BSc (Hons) Computer Science

University of Portsmouth

Second Year

Operating Systems and Internetworking

M30233

Semester 1

Hugh Baldwin

hugh.baldwin@myport.ac.uk

Contents

Part I

Operating Systems

Operating Systems Cheatsheet

Term	Definition
Kernel Mode	The CPU mode in which all assembly instructions can be used. This mode is usually used by the Operating System
User Mode	The CPU mode in which some assembly instructions cannot be used, such as IN and OUT. This mode is usually used by Application Software
System Software	The operating system and related utilities that control the computer, and provide essential functionality, such as keyboard and mouse input handling
Application Software	Programs that allow the user to perform specific funtions with a computer, such as word processing or internet browsing
Interrupts	A signal sent to the CPU, usually from an I/O device, that requests the attention of the CPU to handle an urgent operation

Lecture - Introduction

13:00 26/09/23 Tamer Elboghdadly

Operating Systems

- The Operating System sits inbetween the hardware and application software
- It usually is not the actual GUI, but provides functionality for the applications which implement it
- OSes typically provide abstractions for applications so that they can run on different hardware

User and Kernel Mode

- User mode
  - The programs that a user directly interacts with
  - Uses an API to access hardware, rather than having direct access
- Kernel mode
  - The programs that run the operating system
  - Has direct access to hardware

System and Application Software

- Application software
  - Programs that allow a user to perform a task
  - Requires the support of system software to run
- System software
  - Software directly related to the operating system
  - Manages the boot process
  - Hardware drivers
  - File system management

The main functions of an Operating System

- Resource management
  - Manages the memory, CPU and other hardware to allow multiple programs to run concurrently
  - Handles requests from applications to allocate more resources
- ”Extended machine”
  - Handles reading and writing to control registers, handlling interrupts, etc
  - Provides higher level APIs for other software to interact with the hardware

CPU Organisation

Registers

Name	Use	Description
EAX	Accumulator	The default register for many addition and multiplication instructions
EBX	Base	Stores the base address during memory addressing.
ECX	Count	The default counter for repeat (REP) prefix instructions and LOOP instructions.
EDX	Data	Used for multiply and divide operations
ESI	Source Index	Store source index
EDI	Destination Index	Store destination index
EBP	Base Pointer	Mainly helps in referencing the parameter variables passed to a subroutine.
ESP	Stack Pointer	Provides the offset value within the program stack.

Figure 2.1: General purpose registers in an x86 Intel CPU

- There are also special purpose registers, such as the Program Counter (PC) and the Program Status Word
- The Program Status Word sets the mode in which the CPU is operating

Assembly Language

- Assembly Language is the lowest-level programming language before Machine Code. It is slightly abstracted from machine code and uses neumonic symbols to represent instructions
- For example:
  - `MOV EBX, EAX`
  - This copies the value in EAX into the EBX register
  - `ADD EBX, 4`
  - This adds the value 4 to the EBX register
  - This is equivalent to $\mathrm{b} = \mathrm{a} + 4$
- The MOV instruction can also be used to move values between registers and the main memory
- I/O devices such as hard disks have a set of ports, which can be accessed to control the device and transfer data
- The special instructions IN and OUT are used to read from or write to ports
- For example:
  - `IN EAX, 368`
  - This copies the value from the port 368 into the EAX register
- The IN and OUT instructions can only be used when the CPU is running in Kernel Mode

User and Kernel Mode

- CPUs support running in two different modes: Kernel mode and User mode
- When running in User mode, some instructions (such as IN and OUT) cannot be used
- Typically, the Kernel of an operating system will run mostly or entirely in Kernel Mode

Interrupts

- When an external device needs to gain the attention of the CPU, it sends an interrupt
- A couple of examples are as follows
  - When a disk has requested data in it's buffer
  - When the key on an old PS/2 keyboard is pressed
- When the CPU recieves an interrupt, it must abandon whatever it is currently doing and run an Interrupt Handler routine
- Interrupt Handlers run in Kernel mode
- The CPU places whatever resources it was using onto the stack and executes the interrupt handler. Once the handler has finished, it returns to the execution path stored on the stack

Lecture - Concurrency

13:00 03/10/23 Tamer Elboghdadly

- A sequential system is one where multiple threads are exectued to completion, one after the other
- A concurrent system is one where multiple threads are exectued at the same time
- A parallel system is one where multiple threads are executed in parallel, usually on multiple hardware threads
- There are several issues that may arise from concurrency, especially in cases where threads of the same process are run on multiple processors in parallel
  - A thread is a sequence of instructions defined by a program or part there of
  - A process has one or more threads, each of which may be run in parallel
  - Threads within a single process can share resources with each other, but have their own thread of execution
- When multiple threads are run concurrently, the result is usually non-deterministic
  - This means that it is impossible to predict the order in which the instructions will run
  - This can lead to a program giving completely different results depending upon which order the instructions run
- Non-determinism can also lead to race conditions if multiple threads access the same variable
  - A race condition can lead to counting errors, data corruption or a complete deadlock of the program, depending upon which order the instructions execute
  - A specific type of race condition is interference, which is when multiple threads operate on the same variable, and interfere with each others results, such as two threads adding to a value, and the second thread overwriting the value of the first thread
- One method of avoiding non-determinism is to avoid threads sharing any variables at all. This works perfectly for some applications, but is very restrictive and can greatly reduce the efficiency of certain algorithms
- Any part of a program which requires using shared variables is known as the critical section, which must not be run at the same time as the critical section of any other thread

Lecture - Mutual Exclusion

13:00 10/10/23 Tamer Elboghdadly

- Techniques to ensure critical sections do not execute concurrently
- An atomic instruction or code fragment is a piece of code that executes without interruption, meaning that other threads cannot interfere

Naïve ME Solutions

Locks

- This works by using a single boolean variable to determine if the variables are locked by a thread or not
- When the variables are locked, in theory only one thread has access to them
- However, because the threads are running in parallel, there is a chance that multiple threads will read the lock variable as false simultaniously, resulting in two threads entering the critical section at once
- Therefore, this does not achieve Mutual Exclusion

