50005 - Networks and Communications - Lecture 1

Oliver Killane

18/01/22

Lecture Recording

Lecture recording is available here

Introduction

Module Timetable

Week	Topic
1	Introduction
2	Basic Concepts
3	Application Layer
4	Transport Layer
5	Security
6	Network Layer
7	Practical Applications
8	Data Link Layer
9	Physical Layer & Coding & Network Simulation
10	The Future & Revision

Coursework runs from Monday 7/2 → Friday 25/2.

Exam occurs in the summer term, covering principals, design tradeoffs (not a low-level/technical style).

Lecture Recording

Lecture recording is available here

Lectures

- All lectures are pre-recorded and released a week before the Monday Q&A session.
- Monday Q&A session is recorded and run on teams.

How to not fail the module

- Attend Q&A sessions.
- Complete the weekly worksheets.
- Read up on the links in the slides.
- Complete the coursework.
- Ask and help to answer questions on EdStem.
- Revise during term.

Bibliography

Books to aide with the course, you are not expected to read them completely however you may find them useful to address topics from the slides.

- **"Computer Networks" by Andres S. Tanenbaum** 5th or 4th edition suffice.

- **"Computer Networking: A Top-Down Approach" James Kurose and Keith Ross** 7th or 6th edition suffice, E=Book available on Imperial College Library Website.

What is Computer Networking

Definition: Computer Networking

The process of interconnecting computer systems via telecommunications methods to share data and resources.

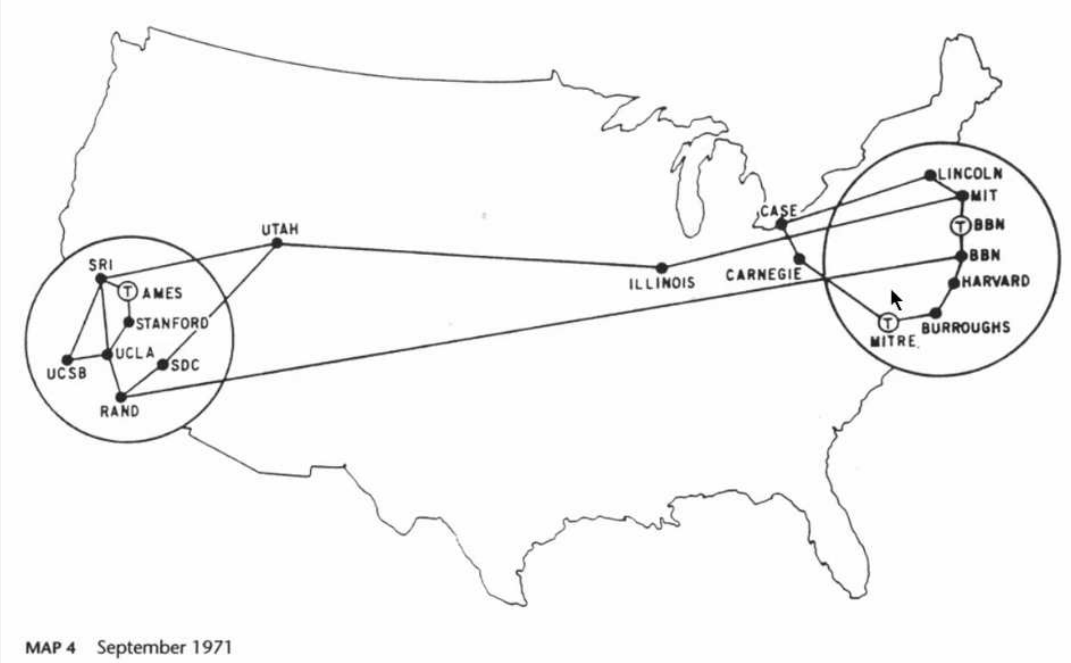
- Networks are becoming eprvasive (everywhere, always on).
- Most mainstream softwarew systems are distributed (cloud computing).
- Peformance often depends on network usage (can be a bottleneck or on critical path)

First Internet Connection

Arpanet (first part of the internet) was created on september 1st 1969 with a single node.

First message as "login", after "lo" was transmitted it crashed, but sent the resul after rebooting an hour later.

Greatly expanded afterwords, connecting several universities.



Vocations

- **Network Engineer/Architect** Design, build and maintain networks.

- **Server Application Developer** Server Backend and communication for cloud applications.
- **Network Software Engineer** Networks + Software Engineering
- **Data Center / Cloud Platform Admin** Networks + Cloud Computing
- **Network Secuity Engineer** Networks + Computer Security

50005 - Networks and Communications - Lecture 2

Oliver Killane

20/01/22

Basic Concepts in Computer Networking

Lecture Recording

Lecture recording is available here

Networking Stack

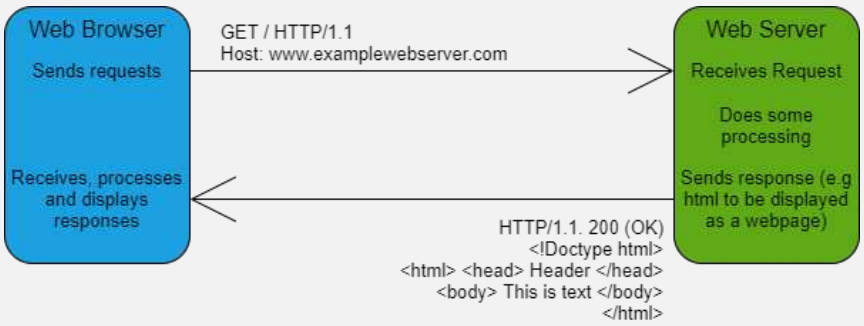
Definition: Application Layer

Applications send and recieve data in a format they specify. Implementations detais of **OS**, **packet types**, **network** setups and hardware models are abstracted away.

Applications use protocols which define structure of data (requests and responses), as well as port numbers and other conventions.

Example: World Wide Web

Part of the internet, invested by Tim Berners-Lee while working at CERN. It uses **HTTP**(HyperText Transfer Protocol). Early versions use plain text (newer & more advanced no longer always true).

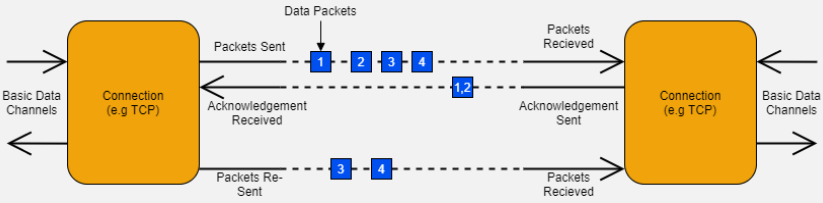


HTTP exists in the application layer of the **TCP/IP** stack. The level of abstraction on which we consider protocols, agreements and transfer of application data.

Definition: Transport Layer

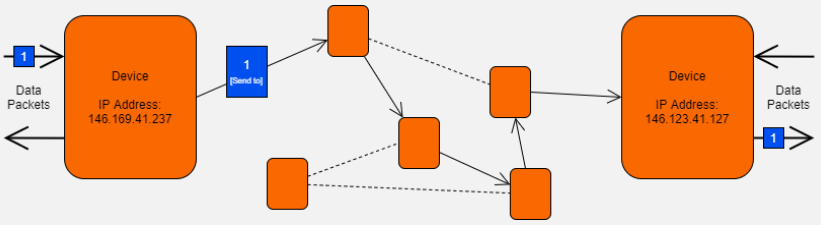
Establishes basic data channels, taking data to be sent or being recieved and converting to/from data packets. Networking can be:

- **Connection-oriented TCP** - Transmission Control Protocol. Packets not acknowledged to have been received are re-sent.
- **Connectionless UDP** - User Datagram Protocol. No checking, packets sent once, more performant.



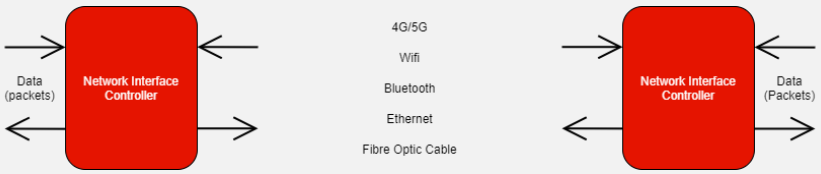
Definition: Network Layer

The internet protocol is used to add **IP Addresses** and other information to packets, and then route them through a mesh network of hosts to reach the destination. The path taken frequently changes and is per-packet.



Definition: Data Link Layer

NIC (network Interface Controller) hardware controlling communication over standards to allow physical communication of data (e.g ethernet, wifi, bluetooth, coaxial cable) to transfer data (packets) between devices.



Definition: Physical Layer

The actual hardware transfering data.

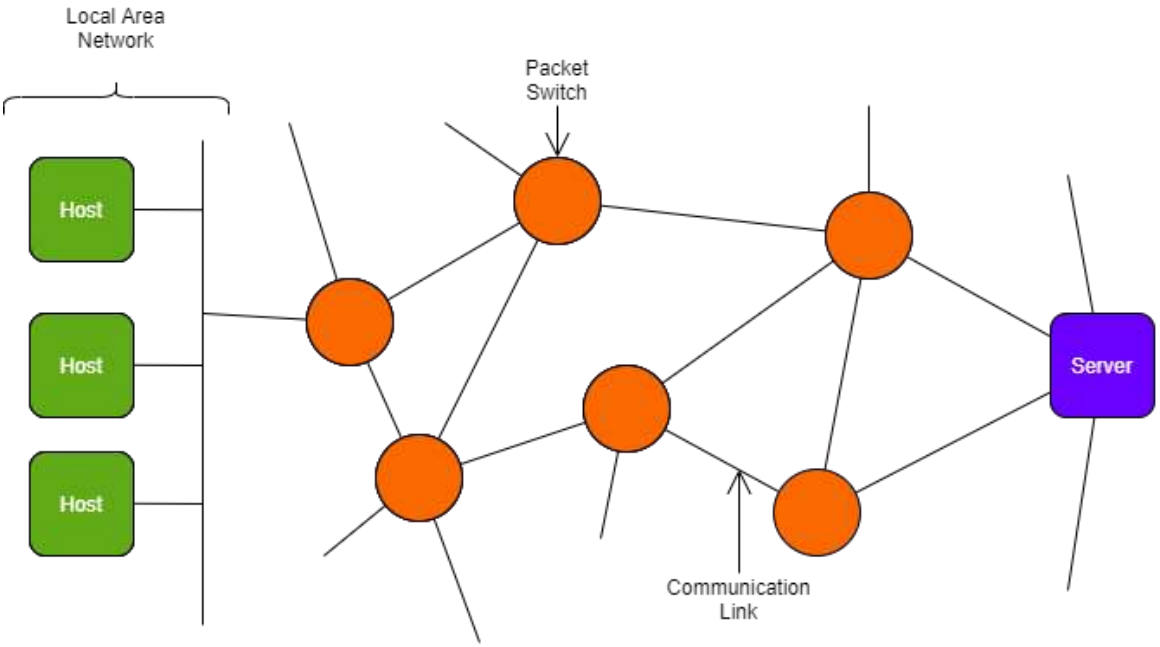
- Fibre-Optic cable
- Twisted-pair copper cable
- coaxial cable
- wireless links (wifi: 802.11, bluetooth)

Internet Structure

Definition: Host/End System

A computer system that is the source, or destination of communication.

- **Smartphone** Send and recieves data (e.g to browse internet)
- **Home Security System** To send and recieve security footage



Some simple terms (for now):

Packet Switch	A **Data Link Layer** switch or router (routes data through a network)
Communication Link	A connection between packet switches and/or end systems (hosts).
Route	A sequence of switches a packet traverses to go from source to destination.
Protocol	A standard concerning the control and format of sending and receiving data to and from end systems.

Packet Switching

Lecture Recording

Lecture recording is available here

Packet Switching

- Data is split into packets which are independently routed through the network.
- Switches & routers use packet information, and network status to determine which next router/end system to forward a packet on to.
- If any links in the network become slow or disconnected, packets are rerouted.
- No setup cost.
- Processing cost associated with forwarding each packet.
- Space cost associated with containing independent information in each and every packet.
- Quality of service is difficult to guarantee (no connection, processing and switching overhead, others can start using links as no reservation).
- High network resource utilisation (can send packets different routes in parallel, two connections can work on shared communication links)
- **Example** The internet.

Circuit Switching

- At the start of a connection, a path is specified and connected.
- Connection stays on the path for the entire duration of the communication.
- High setup cost to create path.
- No processing or space cost as data can be sent straight down the link.
- If the link becomes slow (over-saturated) or breaks a new link must be obtained (slow).
- Network Resources are reserved at connection start, so quality of service is guaranteed.
- Reservation of a route leads to inefficient network resource utilisation.
- **Example** Older telecommunicaton systems (landline).

Telephone Network

The old telephone network (modern is digital, Voice over IP) ius a circuit switched network. A circuit (path through the network) is connected and maintained for the call's duration.

Internet Protocol Stack

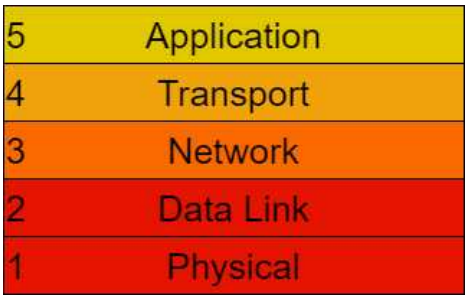
Definition: Communication Protocol

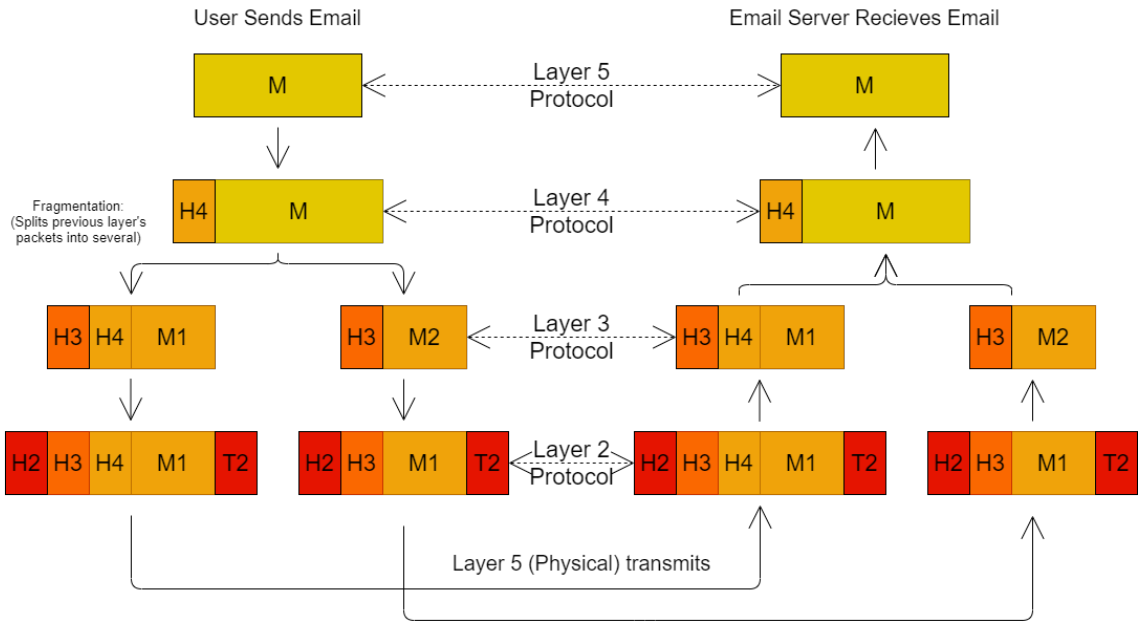
A network protocol is an established set of rules that determine how data is transmitted between different devices. (e.g describe layout and meaning of packets and the order they should be sent).

Phase	**Description**
Handshake	Establishes identities, and the context to begin the communication.
Conversation	Communication, exchanging data in the format & way specified by the protocol.
Closing	Terminates the conversation, performing any necessary cleanup/notification to other.

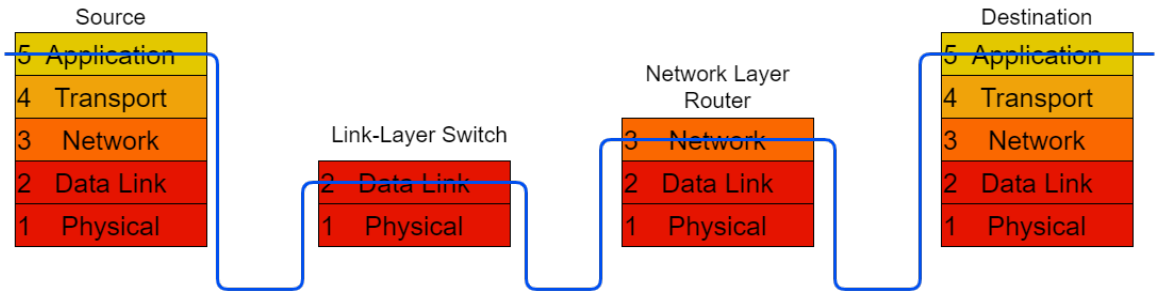
Hybrid 5-Layer Model

For this module we consider the 5 layer model of the networking stack.





For example when communication through other switches:



Alternative Models

In this course the 5 layer model is used. The reason being that Dr Gkoutzis claims it is the best. I (& you) have no reason to doubt this.

There is also:

7-Layer OSI

7	Application
6	Presentation
5	Session
4	Transport
3	Network
2	Data Link
1	Physical

4-Layer TCP/IP

4	Application
3	Transport
2	Internet
1	Network Access

Definition: Service

A set of primitives that the layer provides to the layer above. For example the **Transport Layer** packeting the data sent from the application layer.

Internet Protocol Design

When designing a protocol, one must consider:

- **Addressing** How to denote intended recipient.
- **Error Control** Detection and possible correction of inevitable transmission errors.
- **Flow Control** Prevent fast sender *swamping* slow receiver.
- **Demulti/Multi-plexing** Supporting parallel communications.
- **Routing** How to route packets to destination via best route with low processing/space overhead.

Most **network layers** have both connection-oriented and connection-less protocols:

Connection-oriented	Setup Connection with client, transmit data over channel. (e.g cirtcuit switch, TCP on IP)
Connectionless	Send data to destination address, no formal connection created (*postal mode*). (e.g packet switching, UDP on IP)

Application Layer Protocols

- **Traditional** Name Services (DNS), Email (SMTP), FTP, Telnet, SSH, HTTP/S
- **Modern** Middleware to support distributed systems (Java RMI, Apache Thrift, Google Protocol Buffers - used for sending serialized data)
- **High Level** e-commerce, banking (visa) etc.
- **Peer-to-peer** BitTorrent, skype (old protocol)

Transport Layer

Offers both connection-oriented and connectionless protocols.

- Often provides network interface through sockets (e.g **UNIX** sockets).
- Provides support for secure connections.
- Support for **datagrams** (unreliable but fast per-message basis sending - connectionless, e.g **UDP**).
- Provides mechanisms to prevent fast senders overwhelming slow receivers.

Network Layer

Describes how routing and congestion is done.

- Determining best route.
- Dealing with router unreliability (e.g connection goes down).
- Supporting multicasting/broadcasting.
- Dealing with packet dropping (e.g when a router is overloaded).

Multicasting

Sending information to many recipients from a single source. Useful to reduce network traffic, for example a CCTV system sending a single video stream to be recieved by many screens, backups.

Data Link Layer

Reducing, detecting and rectifying bit transmission layers.

- Adding parity bits, checksum (e.g Cyclic Redundancy Check).
- Specifying how computers can share a common channel (**MAC** (Media Access Control) addresses).
- Specifying how network connects together (e.g Ethernet, FDDI (Fibre Distributed Data Interface)), and token rings (one holds token and listens at a time)

Physical Layer

Describe transmission of raw bits in terms of mechanical, electrical, optical means.

e.g set $0 : +4V, 1 : -3V$ change at frequency of $20KHz$.

Network Performance

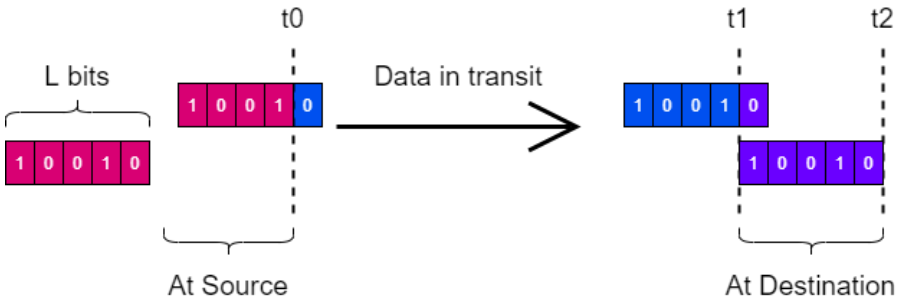
Digital Units

×1000		
Term	**Bytes**	
KiloByte	KB	1000
MegaByte	MB	1000^2
GigaByte	GB	1000^3
TeraByte	TB	1000^4
PetaByte	PB	1000^5
ExaByte	EB	1000^6
ZettaByte	ZB	1000^7
YottaByte	YB	1000^8

×1000		
Term	**Bits**	
Kilobits	Kb	1000
Megabits	Mb	1000^2
Gigabits	Gb	1000^3
Terabits	Tb	1000^4
Petabits	Pb	1000^5
Exabits	Eb	1000^6
Zettabits	Zb	1000^7
Yottabits	Yb	1000^8

×1024		
Term	**Bytes**	
KibiByte	KB	1024
MebiByte	MB	1024^2
GibiByte	GB	1024^3
TebiByte	TB	1024^4
PebiByte	PB	1024^5
ExbiByte	EB	1024^6
ZebibByte	ZB	1024^7
YobiByte	YB	1024^8

Speed and Capacity



Term	Description	Formula
Throughput	Total data received per time (link bandwidth).	$R = \frac{L}{t_2 - t_1}$
Latency	Time taken for a single bit to be transmitted (propagation delay).	$d = t_1 - t_0$
Packetization	Time per bit to be received (transmission delay).	$\frac{L}{R}$
Bandwidth	Maximum possible throughput.	
Transfer Time	Send time per bit	$\Delta = d + \frac{L}{R}$

Example: Small Email

We are transfering a file from London → Edinburgh.

$$L = 4KB \quad d = 500ms \quad R = 1MB/s$$

What is the transfer time (Δ)?

To get transmission delay. $\frac{4KB}{1MB/s} = \frac{4}{1000}s = 4ms$
Hence transfer time is: $500ms + 4ms = 504ms$.

Example: Big File

We are transfering a large file:

$$L = 700MB \quad d = 500ms \quad R = 1MB/s$$

Find the transfer time (Δ).

To get transmission delay: $\frac{700MB}{1MB/s} = 700s$
Hence transfer time is: $500ms + 700s = 700.5s$

Processing Delay

Processing delay d_{proc}:

- Check for bit errors.
- Determine output link.
- Negligible (¡msec).

Queueing delay d_{queue}:

- time waiting at output link for transmission.
- If link is congested, packet might be queued for a long time before being sent.
- If in queue too long, packet may be dropped.

$R:$ link bandwidth(bps) $\quad L:$ packet length (bits) $\quad a:$ average packet arrival rate $\quad \frac{L \times a}{R}:$ traffic intensity

$$\text{Delay} = \begin{cases} \text{small} & \frac{L \times a}{R} \approx 0 \\ \text{large} & \frac{L \times a}{R} \to 1 \\ \infty & \frac{L \times a}{R} > 1 \end{cases}$$

If more work is arriving that can be processed, the delay becomes infinite (and packets will likely be dropped).

50005 - Networks and Communications - Lecture 3

Oliver Killane

21/01/22

The Web

Lecture Recording

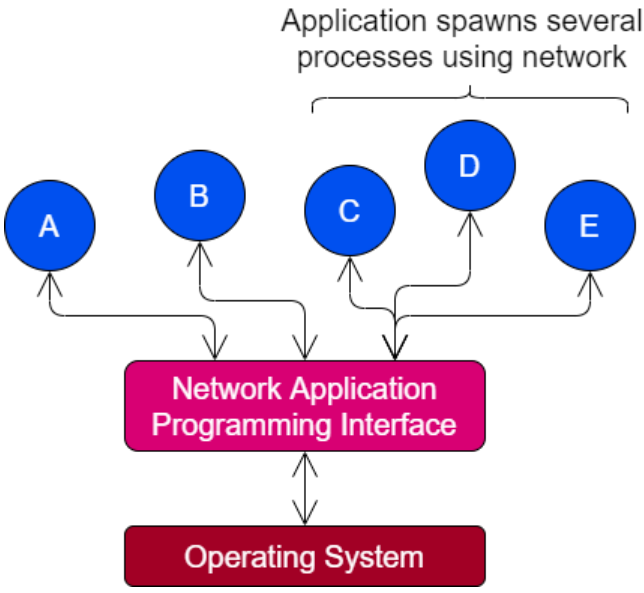
Lecture recording is available here

End Systems Applications

Internet applications are end system applications (or processes).

- Processes run of different Hardware, Operating Systems, etc.
- Protocols offer a layer of abstraction, only need to comply with protocol, all the rest can be different.
- Processes need to be able to address each other to communicate.
- Different processes use different ports, only one process can use one port on one ip at a time.
- The operating system provides networking primitives. This usually comes in the form of sockets.

An **end-system/host** may run multiple programs running multiple processes using the internet/networking. A process can be addressed within its host using the **port number** (used by transport layer) it is using.



Two communicating processes the roles:

Client

- Initiates communications.
- If on a connection-oriented service, the client establishes the connection.

When using sockets:

1. Creates a socket C by connecting to server (e.g to host H on port P).
2. Use socket C by writing/reading to/from it (body of client app protocol).
3. Disconnect and destroy the socket.

Server

- Waits for connections.
- If on a connection-oriented service, the server passively accepts connection requests.

When using sockets:

1. Create a server socket S by accepting a connection on port P
2. Read/Write data from socket to use it (body of server app protocol).
3. Disconnect and destroy S.

Peer-to-Peer

In **Peer-to-Peer** (P2P) networking both processes act as both clients and servers. For example BitTorrent ($\mu Torrent$), Gnutella, even Skype and Spotify for some time.

