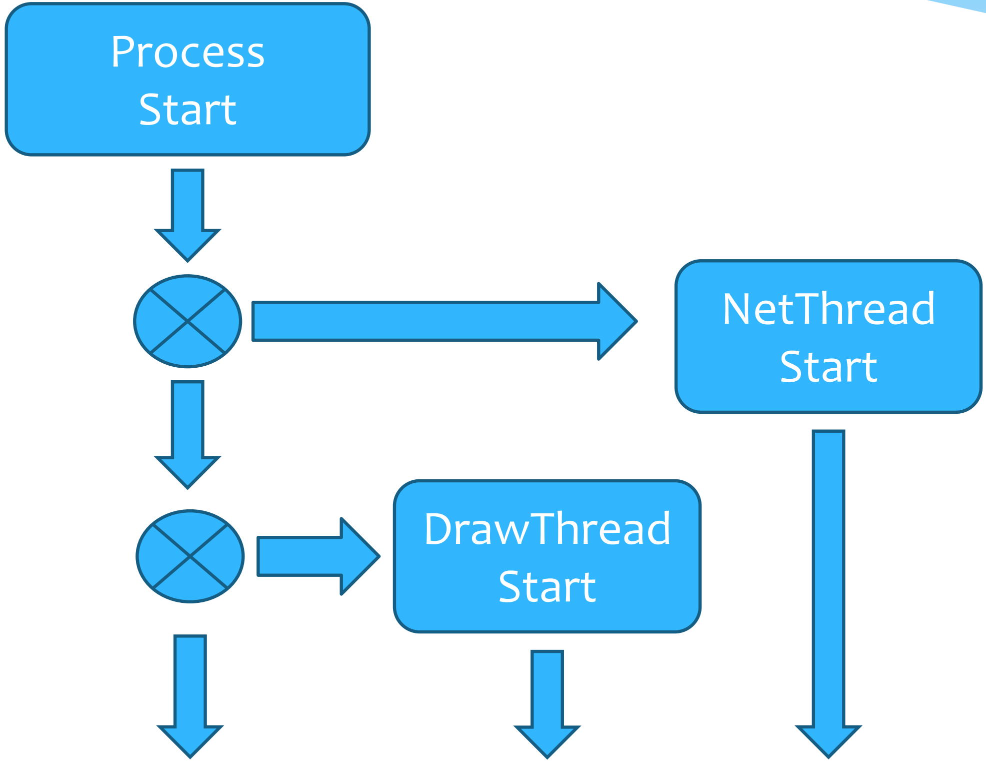
Process vs Thread

What is a thread?

- Makes multitasking more efficient
- A Thread …
  - is also known as a "Lightweight process"
    - Easier to manage context switches
  - can only exist as a child of a process
  - runs independently of other threads or the process
- Before threads we had multiple "child processes"

Process vs Thread

Threads illustrated



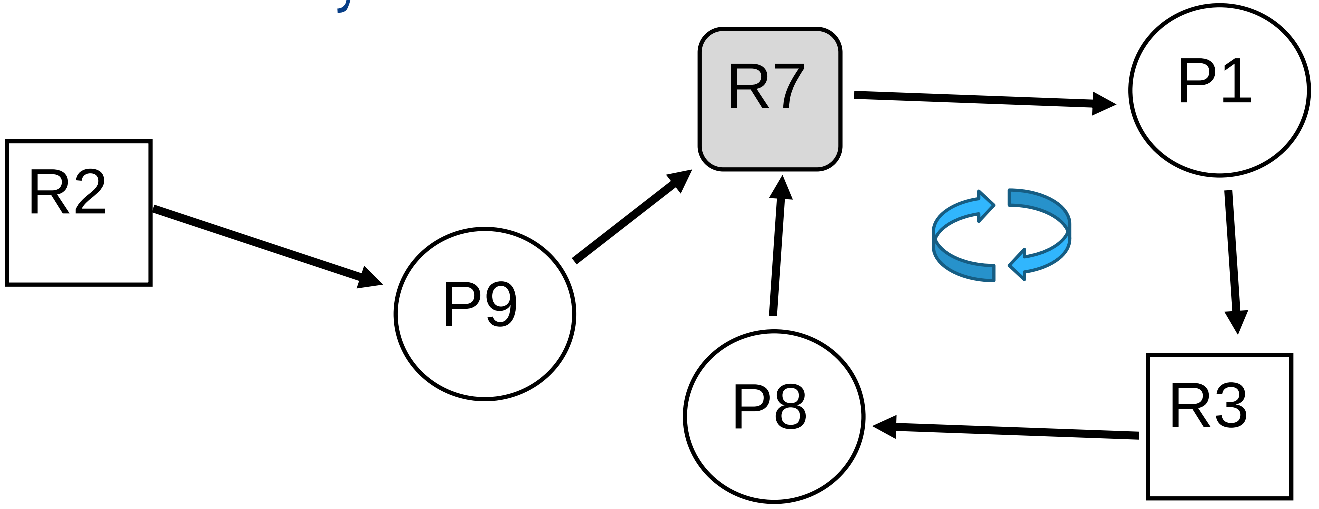
Processes can start multiple threads. E.g. a Browser loading a page may start different threads to:

1. Control Tabs
2. Read data from network
3. Draw HTML text
4. Draw images

Process vs Thread

Deadlock

Deadlock happens when two processes or threads need to wait for each-other to finish. In the example, process P8 needs to wait for resource R7 which is in use by P1 which in turn waits for R3 which is use by P8.



Process vs Thread

Threads vs Hyper-Threading

- Do not confuse threads with Hyper-Threading
- Hyper-Threading is a hardware feature of Intel CPUs
- Sharing of the main processor components
  - e.g. CU and ALU
- Two sets of the context components
  - e.g. registers, PC, IR
- OS views this as a two-core processor

File Systems

Mounting and Mapping

Mounting and Mapping

Introduction

- Mounting or Mapping is the process of attaching a disk partition (containing a file system) to a location accessible by users and apps
- In Linux the partition is represented in a file name and this can be mounted to a directory. E.g.:

```
mount -t ext2 /dev/hdb1 /new-disk
```

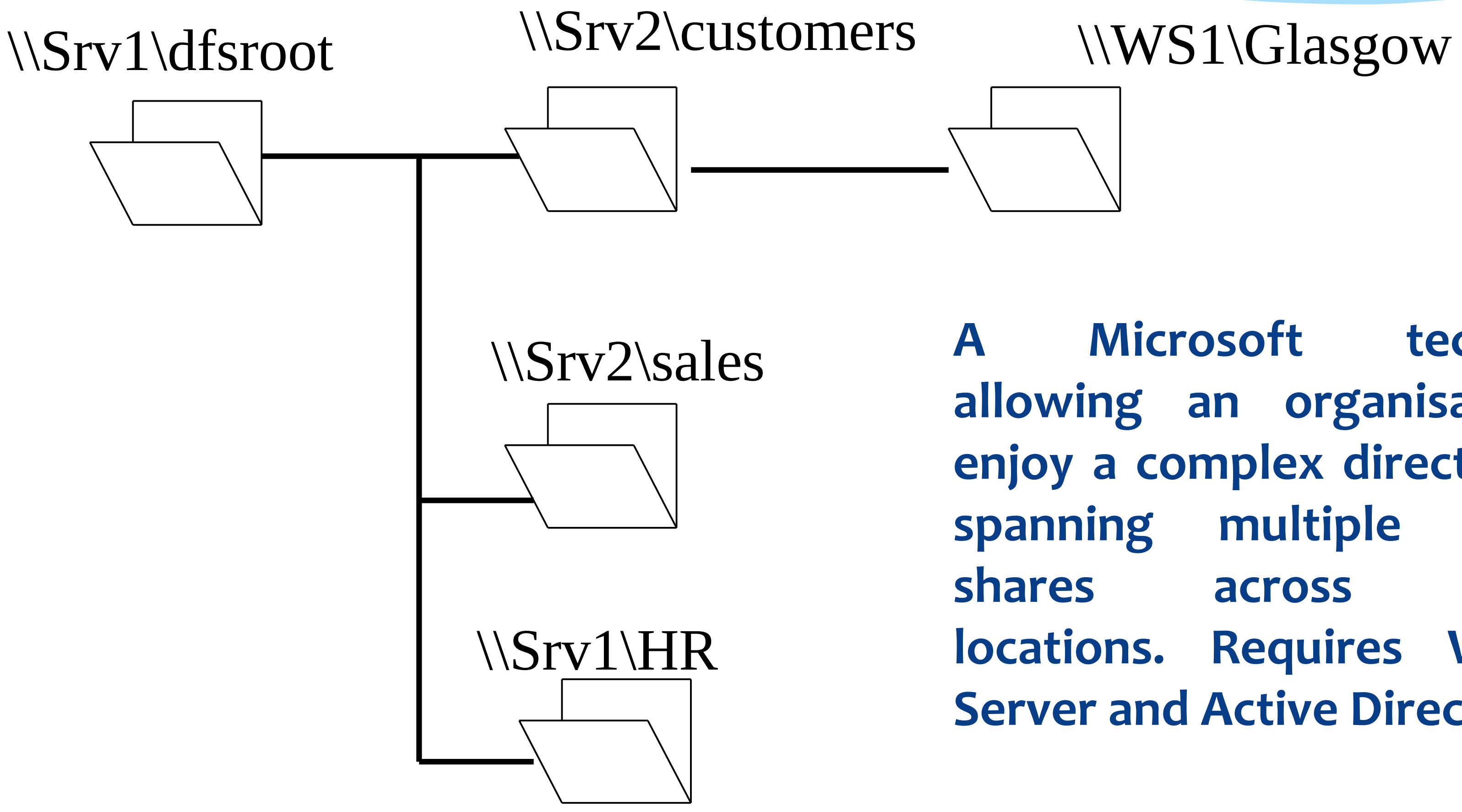
- In Windows, mapping is done through a GUI and into a drive letter (**though mapping to folders or via CMD is possible**)
- Mapping network drives is also possible