Taking Turns

- This works by using a single variable to store which thread should get the next turn
- Each thread sits in a while loop until its their turn to enter the critical section. After finishing it's critical section, the thread assigns the variable to the next thread
- While this does guarantee Mutual Exclusion, it doesn't guarantee progress, as if any of the threads stalls at any point, all of the other threads will be left waiting for their turn

Interested Flags

- This works similarly to a lock flag, but rather than having only one lock, each thread is assigned an interested variable
- When a thread wants to enter the critical section, it sets it's own interested flag to true, and then waits until all other interested flags are false
- However, it is possible for both threads to set their interested flags at the same time, and therefore both enter a while loop
- Once again, this does provide Mutual Exclusion, but doesn't guarantee progress

Peterson's Algorithm

- This method combines interested flags with taking turns
- Each thread has an interested flag, and when it wants to enter the critical section, it sets it's interested flag, and then foregoes it's turn, assigning it to the other thread
- It then waits for the other thread to take it's turn, or in the case that neither thread is in the critical section, the other thread will reach the critical section and then forego it's turn
- This algorithm does work in theory and practice, but there's no way to generalise it for an arbitrary number of threads. It's very easy to implement for just 2 threads, but it gets exponentially harder as more threads are added

Practical Solutions

Hardware Test and Set

- This is an atomic instruction built into the instruction set of most modern processors
- An atomic instruction is one which performs a complex action, but acting like a single instruction
- That means that it's impossible for more than one thread to write using an atomic instruction at a time
- This basically allows us to use a lock variable, but without causing another race condition
- However, this is not really suitable in a multitasking environment, as it wastes a lot of CPU cycles continuously checking if the lock is available

Semaphores

- A semaphore is an integer variable that can be accessed by any thread, but only using two methods - P and V
- P decreases the value of the semaphore by one, but can never go below zero
- V increases the vaule of the semaphore by one
- If P failes to decrease the value, the thread calling it blocks until another thread increases it by one, allowing it to exit the block
- Since the semaphore starts at one, the first thread to reach the critical section decreases it by one and enters the critical section
- If another thread reaches the critical section, it has to block until the first thread has exited the critical section and increases the semaphore
- When using as a method for mutual exclusion, a binary semaphore is usually used (as the name suggests, this can only be 1 or 0)

Lecture - Synchronisation and Deadlock

13:00 17/10/23 Tamer Elboghdadly

Synchronisation

- Mutual Exclusion is just one type of synchronisation, but there are others, such as:
  - Messaging - Threads can send messages to each other. The recieving thread cannot continue until it gets a message from the sending thread
  - Join synchronisation - The parent thread can wait for the child thread to terminate
  - Barrier synchronisation - All threads must wait for all other threads to reach a barrier operation before they can continue
- Semaphores can be used to achieve multiple types of synchronisation
  - When using for Mutual Exclusion, the semaphore usually starts at 1
  - It could start at 0, and when a thread completes an operation, it raises the semaphore using V, allowing another thread to know it has completed an operation
  - This would allow you to guarantee the order in which the operations are run, rather than just that they don't run at the same time

Deadlock

- The resources of a computer can only be accessed by one thread at a time
- Mutual Exclusion is one example of this, as only one thread can access the areas in memory belonging to a data structre at a time
- Other examples are devices such as a printer or scanner
- A situation which could cause a deadlock is as follows
  - Two threads, A and B, need access to resources R and S
  - They both need access to the resources, but attempt to acquire them in different orders
  - Thread A acquires R and B acquires S
  - Thread A now attempts to acquire S, and B attempts to acquire R
  - Each thread is waiting for the other thread to release the resource
  - This causes a complete deadlock as neither thread will finish and release the resource

Deadlock Modelling

- Resource deadlock can be modelled using a resource allocation graph
- This shows
  - Which processes are requesting which resources

  - Which resources have been assigned to which processes
- If this graph contains a cycle, there will be a deadlock. If not, there won't be a deadlock

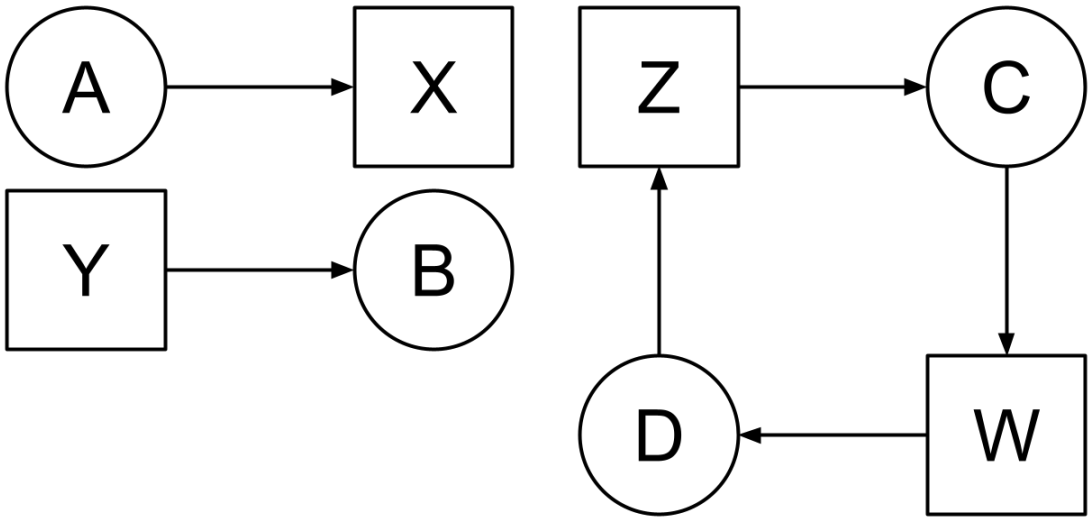

Figure 5.1: A resource allocation graph - Process A is waiting for resource X; Resource Y is assigned to process B; Process C and D are in a deadlock, waiting for resources Z and W