World Wide Web

Based on concepts of **hypertext** and **hyperlinks**.

- Basically glorified FTP, transferring plain text.
- HTTP and old concept (based on proposal by William Tunnicliffe in the 60s).
- HTML language is very simple.
- HTTP protocol is stateless & simple.
- Low barrier of entry to learn and use.
- GUI browsers make it more accessible, 3rd party graphical applications appreciate the ecosystem.

Web Terminology

document	A webpage, a website containing several.
objects	A file, a document may contain several (HTML, JS, video, images).
URL	Uniform Resource Locator (specifies the address of an object).
browser	Program to request, receive document and process the document to display graphically.
web server	An application containing document and objects, serving them to clients over HTTP.

Definition: HTTP

HyperText Transfer Protocol used fro transfering web objects. It uses a connection-oriented mechanism (**TCP**) but can also work over connectionless (e.g **UDP**).

- Each request and repsonse is a single unit.
- No request depends on a previous one (stateless), everything is self contained.
- If a request is dropped, others are not affected.

$$HTTP/1.0 \quad HTTP/1.1 \quad HTTP/2.0 \quad HTTP/3$$

1.1 is the most popular, with 2.0 being faster and influenced by projects at google. 3 is in final draft.

80 is the port for HTTP requests.

HTTP Connections

HTTP/1.0

Used one **TCP** connection per object. This is inefficient and requires may objects to be spawned and destroyed.

HTTP/1.1

- Same **TCP** connection is used to issue multiple requests and receive multiple responses (can receive multiple objects).
- Default behaviour is to use persistent connections (keep open to send further requests).
- Request containing $Connection: close$ closes the connection, done after all responses/requests have been sent.

HTTP/2

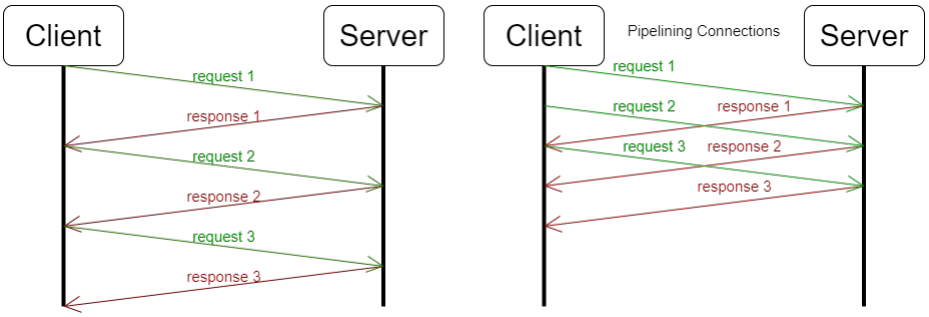
Expected to replace HTTP/1.x completely within a few years.

- Exchanges content in binary, allowing for more compact representation and higher speed (less data to transfer).
- Connection is fully multiplexed (not ordered or blocking)
- Can use a single **TCP** connection with requests in parallel.

HTTP/3

Uses **UDP** for exchanges (faster).

Persistent Connections



Protocol Features

Request

- Protocol Version
- URL Specification
- Connection Attributes
- Content/Feature Negotiation

Response

- Protocol Version
- Reply Status/Value
- Connection Attributes
- Object Attributes
- Conent Specification (type, length)
- Content (Objects)

HTTP Methods

- **GET** retrieve object using **URL**.
- **POST** Submit data to server (e.g a form, message).
- **HEAD** Like get, but only recieve the header, used for testing link validity.

Anatomy of a Response

```
<!-- Status Line -->
HTTP/1.1 200 OK

<!-- Header Lines -->
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT
Content-Length: 88
Content-Type: text/html
Connection: Closed

<!-- Empty Line -->

<!-- Object Body (can be empty) -->
<html>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

Definition: Status Code

A 3 digit code:

$1xx$	Informational.
$2xx$	Successful Operation (e.g 200 $\rightarrow$ OK).
$3xx$	Redirection (object has moved either temporarily or permanently).
$4xx$	Client Error, e.g 400 (Malformed Request), 401 (Unauthorized), 404 (Object not found), 405 (Method not allowed).
$5xx$	Server error, e.g 500 (internal server error), 503 (service overloaded).

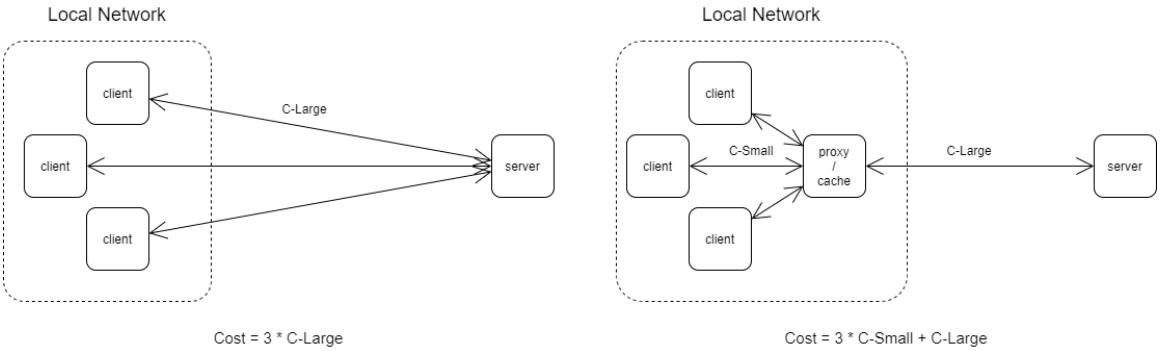
We can use **telnet** to send plain-text commands directly to a server listening on a specific port (80 for HTTP).

```
1 > telnet www.imperial.ac.uk 80
2 > GET /computing/ HTTP/1.1
3 > Host: www.imperial.ac.uk
```

Web Caching

Lecture Recording

Lecture recording is available here



A proxy can be used to speed up common requests.

1. Get request from client.
2. Check if request is cached.
3. If cached, take cached response.
4. If not, forward request to server acting as a client, cache the response for some time.
5. Forward the response to the client.

Benefits

- Reduced latency for requests.
- Reduced network traffic.
- Better security, server only sees proxy.
- When using a firewall on proxy, **LAN** protected from intrusions.

Problems

- Latency associated with finding entries and caching.
- Complexity (need a proxy to be setup).
- Need to determine how long cache entries last for data freshness.

Proxy/Cache is central to several HTTP features.

- HTTP is defined as a request/response protocol, where requests and responses are explicitly passed through the response chain.
- Explicitly specifies how protocol version are handled on the request chain.
- Determines how each method should be handled (e.g only cacheable if indicated by **Cache-Control** or **Expires** in the header, *OPTION* requests are not cacheable).
- Cached pages can become stale. A HEAD request can be made to see if an object has been updated (and cache needs to be invalidated).
- Servers can specify explicit expiration times using either the **Expires** header, or the **max-age** directive of the **Cache-Control** header.
- A client or proxy can use a condition GET request including an **If-Modified-Since** header.

Example: Requests

```
1 GET /this/2122/are/of/site.html HTTP/1.1
2 Host: www.mywebsite.ic.ac.uk
3 Cache-Control: no-cache
4
5 GET /this/2122/are/of/site.html HTTP/1.1
6 Host: www.mywebsite.ic.ac.uk
7 Cache-Control: max-age=30
```

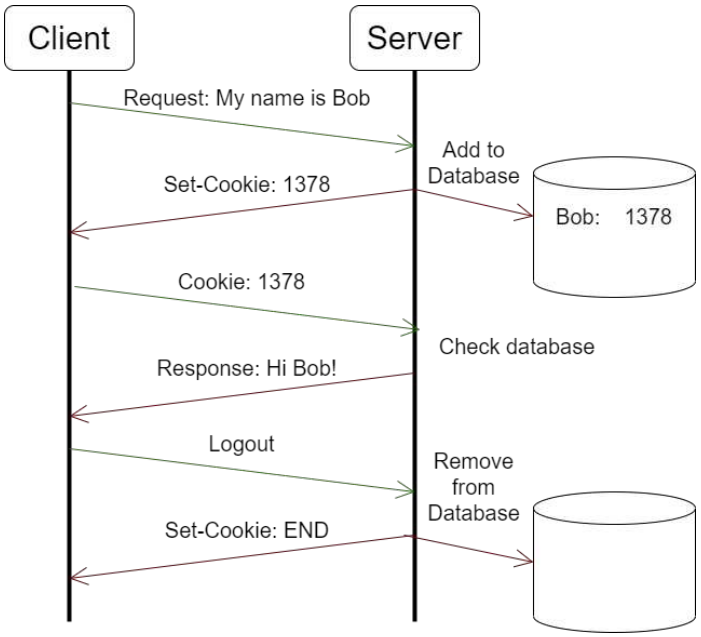
Example: Responses

```
1 <!-- Cache for at most 100 seconds, then revalidate -->
2 HTTP/1.1 200 OK
3 Cache-Control: max-age=100, must-revalidate
4
5 <!-- Do not cache -->
6 HTTP/1.1 200 OK
7 Cache-Control: no-cache
```

HTTP Sessions

HTTP is a stateless protocol, however we need stateful applications (e.g shopping cart for website, playlist next track identifier).

This is done through the **Set-Cookie** and **Cookie** headers:



Some websites keep cookies between visits to track users. Others only use cookies for the extent of a session (given cookie at login, cookie deleted at logout).

Dynamic Web Pages

Instead of storing and serving static web pages, generate webpages for a given user's request (e.g chat, profile, recommendations based on account or session).

Definition: Common Gateway Interface

Allows a program to identify parameters from the url.

https://www.mywebsite.com/page.html?name=oliver&age=19&day=monday

The webserver gets the request url, and can then process it in any way it chooses (e.g for non-existent pages, returning a 404 page).

Servlets

A Java based solution to state, the webserver creates new instances of the JVM to run & process requests for each client connecting.

An alternative approach is to execute code on the client side. The code is sent to the clients browser to run, rather than the server creating new pages to send.

Example: Javascript and PHP

PHP (PHP Hypertext Processor) is server side, generating pages to be sent to the client.

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php echo "Hello World!"; ?>

</body>
</html>
```

Javascript is client side, the code is sent to the client (embedded in the web page) and run on the client side to create the page.

```
<!DOCTYPE html>
<html>
<body>

<h1>My first Javascript page</h1>

<script>
    document.write("Hello World!");
</script>

</body>
</html>
```

Domain Name System (DNS)

Lecture Recording

Lecture recording is available here

IP Addresses

Uniquely identifies an end system by an address.

Type	Size	Example
PIv4	32 bit address	146.169.41.237
PIv6	128 bit address	$fe80::211::43ff::fecd::30f5/64$

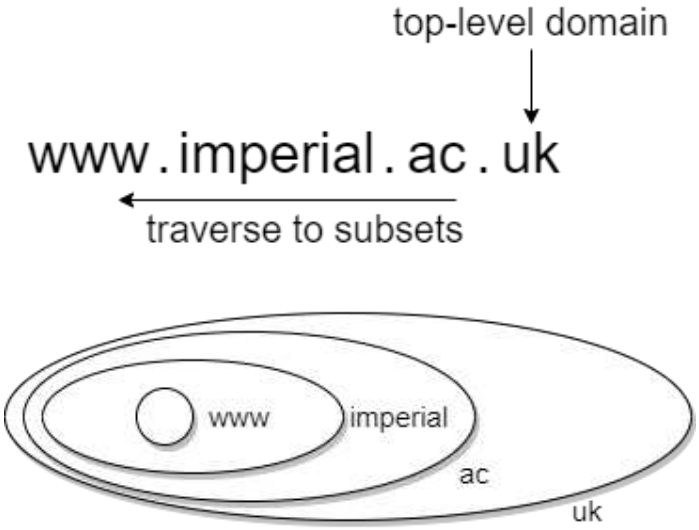
- Easy format for routers to process quickly.
- Not practical for use by people.

Pre-1983

Users can use a file containing mappings of host mnemonics to IP addresses.

Domain Name System

A distributed lookup facility for mapping hostnames to IP addresses.



- **Root Servers**
  Each top-level domain (e.g .com, .edu, .org, .uk etc) is associated one of 13 root DNS servers operated by one of 12 independent organisations.
- **Top-Level Domain Servers**
  A DNS server associated with a top-level domain.
- **Authoritative Servers**
  For each domain, a server holds the master copy mapping all public hosts within that domain.

Most root servers as well as lower level servers are implemented as a distributed set of machines.

By distributing copies of DNS maps (that are constantly updated) across the world traffic can be load balanced, latency can be lower (have servers geographically near) and there is redundancy (e.g if a server is damaged or down for maintenance).

DNS Caching

Important to reduce the load on DNS infrastructure while improving performance.

- Cache can go stale, may needs to be updated from authoritative server.
- **DNS Cache Poisoning**

DNS Cache Poisoning

also called DNS spoofing, entering incorrect mappings into a DNS cache to direct users to the wrong site.

DNS Features

DNS can be described as a directory service database. Each entry is a **Resource Record** describing a translation of a name.

Name	**Value**	**Type**	**TTL - Time to Live**
www.imperial.ac.uk	146.179.40.148	A	...
shell3.doc.ic.ac.uk	146.169.21.39	A	...
www3.imperial.ac.uk	www.imperial.ac.uk	CNAME	...
imperial.ac.uk	ns0.ic.ac.uk	NS	...
imperial.ac.uk	mx1.cc.ic.ac.uk	MX	...

- **Time to Live**
  Specifies how long the mapping should be cached before being invalidated.
- **Type**

Type	**Name**		**Value**
A	host name	→	IP Address
NS	domain name	→	authoritative name server
CNAME	host name alias	→	primary/canonical host name
MX	host name	→	server to receive incoming mail (MX - Mail eXchange)

More of the types can be found in the "Resource Record (RR) Types" table here.

DNS Protocol

- **Connectionless**
  Runs on top of UDP (User Datagram Protocol in the transport layer) on port 53.

  This is as getting a hostname translation only requires two packets (request with name & reply with the value), so the overhead of setting up and closing a TCP connection would be significant comapred to the message time.
- **Messages**
  Has query and reply messages, an identifier is contained in both so messages can be associated.
- **Same format**
  Queries and Replies have the same basic format for simplicity.

Round Robin DNS

A load-balancing technique for geographically distributed web servers.

1. DNS server requests translation of hostname from an authoritative DNS server.
2. A DNS request (to get mapping) is responded to with a list of IP Addresses.

3. DNS server round robins through each address (using each a specific number of times before moving to the next) to make clients send requests to many IPs.
4. Requests to hostname balanced across many servers.

Note that when using this technique, TTL should be low (< 18 seconds), so that the DNS server updates its list often and hence can get the most up to date list of servers available/not snowed under with requests.

e.g

1. Send list with mapping to server A to DNS server 1.
2. Server A becomes overloaded.
3. DNS server 1 requests and update.
4. New list does not contain A.

Manual DNS Lookup

Definition: Name Server Lookup (nslookup)

A tool to find DNS information for a hostname.

```
> nslookup www.imperial.ac.uk
Server:         172.24.128.1
Address:        172.24.128.1#53

Non-authoritative answer:
www.imperial.ac.uk      canonical name = wrpwww.cc.gslb21.ic.ac.uk.
Name:   wrpwww.cc.gslb21.ic.ac.uk
Address: 146.179.42.148
Name:   wrpwww.cc.gslb21.ic.ac.uk
Address: 2a0c:5bc0:88:100:1::172
```

- The first line specifies the DNS server used.
- Non-authoritative specifies the address was extracted from the DNS server's cache.

```
nslookup -type=NS imperial.ac.uk
Server:         172.24.128.1
Address:        172.24.128.1#53

Non-authoritative answer:
imperial.ac.uk  nameserver = ns0.ic.ac.uk.
imperial.ac.uk  nameserver = ns1.ic.ac.uk.
imperial.ac.uk  nameserver = ns2.ic.ac.uk.
imperial.ac.uk  nameserver = auth0.dns.cam.ac.uk.

Authoritative answers can be found from:
```

Definition: Domain Information Groper (dig)

Provides more information on name servers.

```
> dig www.imperial.ac.uk
; <<>> DiG 9.16.1-Ubuntu <<>> www.imperial.ac.uk
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18175
;; flags: qr rd ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;www.imperial.ac.uk.            IN      A

;; ANSWER SECTION:
www.imperial.ac.uk.     0       IN      CNAME   wrpwww.cc.gslb21.ic.ac
    ↪ .uk.
wrpwww.cc.gslb21.ic.ac.uk. 0    IN      A       146.179.42.148

;; Query time: 30 msec
;; SERVER: 172.24.128.1#53(172.24.128.1)
;; WHEN: Sat Jan 22 19:04:30 GMT 2022
;; MSG SIZE  rcvd: 134
```

Much like **nslookup dig** can query for types of DNS records.

```
> dig MX imperial.ac.uk

; <<>> DiG 9.16.1-Ubuntu <<>> MX imperial.ac.uk
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11601
;; flags: qr rd ad; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;imperial.ac.uk.                        IN      MX

;; ANSWER SECTION:
imperial.ac.uk.         0       IN      MX      10 mx1.cc.ic.ac.uk.
imperial.ac.uk.         0       IN      MX      10 mx2.cc.ic.ac.uk.
imperial.ac.uk.         0       IN      MX      10 mx3.cc.ic.ac.uk.
imperial.ac.uk.         0       IN      MX      10 mx4.cc.ic.ac.uk.

;; Query time: 40 msec
;; SERVER: 172.24.128.1#53(172.24.128.1)
;; WHEN: Sat Jan 22 19:07:43 GMT 2022
;; MSG SIZE  rcvd: 170
```

Content Delivery Networks

Lecture Recording

Lecture recording is available here

When storing large files (e.g videos) there are two solutions:

Store on a single powerful server.

- If server down, the file is inaccessible.
- Server can over overwhelmed (run out of sockets or system resources) and become slow.
- Local network can become congested (switches connected to server become overwhelmed, slow donw & drop packets).
- In a single location, so clients may be very far away, so latency is high.

Store and serve many copies from many geographically distributed servers. (The **CDN** approach.)

- Clients can be closer to servers (lower latency).
- Lots of redundancy.

Example: CDN Usage

Video Stored at *http://CDN.com/as8f1324kje12i2*, but requested from *http://notNetflix.com/coolvideo*.

1. Client requests *http://CDN.com/as8f1324kje12i2* from local DNS to get 172.24.128.1.
2. Client connects to 172.24.128.1 over HTTP to get web page.
3. Web page is received, it contains a video at address *http://CDN.com/as8f1324kje12i2*.
4. Client requests *http://CDN.com/as8f1324kje12i2* from local DNS.
5. Local DNS has authoritative DNS stored (CDN's DNS), so connects to CDN's DNS.
6. CDN's DNS uses the local DNS' location, determines which server to connect, and the IP of the video on that server: 142.25.228.77.
7. Client connects to 142.25.228.77 to get video, streams over HTTP.

Main CDN approaches

Enter Deep

Place CDN servers inside many access networks (e.g inside ISP's own networks).

- Very close to users, so low latency.
- Very large number of servers to maintain on many sites.
- Need to get access to other organisation's networks.

Akamai

A large **CDN** network using the "enter deep" approach. According to their website 85% of the worlds internet users are within a single hop of an **Akami CDN** server.

Bring Home

Place a smaller number of CDN servers in large clusters at **Pop** (point of Presence) locations very close to, but not inside, access networks.

Limelight

A very large CDN using the "bring home" appraoch. Their private network extends globally to connect to thousands of ISPs. According to their website they have 123 points of presence.

CDN Performance

To lower latency, the CDN Node (server) used must be the closest to the client requesting the resource.

- CDN will only see the local DNS server's address (difficult to use).
- As a result for some *faster* DNS services such as google's or cloudflare's CDNs will often pick sub-optimal nodes.

Alternatively the client can be given a list of **CDN** servers, it can then pick the best (by pinging to get latency) & then choose the best (this is the approach used by Netflix).

Netflix

Originally on Amazon Webservices, but now on their own **CDN**. They use a hybrid between the "bring home" and "enter deep" approaches.

Email

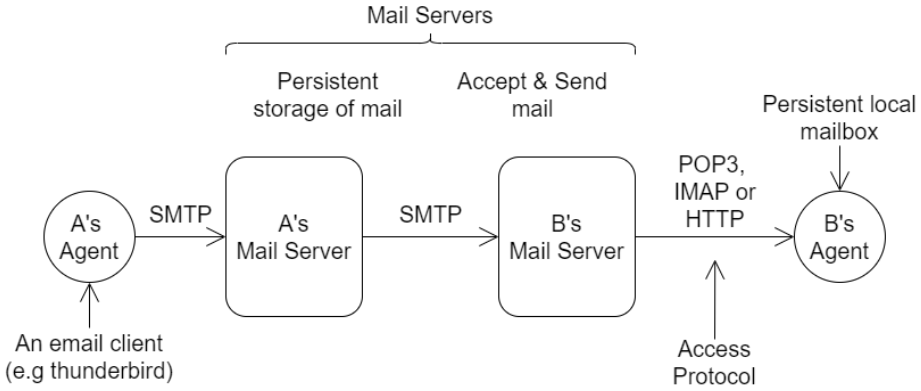
Lecture Recording

Lecture recording is available here

Definition: Email

Text based (with attachments) communication:

- **Asynchronous** Can send messages to users when they are offline.
- **One-to-Many** Can send the same email to many recipients.
- **Multimedia** Can attach small files such as images or video.
- **No Authentication** Messages can be forged or modified.
- **No Confidentiality** Plain text that can be read by snoopers.
- **No delivery Guarentee** Can be accidentally dropped or intentionally blocked, no reliable system to acknowledge recepit.



User Agent Allows users to read, compose, reply to, forward and save messages. Can often offer searching and sorting features, as well as multiple mailbox management.

Mail Servers Accepts messages for remote (sending) and local (receiving) delivery.

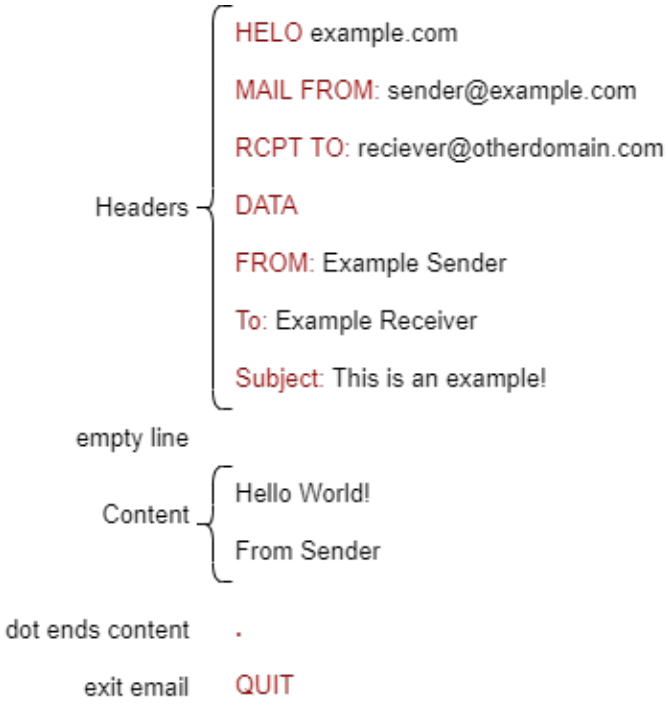
- Has persistent storage of remote delivery messages in a queue.
- Messages for local delivery persistently stored upon receipt.
- user agents can access local mailbox through *access protocol*.

Address is found using **DNS**, the MX type is for mail exchange addresses.

Simple Mail Transfer Protocol (SMTP)

A very simple (and old) protocol working using TCP connections on port 25.

- **Simple**
  1. Set up TCP/IP connection from client to server.
  2. Client requests server to accept messages.
  3. Server responds, if accepting, client sends message.
- **Restrictive**
  Lines must be $\leq$ 1000 characters and only supports ASCII (7 bit characters) (this has been fixed with extensions).
- **Insecure**
  As it is very simple, so easily spoofable and can be used by malicious parties.



Single dot emails

As a "." is used to terminate the email, and this is a trext character.

For each line, if the first character is a "." another is prepended to the line.

When recieving, a line with a single "." is considered terminating, otherwise if the line contains a "." followed by other characters, the "." is removed.

e.g: "." → ".." → "..." → "."

SMTP Email Headers

To	Email address/es of main destination/s.
Cc	(Carbon Copy) send copies to addresses.
Bcc	(Blind Carbon Copy) send blind copies (cannot see other Bcc'd).
From	Name of sender/s.
Sender	Email address of sender.
Received	Added by transfer agent when being received by mail server.
Return-path	Return address.
Date	Date and time the email was sent.
Subject	Short summary of the message.
Reply-To	Email address to send replies to (typically the sender).

Sender

Often the same as "Reply-To", so usually can be left out.

SMTP does not process message content, and should only add the "Recieved" header.

Extensions

- **SMTPS** **SMTP-S**ecure
  SMTP is plain-text, **SMTPS** adds encryption (**TSL/SSL**). Uses **STARTTLS** as the start word instead of **HELO**.

  This can be done over the same port (25) though some servers use different ports
- **ESMTP** **E**xtended **SMTP**
  Adds more methods for XML, html and images. These can be found here.

  Uses **EHLO** as the start word rather than **HELO**. If the reciever responds in the correct way you can use **ESMTP**'s extra methods, else you can fall back to **SMTP** and send a **HELO**, or the server will disconnect you.
- **MIME** Multipurpse Internet Mail Extensions
  Can use provided methods to encode non-ascii as ascii characters to send over **SMTP**.

  MIME types include:

  - **text/plain** Normal plaintext.
  - **text/html** A HTML-Formatted Message.
  - **image/jpeg** Message contains only an image.
  - **multipart/mixed** Message consists of multiple parts.

POP3

Post Office Protocol 3, used to retrieve emails from the mail server.

- Can do basic mail retrieval.
- Implicitly assumes retreived mail is deleted from mails server.
- Uses port 110 (unencrypted) or 995 (**POP3S** - encrypted).

IMAP

Internet Message Access Protocol, it replaces **POP3**.

- Mail is kept on the server, and read online.
- Allows for multiple mailboxes, backed up by the ISP.
- Gives user control over downloading mail.
- Can be encrypted (**IMAPS** port 993) or unencrypted (port 143, rarely used).