Mounting and Mapping
Network file systems

* A network file system is a protocol allowing a file system existing on a different computer to be mapped locally. Examples are SMB and NFS
  * SMB (Server Message Block) is used by Windows
  * NFS (Network File System) is typically used by *nix
* Network file systems are also mapped on folders (or virtual drives in the case of Windows)
* Your H: and I: drives are examples of these

Mounting and Mapping

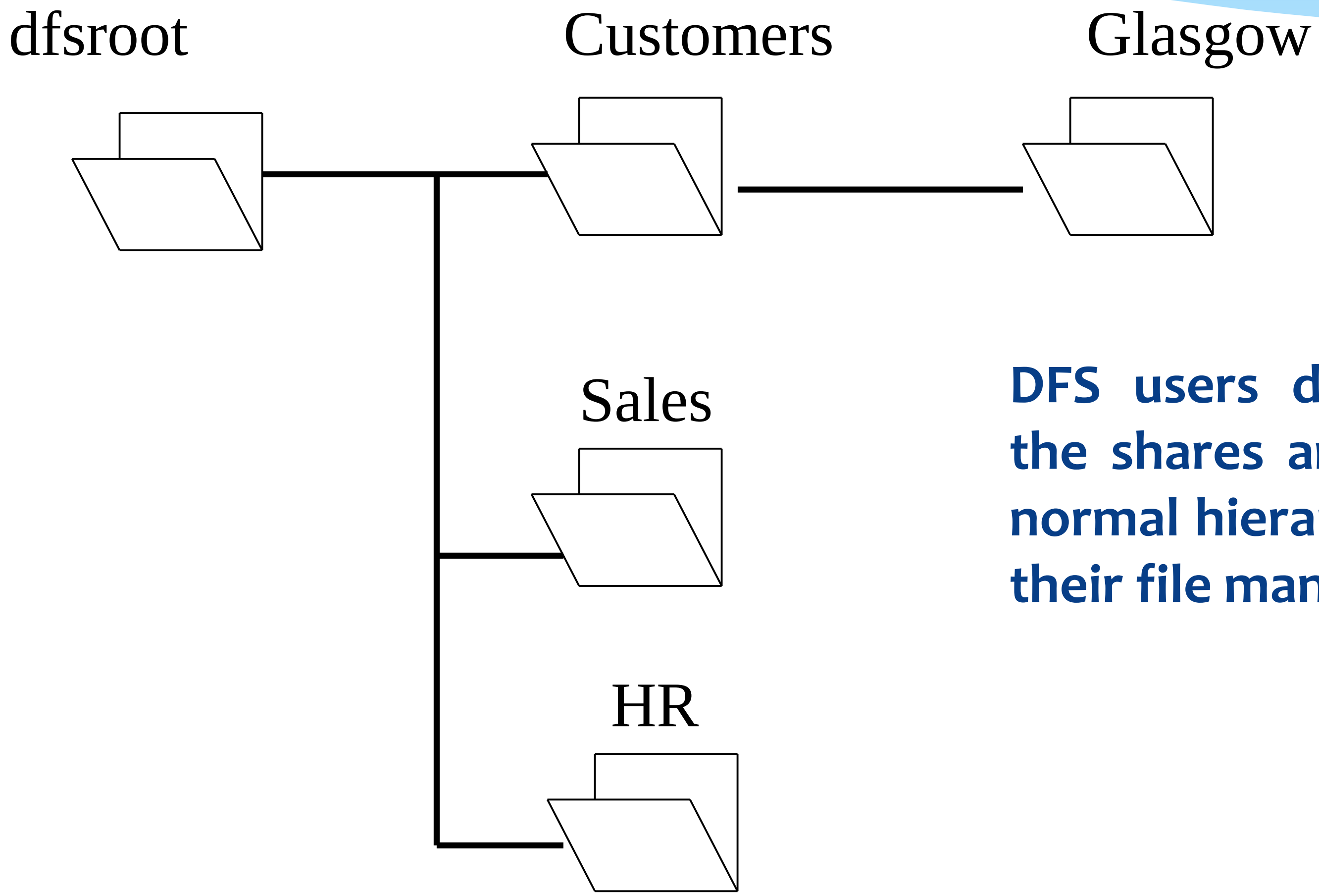
Distributed file systems



A Microsoft technology allowing an organisation to enjoy a complex directory tree spanning multiple network shares across multiple locations. Requires Windows Server and Active Directory.