Deadlock Detection and Recovery

- This is a process which allows the system to enter a deadlocked state
- A deadlock detection algorithm is run periodically
- If a deadlock is detected by finding a cycle in the resource allocation graph, process that make up the cycle are randomly killed until the deadlock is resolved
- Any remaining processes are allowed to use the requested resources
- This is very inefficient and may result in data loss depending upon the type of processes being killed
- It is commonly used in relational databases, as any transactions which create a deadlock can be rolled back, allowing the system to continue

Deadlock Avoidance

- It would be better if the system never entered a deadlock state
- This works as follows:
  - The system analyses the resource usage of all threads of the system
  - This allows it to predict when threads will need access to a specific resource
  - If two threads need access to two resources in confliting orders, the system will pause one thread until the other is finished with both resources
  - This avoids entering an unsafe execution path in which a deadlock is guaranteed
- One such deadlock avoidance algorithm is Dijkstra's Banker's algorithm
  - All processes declare which resources they may need access to, and how much of said resources they may need
  - The algorithm then keeps track of the current allocation for each process

- When a request for more resources is received, it pretends to honor the request, before attempting to fulfill the needs of all other processes
- If it is unsafe to do so, deny the request, else grant it

Lecture - Processes and Scheduling

13:00 31/10/23 Tamer Elboghdadly

Processes

- A process is a collection of threads which belong to a program
- Several copies of the same program can be running at once, using separate processes
- A process has a space in memory which is split into multiple sections
  - Program code - The actual code for the program
  - Data section - Global and static variables
  - Heap - Memory which can be dynamically allocated during runtime
  - Stack - Memory used temporarily during runtime when functions are called
- Part of this memory is known as the execution state, which includes the program counter, stack and data section

Process Control Blocks

- The operating system maintains a process table
- Within this process table, each process has a Process Control Block
- For processes which are not currently running, the PCB contains
  - The contents of all registers at the time the process was last running
  - Pointers associated with memory management for the process
  - Any other information which may be needed to restore the process to a running state, exactly as it was before
  - It does not contain variables, as these are assumed to be in the processes' main memory block

Dispatcher

- The dispatcher actually gives control of the processor over to the process selected by the scheduler
- This involves
  - Stopping the currently running process
  - Storing it's information in the PCB associated with it
  - Re-loading the information in the new process' PCB
  - Switching between Kernel and User mode
  - Jumping to the correct point in memory to resume execution
- This set of tasks is collectively known as context switching

Scheduling

- The scheduler is responsible for selecting the next process to run on the processor
- To do this, it uses a scheduling algorithm, which takes into account the following:
  - When should processes be swapped?
  - Under what circumstance should the process be swapped? (Time used, if a process stalls, etc)
  - How long should threads be given the CPU?
  - Which order should processes run in?
- Processes which are ready to be run, but waiting for processor time are placed into the ready queue
- The scheduling algorithm is responsible for deciding when to swap the currently running process and the one at the start of the ready queue
- Typically, when a process is stopped, it is placed at the rear of the ready queue, unless it was stopped to wait for a resource
- There is usually also a waiting queue, where processes which are not yet ready to continue go to wait for whatever is stalling them

System Calls

- Since user processes are run in user mode, they do not have direct access to hardware or other resources
- Therefore, the must user so-called system calls to request that the operating system performs a function on their behalf
- System calls are usually implemented using software interrupts (sometimes called traps) so that the running process is immediately suspended while the operating system runs the function
- System calls include the following functions:
  - Creating a new process
  - Waiting for another process to exit
  - Exiting the current process
  - Creating or opening a file
  - Closing a file
  - Deleting an existing file
  - Reading from or writing to a file
  - Creating or removing a directory
  - Setting the working directory
  - etc

Lecture - Interprocess Communication

13:00 07/11/23 Tamer Elboghdadly

- Threads can pass information between them using shared variables, as they use the same address space
- Since each process has it's own address space, they can't share variables with other processes
- This means that a specific mechanism to allow communication between processes, which is known as Interprocess Communication or IPC
- There are many different methods and implementations of IPC, which are designed for different purposes, such as communicating between threads on different computers
  - Pipes and Sockets (Stream oriented)
  - Shared Memory (Memory oriented)
  - Message Passing (Message oriented)
  - Remote Procedure Call (Procedure oriented)

Single System IPC

- Communication between two or more processes within the same computer

Pipes

- One of the simplest forms of IPC
- Commonly used in Unix-based systems, such as Liunx
- A pipe has an input and an output
  - A stream of bytes from one process is written to the output
  - The same stream of bytes can then be read from the input to the second process
- The pipe is a one-way file, which can be read only by one process and written by the other
- The kernel buffers data between the two threads, specifically using a bounded buffer (typically 64KB in size) with blocking (when a process is reading or writing, the other process must wait)
- The pipe is created by a parent process, and can be used to communicate one-way between two child processes
  - The parent process is usually the user's shell, with which they can pipe information from one command into another

Shared Files

- Multiple process access one file to communicate
- The file requires a lock to ensure only one process is accessing it at a time

Multi-System IPC

- Communication between two or more processes on two or more computers

Message Passing

- Similar to a pipe, but the data is split into messages of a certain size
- It uses connectionless messaging
- This can cause issues as there is no guarantee that a message was recieved, or that they were recieved in the correct order
- This can be solved using a buffer on the sender side
  - Zero-capacity buffer - The Sender will always wait for the reciever to acknowledge the message
  - Finite-capacity buffer - The Sender will continue to send messages until the buffer is full, messages are removed from the buffer when the reciever acknowledges them
  - Infinite-capacity buffer - The Sender will continue to send messages forever, regardless of acknowledgement
- There are a few implementations available, but the standard is the Message Passing Interface, which is used by open source projects, as well as hardware vendors