Other Protocols

Lecture Recording

Lecture recording is available here

- **FTP** File Transfer Protocol
  For exchanging files across the network. Can be combined with **SSL** encryption (**FTPS**).
- **SSH** Secure Shell
  Direct encrypted communication, can also be used to transfer files (**SFTP**).
- **Telnet**
  Plain text direct communication for non-sensitive data exchange.
- **Crypto**
  Protocols such as **Bitcoin Protocol (BP)** and **Lightning Network Protocol** (**LNP**) are becoming more used and supported.
- **SNMP** Simple Network Management Protocol
  Administrator management of network and its devices.
- **NFS** Network FIle System
  Developed by Sun (bought by oracle), enables file access over a network.
- **DHCP** Dynamic Host COnfiguration Protocol
  Allows all networked devices to get an **IP** address.
- **IRC** Internet Relay Chat
  A live chat system for chatrooms designed in 1988, now rarely used.

Tor

"The onion router", using layers of encryption to enforce anonymity online. A basic explanation is here.

50005 - Networks and Communications - Lecture 4

Oliver Killane

14/02/22

Lecture Recording

Lecture recording is available here

The Transport Layer

Provides connection (**TCP**) and connection-less (**UDP**) services to allow communication between **end-systems/hosts**.

- Connection decision is made at this level.
- Only runs on **end hosts**, not on **routers** or **switches**.
- Requires the lower layers in order to operate (Network, Data-link and Physical).
- Protocols in this layer work on the assumption that the lower levels are working, however must consider that **IP** is best effort, and gives no guarantees on data integrity or order of delivery for packets.

Other layer-4/Transport Layer protocols include:

QUIC A **UDP** based transport layer designed by google employees to replace **TCP** using multiple multiplexed connections using **UDP**, focused on improving **HTTP** performance. The wikipedia article goes into more detail.

UDP-Lite A **UDP** like connectionless protocol that allows potentially damaged data payloads to be propagated to the application layer, and hence allows the application layer to discern data integrity and act accordingly (Wikipedia article).

DCCP The **Datagram Congestion Control Protocol** is a message-oriented protocol that uses reliable connection setup, close and has explicit congestion notification (Wikipedia article).

SCTP The **Stream Control Transmission Protocol** is a message-oriented protocol based on **UDP** (Wikipedia article).

RSVP The **Resource Eservation Protocol** is used to reserve network resources to ensure quality of service (Wikipedia article).

Terminology

Layer	No	Data Name
Application	5	Data
Transport	4	TCP Segments (created by segmentation) or UDP Datagrams
Network/Internet	3	IP Datagrams (a.k.a. packets) (created by fragmentation)
Data Link	2	Frames
Physical	1	Bits

Port Numbers

Definition: Ports

Used to connect applications together/ separate different application's connections.

The transport layer uses port numbers to differentiate between many different network communications. Each application on a host uses a unique port number.

Port numbers are cross-platform, meaning on many different devices, computer architectures and OSes they are the same for the same types of applications (e.g HTTP, IMAP).

Ports			Use
0	→	1023	(well known/reserved for certain protocols, e.g HTTP → 80, SMTP → 25, SSH → 22)
1024	→	49151	(for any user application to use or register)
49152	→	65535	(dynamic/ephemeral/private) and are used by clients temporarily)

TCP

Definition: Tranmission Control Protocol (TCP)

A connection-oriented transport layer protocol.

- Data is split into **segments**.
- Reliable data transfer (integrity of data and (possibly) ordered delivery)
- Not secure (other mechanisms need to be used to ensure security)
- Can offer stream connections (ordered delivery, only accept segments in order, e.g received 4, waiting for 5, but received 6, 7, ignore 6 and 7 until 5 is received.)
- Congestion Control (avoids destructive congestion on the network)
- Requires A handshake to start the connection.
- **Full-Duplex** so both sides can send and receive at the same time.

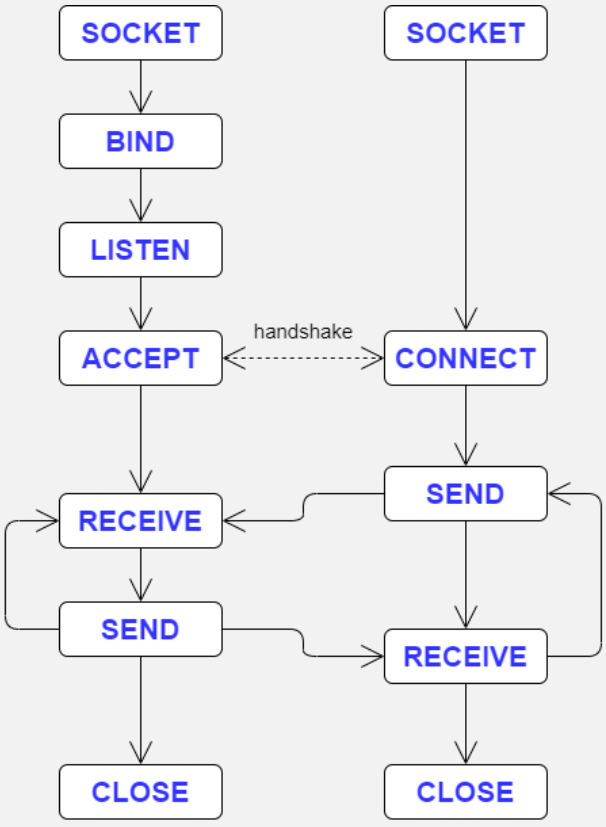
To identify a socket connection we use the **IP Address**, port number and protocol (**TCP**/**UDP**).

Definition: Berkely Socket Interface

An interface adpoted by all **UNIX** systems and windows.

1. **SOCKET** Create a new communication endpoint.
2. **BIND** Attach a local address to the socket. The client and server both bind a transport level address and name to the locally created socket.
3. **LISTEN** Prepare for / Annouce ability to accept, n connections. The kernel now waits for connections from clients.
4. **ACCEPT** Block until some remove client wants to establish a connection, hence the server can now wait, receive a request and choose to accept or deny a connection.
5. **CONNECT** Attempt to establish a connection. When a client connects it must provide the full transport-level address to locate the socket.
6. **SEND** Send data over the connection.
7. **RECEIVE** Receive data over a connection.
8. **CLOSE** Release the connection, communication ends when the socket is closed.

A connection-oriented example:



In a connection-less scenario, **LISTEN**, **ACCEPT** and **CONNECT** are not required.

Example: Simple Java Web Client

```
1  import java.io.BufferedReader;
2  import java.io.IOException;
3  import java.io.InputStreamReader;
4  import java.io.OutputStreamWriter;
5  import java.io.PrintWriter;
6  import java.net.Socket;
7  import java.net.UnknownHostException;
8
9  public class exampleTCPClient {
10     // connect to localhost (this machine) on port 2251
11     static final int port = 2251;
12     static final String ip = "127.0.0.1";
13
14     public static void main(String[] args) throws UnknownHostException
           ↪ , IOException {
15         // Create a socket and implicitly connect.
16         Socket socket = new Socket(ip, port);
17
18         // create reader and writer for sending and receiving
19         BufferedReader receive = new BufferedReader(new
               ↪ InputStreamReader(socket.getInputStream()));
20         PrintWriter send = new PrintWriter(new OutputStreamWriter(
               ↪ socket.getOutputStream()), true);
21
22         // send a message from console input
23         send.println(System.console().readLine());
24
25         // print a received message from the socket to the console
26         System.out.println(receive.readLine());
27
28         // close the socket
29         socket.close();
30     }
31 }
```

Example: Simple Java Web Server

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;

public class exampleTCPServer {
    static final int port = 2251;

    public static void main(String[] args) throws IOException {
        // Bind and set socket to listen.
        ServerSocket serverSocket = new ServerSocket(port);

        // status message
        System.out.println("Started listening on port " + port);

        while (true) {
            // accept a new connection and create socket to handle
                ↪ connection.
            Socket socket = serverSocket.accept();

            // create reader and writer for sending and receiving
            BufferedReader receive = new BufferedReader(new
                ↪ InputStreamReader(socket.getInputStream()));
            PrintWriter send = new PrintWriter(new OutputStreamWriter(
                ↪ socket.getOutputStream()), true);

            // send a message from console input
            send.println(System.console().readLine());

            // print a received message from the socket to the console
            System.out.println(receive.readLine());

            // close the socket
            socket.close();
        }
    }
}
```

To handle many clients, a thread must be created per client, rather than the basic forever loop as above.

Segments

Definition: TCP Segment

A wrapper for **TCP** data, transmitted within the Network Layer protocol (e.g **IPv4** or **IPv6**)

Definition: Maximum Segment Size (MSS)

The maximum amount of application data transmitted in a single segment (header size is not included).

Usually related to the **MTU** of the connection to avoid network level fragmentation (splitting segments in the network layer into multiple packets).

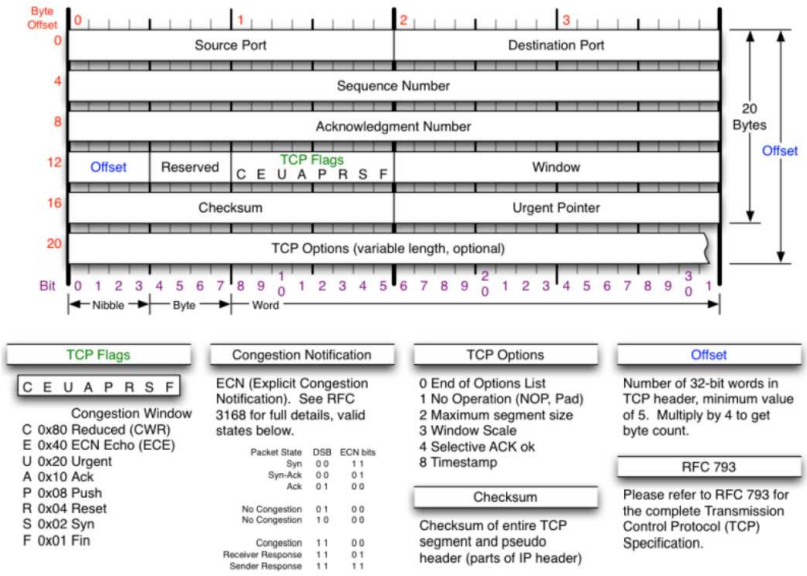
Definition: Maximum Transmission Unit (MTU)

The largest link layer frame available to the sender. Consider it as the largest unit of data that can be transmitted through all links to the receiver without requiring it to be split.

Path MTU Discovery determines the largest frame that can be sent on all links from the sender to receiver.

TCP Header

From the NMap book:



Points of note:

- Source and destination ports are 16 bit identifiers.
- Sequence number and Acknowledgement Number (32 bits) is used for reliable data transfer (identifies a segment, so any segments missing in the sequence can be detected)
- Receive window (16 bits), the amount of data that can be sent before an acknowledgement is received (if the receiver cannot process data as fast as it arrives, it will ask to reduce the TCP window), more here.
- Header length determines the size of the **TCP** header in 32 bit words.

- The optional/variable length field is used to negotiate protocol parameters such as window scale, or **maximum segment size**.

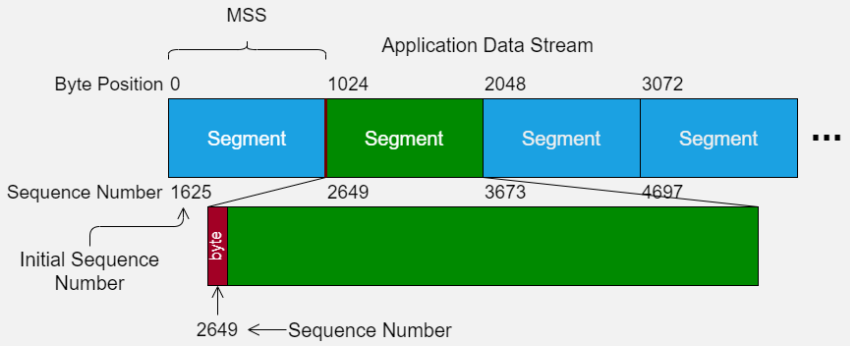
There are several header Fields:

Field	Bits	Description
URG	1	Signals the data as urgent, location of urgent data marked by urgent data pointer field. Note that some software will ignore it.
ACK	1	Signals that the acknowledgement number is a valid acknowledgement.
PSH	1	Push flag, asks the receiver to push data to the application immediately.
RST	1	Resets the connection, often used to shutdown a connection when some unexpected error occurs.
SYN	1	Synchronisation flag, used as part of the handshake.
FIN	1	Signals connection to finish/shutdown.
Checksum	16	Used for error detection.

Definition: Sequence Number

Each byte in the data stream has a sequence number (byte, **not** segment).

The sequence number in a **TCP segment** indicates the position of the first byte carried by that segment.



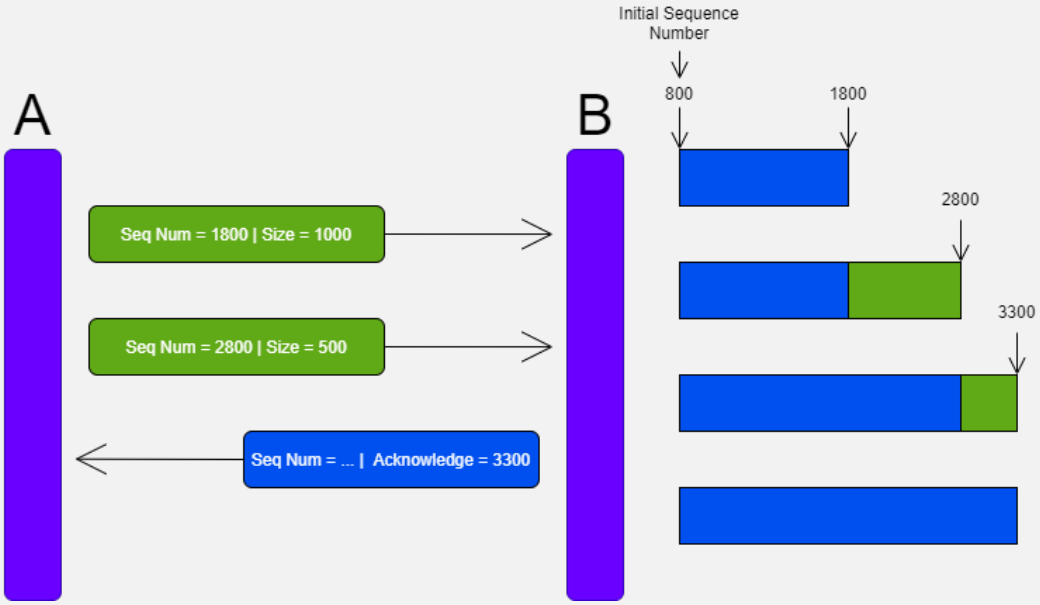
When the **TCP** connection is setup, a random **Initial Sequence Number** is decided upon to avoid any leftover segments being received by mistake.

Hence when creating a new connections, even with the same data the sequence numbers will be different.

Definition: Acknowledgement Number

An **acknowledgedment number** represents the end of the data received, or the first sequence number of the data waiting to be received.

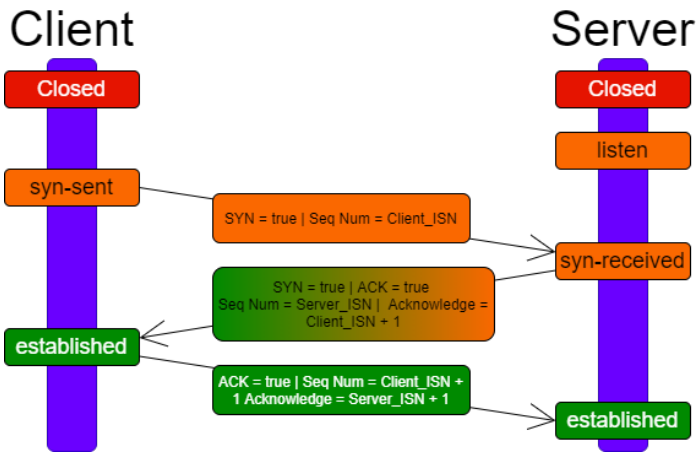
- TCP acknowledgements can be cumulative (recieve segments $1, 2, 3$, ackenowledge (wait for) 4).
- Typically acknowledge every other packet.



- As **TCP** is full duplex, multiple streams/sequences can be received, and acknowledged at the same time.

3-Way Handshake

1. Client sends a **TCP segment** with the **SYN** flag set to **true**, and the **initial sequence number**.
2. Server responds with another **SYN TCP segment** which also has **ACK** as **true** and the first unseen client Sequence number.
3. Client responds with an **ACK** with first unseen server **sequence number**, and a new **sequence number**.



Connection termination is similar, but uses **FIN**.

UDP

Definition: User Datagram Protocol (UDP)

A connection-less transport layer protocol.

- Data is split into **datagrams** (think like telegrams).
- **Datagrams** cannot be larger than $65,507$ bytes ($20B$ IP Header $+8B$ UDP Header $+$ $65,507B = 65,535B$ which is the maximum IP packet size).
- In practice smaller $500B \rightarrow 1KB$ datagrams are used to increase the proportion of packets that are intact (any small error only effects a small datagram, does not invalidate a large datagram).
- Application identification is provided (multiplexing/demultiplexing).
- Integrity of data is checked by a **CRC**-type checksum.

UDP is very simple:

- no flow Control
- no error Control
- no retransmissions

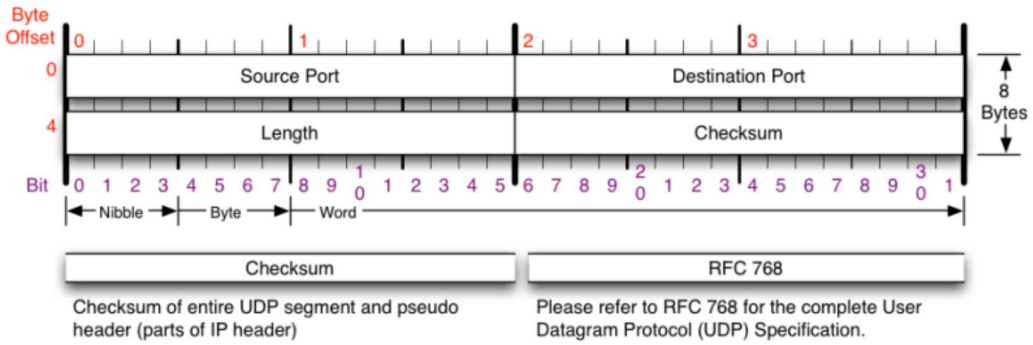
Why use **UDP**?

- Finer level of application layer control over when and what data is sent (e.g real-time such as skype).
- No connection needs to be established (faster than TCP).
- No connection state needs to be stored.
- Very small packet overhead (only a small bit of the packet is not payload).

It is also very useful in **client-server** interactions.

A client can send a short message to a server, and get a quick response, on failure can time out or try again. The resulting code is simple and fewer messages are needed (no connection setup/teardown).

UDP Header



Example: Simple Java Web Client

```java
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

public class exampleUDPClient {
    // connect to localhost (this machine) on port 2251
    static final int port = 2251;
    static final String ip = "127.0.0.1";

    public static void main(String[] args) throws IOException {
        // Create a buffer to store data to send & receive, place
            ↪ contents from
        // the console.
        byte buffer[] = System.console().readLine().getBytes();

        // Create a new packet sourced from the buffer to the target
            ↪ ip and port.
        DatagramPacket packet = new DatagramPacket(buffer, buffer.
            ↪ length, InetAddress.getByName(ip), port);

        // Create a datagram socket to send & receive packets
        DatagramSocket socket = new DatagramSocket();

        // send the packet, the ip and port and inside the packet.
        socket.send(packet);

        // reallocate the buffer to take the response packet
        buffer = new byte[256];

        // take the response from the socket and store in buffer.
        packet = new DatagramPacket(buffer, buffer.length);
        socket.receive(packet);

        // print the received data to the console.
        System.out.println(new String(packet.getData()));

        // Socket no longer used, so close.
        socket.close();
    }
}
```

Example: Simple Java Web Server

```java
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

public class exampleUDPServer {
    static final int port = 2251;

    public static void main(String[] args) throws IOException {
        // Create a socket to receive datagrams on.
        DatagramSocket socket = new DatagramSocket(port);

        // Server runs in forever loop to deal with packets.
        while (true) {
            // Allocate a buffer to store packet data
            byte buffer[] = new byte[256];

            // Create a packet to use buffer.
            DatagramPacket packet = new DatagramPacket(buffer, buffer.
                ↪ length);

            // Receive data, write to buffer.
            socket.receive(packet);

            // Print the data received to the console.
            System.out.println(new String(packet.getData(), 0, packet.
                ↪ getLength()));

            // Take response from standard input.
            buffer = System.console().readLine().getBytes();

            // Get the address of the sender from the received packet.
            InetAddress clientAddress = packet.getAddress();
            int clientPort = packet.getPort();

            // Create a new packet from the buffer.
            packet = new DatagramPacket(buffer, buffer.length,
                ↪ clientAddress, clientPort);