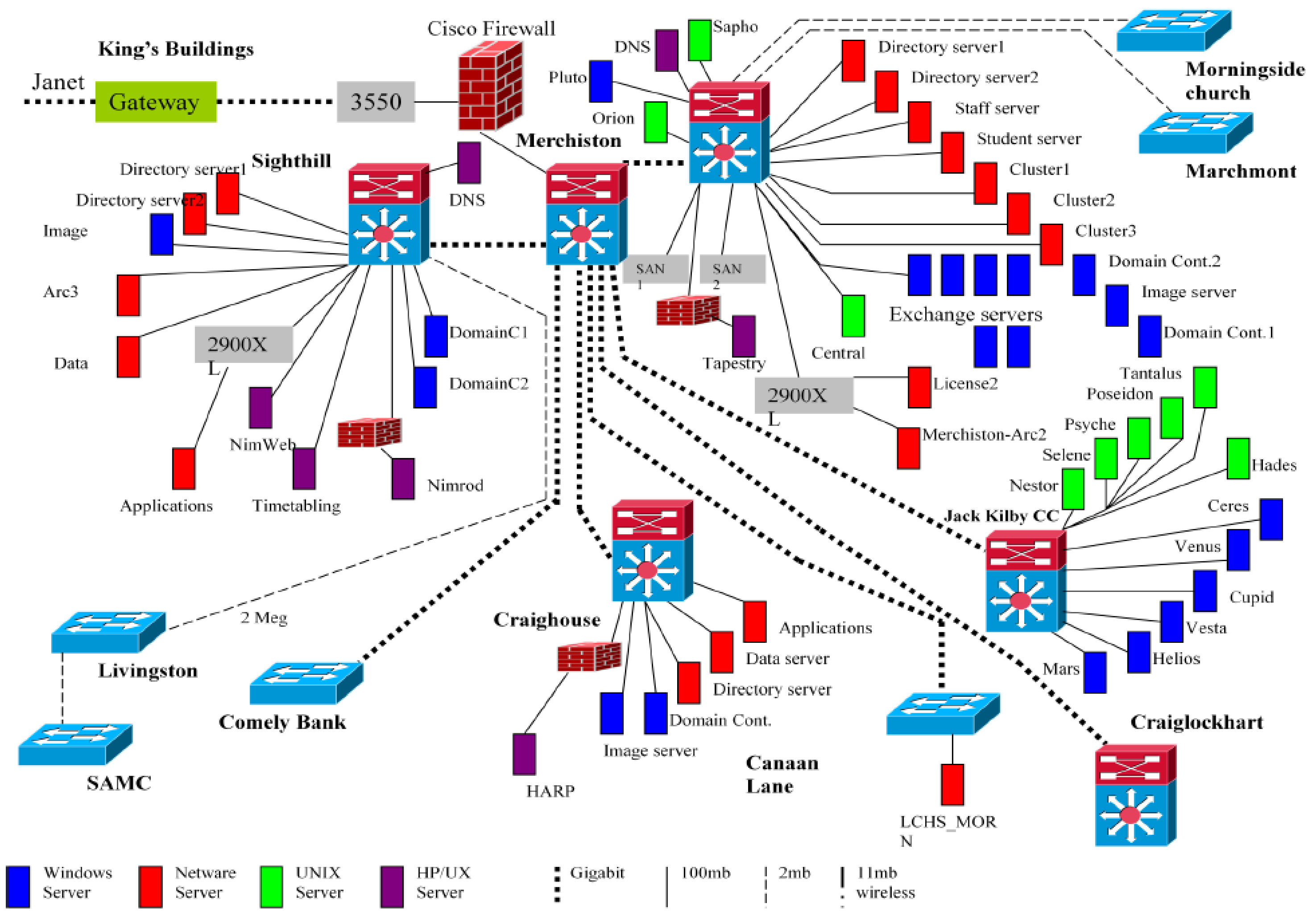
Mounting and Mapping

Distributed file systems



DFS users don’t know where the shares are, they just see a normal hierarchical structure in their file manager

Mounting and Mapping
Distributed file systems



This old snapshot of the ENU network reminds us why it is useful to see network shares as normal folders in a file manager. IT also illustrates how complex organisational networks can become.

CSN08601

End of Lecture

Questions?

Tutorial:	Types of OS
Lab:	Multitasking