Sockets

- Sockets are a form of message passing more similar to pipes
- They can connect processes on different computers, including different operating systems or over the internet
- They are more similar to pipes as they can be used in UDP mode, allowing for stream communication rather than the message-based communication of TCP
- The reciever creates a socket with a certain port, and the sender opens (connects to) the socket based upon it's IP address and port
- If the connection is accepted, the sender can start to send a stream of data over the network
- This means that it is a connection-based protocol

Remote Procedure Call

- A server exposes a set of procedures (functions) which other computers are able to call
- A client is then able to call a remote procedure on a server, and recieve the results of the procedure
- This allows for low-powered clients to run high-power functions on a remote server
- It also makes it easier to write client/server based programs, as the programmer only needs to write the application itself, and not the protocol that the application will use for communication
- There are some issues with RPC
  - If the two computers use a different architecture or operating system, they may need to convert results between different formats
  - This requires External Data Representation (XDR), which is a format both computers are able to use, which is converted to/from on both ends
- There are several implementations

- CORBA, Common Object Request Broker Architecture
- RMI, Java Remote Method Invokation
- SOAP, Simple Object Access Protocol
- WCF, Windows Communication Foundataion

Lecture - File Systems

13:00 14/11/23 Tamer Elboghdadly

- There are many methods of physically storing a file
- Operating Systems reduce the complexity of programs by abstracting the way that each file is accessed
- Files typically have some content, and metadata
- The main functions that can be performed with a file are
  - Creating a new file
  - Rename an existing file
  - Deleting an existing file
  - Opening an existing file into memory
  - Closing a file in memory
  - Read data from a file in memory
  - Write data to a file in memory
  - Seek to a specific position in a file in memory
  - Get information from the metadata
  - Set information in the metadata
- Metadata usually contains information such as
  - File Type
  - File Size
  - Ownership and Permissions
  - Time and Date of last modification
  - Location of the file

Directories

- A directory is a special type of file which contains a list of references to other files
- Entries in a directory always point to another file, which can either be a regular file or a directory
- Directories contained within another directory are known as child directories
- Referenced files or directories are considered to be contained within the directory

The Root Directory

- Every file system has a root directory
  - In UNIX-based systems, this is known as ”/”
  - In Windows, this is known as ”\”, but also depends upon the drive letter, so the root of the boot drive would be ”C:\”
- Neither files nor directories contain an absolute path to themselves or their children
- This means that you have to traverse the file system from the root to get to a file
- To locate the file ”/home/hugh/cs-notes/README.md”
  - Go to the root directory, /, and find the directory ”home”
  - Go to ”home” and find the directory ”hugh”
  - Go to ”hugh” and find the directory ”cs-notes”
  - Go to ”cs-notes” and find the file ”README.md”

Blocks and Clusters

- Most file systems store file contents in the unit of storage of the drive it will be stored on
- In the case of a HDD, a unit may consist of one or more consecutive sectors
  - In UNIX-based systems, these units are known as blocks
  - In Windwos, these units are known as clusters
- Either way, a the size of a block is usually some binary multiple of the physical sector size of the disk (e.g. 1KB, 2KB, 4KB, etc)
- Each file is allocated a whole number of blocks to store it’s content

Types of File System

- On the user level, most file systems look very similar
- This is because the operating system abstracts most of the complexity of a file system, and usually supports more than one file system
- Different file systems exist, and the one which should be used depends on many factors, such as
  - The type and characteristics of the physical storage
  - The operating system in use
  - The version of the operating system
  - The preference of the person who setup the file system
- A few examples are
  - FAT (File Allocation Table) - Legacy DOS file system, but still widely used in it’s 32-bit variant for smaller devices
  - NTFS (New Technology File System) - Default boot partition format for modern Windows versions, much more reliable and configurable than FAT
  - Ext (Extended file system) Family - Default format for most modern Linux OSes
  - HFS+ (Hierarchical File System Plus) - Default format in modern versions of macOS

File Systems in Detail

FAT (File Allocation Table)

- The original file system used for MS-DOS, circa 1980
- Designed by Bill Gates and used in Microsoft Basic
- Updated versions of FAT were still used by Windows up until Windows Millenium Edition
- Windows ME replaced FAT with NTFS
- Updated versions, such as FAT32, are still used on smaller storage devices as it is usable in all modern operating systems, so allows easy file transfer between them
- A FAT32 directory is a 32 byte string and contains the file name, file metadata and a reference to the first cluster of data
- Since the directory only contains a reference to the first cluster, all remaining clusters of a file are located in a separate data structure, the File Allocation Table
- This uses a linked list, where the links are stored on their own in a dedicated area of the disk
- The FAT entries contain only a reference to a single cluster in the data area
- They act as a data structure known as a cluster chain, but for all intents and purposes, it's just a linked list

Ext (Extended File System) Family

- Used by various UNIX-based systems, such as Linux
- The current default format for Linux systems is Ext4
- Everything in UNIX is a file, including physical devices such as USB peripherals, DVD drives, and even displays
- Allocation follows an inode (index node) approach
  - This means that any block can be in an allocated or unallocated state
  - Every file or directory has exactly one inode
  - The inode is a small (typically 128 byte) data structure containing metadata and block pointers for the contents of the file
  - Within the file system, the inode number is the main method of refering to a file or directory
- Since directories are also files, they have their own inode
- This inode contains a list of names and inode numbers for the file and directories inside the directory, and basically nothing else
- Every directory also contains two files, "." and ".." which are a reference to the directory itself, and the previous directory respectively

NTFS (New Technology File System)