            // Send the response.
            socket.send(packet);
        }
    }
}
```

TCP vs UDP

- **(UDP) - A PvP Game sending short bursts of data to players**
  Data transmission is time critical, if a message is lost, we can simply recover our own way (e.g list of messages, retry).

  We get to control the implementation. While we may mimic TCP in some error recovery, we can decide what features we want and don't for the best experience.

- **(TCP) - An online card game**
  Speed is not a concern, **TCP** is just fine.
- **(TCP) - Movie player application**
  We want it to be fast, however we do not want to drop frames.
  - Pre-Buffer the video, and constantly use connection to get next few seconds as the movie plays.
  - TCP manages errors to reduce dropped frames.

Data Transfer

Lecture Recording

Lecture recording is available here

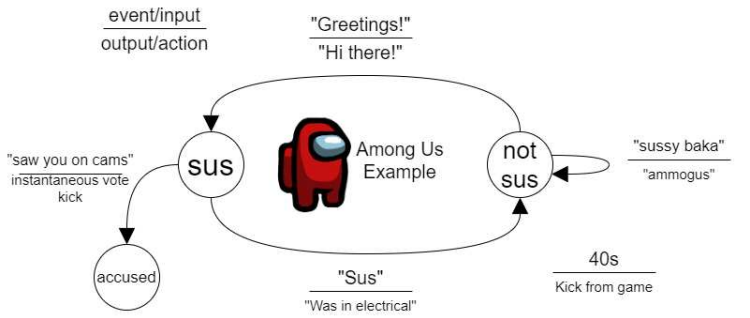
Definition: (FSM) Finite State Machine

A mathematical abstraction used among other uses, to describe network protocols.

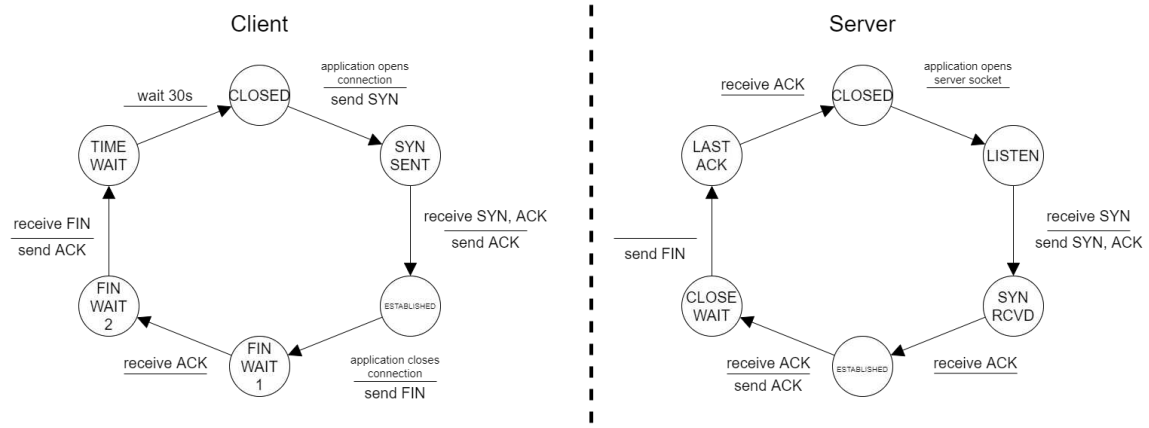
FSM	Finite State Machine
FSA	Finite State Automata
DFA	Deterministic finite-state automaton
NFA	None-deterministic finite state automaton

We can describe transitions between states for a protocol by the event and action.

Example: Among Us Basic Group Meeting



TCP FSM



View TCP states on your device

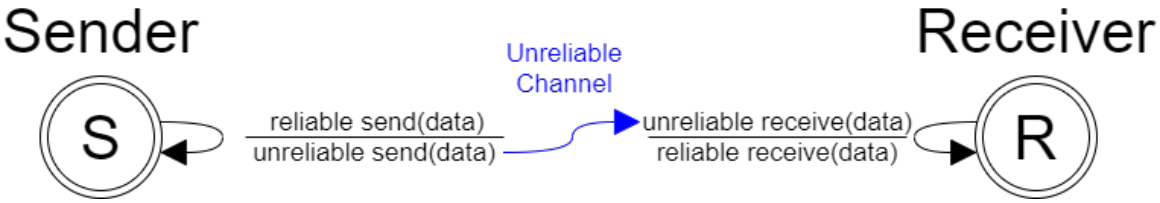
On windows can use the **netstat -a** commend, tcpview or currPorts.

On linux there is htop and iptraf.

Data Transfer FSM

TCP provides mechanisms to ensure reliability in data transfer. While **IP** in the **Network Layer** is a best-effort protocol and is unreliable, by going through **TCP** we can create a reliable connection.

we can generalise this as:



Error Detection

Bits may be flipped in transmission (due to noise/interference and imperfect physical hardware).

Error Detection	Receiver must be able to check if packet is corrupted.
Receiver Feedback	Receiver must be able to tell sender the packet sent was corrupted.

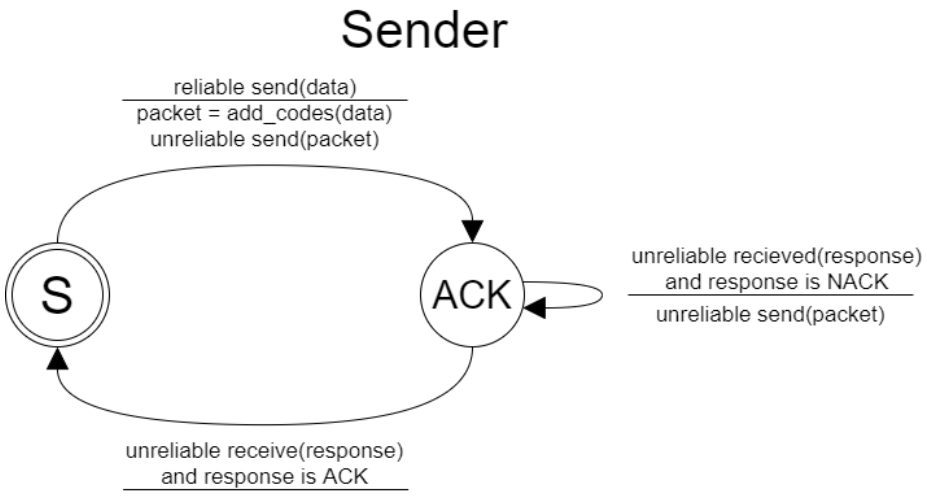
Definition: Parity Bit

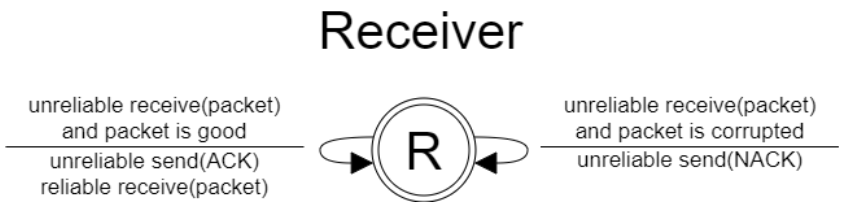
A very simple type of error dectection code, where a single bit is set based on the parity of the rest of the packet. Typically this is the **XOR** of all the bits.

$$\begin{aligned} 1001 &\rightarrow 1001\ 0 \\ 1101 &\rightarrow 1101\ 1 \end{aligned}$$

Stop and Wait with Error detection

We can express the data transfer **FSM** to include error detection. In this setup the protocol is synchronous, meaning for each segment the sender must receive back an acknowledgement before the next segment is sent.





The main issue with this approach, is that **ACK**s and **NACK**s can also get corrupted. If we use the same scheme to reliably transfer the **ACK**s and **NACK**s we end up with potentially no termination (sender and receiver stuck in a loop of replying with **NACK**s at each other, that the other does not receive uncorrupted on a noisy channel).

Assume NACK and retransmit

We can add a sequence number to each packet, so that packets can be retransmitted, and the receiver knows which packets are retransmissions.

If we use **stop and wait** we only need 1 bit for the sequence number, as 0 is original, 1 is retransmission.

Use Sequence Numbers instead of NACKs

We can instead use sequence numbers. If the packet is not acknowledged then an **ACK** is not sent. **ACK** is sent for the last good packet, hence the receiver can use a lack of **ACK**s to determine that it must retransmit some data.

Note that with TCP the **ACK** contains the start of the packet to be sent (or resent) next/ the end byte sequence number of the data received.

Out of Order Sequence Numbers

Rather than use stop and wait, a sender may send many packets.

- **Delayed ACK**
  The receiver only accepts in order, so ignores any packets sent out of order. After receiving a packet in order, it waits some time (e.g $500ms$) before sending the **ACK**, allowing for more in order-packets to be received in this time (resetting the wait).

  1. Received 0, start wait
  2. Wait interrupted, received 1, start wait again
  3. Wait interrupted, received 2, start wait again
  4. Received 7, ignored
  5. Received 6, ignored
  6. Wait from (3.) over, send **ACK** (have recieved up to 2, please send me 3)
  ...
- **Cumulative ACK**
  Received an in-order segment with the expected number, waiting on the next segment.

  Immediately send a cumulative **ACK**.
- **Duplicate ACK**
  Send an **ACK** for the next segment. Then an out-of-order segment arrived with a higher than expected sequence number, there is a gap.

  Immediately send another (duplicate) **ACK**.
- **Immediate ACK**
  A segment received partially or completely fills a gap in the received data.

  Immediately send an **ACK** for the lower end of the gap (to fill).

Timeouts

We can set a timeout for receiving **ACK**s. When the sender does not receive an acknowledgment within the time, it assumes the packet was not received and can try again (retransmit).

- If the timeout is too long, when packets are lost retransmission will have to wait a long time and hence slow down the connection.
- If the timeout is too short, packets may be needlessly retransmitted.

TCP & Checksums

Used by **TCP**. A checksum is calculated from the payload data.

- When received, if the checksum does not match the recalculated checksum, then corruption has occurred.
- Retransmission allows for error recovery.
- **ACK**s and **NACK**s are also protected by error detection code.
- Corrupted **ACK**s are used as **NACK**s
- Sequence numbers allow the receiver to ignore duplicate segments.

Network Simulation

We can use network simulation to check, test and optimise parameters for protocols and network designs.

- **Cisco Packet Tracer**
  Packet tracer is a lightweight network simulator with a user friendly GUI.
- **GNS3 Network Emulator**
  GNS3 is network simulation tool available for free.
- **Opnet Modeler**
  Opnet (now riverbed) modeller is a commercial (paid for) network simulation tool.

Detecting Congestion

Lecture Recording

Lecture recording is available here

Definition: Congestion

So far we have roughly described the **TCP Reno** protocol. However there are many other variants to deal with congestion control.

Routers have a limit to how many packets they can route. Packets are held in a queue.

If too many packets are sent to one of the routers between a sender and reciever, its queue will overflow, resulting in some segments being dropped.

Hence the server assumes the network is congested when it detects segment loss from:

- timeouts (no **ACK** received)
- multiple **ACK**s (or equivalent acknowledgements) can be considered a **NACK**

There are many different congestion control algorithms:

Algorithm	Affects
TFRC	Sender, Receiver
RED	Router
CLAMP	Router, Receiver
XCP	Sender, Router, Receiver
VCP	Sender, Router, Receiver
MaxNet	Sender, Router, Receiver
JetMax	Sender, Router, Receiver
ECN	Sender, Router, Receiver
Vegas	Sender
High Speed	Sender
BIC	Sender
CUBIC	Sender
H-TCP	Sender
FAST	Sender
Compound TCP	Sender
Westwood	Sender
Jersey	Sender
BBR	Sender

- **Linux**
  Usually CUBIC, but can be found at /proc/sys/net ↪ /ipv4/tcp_congestion_control.
- **Windows**
  Can be found at netsh interface tcp>sh gl, if nothing then it is using the windows default.
- **Custom**
  It is possible to force any socket to use any variant
- **Characteristics**
  Most have variable characteristics combined. For example:

Tahoe	Slow start, AIMD, Fast Retransmit
Reno	Fast Recovery
Vegas	Congestion Avoidance

Definition: TCP Vegas

A popular **TCP** implementation.

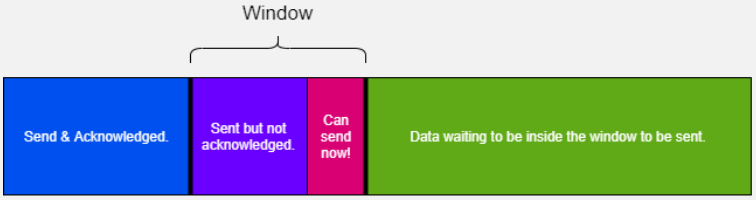
- Attempts to detect congestion before losses occur.
- Predicts packet loss using **RTT** (round trip time)
- Larger **RTT** $\Rightarrow$ greater congestion

Definition: TCP CUBIC

Used by linux as the standard.

In order to avoid advantaging smaller **RTT**s (as can happen with **TCP Reno**), grows **window** as a function of time rather than **RTT**

Definition: Congestion Window



The **congestion window** is the number of bytes that can be sent before blocking to wait for acknowledgements.

Both the sender and receiver can define the window size, the size used is the minimum of both.

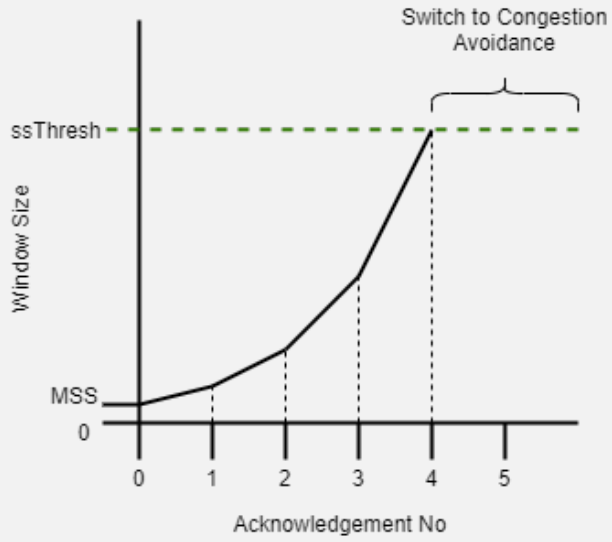
$$W = min(\text{Congestion Window}, \text{Receiver Window})$$

Hence with a given RTT and window size W:

$$\text{maximum rate} \quad \lambda \approx \frac{W}{RTT}$$

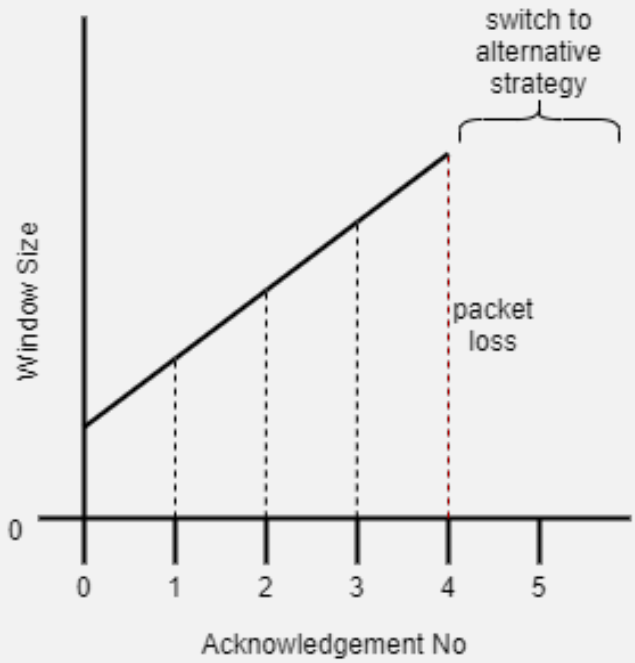
Congestion Methods

Definition: Slow Start



1. Set initial window size to **MSS** (maximum segment size) (quite small for high-speed networks).
2. For every good acknowledgement, increase the window size by the size of data acknowledged (meaning window size is roughly doubled every **RTT**).
3. Continue this exponential increase until window size reaches the **ssthresh** (segment size threshold).
4. The use **Congestion Avoidance**.

Definition: Congestion Avoidance



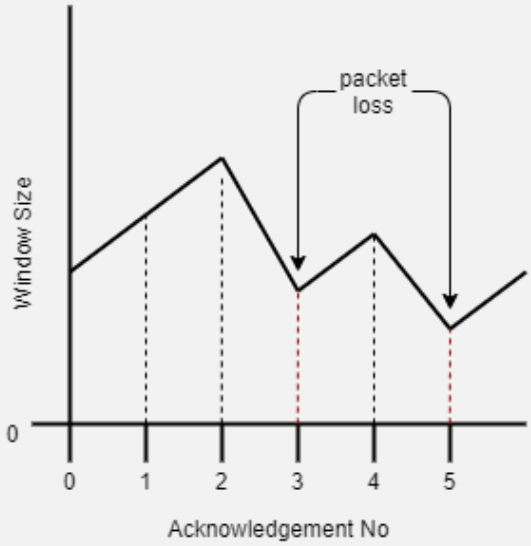
The window size is increased roughly linearly (≈ 1 MSS per RTT).

For each good acknowledgement:

$$W = W + \frac{MSS^2}{W}$$

When congestion is detected (packet loss) switch to a different strategy.

Definition: (AIMD) Additive Increase / Multiplicative Decrease



- For every good acknoledgement: $W = W + \frac{MSS^2}{W}$
- For every packet loss event: $W = \frac{W}{2}$

Definition: timeout

We need to detect packet loss, when no **ACK** is send back.

- **timeout interval** T must be larger than the RTT otherwise we will retransmit data unnecessarily.
- T cannot be too large, otherwise it will be slow to retransmit.
- **TCP** continuously estimates the RTT.
- **TCP** sets T using the **smoothed RTT** ($SRTT$) and the RTT Variation $RTTVAR$ (exact computation can be found in section 2.2 & 2.3 here.)

$$T = SRTT + 4 \times RTTVAR$$

Definition: Fast Retransmission

Three duplicate **ACK**s are interpreted as a **NACK**. The number 3 is agreed upon in section 3 here as a tradeoff between fast retransmission and unnecessary premature retransmission.

- timeout suggests congestion
- 3 duplicate **ACK**s suggests the network can still transmit,

Definition: Fast Recovery

Given the current window size is $\overline{W}$:

If **timeout** occurs:

1. $W = MSS$
2. Run slow start until $W = \dfrac{\overline{W}}{2} = ssthresh$.
3. Switch to collision avoidance.

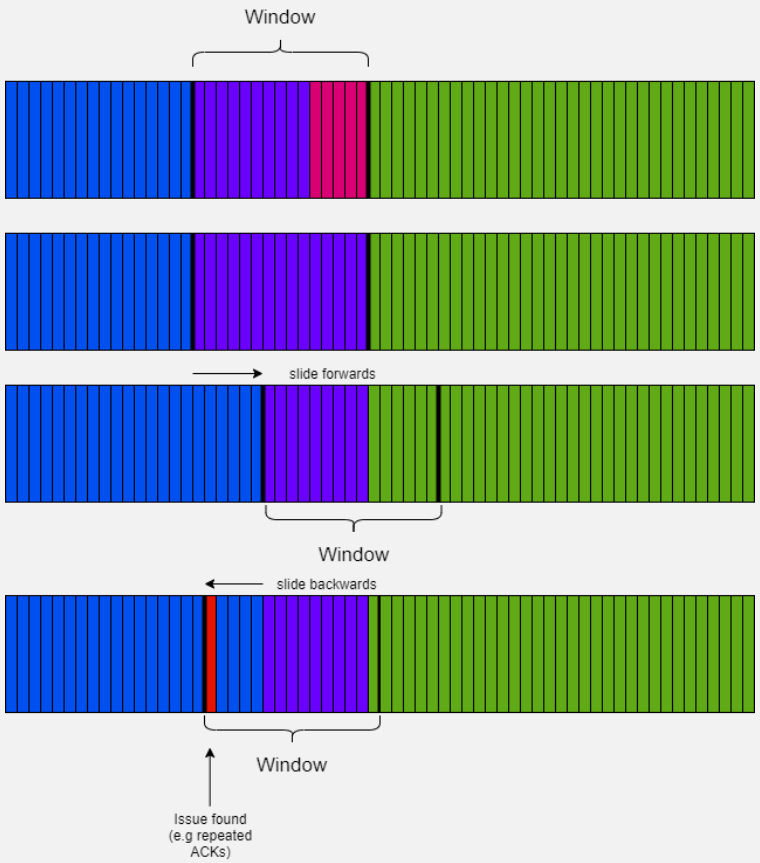
If it is 3 duplicate **ACK**s (a **NACK**) then run **Fast Recovery**:

1. $W = \dfrac{\overline{W}}{2}$
2. Switch to collision avoidance.

Fast recovery is *fast* as the window size is not reset all the way back to MSS, so can ramp up the window size more quickly.

Window Strategies

Definition: Sliding Window - Go Back N



Sender transmits multiple segments without waiting for acknowledgement.

- The sender can have up to W bytes of unacknowledged segments in its pipeline.
- Sender's state is a queue of acknoledgements.
- When we receive some acknoledgements, we can move the slide the window along.

Definition: Sliding Window - Selective Repeat

Sender only re-transmits those segments it suspects were dropped or corrupted.

- Sender keeps a list/vector of acknowledgements.
- Receiver keeps a list/vector of acknowledged segments.
- When segments received out of order, they are kept to be added into the data once the missing/gap segments arrive.
- Sender keeps a timer for each segment it is waiting for acknowledgement of, resending only when the timer expires.
- Sender slides the window when the lowest pending segment is acknowledged.

Definition: Flow Control

Flow control attempts to prevent the receiver from being overwhelmed/overflowing (rate of sending is too high for it to cope).

- The receiver sends the **RecieverWindow** size along with acknowledgements.
- This typically is the size of buffer left to fill.
- When a buffer is full and the receiver can take no more, it sends an acknowledgement with **RecieverWindow** set to 0, and repeats a 1-byte ping to the sender to indicate it is not down or deadlocked, but rather just processing.

Wireless TCP

TCP was designed before the popularisation of wireless networks.

Wired Network

When packets are lost, this indicated congestion.

- Reduce packets sent.
- Use congestion avoidance and recovery strategies.

Wireless Network

When packets are lost, this is most likely a channel reliability issue.

- Resend packets as much as possible.
- Gives best chance of one getting received correctly.

We can fix these conflicting requirements in two ways:

- **Split TCP Connections** If we use separate connections for wired and wireless we can distinguish between the two and hence use different algorithms for congestion avoidance.
- **Use Base Station** Have the wired base station do some retransmissions without informing the wireless source.

  Here the base station tries to improve wireless IP reliability using **TCP**.

Network Usage

$$\text{Utilisation Factor} = \frac{\text{network use}}{\text{maximum theoretical usage}}$$

When we have the RTT, packet size L and transmission rate R, we can also use the time on the connection used out of the possible time length:

$$d_{trans} = \frac{L}{R}$$

$$\text{Utilisation Factor} = \frac{d_{trans}}{RTT + d_{trans}}$$

50005 - Networks and Communications - Lecture 5

Oliver Killane

15/02/22

History and Terminology

Lecture Recording

Lecture recording is available here

H/P/V/A/C

- **Hacker**
  Highly competent computer ethusiast/engineer

White Hat	Informs organisations of vulnerabilities before going public.
Grey Hat	Only informs if paid.
Black Hat	Malicious, uses findings to do illegal activity.

- **Phreaker**
  Phone hacker, as phone network has become more digital they have been more often in the hacker category.

- **Virii**
  Computer virus creators.

Ransomware	Encrypt files, decrypt for ransom.
Spyware	Keyloggers, browser addons to track, often include adware.
Trojans	Software for botnet zombies, often appear as legitimate software.

- **Anarchist**
  Politically active hackers, when peaceful called hacktavists, when not so much they are anarchists. Some parts of the anonymous movement could be considered anarchists.

- **Crackers**
  Make use of tools built by others (e.g purchasing virus software, infiltration tools). Most modern digital organised crime would be in this category.

- **DDoSers**
  Someone who participates in **Distributed Denials of Service** attacks. **Low Orbit Ion Cannon** is an example, it is used to stress tests networks.

- **Spammers/Botters**
  Send unsolicited messages (often advertisements) en-masse, usually using botnets.

- **Warez**
  Information piracy, distribution software, images & videos without the legal right to do so. Examples include the priate bay piracy site.

- **Whistleblowers**
  Former employees of an organisation that leak/"blow the whistle" on often malicious, illegal, or immoral activity even when it is illegal to do so (e.g have signed a **Non-Disclosure Agreement**).

- **Social Engineers**
  Use of social manipulation to compromise the human security of an organisation.

Phising	Usually over email, pretending to be some organisation they are not.
Vishing	Via voice messages.
Smishing	Via **SMS**.
Catfishing	Via impersonation (e.g fake social media profiles).

Black Hat Methods

- **Credential Reuse/Stuffing**
  Previously leaked login credentials are used against many sites (often people use the same email and password comnbination on many sites).

  The site have I been pwned can be used to check if your own details are included in any leaks.

- **Packet Sniffing**
  Monitoring network traffic not intended for your **NIC**, e.g over a wireless network, on a router or switch you control.

- **Code/SQL Injection**
  Using data infput to get a system to runb your code. If the system does not sanitise its input, some keywords or code contained may be executed. This is often SQL Injection, as most opportunities to input data are related to databases. An example is the log4j bug/exploit expained well by this video.

- **Session/Cookie Hijacking**
  Stealing a session cookie to be authenticated as them in an ongoing browser session.

- **Wardriving**
  Searchinbg for and abusing open/unsecure **WiFi**s.

- **Trashing/Dumpster Diving**
  Checking physical waste for useful informatrion (e.g bank statements, official records).

- **Clickjacking**
  Using hidden html divs, popups to force a user's click to redirect to a malicious destination.

- **Bait & Switch**
  Luring a user to click with a seemingly legitimate advertisement, only to redirect them to something else.

- **Spoofing**
  Falsifying identification to receive packets intended for another recipient.

IP	Fake your IP as another (Layer 3).
MAC	Media Access Control address distinguishes between different **NICs**.
DNS	DNS cache poisoning (falsifying the cache to pretend to be at a given domain).

- **Rootkits**
  Allows attackers to secretly enter external systems, often installed as part of a virus.

- **Keyloggers**
  Records all keyboard input, sending key-logs back to the hacker, or allowing them to be remotely accessed.

  A potential advantage of password managers is that keyloggers cannot be used effectively to detect passwords if they are not typed.

- **Trojans**
  Allow a hacker to remotely control an entire system, often as a zombie on a botnet.

- **Evil Twin**
  Where a hacker attempts to lure victims into using their network, gaining information from the victim (e.g network histroy) & potentially sending malicious data to the user when they use it.

White Hat Tools

- **Tails**
  Tails is a portable operating system designed to be usable from a usb drive, and to never store data, thus removing data integrity related security issues.

- **Kali Linux**
  Kali is an operating system designed for penetration testing and other security related work. It comes bundled with many useful tools such as **Metasploit** and **Nmap**. It is supported on ARM, as well as by the windows subsystem for linux.

- **Metasploit**
  Metasploit is a tool use to automatically scan systems for vulnerabilities based on a large database of known vulnerabilities and exploits.

Cybercrime Laws

In the **UK** many laws (listed below) apply. Physical locations of hosts is used to determine which nation's laws are used, meaning **US** law is also very important (common country to host from).