- Introduced and used for versions of Windows after NT (New Technology)
- This includes all modern versions of Windows, such as 10 and 11
- A much more complex file system than FAT, which has native support for
  - Long unicode file names
  - Ownership and Permission structures
  - Encryption
  - Journaling
  - Etc
- The primary metadata storage is in the Master File Table (MFT)
- It contains at least one entry, or file record, for every file and directory
- A file record is similar to an inode from Ext
- Every file record in the MFT has a fixed size, which is configurable in the boot sector but is usually left at the default of 1KB
- The value of any attribute can be resient or non-resident
  - Attributes with a short, fixed length will usually be resident, and therefore stored entirely within the file record
  - Attributes with large values (including $DATA attributes which store the actual file content) will usually be non-resident, and therefore stored outside the file record with only references to the actual location
- Unlike Ext where inodes are stored in a separate space, the entire NTFS file system is able to store clusters
- This means that the MFT is paradoxically stored within a file itself
- It is also not stored in any specific physical location, and can be spread around the disk

Part II

Internetworking

Lecture - Network Services

09:00 25/09/23 Athanasios Paraskelidis

DHCP

- DHCP stands for Dynamic Host Configuration Protocol
- The main motivation of DHCP was to provide a set of configuration parameters to automatically configure new devices as they are added to the network. The main parameters are as follows:
  - IP address
  - Gateway address
  - Subnet mask
  - DNS server address
- Before DHCP, either devices were configured manually, or the bootstrap protocol (BOOTP) was used
- The main improvements over BOOTP are:
  - Support for temporary allocation of IP addresses
  - DHCP clients can automatically discover the local DHCP server
  - Once the server is setup, there is almost 0 human interaction unless something goes wrong
  - Still compatible with BOOTP clients
- A lease is the length of time that an allocated IP address can be used before either a new address is needed, or a request to continue using the current address needs to be approved
- An IP address can be released by the client if it is nolonger needed, e.g. the device shuts down or nolonger needs to communicate on the network
- Advantages
  - Saves manually configuring every single device
  - Ability to move to a different network without having to reset any network settings on the device
  - Allows more efficient use of the IP space, as inactive devices do not need to keep a lease on an address
- Disadvantages
  - DHCP uses UDP to configure devices, so the communication is unreliable and insecure
  - Possibly allows unauthorised clients, but this can be avoided using MAC address white/blacklisting
  - Potential for malicious DHCP servers setup on a network that provide incorrect network settings

DNS

- DNS stands for Domain Name System
- This is the system which devices use to convert the human-readable domain names into IP addresses which computers can use to communicate to the server
- DNS is a globally distributed mapping database between domain names and IP addresses
- There are 3 main components
  - A **name space**
  - **Servers** that make the name space available
  - **Resolvers** which make the request from clients to servers
- As DNS is globally distributed, some data is maintained locally, but also retrievable globally
  - No single server holds all DNS records
- A DNS lookup can be performed my any device
  - Remote DNS data is usually cached locally to improve performance
- DNS has 'loose coherency'
  - The database is always internally consistent
  - Each version of the database has a serial number, which is incremented every time the database changes
  - Changes to the master copy of the database are replicated to secondary servers regularly, depending upon the timing set by the zone administrator
  - Cached data expires depending upon timing set by the zone administrator
- There is no limit to the size of the database
  - Having a very large number of records on one server would decrease performance
  - Therefore, the database is spread across many servers around the internet
- There is no limit to the number of queries which can be made at any time, or by a single user
- Queries are usually distributed between multiple DNS servers as well as local caches
  - e.g. nameserver1 and 2
- Clients can query and use the data from any server, primary or secondary
- Clients will typically have their own local cache of more frequently accessed records
- DNS uses both TCP and UDP
  - TCP is used for communication between servers, for example when replicating records from a primary to secondary server
  - UDP is used for communication between clients and servers
- The database can be updated dynamically
  - Add, delete or modify any record on the server
  - These only need to be performed on the main server, as the secondary servers will replicate the changes over time

- There are two main types of servers
  - Authoritative
    - Primary server - Where data is added and modified
    - Secondary server - Servers which replicate the primary server to share the load
  - Non-authoritative
    - Caching servers - temporarily retain records from authoritative servers to improve resolving performance and reduce load on authoritative servers
- Domains can be resolved either Iteratively or Recursively
  - Iteratively
    - The client's domain name resolver starts by querying the root nameserver
    - The root nameserver responds with the address of the nameserver on the next level down
    - The domain name resolver then queries this nameserver, and so on until the nameserver with the full domain is found
  - Recursively
    - The client's domain name resolver queries the root nameserver
    - The root nameserver itself queries the nameserver on the next level down
    - This process repeats until the nameserver with the full address is found
    - The IP address of the domain is then passed back up the chain and to the requester

Domain Names

- A domain name is the sequence of labels from a node to the root, separated by dots
  - e.g. port.ac.uk has 3 labels
    - port
    - ac
    - uk
  - There can be up to 127 labels in a domain name
  - But there can only be 255 characters in the domain overall
- The root domain or Top Level Domain (TLD) of a domain is the final label, e.g. the TLD of port.ac.uk would be uk
- A subdomain is any domain which resides below the TLD. In the case of port.ac.uk,
  - uk is the TLD
  - ac is the Second Level Domain
  - port is the actual domain name
- Name servers store information about domains in units called zones
  - Each zone usually corresponds to a subdomain, for example the .uk TLD has many sub-zones, such as .ac.uk, .gov.uk and .co.uk

Lecture - IP Addresses and Subnetting I

09:00 02/10/23 Athanasios Paraskelidis

- Layer 3 networking is the layer that handles routing data between networks, such as a local and wide area network
- Layer 3 functionality is spread over the network - in routers as well as client devices

Internet Protocol

- Connectionless
- Best effort delivery
  - Unreliable
  - Packets may not arrive at all, or in the wrong order
  - There is no built-in error handling other than error checking
- When transmitting
  - Encapsulates data from the transport layer in datagrams
  - Adds header data to the datagrams (source and destination addresses, time to live, etc)
  - Apply the routing algorithm once on the internet
  - Will continue to route the packet until either it reaches the target network card, or exceedes it's time to live
- When receiving
  - Check the validity of all datagrams
  - Reads the header and checks if forwarding is needed
    - If the destination address is on the LAN, send the datagram to the destination
    - If it isn't, relay it to the next router on it's route

IPv4 & Subnetting

- 32-bit string
  - 4*8 bit integers (1 byte, or 1 octet)
  - Each byte can be 0-255
  - Intended to be human-readable
  - Allows for 2^{32} or $4,294,967,296$ theoretical addresses, less in practice
- The IP address is split into two fields
  - The Network Prefix
  - The Host ID

  - Any devices with the same network prefix are on one network
  - Two devices cannot have the same Host ID
- The network prefix can be 8, 16 or 24 bits in length, leaving 24, 16 or 8 bits for Host IDs
- There are 3 main classes of IP addresses
  - Class A: Network prefix always starts with 0, followed by 7 bits, and 24 bits for the Host ID (1.X.X.X - 126.X.X.X)
  - Class B: Network prefix always starts with 10, followed by 14 bits, and 16 bits for the Host ID (128.0.X.X.X - 191.255.X.X)
  - Class C: Network prefix always starts with 110, followed by 21 bits, and 8 bits for the Host ID (192.0.0.X - 223.255.255.X)
- There are also 2 more reserved classes
  - Class D: Network prefix starts with 1110, followed by a 28 bit multicast ID (224.X.X.X - 239.X.X.X)
  - Class E: Network prefix starts with 1111, followed by 28 bits reserved for experimentation (240.X.X.X - 255.X.X.X)
- The Class A network 127.X.X.X is reserved for loopback addresses, every computer has 127.0.0.1 as a loopback address, some computers may have more than 1
- Each class of network has 2 less usable host addresses than would be expected, e.g.
  - A class A network provides $2^{24} - 2$ host addresses $(16,777,214)$
  - A class B network provides $2^{16} - 2$ host addresses $(65,634)$
  - A class C network provides $2^{8} - 2$ host addresses (254)
- This is because X.X.X.0 is the network address, which cannot be assigned to a computer, and X.X.X.255 is used as the network's broadcast address
- There are 3 ranges of IP addresses that can be used by anyone on a local network, one in class A, B and C
  - Class A: 10.0.0.0 - 10.255.255.255
  - Class B: 172.16.0.0 - 172.32.255.255
  - Class C: 192.168.0.0 - 192.168.255.255
- In order to connect to devices on the internet, a NAT (Network Address Translation) service is needed on the network (usually on the router)

Subnet Masks

- A subnet mask tells the computer which parts of the IP address are reserved for the network prefix
- The default subnet mask for each class is as follows:
  - Class A: 255.0.0.0
  - Class B: 255.255.0.0
  - Class C: 255.255.255.0
- The subnet mask also determines how many devices can be on the network at a time, as it sets how many bits are left for the Host ID
- A network address is the network prefix, with all host bits set to 0, e.g.

  - The network address of 10.128.47.87 is 10.0.0.0
  - The network address of 192.168.10.104 is 192.168.10.0

- But why do we need subnets?
  - If you are assigned a Class B network address (e.g. 128.147.0.0), you have up to 65534 host addresses available
  - It's not possible to effectively manage that many clients on a single physical network, and would be a waste of most of those IP addresses
  - Therefore, we should split the network into subnets which each have a much more managable number of devices
  - e.g. we could use the 3rd byte of the address as a subnet ID and have 256 subnets, each with 254 hosts
  - Since the subnet ID is configurable, we could also use 4 bits for the subnet ID, allowing 16 subnets with 4094 hosts each
- There is an alternative way to write subnet masks, which is called the slash notation
  - This is noted by adding the number of bits reserved for the network address after a slash at the end of the IP
  - For example, you could say that you have a Class B network with the ID 148.197.0.0 and therefore subnet mask 255.255.0.0, or you could simply write it as 148.197.0.0/16
- This allows us to use custom length subnet masks when creating subnets
- For example, you may have a Class B address and want to have 32 subnets. To do this, you would need an additional 5 bits for a subnet ID, and therefore should use the subnet mask /21. This allows for 32 subnets with $2^{11} - 2$ or $2,046$ hosts each

Lecture - Variable Length Subnet Masks

09:00 16/10/23 Athanasios Paraskelidis

Subnetting with VLSM

- Using classful subnetting wastes a lot of IP addresses, so a new approach was needed - VLSM
- VLSM allows for more than one subnet mask within the same network, allowing for different subnets to be different sizes, based on their needs
- This also allows for much more efficient used of the IP address space
- Also allows route aggregation, meaning that less routing information is needed
- Links between routers are in and of themselves networks - although they only need enough IP addresses for the routers, so a direct link only needs 2 usable addresses
- The smallest subnet you can have is a /30 subnet, which has 4 IP addresses, with only 2 usable
  - This is usually used only for point-to-point links, such as a connection between 2 routers
- The routing protocol needs to support VLSM, by carrying extended network prefixes
  - Some routing protocols that support VLSM are: RIPv2, OSPF, EIGRP, BGP
- When using VLSM, we can completely ignore the standard classes of IP addresses, as long as we have been assigned the full address range
- When assigning subnets with VLSM, you should consider the largest subnet first, and then use the remaining space to create any additional subnets
- If needed, you can create sub-subnets within larger subnets to allow creating more, smaller subnets
- VLSM uses a longest match forwarding algorithm
- This means that it looks through the routing table to find the longest match
  - The longest match is the entry which reduces the size of the network as much as possible - find the smallest network to route to

Supernetting

- Supernetting is combining multiple small (usually class C) networks into one larger one
- This creates a single address space with more addresses
- You can only use supernetting if the networks you wish to combine are contiguous
- Convert the addresses into binary, and count from left to right the number of identical bits. This gives you the supernet mask for the network

- The supernet address will always be the network address of the first of the contiguous networks
- You need to make sure that the supernet mask does not allow you to address IPs outside of your contiguous networks