1964	Obscene Publications Act (In reference to spam)
1978	Protection of Children Act (In reference to online abuse & spam against children online)
1988	Copyright, Designs and Patents Act
1990	Computer Misuse Act
1999	Amendment to the Protection of Children Act (Still being changed)
2000	Freedom of Information Act
2000	Regulation of Investigatory Powers Act (In reference to computer/phone surveillance)
2002	e-Commerce Regulations Directive
2003	Criminal Justice Act
2005	Disability Discrimination Act (In reference to online abuse & spam)
2010	Amendment to the Copyright, Designs and Patents Act
2013	Defamation Act (In reference to online abuse & spam)
2017	Digital Economy Act
2018	Data Protection Act

In the **US** there is the DMCA (Digital Millennium Copyright Act).

Standards Organisations

IANA	Internet Assigned Numbers Authority, deals with **DNS**, **IP** Addressing and more standards.
ICANN	A nonprofit organisations responsible for coordinating standards for the maintenance and running of namespace and numerical space databases for the internet.
IEFT	Internet Engineering Task Force, a collection of working groups (e.g routing, transport, security) concerned with developing the internet.
ISOC	Internet Society, dedicated to furthering beneficial use of the internet.
EFF	Electronic Frontier Foundation, a politically active nonprofit dedicated to defending privacy, free speech and freedom to innovate online.
W3C	The World Wide Web Consortium develops standards to help developers build tools on the web smoothly.
ISO	International Organisation for Standardization, you can find their standards for information technology here.

Attack Examples

Definition: Heartbleed

A bug in OpenSSL 1.0.1 first identified onm 14/03/2012 and patched on 07/04/2014.

OpenSSL is an implementation of the TLS (transport Layer Security)/SSL (Secure Sockets Layer) protocol that allows for secure website access (https://)

The bug allowed users to gradually reveal server memory in chunks of $64KB$. It is not known if it was used in any exploits.

Definition: KRACK

The **WPA2** (wireless protected access protocol) used in Wifi. Android devices could be forced to use a zero-based key, rendering the encryption useless.

The full explanation can be found here.

It has been patched, but the next version (**WPA3**) was released in 2020 which was meant to be more secure also has issues.

Definition: WEP

Wired Equivalent Privacy is a security algorithm for wireless networks, it has been shown to be vulnerable many times.

Network Security Issues

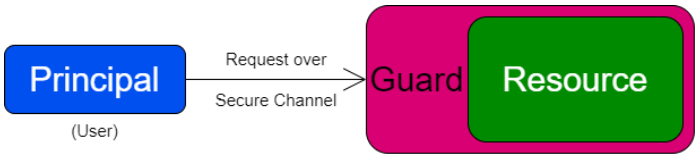
Lecture Recording

Lecture recording is available here

Basic Security Concepts

Access Control	Only certain users can access certain resources.
Authentication	User knows resource is as identified, and resource that user is as identified.
Confidentiality	Users can limit access to their resources and information, and limit access to see their traffic over a channel.
Data Integrity	Users cannot damage the integrity of a resource (e.g crash a webserver by visiting).
Non-Repudiation	User cannot deny communication occurred, secure logs held and can be audited.

Access Control



Assuming the channel used for communication is secure, the guard needs to determine:

- Which principals (users) can access the resource.
- Where principals can be located (e.g user's **IP** address is outside the organisation's network).
- What requests principals can make for this resource (e.g can view database, but not send mutating **SQL** commands).

Security can be difficult as:

- Many systems used by an organisation can be different (Heterogeneous systems) (e.g bank has different OSes several models of ATM).
- Users can be careless (e.g reusing passwords), this include system administrators and managers.

Firewalls

Definition: Firewall

A security barrier between internal and external networks.

- **Application Level Gateway**
  An application that runs, checks requests in the application layer. Can also be a proxy, using an extra set of rules to decide if to share requests or responses, or to send on requests.

  Examples include **SOCKS** and netfilter's iptables.
- **Proxy Server**
  Runs on the network protecting it by making requests and receiving responses on its behalf (to the external network). Can also include caching of results.
- **Circuit Level Gateway**
  Creates a circuit of proxies, sending data between each node in the circuit (for example **Tor**).
- **Packet Filtering**
  Filtering packets with a set of rules based on contents, source and destination IP address/port, only alowing non-suspect packets through. Can also be stateful, considering not just a single packet traffic to a host over some time period.
- **Hybrid**
  Use a combination of all the above.

They can be software or hardware based, with hardware solutions being faster, but more difficult to change (e.g if a vulnerability is found).

Definition: Proxy

Makes requests and recieved responses on behalf of a client, can filter in and outgoing traffic.

- **Normal**
  Client is aware of proxy, and connects to it to use it.
- **Transparent**
  Client is unaware, for example a local router could act as a proxy. Requires not intervention from client.
- **Reverse**
  Runs on the receiving side, impersonating a server and protecting a server from the external network. (Much like **CDN** load balancing)

Definition: Bastion Host

A server that expects to be attacked.

- Runs a minimal trusted/secure OS.
- Only essential applications (e.g no window manager needed).
- All possible limits enabled (readonly file system, no mounts, file permissions all set, no normal user accounts)
- Typically managed over a dedicated terminal

It passes requests on from the external network, and acts as a proxy firewall.

It drops any connections it determines are suspect using packet filtering (usually stateful) and other techniques.

Example: iptables

iptables can be used on linux to set packet filters. It consists of several tables, each containing chains of rules on managing network packets.

Note that it requires root as it interfaces directly with the linux kernel's firewall.

Example: tcpd

The linux **TCP** Daemon controls access to unix services & can monitor requests to services (e.g **ftp**, **exec**, **rsh**, **telnet**).

It uses two files, `/etc/hosts.allow` and `/etc/hosts.deny` to determine access.

Firewall Avoidance

SSH

We can attempt to avoid a firewall by tunneling through with an allowed protocol, to then use the internal.

An example of this is with **ssh**. We can get through the firewall on ssh, and then send our requests through ssh to the internal network, to get responses, and use services the firewall may normally block.

Spoof MAC Address

Can re-write the **MAC** address if the firewall is blocking requests based on it (MAC address blacklisting or whitelisting).

Spoof IP Address

Much like with **MAC** address, however stateful firewalls will most likely detect this.

(VPN) Virtual Private Network

Much like with **SSH**, we can tunnel through the firewall. Provided the tunnel is secure (e.g using **SSL**) the firewall will not be able to decipher your traffic.

Other Security

Definition: (IDS) Intrustion Detection System

Detects intrustions to inform the system (e.g a DDoS attack), however does not perform actions to stop the detected intrusion.

Definition: (IPS) Intrusion Prevention System

Actively prevents intrusions (e.g blocking **SYN** flooders attempting to perform a DDoS attack), can work with an **IDS**.

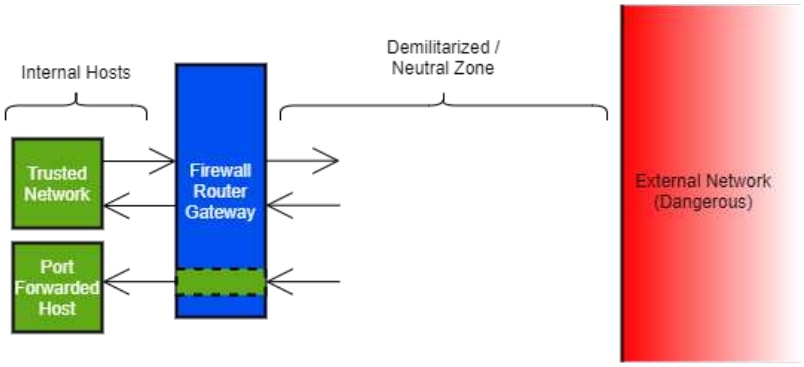
Definition: (NGFW) Next Generation Firewall

A stateful firewall that comes with an **IPS** / **IDS** system.

Definition: (UTM) Unified Threat Management

Similar to a **NGFW** but with added features such as spam filters, antivirus etc.

(DMZ) Demilitarized Zone



Definition: (NAT) Network Address Translation

Rather than expose the **LAN IP** address of an internal host, routers translate the IP addresses to their own public IP to send, and when receiving from their public IP back to the IPs of internal hosts.

Definition: Port Forwarding

To expose an internal host to the external network without placing it in the **DMZ** we can set the router to forward all packets arriving at a given port straight to the internal host.

For example we could specify any packet recieved on the router's IP at port 3472 should be immediately forwarded to the **NAT** based **LAN IP** of "host A" on port 80.

Useful for hosting servers, even for games (e.g minecraft servers require port forwarding).

Logging and Auditing

Most systems keep logs, they are useful for:

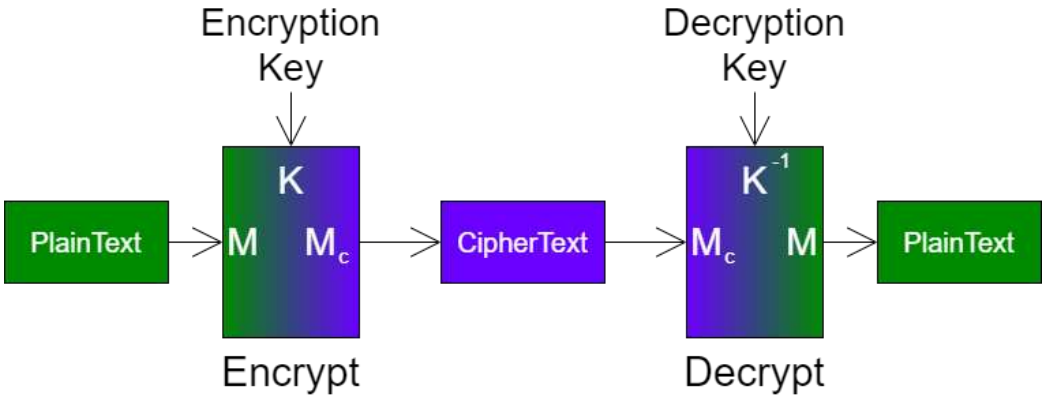
- Checking for missed breaches. An attack may only be discovered in logs after the fact.
- Forensics, providing evidence to law enforcement to discover who the purputrators of an attack are.
- Determining how a system was exploited in an attack, in order to patch it.
- Ensuring good practices are being followed (e.g if an unsafe features starts to be used).
- Detecting other network issues (e.g congestion).

Logs can be found on linux at `/var/log/` and the event viewer in Windows.

Cryptography

Lecture Recording

Lecture recording is available here

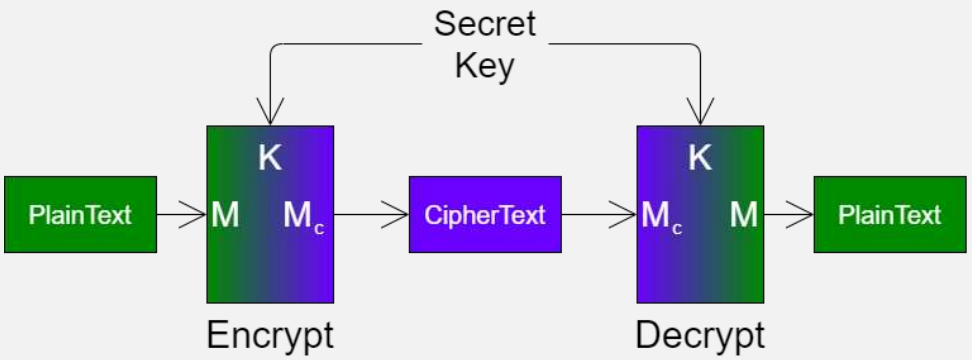


A good encryption algorithm should ensure:

- Given M_C it is only possible to find M by going though all possible values of K^{-1} (brute force attack).

- Given M and M_C it should be difficult to get the values of K and K^{-1}. (e.g caesar cipher is poor as we can just calculate the shift)

Definition: Symmetric and Secret Key Encryption



The same key is used for encryption and decryption (symmetric), and this key is secretly shared between sender and receiver (not on an unsecure channel) (secret).

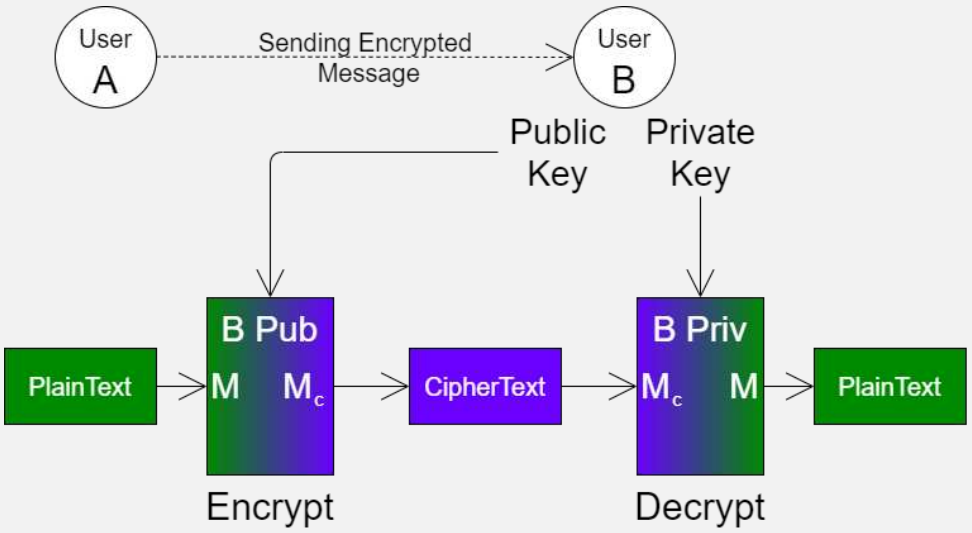
- Must secretly disclose key to communicate.
- Faster encryption/decryption than **Asymmetric**.

An example is **DES** (**Data Encryption Standard**), though this has a short key length and is now too insecure for general use (wikipedia article here).

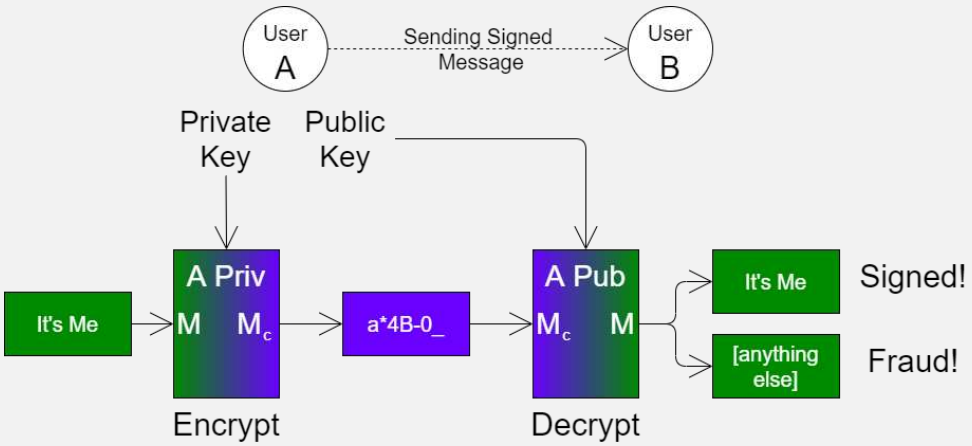
Definition: Asymmetric and Public Key Encryption

Each user has a **public** and **private** key.

For confidentiality: Sender encrypts with receiver's public, receiver decrypts with their private.



For signing: Sender encrypts with their private key, receiver decrypts with sender's public key, if value was successfully decrypted then we know the message was from the sender with the public key we used.



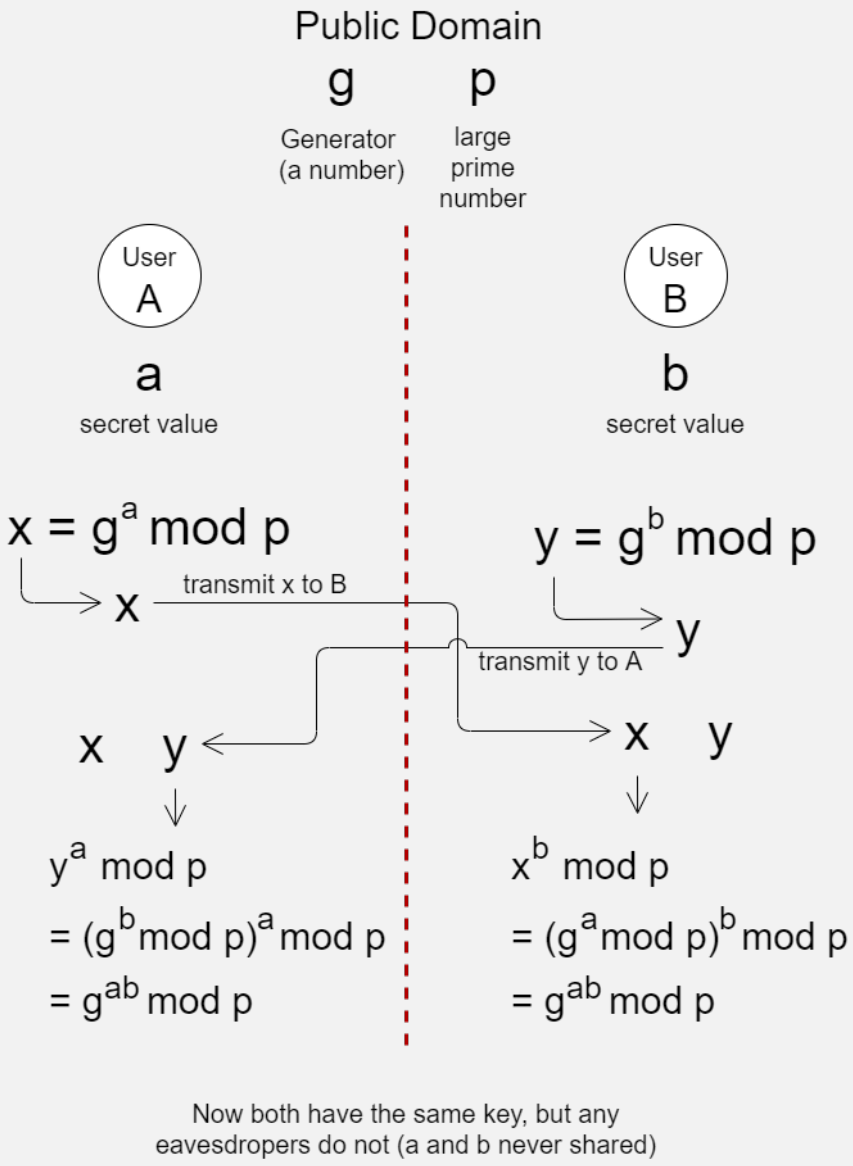
We can then combine these, encrypting a message, but including a signed segment inside to verify the sender. We can also combine with symmetric encryption to sign symmetrically encrypted files (e.g check if a password protected file is from the correct sender/is not tampered with). Foe example GPG.

The main points are:

- No need to disclose private information to communicate securely so, can start communication on unsecure channel. 11
- Hence more secure than **Secret Key** encryption.
- However is slower to encrypt and decrypt.

An example of this is **RSA** which uses current difficulty in prime factor decomposition to ensure brute force attacks are computationally intractable (wikipedia article is here).

Definition: Diffie-Hellman Key Exchange



Definition: Kerberos (Needham & Schroeder)

A key distribution system for secret keys, using a trusted server.

- Kernberos authenticates you with a password.
- It can also authenticate the user/resource you intend to communicate with.
- Generates a ticket which allows for communication.
- Ticket can be used up to a time limit, after which you must get another ticket.
- Originally vulnerable to **Man/Monster in the Middle** attacks, though these have since been addressed.

Definition: Hashing

In cryptography, a hash function converts some data into a fixed size alphanumeric string.

- The same input data always produces the same hash value.
- Not possible to derive the original input from the hash value (unlike encryption).
- Can be used as a checksum to verify data (e.g that the contents of a file are unchanged/not tampered with).
- Many old hash functions have been broken (either algorithm issues, or all possible hashes now determines - rainbow tables).

50005 - Networks and Communications - Lecture 6

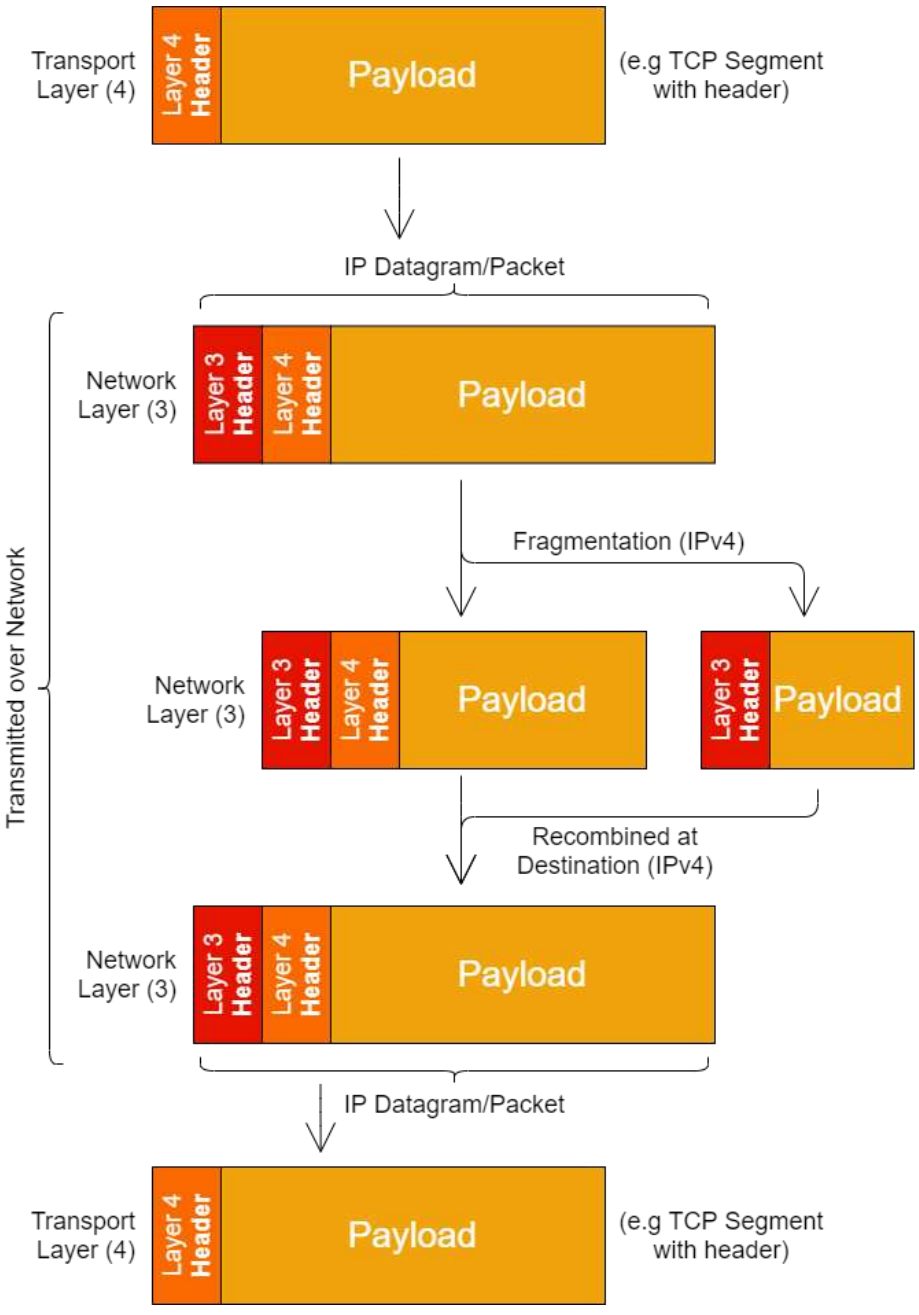
Oliver Killane

17/02/22

Network Layer

Lecture Recording

Lecture recording is available here

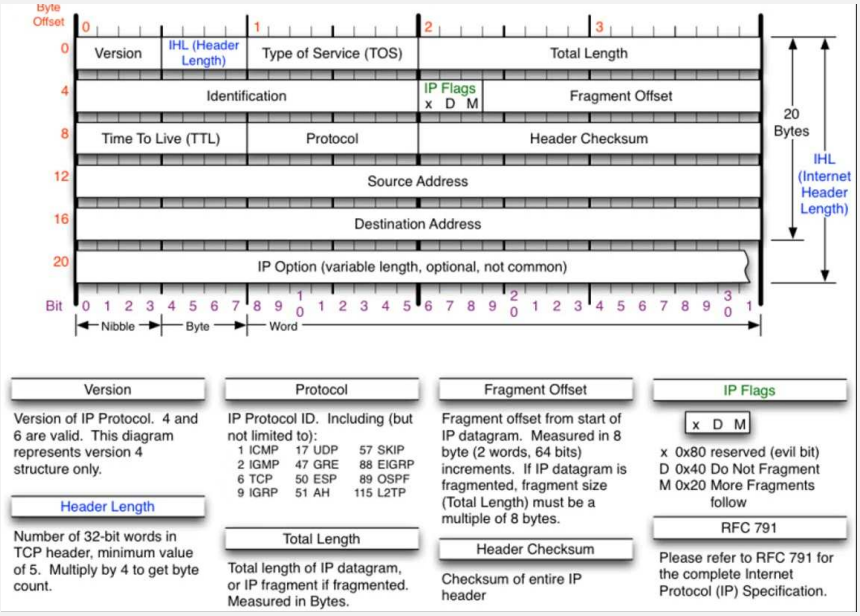


The network layer contains the **Internet Protocol** and is responsible for routing packets though the internet, and across networks with differing hardware, protocol stacks.

Definition: (IP) Internet Protocol

The main protocol use for this layer.

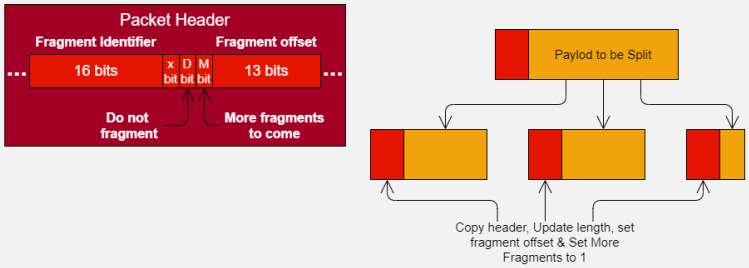
- Datagram Format
- Fragmentation (IPv4 only)
- IP addressing
- Packet handling



Note that:

- Type of Service is now called **DiffServ**.
- Most IP Options are not used (security issues).

Definition: Fragmentation

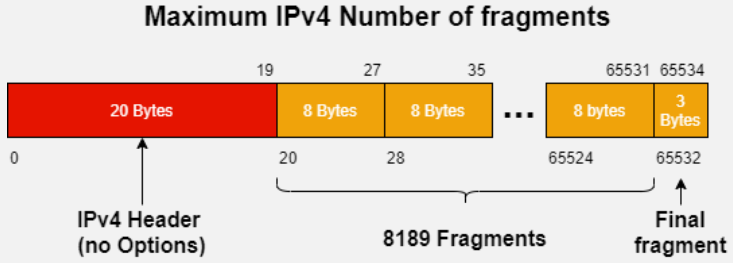


When the data sent to **IPv4** is larger than the **MTU** (maximum transmission unit) of the output link it is being forwarded through the **IP datagram** needs to be split (fragmented).

- Fragmentation at the start, or any intermediate routers.
- Only reassembled at the destination.
- Each fragment is identified by its 16-bit **fragment identifier**.
- Each fragment offset is the offset in units of 8-bytes (all fragments will be multiples of 8 bytes, plus a last byte).
- The **more fragments** bit (**M**) informs the receiver if it should expect more fragments to arrive, this is set when an intermediate router fragments a packet.

Maximum Fragments

It is not possible to fit the maximum number of fragments allowed by the 13-Bit fragment offset (8192) inside an **IP Datagram/Packet**.



- **Total Length** in the **IP Header** is 16-bits, hence maximum size is $2^{16} - 1 = 65535B$ (65536 sizes including 0).
- The minimum header (for IPv4) with no options is $20B$ long.
- Hence the maximum amount of data that can be stored as payload is $65515B$
- We can calculate the maximum number of $8B$ fragments as $\left\lfloor \frac{65515B}{8B} \right\rfloor = 8189$.

Hence it is only possible to have 8189 $8B$ fragments, with a single $3B$ fragment on the end.

Terminology

Networks Types

PAN	Personal Area Network	(e.g phone connected to PC, connected to bluetooth speakers)
LAN	Local Area Network	(Small network in a single geographical location, e.g home PC connected to home wireless network)
MAN	Metropolitan Area Network	(City wide network, e.g subway digital signalling network spanning large parts of the city.)
WAN	Wide Area Network	(Over a large geographical area, largest example is the **internet**.)

Devices

Definition: Repeaters/Hubs

In the **Physical Layer**/**Layer 1** and simply repeat wireless network traffic to boost signal. They do not processing of any kind, and just repeat any signal they intercept.

Definition: Switches/Bridges

In the **Data Link Layer**/**Layer 2** and make interconnections based on **MAC** addresses (which identify a given **NIC**).

Definition: Gateways/Multi Protocol Routers

In the **network Layer**/**Layer 3** to mak e decisions on forwarding packets (as well as splitting them - e.g fragmentation) based on **IP Addresses**.

To connect two **IP**-based networks together, a gateway (or router acting as one) is required between them.

Other Internet Protocols

Definition: The Internet

A collection of **Autonomous Systems** (separate networks, can run independently of each other) connected together by **backbones** (large long distance network infrastructure to link networks).

Designed in accordance with the principles of RFC 1958.

- Applications send data through a connection-oriented or connectionless transport layer protocol.
- Transport layer creates TCP Segments or UDP datagrams.
- Network layer converts TCP/UDP into **IP Datagrams**.
- Data Link Layer pass datagrams between routers, across networks.
- Physical layer (cables) transmits data.

Definition: (ICMP) Internet Control Message Protocol

Used for sending standardised control messages inside **IP Datagrams** (e.g ping is an ICMP Echo Request, Destination host unreachable).

- Each message has a type (e.g destination unreachable, time exceeded and more)
- Each message type also has a code (e.g destination unreachable (3), unsupported protocol (2))

For example **ping** sends an **ICMP** (type = 8, code = 0) which is responded to with an **ICMP** reply of (type = 0, code = 0).

Definition: Dynamic Routing Protocols

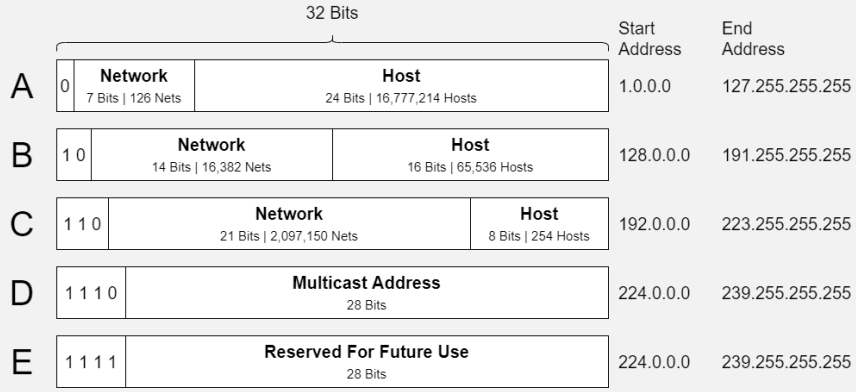
Include RIP (Routing Information Protocol), EIGRP (Enhanced Interior Gateway Routing Protocol), OSPF (Open Shortest Path First), BGP (Border Gateway Protocol). They determine how a packet is routed through a network, and create/manage routing and forwarding tables

IPv4 Addressing

- Addresses are contained in 32 bits.
- Displayed as $XXX.XXX.XXX.XXX$ where $XXX \in [0, 255]$.
- Each IP address is associated with an interface (not a host), so hosts may have more than one address.

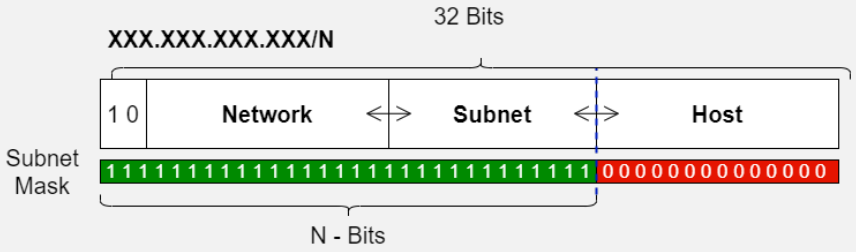
Definition: Classful Addressing

IP Addresses are split into several classes, each with a different length prefix to denote the organisation.



Critical Issue: All hosts on the network must share the network address section, so if an organisation hast hosts with several different IPs, it must publicly announce/claim multiple network identifiers.

Definition: Classless Addressing



A single network address is used for the entire organisation, internally addresses are divided into subnet addresses and host identifiers.

- External routers only consider the network address, and forward to a router of the associated organisation.
- Subnet routers apply the subnet maskand check if the IP is in their subnet, or if they need to forward to another subnet in the organisation.
- Once a host is found, routers know which interface to forward packets to.
- Network, Subnet and Host can have their sizes different for each network, according to the prefix length (N).
- The any-length prefix scheme is called **CIDR** (Classless Inter-Domain Routing).
- Routers attempt to match the longest prefix in order to select the correct network to pass a packet onto.

A simple python script for conversion is available with this lecture:

```python
# Super basic IP mask creation, string conversions and range.
from typing import Tuple

def ipv4_to_str(ip: int) -> str:
    assert(0 <= ip < 2**32)
    return ".".join([str((ip // (2**(8 * i))) % 256) for i in range(3,-1,-1)])

def get_mask(prefix_len: int) -> int:
    return (2**(prefix_len+1) - 1) * 2 **(32 - prefix_len)

def get_ipv4(ip: str) -> int:
    ip = ip.split(".")
    assert(len(ip) == 4)
    ip_num = 0
    for sub in map(int, ip):
        ip_num *= 256
        assert(0 <= sub < 256)
        ip_num += sub
    return ip_num

def apply_mask(ip: str, mask: int) -> str:
    return ipv4_to_str(get_ipv4(ip) & get_mask(mask))

def get_range(ip: str, mask: int) -> Tuple[int, int]:
    ip = get_ipv4(ip)
    subnet_mask = get_mask(mask)

    return (ip & subnet_mask | ((2**32 -1 ) & (~subnet_mask)), ip & subnet_mask)

# Example code
# print(apply_mask(input("Input IP:  "), int(input("Prefix:    "))))
# (bot, top) = get_range(input("Input IP:  "), int(input("Prefix:    ")))
# print(f"From {ipv4_to_str(top)} to {ipv4_to_str(bot)}")
```

It is also covered in the lecture below:

Lecture Recording

Lecture recording is available here

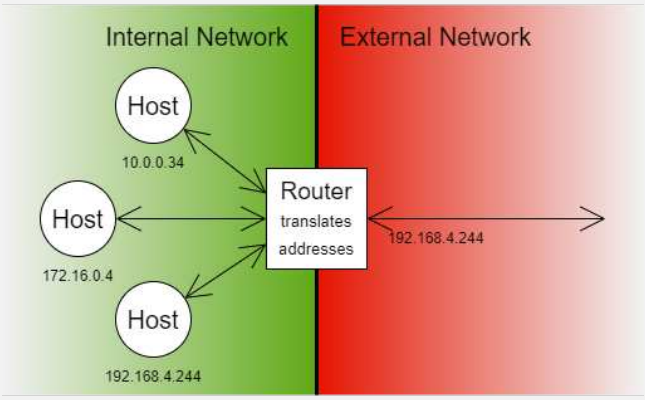
Definition: (DHCP) Dynamic Host Configuration Protocol

Allows hosts' interfaces to safely be assigned an **IP Address**.

- On boot, host broadcasts a **DHCP Discover** packet, a listening **DHCP** server will respond with an assigned **IP Address**.
- **DHCP** servers can maintain static mappings (hosts to addresses), and also assign different addresses each time a host connects.
- Hosts lease an **IP**, and must refresh periodically (to prevent hosts hogging an **IP**).

For example your router requests an **IP** from your **ISP**'s **DHCP** servers periodically.

Definition: (NAT) Network Address Translation



An attempt to solve the **IPv4** address shortage. Translates many private **IP**s into a single public **IP address**.

- Translation occurs when packets leave or enter the local network.
- On the local/internal network, every computer gets a unique **IP address**.

This is managed with a table of mappings between hosts & their processes (**Transport Layer**/**Layer 4** header contains this information) and ports on its own **IP**.

The following address ranges are also *private* and can only be used in local networks:

10.0.0.0	$\rightarrow$	10.255.255.255/8	16,777,216 addresses
172.16.0.0	$\rightarrow$	172.31.255.255/12	1,048,576 addresses
192.168.0.0	$\rightarrow$	192.168.255.255/16	65,536 addresses

There is much valid criticism of **NAT**:

- It violates the **IP Model** (that each IP Address uniquely identifies a host).
- It changes the internet from connectionless to connection oriented (as router must keep track of connections, associate with a mapping for translation).
- It violates the fundamental rule of the protocol stack: That layers do not make assumptions about protocols above. As **NAT** uses **Transport Layer** information, if new transport protocols are used, **NAT** does not work.
- It cannot easily support new transport protocols (due to the previous port).
- Many Peer-to-Peer protocols require full connectivity between hosts which **NAT** cannot provide. Hence prot forwarding, TURN relays, NAT punching holes, 3rd party servers and other solutions are required.

Special IP Addresses

0.0.0.0/0	The **default route**, used when no other **IP address** matches.
0.0.0.0/8	This host on this interface. Must not be sent, only used to acquire an **IP Address**.
127.0.0.0/8	"Loopback" (reference to the host), that can be sent (127.0.0.1 is localhost).
169.254.0.0/16	"Link Local" (something went wrong which acquiring an **IP Address**).

Ipv6

Definition: IPv6

Intended to fix **IPv4**'s address shortage (**IPv6** has $\approx 3.8 \times 10^{38}$ addresses), while also:

- 128 bit addresses, displayed in hexadecimal (e.g 2001:630:12:600:1:2:0:10b)
- Reducing routing table size and simplifying the protocol for higher performance.
- Improving security.
- Better support for "type of service" (now DiffServ in **IPv4**).
- Support scopes with multicasting (sending a packet to many hosts in a certain scope, e.g network).
- Support roaming hosts without address changes (more on roaming scopes here)
- Better support coexistence of old and new protocols, while making it easier to develop new ones.

It has several key differences with **IPv4**:

- Fragmentation is done by end-systems. Hence packets do not have to deal with fragmentation.
- No header checksum, it is redundant as both the **Transport** and **Data Link** Layers have error detection features.
- Fixed length header is easier to process, **IPv4**'s options were almost always unused.
- Better modularity for extensions.

Extensions are done by placing an extending header after the **IPv6** header.

- Hop-by-hop options (provides information to routers, e.g quality of service)
- Routing (Provides a full or partial route to follow)
- Fragmentation (Information for end systems)
- Authentication (Sender identity verification)
- Encrypted payload (Info on the payload)
- Destination Options (extra information, e.g mobile **IP** (moving networks but maintaining the same IP address))

Routers

Lecture Recording

Lecture recording is available here

Routing Requirements

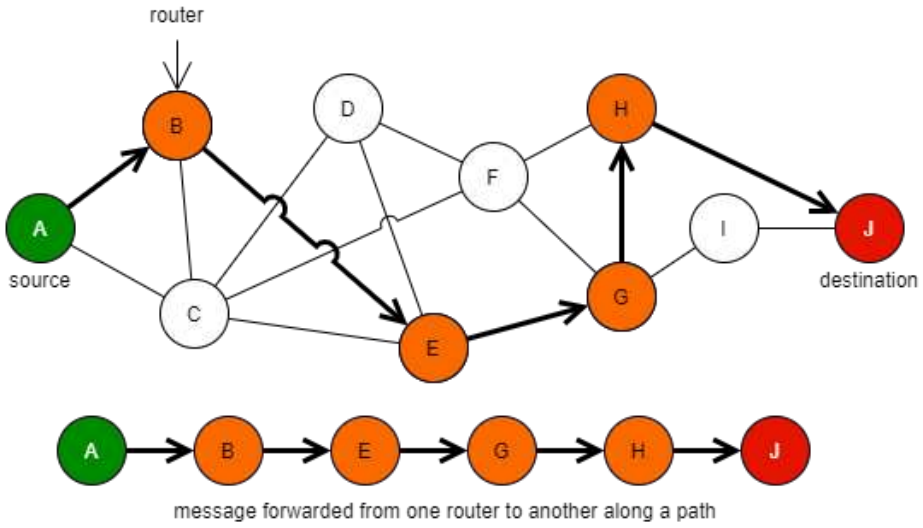
A routing system needs to provide facilities for moving data from source tro destination as well as:

- Allow for multiple hops on nodes in the network.
- Be able to consider the topology of the network to choose appropriate routes.
- Perform load balancing.
- Allow for/deal with network heterogeneity (different types of networks connected).

The internet is a **packet switched** network, providing a connection-less service (no setup/teardown phase, each message is independent and self contained with no steup/tear down phase).

The internet is also a best effort service, and does not provide guarentees of delivery, maximum latency, bandwidth, congestion indication or in-order delivery.

Datagram Networks



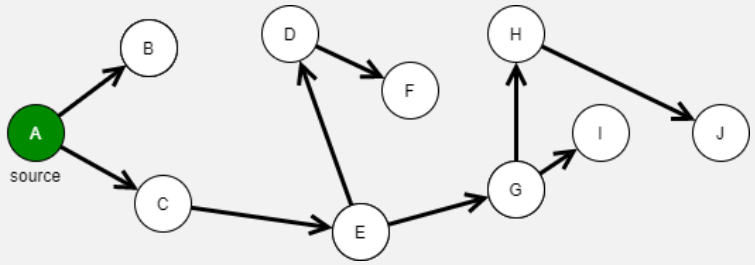
- There are potentially many paths for the same source to destination.
- Paths can be asymmetric (path from $A \to J$ is different from return path $J \to A$).

Each router uses a **forwarding table** to determine which router to forward packets to, based on their final destination.

Routing

Definition: Sink Tree

A tree from a source node, to every destination node, where each path in the tree is the optimal route/shortest path to the destination. As a tree, there are no cycles.



Definition: Djikstra's Algorithm

Each arc is labelled with a cost (e.g delay, hops, some function of parameters potentially including congestion).

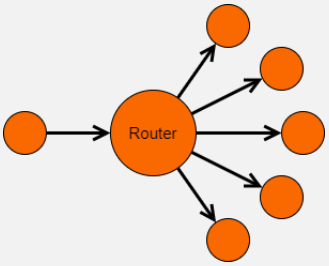
1. Visited = {}
2. Add the start node (at $distance = 0$)
3. Loop wihile there are unvisited nodes:
   (a) Label each fringe node (one arc from a visited node) with the minimum of (weight from visited node) + (weight of connecting arc).
   (b) Add the closest node (based on previous step)
4. The shortest paths are the distances each node is labelled with.

Routers cooperate to find the best routes between all pairs of nodes in the network.

Definition: Shortest Path Routing (SPR)

Use **Djikstra's Algorithm** to determine the shortest routes from each router, to every destination. The forward packets accordingly.

Definition: Flood Routing



Forward incoming packets to every outgoing link. Except for the link the packet was received on.

We can use several strategies to avoid drowning the network in packets:

- **Hop Counter** Disgard a packet after it reaches a maximum number of hops.
  Must decide on a correct number of hops to avoid drowining, but allow packets to reach their destination.
- **Forward Once** If recieving the same packet again, do not forward again.
  This solves the issue where packets are sent though cycles (e.g $A \to B \to C \to A$), however it requires storing sequence numbers per source address to identify packets.
  Furthermore must decide on how lonbg the sequence numbers are stored.
- **Selective Flooding** Flood only in selective directions.
  Rather than flood every outgoing link (except packet source), send to only some of the outgoing links, based on some heuristic (to decide which diurections make the most sense).

Flooding always chooses the shortest path (all paths explored in parallel), however it creates significant overhead (must send many packets at every router in every path).

Use case is when the packet must be received, but when the route to the destination is unknown.

Definition: Distance Vector Routing (DVR/Bellman-Ford)

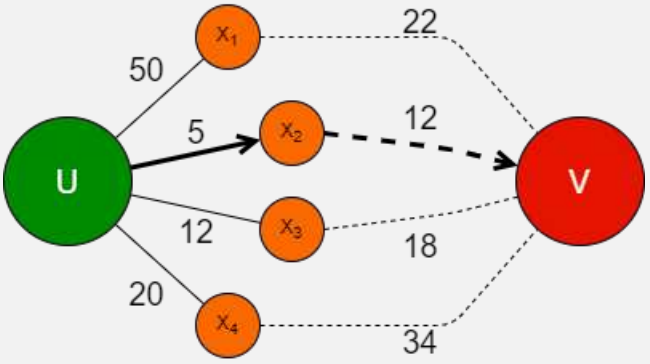
Both **Flood Routing** and **Shortest Path Routing** are static and do not take into account current network conditions (e.g network load). **DVR** is dynamic and does consider this.

- Every router advertises its costs to each destination.
- Router's use their cost to neighbours, and their neighbour's cost to determine how to route packets for the minimum cost, and to then update and advertise their cost.

This is expressed by the **Bellman-Foprd Equation** for the cost from node u to node v:

$$D'_u[v] = \min_{x \in neighbours(u)} (cost(u, x) + D_x[v])$$

(Cost from u to v is the minimum of the cost from u to a neighbour, plus the cost of that neighbour to v)



However there is a **count-to-infinity** problem. When a node goes down, and routers continually update their costs based off eachothers, resulting in the cost incrementing constantly.

$$A \to C = B \to C + 1$$
$$B \to C = A \to C + 1$$

We can resolve this by defining infinite cost as:

$$cost\ \infty = \text{longest acceptable path} + 1$$

Examples of distance vector algorithms include **RIP**.

Definition: Link State Routing

A replacement for **DVR**.

- Broadcasts alll information on network topology to all rputers.
- Each router can use this to calculate a **sink tree**.
- Identifies neighbours using a special "hello" packet, to which neighbours respond with their network address.
- Link costs is determined using a special "echo" packet, and measuring the **rount trip delay**.

The basic algorithm for each router is as follows:

1. Get direct neighbours & their network addresses (so they are uniquely identifiable on the network) ("hello").
2. Calculate the cost of sending packet to each neighbour ("echo").
3. Build a **Link State Advertisement/LSA** describing the router, its connections to its neighbours.
4. Send the **LSA** packet to every router on the network (**flooding**).
5. Recieve **LSA** packets from every other router on the network.
6. Now the router has the status of all links between all routers, it runs dijkstra's algorithm locally, to create a **sink tree** for use in routing.

This algorithm allows better routes to be chosen using current network conditions.

However routers may redirect traffic towards the best routes so much, that these routes become overloaded, and are no longer the best routes.

An example of **Link State Routing** is **OSPF**.

We can compare **DVR** and **Link State Routing**:

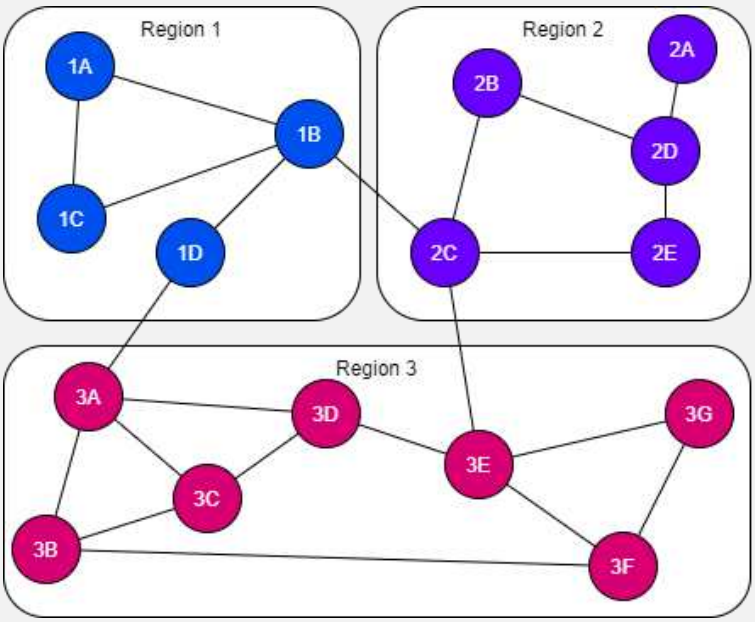
	Distance Vector Routing	**Link State Routing**
Network Info	Local	Global
Computation	Global	Local
Synchronisation	Gradual (routers update & advertise)	Instance (once the **SPR** computation is done)

Definition: Heriarchical Routing

All the previous routing methods are difficult to scale as each router needs to know about all other routers. On the scale of the internet, memory and processing power requirements would be too high.

To solve this the network is split into regions. With different algorithms used for **intra-region** (inside regions) and **inter-region** (between region) routing.

- Can scale the network massively.
- Suboptimal routes chosen between node in different regions, (we just know which region to go to)
- Can use other algorithms within the regions, each group can effectively been its own, autonomous network with its own design, structure, routing algorithms.
- 2/3 levels of regions are generally enough.



We can consider each region a different network, all connected together.

Definition: Broadcast Routing

Another way to solve scaling, send to every host on a network (only feasable for **LAN**s and small **WAN**s), even through we do not know the route to a destination, by sending the packet to every host, it is sure to be recieved.

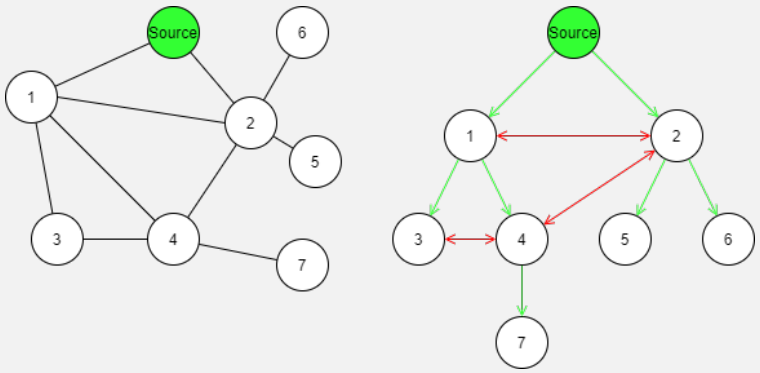
- **Send packets individually** Not efficient
- **Flood Routing** Acceptable if the flood can be limited
- **Multi Destination**
  A list of destinations is sent with the packet. Routers check this list, splitting the list and forwarding the packet to its neighbours.

  However the packet must contain all destinations (size limitations).
- **Multicast Routing** See definition.

Definition: Reverse-Path Forwarding (RPF)

Used to construct spanning trees from a router, for a low cost.

- Every router forwards/broadcasts a packet on every adjacent router, except the router the packet was recieved from (like **flooding**).
- Routers only accept packets if it is on a direct path from the source.
- Hence the paths of packets forwarded, and accepted represents a spanning tree from the source router.
- Note: This can also be used to detect and prevent IP spoofing (as packet will come from an odd path, given the spoofed IP)



Definition: Multicast Routing

Sending a message to a subset of the nodes (groups, each with a group id).

A first solution is to construct a spanning tree at each router, and prune all paths that do not contain members of the group we want to send to.

Alternatively we can use **Core based trees**.

- A single spanning tree per group, with a root (central to the group to reduce cost between it and members of the group).
- To send a multicast message to the group, just send it to the core, which will retransmit to all nodes in the group.
- Not optimal for all sources, however scalable and much lower overhead.

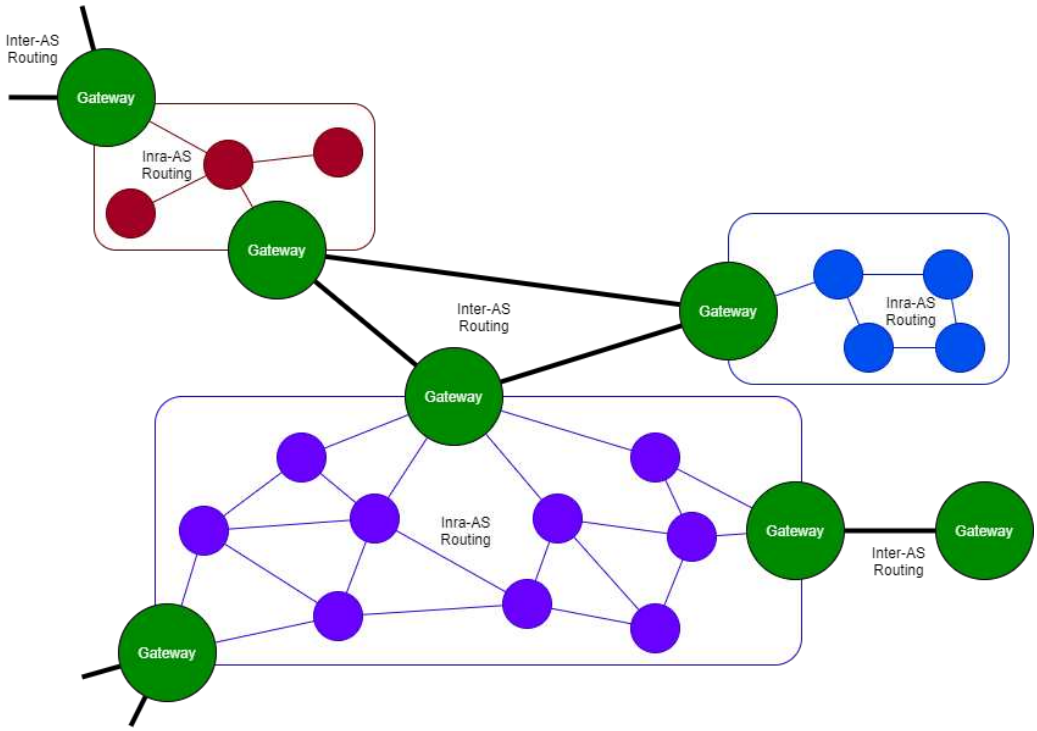
Note this is how it is done in the internet (multicast IP Address/Broadcast Address is effectively the core for an entire network).

Inter-AS Routing

Lecture Recording

Lecture recording is available here

Inter-AS Routing	**Intra-AS Routing**
Routing between autonomous systems (e.g between two different networks)	Routing within an autonomous system (e.g within a **LAN**).
Autonomous systems can be hetergeneous (different protocols, routing algorithms, topologies, hardware) so use **Gateways** to link between.	Within an autonomous system (depending on size) typically uses one design controlled by one organisation.
Cannot support optimal routes at scale, but makes best attempt practical.	Attempts to provide optimal routes on a smaller network.



External → *External*	Gateway (Inter-as router) recieves packet, if it can forward to the next gateway it does, if another gateway in its AS can send it, the packet is routed by intra-as routers through the network to the gateway to send on.
External → *Internal*	Gateway recieves the packet, then sends it on to intra-as routers to route to the destination.
Internal → *external*	Intra-as router sends packet on to a gateway that advertises it can reach the destination, gateway then forwards to the relevant gateway, routing across networks.
Internal → *Internal*	Intra-as routers route packets.

Definition: Open Shortest Path First (OSPF)

A **link state routing** algorithm to replace **RIP** (a distance vector routing algorithm).

- Algorithm is publically available for anyone to implement.
- Supports different distance metrics (hops, delays, etc).
- Can adapt dynamically to changing network topology (nodes added or removed).
- Supports routing based on **ToS** (Type of service).
- Supports load balancing (not overwhelming routers e.g by flooding)
- Offsers some security features (though some have been compromised).

It also supports hierarchical routing. It is possible to split an AS into several "areas", then each area has one or more "area border routers" which are in a "backbone area" (contains all border routers) to route traffic between the "areas".

Definition: Border Gateway Protocol (BGP)

The Inter-AS routing protocol used in the Internet.

- Adjacent routers maintain connections for reliability.
- Gateways transmit reachability information to routers inside an AS.
- Good routes determined based on reachability information and routing policies.
- Routers only check for & discover new paths if allowed.
- Uses a **path-vector** protocol (based on **DVR** but paths instead of distances are announced).

BGP advertises routes/paths to networks:

- Destinations are denoted using the address prefixes (see subnetting).
- ASes may not propagate an advertisement by a gateway, as doing so would imply the network is willing to carry traffic through the AS.
- Routers can aggregate prefixes (merge prefixes together)

Example: Aggregating Prefixes

We can merge several ips (with prefixes) into one, with a shorter subnet mask.

$$\left.\begin{matrix}127.134.126.0/24\\127.134.127.0/24\end{matrix}\right\} \rightarrow 127.134.126.0/23$$

Here the 24th bit is the only difference between the two, and hence we can reduce the subnet mask size.

This is also referred to as *supernetting*.

In **BGP** each AS has a unique identifier (**Autonomous System Number** (**ASN**)) and several attributes:

- **AS-PATH** Sequence of AS identifiers through which the advertisement was sent
- **NEXT-HOP** Next IP address to forward packets towards advertised destination (resolves ambiguity when there are multiple AS reachable through multiple inferfaces)

The **BGP** import policy determines to accept or reject route advertisements.

Router preference is ranked according to:

- Policy used.
- Shortest **AS-PATH**
- closest **NEXT-HOP** router.

The *count-to-infinity* problem is solved by *path exploration/hunting* (actively seeks paths), furthermore routers can send withdrawal messages (e.g before taking a node down, can tell others to remove the path).

This allows routers to identify invalid paths, at the expense of some delays.

50005 - Networks and Communications - Lecture 7

Oliver Killane

18/04/22

WireShark

Lecture Recording

Lecture recording is available here

Definition: Credential Resuse/Stuffing

Using previously leaked/found eail-passoword combinations on other services. Useful when users reuse passwords for mutliple services.

Definition: Network Monitoring / Packet Sniffing

Listening on a network and reading packets where you (your **NIC**) are not the intended recipient.

Definition: Code/SQL Injection

Executing code on a system by passing it through normal data collection.

e.g if text entered in a website is directly substituted into a database query, by writing SQL ocde in the text entry, we can alter (or add another query) to the query generated.

Definition: Session/Cookie Hijacking

Using the cookie/authentication token form another user's session to get authenticated.

Definition: Wardriving

Identifying and compomising unsecured wireless networks.

e.g installing spyware on usecured home routers.

Definition: Wireshark

Wireshark is a network protocol analyser. It allows users to capture, analyse & deconstruct packets to analyse traffic on a network.

WireShark Modes

Promiscuous Mode

- Works for Wired and wireless.
- **NIC** does not drop packets, retains all received packets.
- When wireless, only listens on the connected network.
- Some **NIC**s ignore this (considered *impolite* and easily abused).

Monitor Mode

- Only works on wireless networks.
- **NIC** listens on all networks in range/that it can receive from.
- Wifi networks secured with authentication (e.g password) will appear scrambled (encryption).
- Most **NICs** do not support this, may require new drivers or a special **NIC**.
- **WinPcap** (windows) does not support though **AirPcap** and **Npcap** on linux do.

Sniffing Ethics

When monitoring a network, it needs to be a netywork you have permission to monitor (either wired or wireless)

Lecture Recording

Lecture recording is available here

WireShark Packet Capture

Location	Can Capture
Hub	Local traffic, Broadcast/Multicast, (Promiscuous Mode) Entire Network.
Switch	Local Traffic, Broadcast/Multicast, (Promiscuous Mode) Network connected to the same switch port.
WLAN	Local Traffic, Broadcast/Multicast, (Promiscuous Mode) Entire WLAN, (Monitor Mode) All wireless packets physically receivable/in range.

We can provide wireshark with authentication to allow it to decrypt packets on for protected networks (e.g provide the RSA key for SSL, or password for WPA/WEP).

WireShark Display Filters

Can hide or select packets based on contents, destination & source address and more. And can build up complex filters.

Example: WireShark Capture Filter

```
1 http.request.method == GET &&
2 http contains "password" &&
3 (ip.src != 10.43.54.65 || ip.dst != 10.43.54.65)
```

More Examples and filter building tutorial.

NMAP

Definition: NMAP

A network scanning tool which uses sends raw **IP** packets and monitors responses & determine the services provided by the network and its hosts.

It can be used to detect vulnerable hosts on a network.

We can scan networks using the gui, or by using the command line utility:

```
# Quick scan without checking ports
nmap -sn <ip address>

# Scan a range of ports on a host
nmap -p <start port>-<end port> <ip address>

# Scan all ports on a host
nmap -p- <ip address>

# Scan without discovery (even if the host wont respond to a ping, we can still check
    ↪ its ports)
nmap -Pn <ip address>
```

50005 - Networks and Communications - Lecture 8

Oliver Killane

18/04/22

Lecture Recording

Lecture recording is available here

Preamble

Device Terminology

Device	Layer	Description
Repeaters/Hubs	Physical	Boost signals by repeating all received.
Switches/Bridges	Data Link	Make interconnection decisions based on **MAC Addresses**.
Multi-Protocol Routers/Gateways	Data Link	Forwards (and possibly fragment) packets. Use **IP** addresses.

Transport Layer and **Application layer Gateways** also exist.

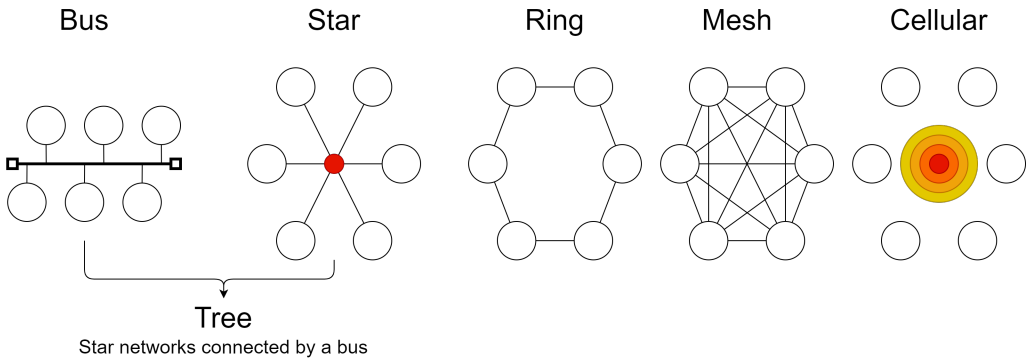
Routers act as **Gateways** to connect **IP**-based networks.

Network Types

Network Types ordered by size (small → large).

PAN	Personal Area Network
LAN	Local Area Network
MAN	Metropolitan Area Network
WAN	Wide Area Network

Network Topologies



Data Link Layer Protocols

802.3	Ethernet	LAN	1-persistent CSMA/CD	Star/Bus
802.4	Token Bus	LAN	Token Passing	Bus/Tree
802.5	Token Ring	LAN	Token Passing	Ring
802.6	DQDB	MAN	Distributed Queue	Bus
802.9	isoEthernet	LAN	Ethernet + ISDN	Star/Mesh
802.11	WiFi	LAN	CSMA/CA	Cellular
802.12	100BaseVG	LAN	Handshaking from hub	Star/Tree
802.15	Bluetooth	PAN	Adaptive FHSS	Cellular
802.16	WiMAX	MAN	Connection oriented	Cellular
802.17	Resilient Packet Ring	LAN to WAN	Distributed Queue	Ring

Ethernet

Definition: Ethernet

A **Data-Link Layer** protocol used for **LAN/MAN/WAN** communications.

- Specification created in 1980.
- Became IEEE Standard 802.3 in 1983.
- Originally coaxial cable (**10BASE5**), $\approx 2.94 Mbps$
- Currently fibre optic, twinaxial (two coaxial) cable $\approx 100 Gbps$.

Ethernet Cables

UTP	Unshielded Twisted Pair
STP	Shielded/Screened Twisted Pair
FTP	Foiled Twisted Pair
SFTP	Shielded & Foiled Twisted Pair

The most popular is **UTP** of which the most used version is **Cat5e**.

Cate6a,**Cat7a** exist and **Cat8** in development.

Cables use shielding to protect against **ElectroMagnetic Interference** (**EMI**) (e.g crosstalk between the wires) causing errors in data transmission.

Lantenna Attack

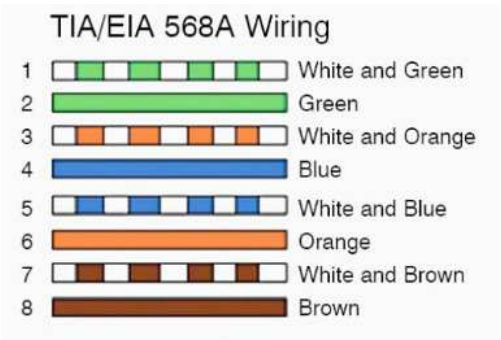
Sheilding can also help to reduce electromagnetic leakage from ethernet cables that can be sniffed and exploited.

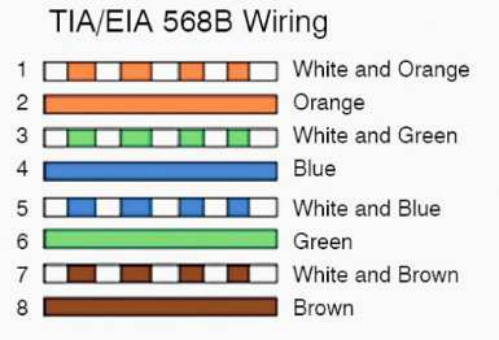
A team from Ben-Gurion Univeristy have demonstrated this by attacking their own basic air-gapped networks. Their paper can be read here.

Cable Code	Name	Cable	Max Length	Topology
10Base5	Thick Ethernet	1/2 inch coaxial cable	500	Bus
10Base2	Thin Ethernet	75-Ohm coaxial cable	180	Bus
10BaseT	Twisted Pair Ethernet	Category 3 UTP	100	Star
100Base-TX	Fast Ethernet	Category 5 UTP	100	Star
100Base-FX	Fast Ethernet	Fiber Optic	185	Star
1000Base-T	Gigabit Ethernet	Category 6 UTP (4 pairs)	100	Star
10GBase-T	10 Gigabit Ethernet	Category 6a UTP (4 pairs)	100	Star

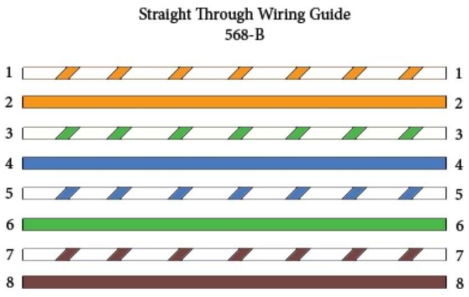
Ethernet Pinouts

There are two main pinout wirings, the only differences are the yellow and green cables are in swapped positions:





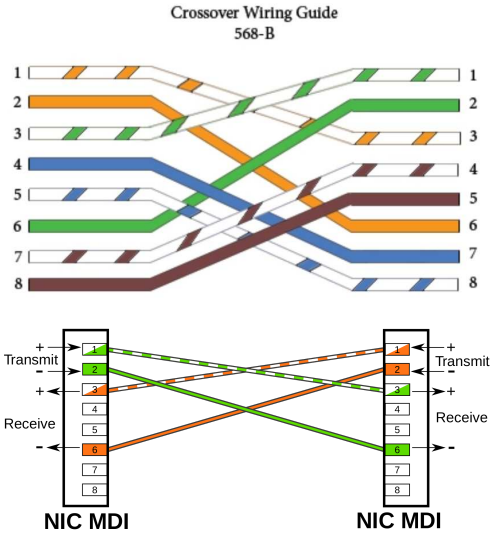
Straight Through



Communication between different OSI layers (e.g switch → router).

This is also called the **Media/Medium Dependent Interface** (**MDI**)

Crossover

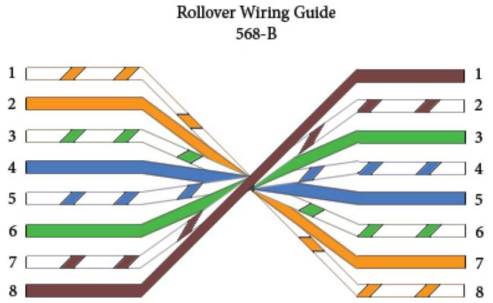


Communication between devices on the same OSI layer (e.g switch → switch).

This is also called the **Media/Medium Dependent Interface with Crossover** (**MDIX**)

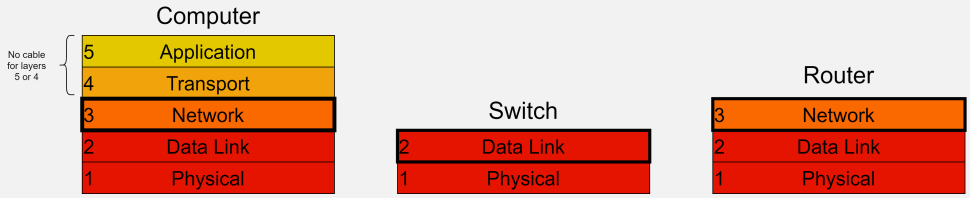
The crossover connects the transmit on one side, to the receive on the other side and vice-versa. As the devices are the same OSI layer, they can communicate back and forth through this.

Rollover



Used to directly tap into a network device (e.g a console to debug a router setup issue)

Example: Connect Computer to Router



When we connect to a router, we will be connecting to a switch (with a router attached).

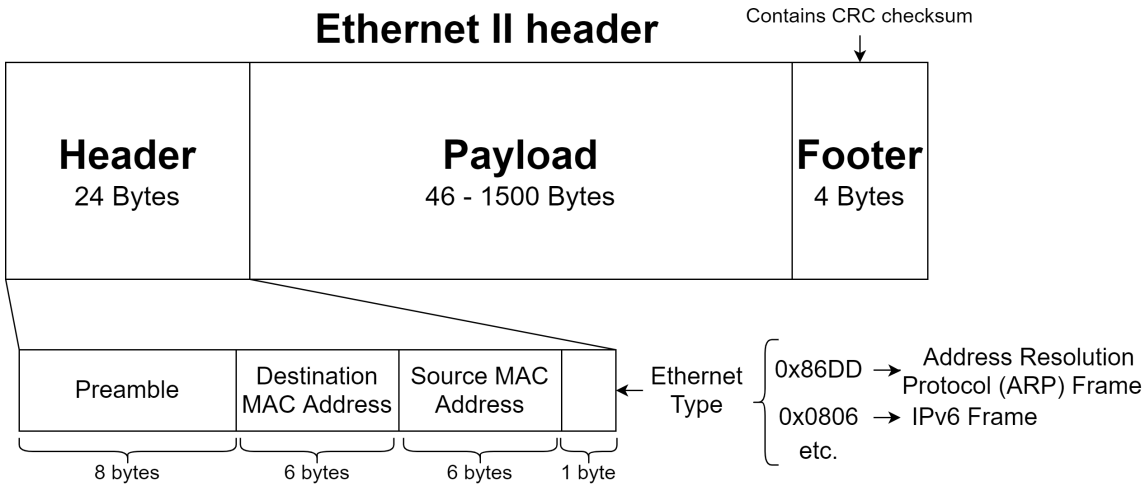
We consider the computer as **Network Layer** as that is the highest we can directly connect, and the switch as **Data Link Layer**. Hence a straight-through connection is used as they are different **OSI Layers**.

Ethernet Frame

Definition: Octet

A byte/8 bits. It is used as an unambiguous term as on older machines the definition of a byte was hardware dependent (e.g like the term word). The wikipedia page for byte contains many interesting sources with examples ranging from 1 bit to 48 bit bytes).

Nowadays a byte is always 8 bits, so can be used interchangably with **octet**

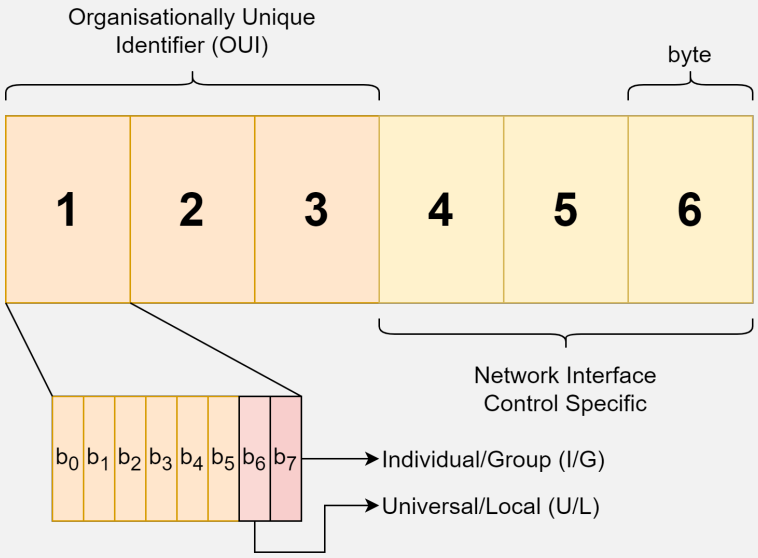


IEEE MAC Addressing

Definition: MAC Address

A 48 bit (6 byte) address etched into IEEE 802 conforming **NIC**s as a unique identifier an **NIC**.

$$< byte >:< byte >:< byte >:< byte >:< byte >:< byte >$$



I/G	0	Unicast, intended for one **NIC**.
	1	Multicast, **NIC** accepts based on other information (e.g list of accepted multicast addresses).
U/L	0	Globally Unique, enforced by the **OUI**.
	1	Locally Unique, e.g new MAC address burned in by network administrator.

A special address is the **Broadcast Address** $FF:FF:FF:FF:FF:FF$ (group Multicast, locally administered).

It is possible to filter sniffed traffic using mac addresses in applications such as **WireShark**.

eth	All Ethernet based traffic.
$eth.addr == 12.00.06.14.3a.fe$	All traffic to/from MAC Address $12:00:06:14:3a:fe$.
$eth.addr != < MAC\ Address >$	All traffic except $< MAC\ Address >$.
!(. . .)	Filter for all except frames traffic matching . . . .
$(eth.dst[0]\&1)$	Multicast only (bitwise and of 1 and the first byte).
$!(eth.dst[0]\&1)$	Unicast only.
$(eth.dst[0]\&2)$	Locally Unique addresses only.
$!(eth.dst[0]\&2)$	Globally Unique addresses only.

Switch

Definition: Switch

Allow many devices to be connected to the same subnet.

- Forwards messages to ports based on the MAC address of the device connected to each port.
- If the switch cannot determine which port to send to, it will send to all (flood).
- Uses a **Forwarding Information Base** (**FIB**) **MAC** table to remember addresses associated with ports.
- Difficult to network-sniff as packets are only directed to intended recipients.
- Can connect them to other **switches** or **hubs**, allowing networks to be connected together.
- Replaced network Bridges.

To allow messages to leave the subnet, a router and an **IP Address** provided by a **DHCP service** or set statically.

Store-and-forward Switching

Once a whole frame is received, check its integrity using the checksum. If it is correct, forward to the correct port based on the frame's destination **MAC Address**.

- Forwarding is slower, as the switch must wait to recieve the entire frame.
- Can check for errors at the switch, and drop the frame if invalid.
- Supported by **bridges** and **switches**.

Cut-through Switching

As soon as the enough information is received (e.g the destination address), start forwarding the packet.

- Faster forwarding.
- Does not error check, so final reciever must check footer checksum.
- Only supported by **switches**.

Wireless

Definition: Wireless Access Point (WAP)

Standardised by IEEE 802.11 for wireless communication.

- Uses $2.4Ghz$ or $5Ghz$ radio (open for unlicensed use).
- Acts as a hub (repeats all recieved)
- Can connect **WPAs** together to extend range.
- Can also act as a bridge to connect to a wired network.

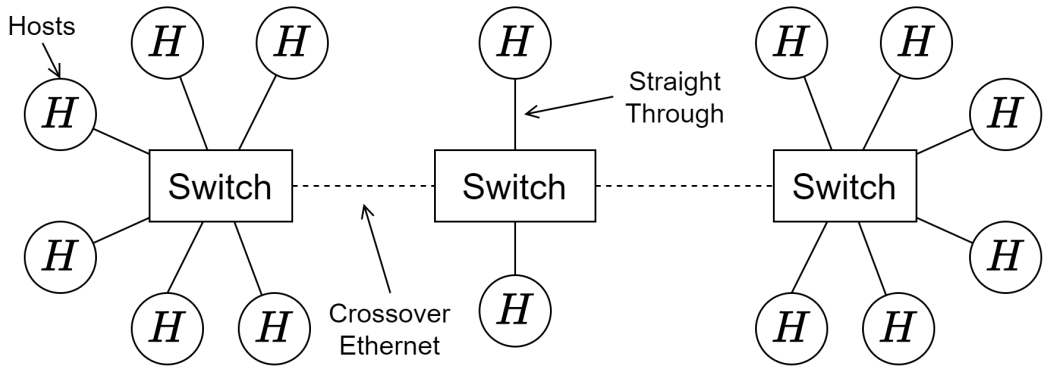
Note that it is very easy to network-sniff, as all devices within range of the **WPA** can receive frames.

Topologies

Lecture Recording

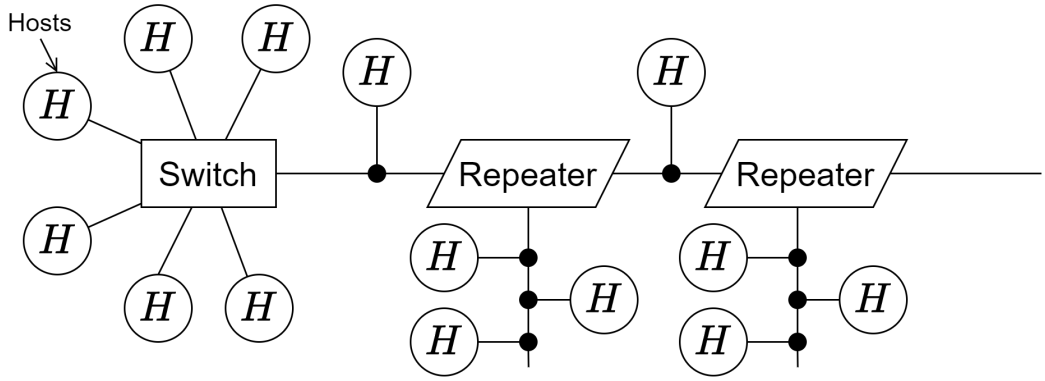
Lecture recording is available here

Switched Ethernet



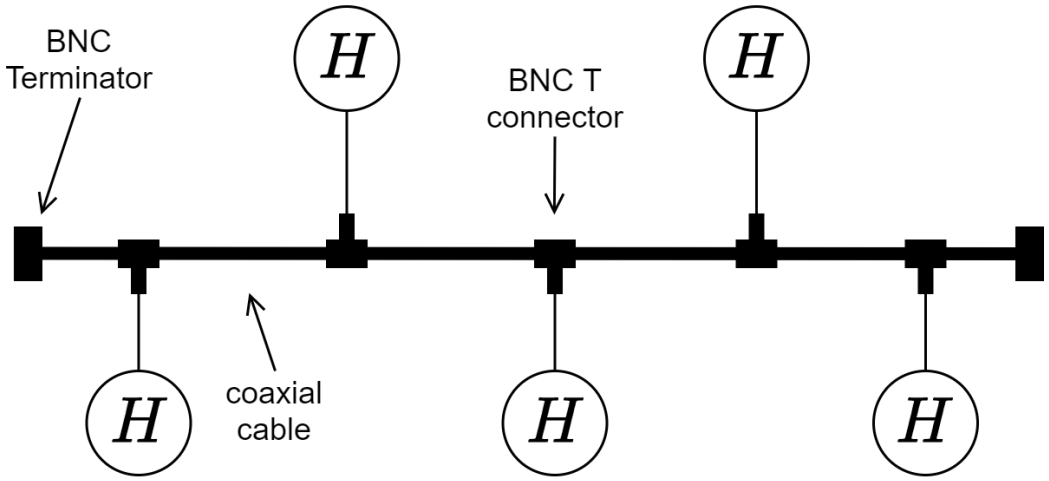
- Each switch port connected to another machine, or a host.
- Collisions avoided using small collision domains (the hosts on a single switch)
- Ideal, but expensive.

Internetworking Ethernet



- A combination of networks sharing a medium (e.g a cable).
- Repeaters boost signal to extend the range of the network (longer length of cables).
- Hubs are used (forward recieved frames out of every port) (use generally discouraged).

Bus

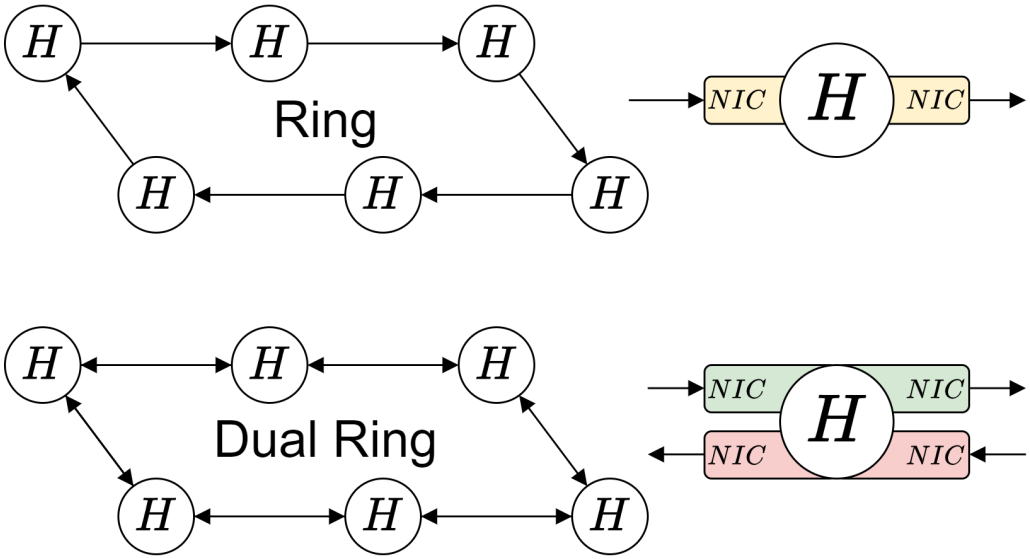


- Data travels up and down the **bus** coaxial cable.
- To add new hosts, the cable must be cut, and a **BNC T** connector used to connect the cable back together with the new host.
- Terminators at the end of the cable absorb signals, preventing them from being reflected back down the cable.

Cheapernet

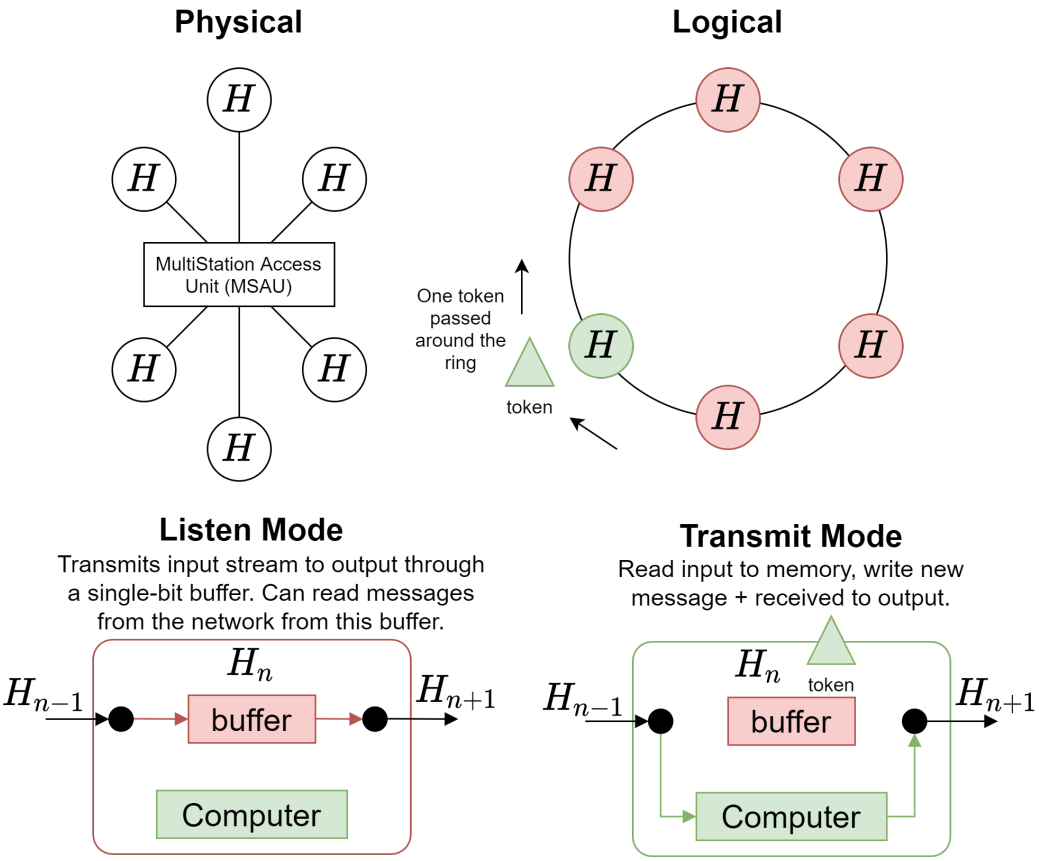
10BASE2 is the code for the coaxial cable used for ethernet on **LAN**s. It is rarely used. The wikipedia page covers the connector types and its replacement.

Ring



- Each host needs two **NIC**s for **Ring** and four for **Dual-Ring**.
- If a link is removed, the network fails (every host is a single point of failure) unless the network is designed to adjust (change flow, become a bus).
- For **single-ring** data flows one way, for **dual-ring** both ways.
- **Dual-Ring** allows for one ring being cut.

Token Ring (IEEE 802.5)



- Hosts connected to the **Multistation Access Unit** (**MSAU**), connected logically as a ring.
- One host has the token at a time. When a host has the token it is in **transmit mode** and can write to the network.
- No collisions as only one host can have the token & write at a time.
- All hosts can listen to the network (**listen mode**).
- Do not have to worry about frames not fitting inside the ring, as the host holding the token buffers with its own memory.

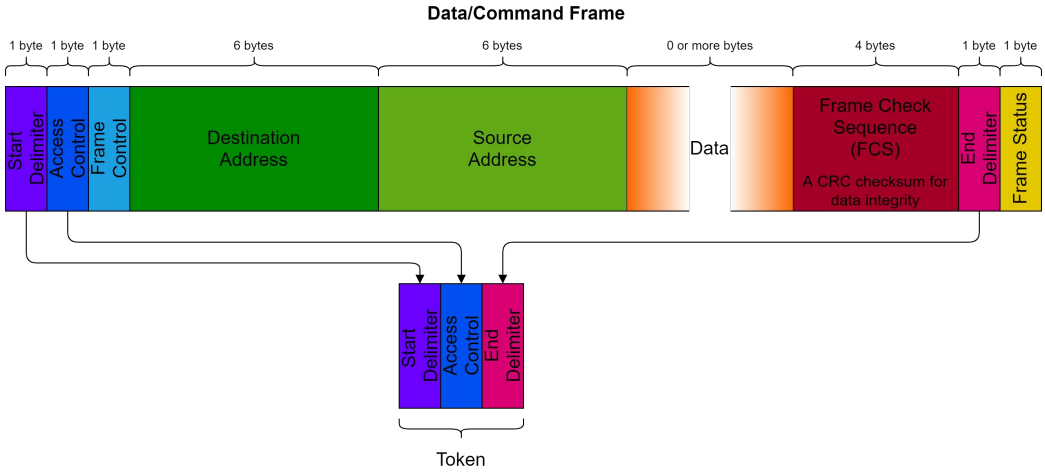
When a host has the token and wants to transmit new data:

1. Direct all recieved frames to memory.
2. Write new frame/s to the output.
3. Wait until the frame written is recieved (meaning it has traversed the entire ring)
4. Write all recieved (except frame written) to the output.
5. Pass token to next host.

Definition: Early Release

Hosts do not wait for a sent frame to traverse the ring before passing on the token. This increases the performance considerably.

Token Ring Frames



When there is no frame to be sent, the token is passed around the circle.

Frame Check Sequence	**FCS**	A **CRC checksum** used to ensure data integrity.
Frame Status	**FS**	Defines if the address was found & frame received/copied from the ring (set by **listening mode** destination host). When set it can be removed from the ring.
InterFrame Gap	**IFG**	Gaps between frames, smaller gap means less inefficiency, but need enough to recognise the start and end of frames. Defined by the protocol used.

Token Ring Priority & Reservation

A priority scheme can be implemented so hosts can only claim the token if the priority level of the the data they want to send is as high as the token's priority.

A host in **listen mode** may have high priority data to send. It can raise the reservation priority in the frame. When a token is created, it is created with the priority of the reservation bits in the frame.