Lecture - Supernetting & Classless Inter-Domain Routing

09:00 30/10/23 Athanasios Paraskelidis

CIDR

- Also known as Supernetting
- Considered as a fundamental solution to reducing the size of routing tables
- Also used as a temporary solution to the lack of internet addresses in IPv4
- A Class C address is too small for most companies, so most requested a class B address, but in the process wasted lots of addresses, as they did not need as many addresses as it provided
- CIDR is similar to VLSM, but applies to the entire internet rather than one specific network
- For CIDR to work, the network prefix must always be specified
- Routers on the internet nolonger use classes, and only the network prefix is important
- Since the network prefix is important for routers to know, the only requirement for CIDR to work is for the routing protocols to forward network prefixes
- CIDR will only provide any routing advantages if topologically significant addresses are used (addresses which all reside within one routing area)

Supernetting

- Supernetting is combining multiple small (usually Class C) networks into a larger one to create a larger range of addresses
- The IP addresses you wish to supernet must be contiguous
- The exact opposite of subnetting, and reserves part of the network address to use as host addresses
- You must have control over enough networks to make up an entire bit (if you wanted to reserve 2 bits, you would need 4 contiguous addresses)
- Your supernet address becomes the first IP address, followed by the new subnet prefix

Lecture - Static and Dynamic Routing

09:00 06/11/23 Athanasios Paraskelidis

- Routing is forwarding packets from a source network to a desintation network
- The source and destination could be a network of any size - anywhere from a single computer to the entire internet
- Routing protocols are typically based around solving a graph, known as graph theory
- There are a number of important factors to consider
  - When should a packet be routed?
  - What is the best route to take?
  - What if the topology of the network changes?
  - What if the destination doesn't exist?

When should a packet be routed?

- Workstation A wants to send a packet to Workstation B
- Workstation A checks if Workstation B is on the same network, but checking it's routing table
- If Workstation B is on a different network, the packet is sent to Workstation A's default gateway, which then routes the packet to Workstation B

Static Routing

- Manually populated routing tables
- Not feasable to maintain in modern networks
- Can still be used for small, stable networks without redundant links
- Dynamic routing protocols used network resources to learn what they can route to
- Static routing can be combined with dynamic routing
- Static routing is still occasionally used for very secure network, as the administrator has complete control over routing

Routing Tables

Code	Network/Mask	AD/Metric	Next Hop	Interface
O	10.0.0.0/8	110/20	200.1.1.1	S0
O	172.16.0.0/16	100/15	200.1.1.1	S0
O	192.168.1.0/24	100/20	200.2.2.2	S1
C	210.1.1.4/30	0/0	Directly Connected	E0

- – Code - What method was used to discover the route
  - Network, Mask - The address and subnet mask of the destination network
  - Administrative Distance, Metric - Used to select the best route. AD depends on the network used, Metric measures the speed of the route
  - Next Hop - If destination network is not directly connected to the current router, what is the next router to send the packet through
  - Interface - Which network or serial port on the current router should be used to route the packet
- Routing tables only contain information about routable networks, not every host on them
- Once the packet reaches the destination network, the network gateway is responsible for getting the packet to it's final destination

Dynamic Routing

- Static routing is impossible to maintain in a network as large as the internet, so a new, automated routing solution was needed
- There are many dynamic routing protocols, but most have the following key features
  - They learn about the network over time
  - They are able to automatically update their own routing tables
- Dynamic routing can be used on a network of any size, but certain routing *protocols* are better suited to very large or very small networks
- The issue with having many different routing protocols is routers which don't share a protocol are unable to communicate

Routing Protocols

Convergence

- If the topology of the network changes, every router needs it's routing table updated
- The time it takes to accomplish this is known as the "Convergence Time"
- This is important as in the event that a network link or router fails, every other router needs to know
- If routers are not informed of topology changes, packets may be routed to inaccessable of failed routers, causing lost packets
- Protocols with a slower convergence time are typically less desirable as any failure will be more impactful
- Slow convergence can also cause *routing-loops*

Metrics

- There are multiple different factors which can influence the metric of any particular route
- Hop Count - The number of routers which have to be traversed to reach the destination network
- Path Length - A refinement of hop count, as it takes the cost of each individual hop into account
- Bandwidth - The speed of the links between routers
- Delay - The time to cross each link

- Load - The congestion on each link due to traffic
- Reliability - Based on the previous reliability of routers and links
- Not every protocol used all of the listed metrics

Types of Dynamic Routing Protocols

- Interior Gateway Protocols
  - Routes packets within large, autonomous networks
- Exterior Gatweay Protocols
  - Routes packets between autonomous networks, such as the a LAN and the Internet
- The vast majority of protocols fall under Interior Gateway Protocols

Distance-Vector Routing

- Routing information is recieved only from immediate neighbours
- Said information is sent between routers as a broadcast of 'update packets'
- Each router compares the information in the update with the existing information in it's routing table, and updates it accordingly
- These routers then pass the information on to their neighbours, and so on until the network is converged
- The metric used in a Distance-Vector protocol such as RIP (Routing Information Protocol) is the number of hops between the current router and the destination
- If multiple paths to a destination are discovered, only the one with the lower hop count is added to the routing table
- Each router only stores the next hop that packets need to take to get to the destination, and each router uses it's own routing table to determine where to send the packets
- This can lead to routing loops, as it's possible that routers with out-of-date information may route packets through a router with no connection to the destination, or only a connection through itself
- On networks with a very consistent delay between nodes, distance vector routing works well, as it picks the route having to pass through the least routers
- On unstable networks with unpredictable delay, distance vector would be sub-optimal as it does not take reliability or speed into account whatsoever

Link-State Routing

- Uses first-hand information to make descisions
- Routes based upon shortest-path-first, usually using Dijkstra's algorithm
- Routing information is transmitted using Link State Advertisement
- The data includes the state of all directly connected links
- This allows every router to have a complete map of the network topology, meaning that they are able to make more intelligent descisions about how to route packets

- As only the link state is sent between routers, the convergence speed is greatly improved
- Table updates are only triggered by the change of a link state, which minimises the use of bandwidth for routing updates, rather than actual traffic
- They also use multicast rather than broadcast, allowing the data to be sent only to specific routers, once again reducing the bandwidth used for routing updates
- It is also less prone to routing loops as each router can determine the entire route a packet should take, rather than only what the next node will be
- Some Link-State protocols are able to hold multiple routes for each destination, but when only one route can be stored, it should be the best path available
- In multipath routing protocols, it is possible to use multiple routes at a time, reducing the bandwidth used on each route, known as load balancing or multiplexing
- Using load balancing or multiplexing can improve both the performance and reliability of the network, as each individual link is not using all of it's bandwidth for a single connection

Hierarchical Routing

- When using a link-state protocol, it is also possible to implement hierarchical routing
- This allows the configuration of a hierarchy of routers, and only routers on the same level or area will be updated at once
- Updates are kept within an area, but if a destination is connected to a different area, specific updates can be communicated to the other area
- This allows for better management of network resources

Route Summarisation

- Route summarisation is defining a single path to multiple subnets
- This allows the routing table to be much smaller, as splitting the data into subnets is done only on the destination router, rather than every router along the way
- By reducing the size of the routing table, you also reduce
  - Bandwidth use
  - Memory use
  - CPU use
  - Routing lookup time
- Summaristaion can be used on the address assignment level, or the organisation level
- Auto-summarisation is available in some routing protocols, but can usually be disabled for specific use cases

Lecture - Routing Information Protocol (RIP)

09:00 13/11/23 Athanasios Paraskelidis

- A very old protocol from the 80s
- Has support for classless networks
- It uses a distance vector algorithm to determine the best path, specifically a variant of the Bellman-Ford algorithm
- Therefore, the main metric it uses is hop count
- It also means that RIP does not consider bandwidth, load or reliability of each hop

Versions of RIP

Version 1

- Class-based routing
- Limited support for classless routing
- Shouldn't be in use any more, as it is very outdated and insecure

Version 2

- Auto-summarisation (route aggregation or CIDR) is on by default
- Multicast is used rather than broadcast when communicating with other routers
  - Routers listen on 224.0.0.9 for updates from other routers, reducing overhead on the network for devices uninterested in RIP updates
- Supports simple password authentication. Not very secure but better than nothing

Advertising Routes

- When the router is booted, a new routing table is created and populated with only directly connected networks, and any statically assigned routes
- After initialisation, the router sends it's routing table to every immediate neighbour on every interface
- It continues to send it's routing table to every neighbour every 30 seconds, even if there is no updated information
- An intentional random delay can be added so that each router sends updates at slightly different times, as to not overwhelm the network
- It is possible to configure a RIP router to only send updates when necessary

Learning Routes

- Once a new route is discovered by recieving updates from other routers, it is added to the routing table
- The router advertises it's new route to all other routers
- Every router learns every path to every network
- Changes or failures are also advertised to neighbouring routers
- The time it takes for this information to spread to all routers is known as convergence time
- Once the information is spread to all routers, the network is considered to be converged
- Every router continues to broadcast their routing protocol every 30 seconds

The Routing Table

- The routing table contains lots of information about different routes that are known by the router
- The first column contains a letter
  - A C means that the route is directly connected to this router
  - An R means that the route was learned using RIP updates
- The second column contains the destination network or host
- The third column contains the network mask of the destination
- The fourth column contains the administritive cost (120 for RIP and 110 for OSPF) and routing cost (in the case of RIP, the number of hops)
- The final column contains the next hop, including it's address, the port to be used, and when it was last updated

Routing Timers

- RIP has 3 timers it uses to update it's routing tables
- Update Timer
  - Determines how often to broadcast the routing table
  - Defaults to 30 seconds unless configured otherwise
- Invalid Timer
  - If a route is not in any updated tables within this time, the route is assumed to be no longer available
  - Defaults to 180 seconds unless configured otherwise
- Flush Timer
  - If a route is still not heard in any updated tables within this time, after the invalid timer, neighbours are informed that the route should be flushed (removed)
  - Defaults to 90 seconds unless configured otherwise
- The invalid and flush timer are there to reduce the frequency that a route is removed and then added again when an interface goes down for a short time

Preventing Routing Loops

- Only maintain the best routes with the lowest metric within your routing table
- Timeout directly connected routes immediately upon failure
- Other methods include
  - Route Poisoning
  - Split Horizon
  - Triggered Updates
  - Poison Reverse
  - Maximum of 15 hops

Advantages

- Widely supported on a variety of hardware
- Still usable in smaller networks
- Reasonable ease of configuration
- Good for networks with few to no redundant paths
- Good for networks where each link has similar speed and latency

Disadvantages

- The maximum hop count is 15, and so it is unsuitable for large networks
  - If the cost for a route is 16, it means the destination is unreachable via RIP
- Prone to routing loops due to slow convergence
- Uses a significant portion of bandwidth for itself (Known as a chatty protocol) as it broacasts the entire routing protocol over the network periodically
- Shortest path is not necessarily the fastest

Preventing Routing Loops

Split Horizon

- Used in routing protocols other than RIP
- Never advertise a route to the router it was learned from
- This helps to prevent loops caused by a dead link that was learned from immediate neighbours

Hold-Down Timer

- The default value for the hold down-timer is 180 seconds
- Once an entry in the routing table is updated, any updates should be ignored until the hold-down timer expires
- This helps to improve the routing stability, as the routing table won't be constantly changing
- In the event of an intermittent failure of one router, it is possible that the network traffic would be significantly disrupted and cause massive overhead
- This is known as route flapping, which can be reduced by the hold-down timer, as it means that the router won't be re-added to the routing table for at least 180 seconds

Triggered Updates

- A trigger update is a complementary rule to route poisoning
- A router that knows of a failed route will not wait for the update timer to inform neighbours of the failure
- This update only includes information about the poisoned routes
- This increases the speed at which poisoned routes converge