- Can claim token if priority of data is as high as token priority.
- Low priority data may be delayed indefinitely.
- High priority data will be sent quickly (good for real-time applications).
- Used for **LAN**s, so it is assumed we can trust hosts to not abuse priority.

Token Ring Acknowledgement

The receiver can alter the **Frame Status**:

$A = 1$	Destination host is working.
$C = 1$	Destination host has correctly read the frame.

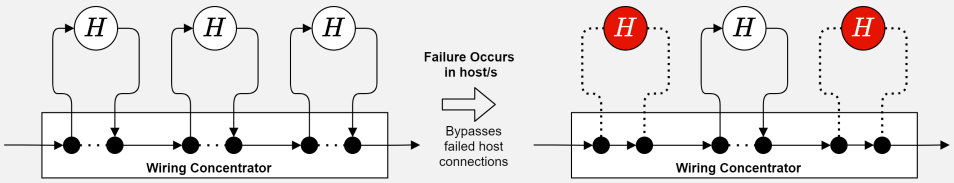
Ring Maintenance Complexity

- The **Frame Control** filed is used to create control frames.
- Frames may become orphaned (e.g never received, end up looping around indefinitely).
- One host is the **Active Monitor** and is responsible for generating tokens and removing orphaned frames.
- **Active Monitor** may fail, so any host must be able to become the **Active Monitor**.
- Contention rules/protocol needed to determine which host becomes **Active Monitor**.

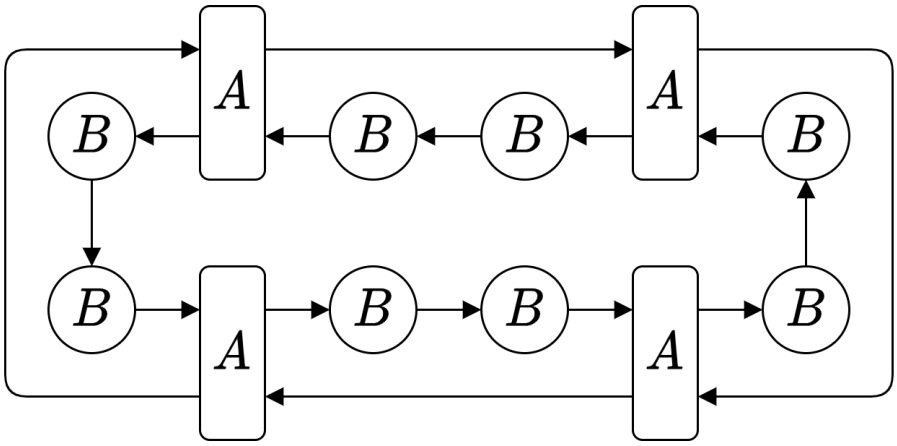
The above points add significant complexity to token passing and reduce reliability.

Definition: Wiring Concentrators

In order to improve reliability, when a host fails it can be bypassed.



Fibre Distributed Data Interface (FDDI)

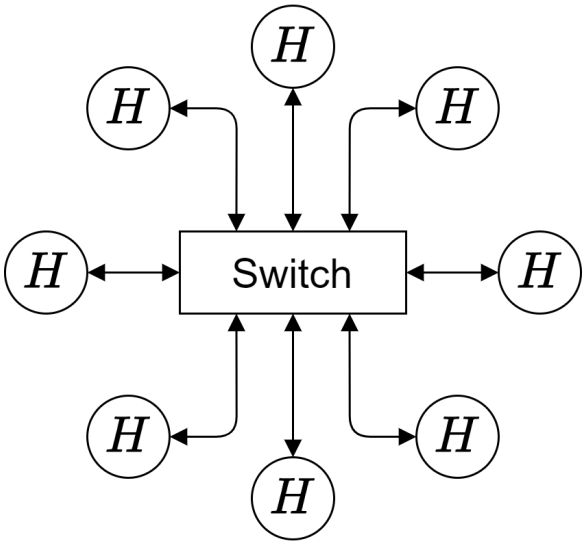


A ring-based token passing topology that was popular in the 1990s.

Hosts are divided into two classes, with one class (in our diagram class A) connected to both rings.

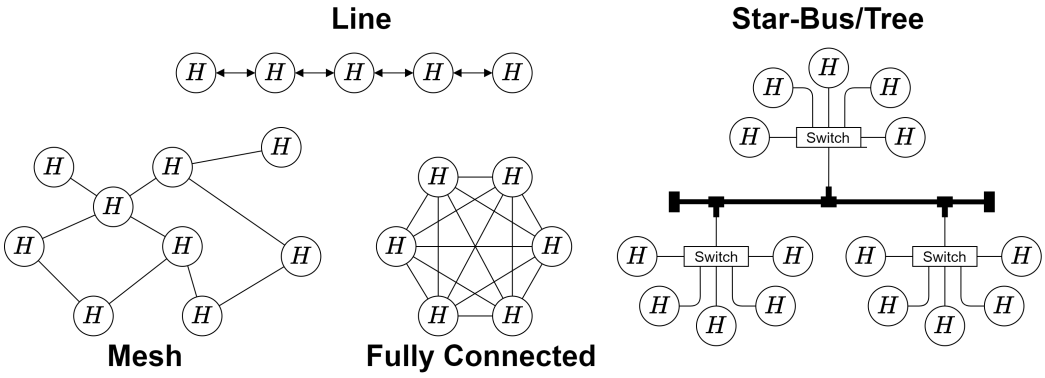
- Optical fibre cabling allows for networks to be geographically large.
- When a class B fails, data can be re-routed through class A hosts and the second ring.
- When a class A fails we can short-circuit two class As to create a single-ring (connecting two rings at disconnected ends).
- Rings can be up to $100km$ long, so FDDI must work with a length up to $200km$.
- No longer a popular.

Star



- All hosts connected directly to a **switch/multiport-bridge**.
- Any host can communicate with any other (provided they have a mechanism to prevent them talking over eachother).
- The central **switch** is a single point of failure (entire network fails it it fails).

Other Topologies

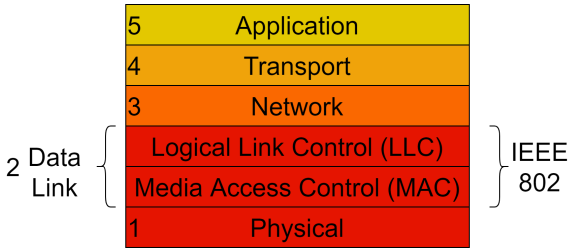


- Line is terrible (Split Ring).
- Star-Bus/Tree is a hybrid.
- Mesh is useful in some scenarios, but can be expensive.
- Mesh is ultra-fast (every host connected directly to every other host), but very expensive & difficult/nearly-impossible to effectively manage.

MAC

Lecture Recording

Lecture recording is available here



In **token ring** only one host can transmit at a time. However with other network types this is not the case.

For example a **broadcast** channel can have multiple receiving hosts & multiple concurrent transmissions can result in **frame** collisions.

We use **Medium Access Control** to coordinate channel access.

Stations

In these notes the term **station** is used to describe a host transmitting on the shared medium.

- In a wired network, frame collisions result in both frames needing to be re-transmitted.
- In a wireless network one transmitter may be stronger/a frame may be recieved even with a collision.

Medium Access Control Strategies

No Control

- When a frame is not recieved, the station retransmits as it pleases.
- Fine if channel ultilisation is low.
- Inefficient when contention is high (lots of transmitting stations → constant collisions & attempted re-transmissions).

Round Robin

- Stations take turns to transmit.
- Used in **token-based MAC** systems (only the station with the token can transmit).

Reservations

- Stations obtain *channel reservations* prior to transmitting.
- Stations can only transmit for the time interval they have reserved.
- Requires a system to manage reservations.
- Used in **slotted** systems.

Static Channel Allocations

Where each station is allocated a fixed schedule of times it is allowed to transmit.

For a channel shared between n different stations:

- **Time Division Multiplexing (TDM)**

  Stations waits for its time slot to transmit. each station's transmission rate limited to $\frac{R}{n}$ where R = maximum channel rate.

- **Frequency Division Multiplexing**

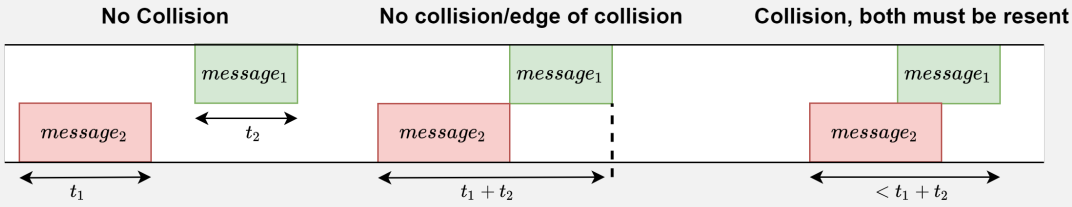
  Stations given a limited frequency band. Each station can use $\frac{B}{n}$ where B = total channel bandwidth.

  Bad for large n or traffic that is in bursts.

Dynamic Channel Allocation

Definition: ALOHA Protocol

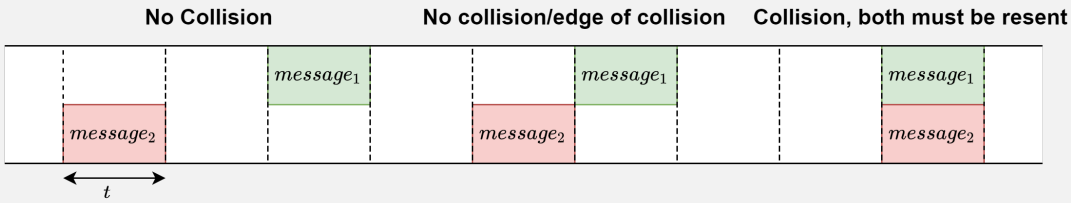
Stations transmit whenever they want to. If a collision occurs, stations wait a random period of time before attempting to re-transmit.



- The protocol suffers from low channel efficiency (worse with more contention) as there is a large **vulnerable period**.
- If a frame transmission is interrupted by another at any point, the both frames must be re-transmitted (new frames can destroy old frames).
- Maximum efficiency of 18% at 50% load.

Definition: Slotted ALOHA Protocol

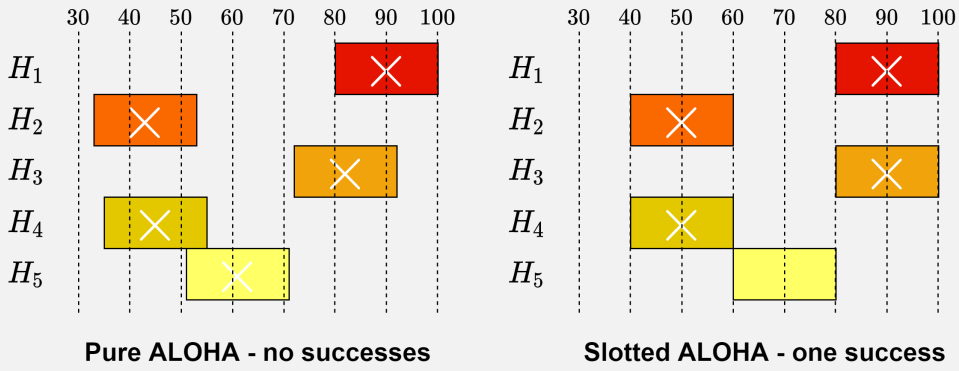
Like **ALOHA** but can only transmit on specific discrete time intervals (slots) (managed by a synchronous global clock).



- Reduces opportunities for a new frame to collide with an old one.
- Can only collide with exact overlap (contention for a slot)
- Maximum efficiency of 36% at 100% load.

Example: ALOHA Comparison

Given hosts start transmitting at times $H_1 : 80, H_2 : 33, H_3 : 72, H_4 : 35, H_5 : 51$ plot their transmissions and determine which transmissions collide given frames sent are $20s$ long.



Carrier Sense Multiple Access (CSMA)

Definition: Carrier Sensing

Listen before transmitting, transmission only occurs when the channel is idle/free.

- Reduces collisions over **ALOHA** as new frames are not sent during another's transmission.
- Collisions can still occur due to transmission delay (e.g two stations see idle channel, start transmitting, or one starts transmitting after another, but signal has yet to reach it).

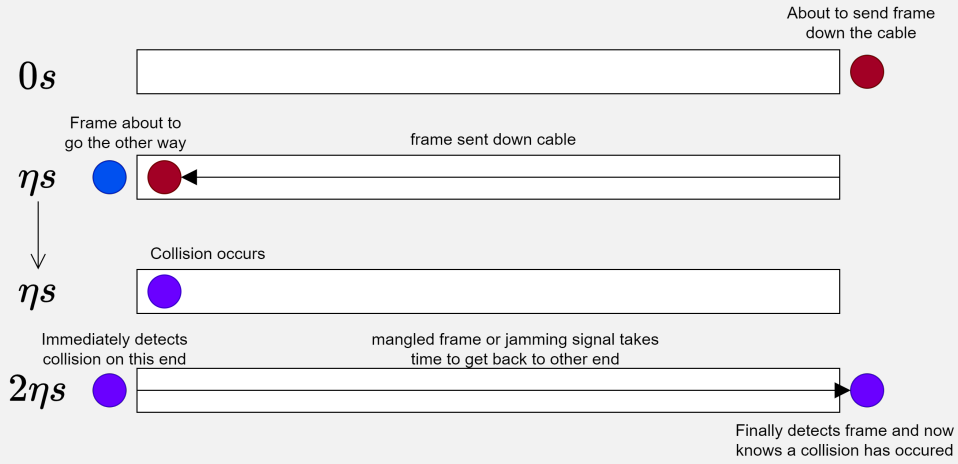
CSMA/CD	Collision Detection	Ethernet	IEEE 802.3
CSMA/CA	Collision Avoidance	WiFi	IEEE 802.11

Carrier Sense Multiple Access / Collision Detection (CSMA/CD)

Used for **ethernet** (wired networking), all collisions result in frames being destroyed.

- Station listens to channel during transmission to check for collisions.
- Transmission stop when collision detected, then sends a **jamming signal** (other transmitter will see the **jamming signal** and hence also know a collision has occurred).
- Host must transmit long enough to be able to tell the frame has not been collided. Hence minimum frame length is 2η where η = end-to-end transmission delay.

Required Delay



The minium size is required as we may not know the first bit (in the above example) has collided until $2\eta s$ after sending.

If the first bit does not collide (reaches end) then we have the line until we stop sending (as other side will not send while channel is not idle/free).

If the first bit does collide, then we should stop sending and attempt to recover (e.g resend using some policy/protocol).

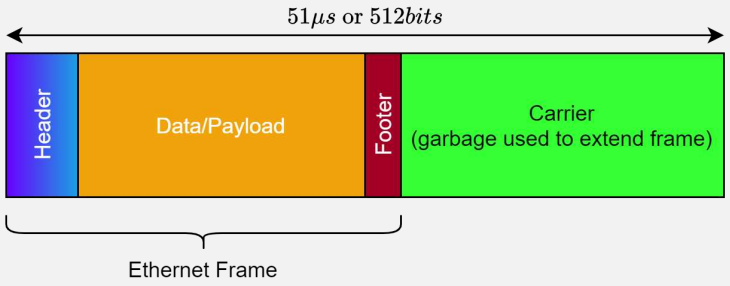
If the frame is not large enough, we will stop transmitting (and hence holding the channel) before we know if a collision may have occurred.

Collisions are inevitable as there is no central authority controlling transmission. Hence it is a best-effort service, in the worst case a frame may be indefinitely delayed.

- Suitable for most **LANs**
- Unacceptable for real-time systems (these require maximum wait time and minimum bandwidth assurances).

Definition: Carrier Extension

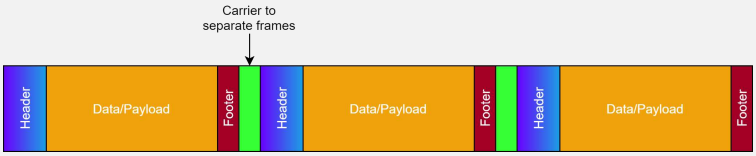
There is a minimum frame size requirement (to hold channel until bits reach destination), so for some frames an extension is required.



The wasted time transmitting the extension makes this inefficient.

Definition: Frame Bursting

Rather than padding with an extension, multiple frames are buffered, then packet together and sent at once.



Channel Back-Off

1-persistent	(Aggressive algorithm) Continually check channel. Transmit as soon as the channel is free. Used by **Ethernet**.
Non-persistent	(Non-Aggressive algorithm) Check channel, if idle translate immediately, else wait a random period of time before checking again.
P-persistent	Continually check channel, if it is free then transmit with probability p (between 1 and Non persistent).

Binary Exponential Back-Off

When network load is high (lots of contention for channels) **binary exponential back-off** is used.

- Slot length is the minimum frame length.
- If a collision occurs in transmission, wait 0 or 1 slots before attempting again.
- After c collisions, wait 0 to $2^c - 1$ slots (up to limit of 1023 slots / 10 collisions)

High contention → lots of collisions → **binary exponential back-off** → re-transmission attempts spread out → fewer collisions

Medium Access through Token Passing

There is a single token, stations can only transmit when they have the token.

- Token transfered with special token frame.
- If a station has the token but no frame to send, pass token on immediately.
- If a station has the token and a frame to send. It sets a timer and transmits until the timer expires or there is no more data to send. Then passes token.

Ethernet became much more popular, and hence has become the standard.

Avoiding Wired Collisions using Switches

A switch can remove the possibility of collisions by buffering frames and retransmitting when a channel becomes available.

- Each channel (ethernet cable) has only two stations (host and the switch)
- Hosts can transmit simultaneously, switch recieves and forwards frames.
- Maximum cable length determined by signal strength.
- **Switches** act as **repeaters**, refreshing the signal to pass it further.

Address Resolution Protocol (ARP)

Lecture Recording

Lecture recording is available here

In order to communicate outside of a network, an **IP Address** is required.

- Switches are in the **data-link layer** and do not use **IP Addresses** (in **network layer**)
- **IP Address** can be set statically (fixed **IP**, set manually) or dynamically (assigned to your **NIC**, e.g by **DHCP** service).
- **IP Addresses** specify hosts on the **internet**, it does not have to be translated when passing through a router (but can, e.g **NAT**).
- **MAC Addresses** specify hosts communicating on the same network/subnetwork. Typically do not change when passing through routers (as packets).

In order to translate **IP Addresses** to **MAC Addresses** and vice-versa we use **ARP**.

ARP Communication

1. **Router** Ask all hosts if they have a given **IP Address**
   Places **ARP Message** query in a Data-link frame and broadcasts.
2. **Host** Checks if it has the requested address, if so sends a reply with its **MAC Address**
3. **Router** Recieves **ARP Message** with **MAC Address** and uses it.
   Will forward **IP Datagrams** (encapsulated in a **Data-Link Frame**).
   Usually also cache the **IP** $\rightarrow$ **MAC** translation

Some optimisations include caching recent **ARP Message** replies, or having all hosts broadcast their **IP** and **MAC Address** on boot/connection (as a network policy).

Source			Destination			Message
Host	**IP**	**MAC**	**Host**	**IP**	**MAC**	
H_2	146.0.4.127	$0:1:2:a3:32:5$	All	146.0.4.98	$ff:ff:ff:ff:ff:ff$	ARP Req
H_1	146.0.4.98	$0:1:2:d5:6b:58$	H_2	146.0.4.127	$0:1:2:a3:32:5$	ARP Resp
H_2	146.0.4.127	$0:1:2:a3:32:5$	H_1	146.0.4.98	$0:1:2:d5:6b:58$	ICMP Echo Req
H_1	146.0.4.98	$0:1:2:d5:6b:58$	H_2	146.0.4.127	$0:1:2:a3:32:5$	ICMP Echo Resp
H_2	146.0.4.127	$0:1:2:a3:32:5$	H_1	146.0.4.98	$0:1:2:d5:6b:58$	ICMP Echo Req
H_1	146.0.4.98	$0:1:2:d5:6b:58$	H_2	146.0.4.127	$0:1:2:a3:32:5$	ICMP Echo Resp

```
1 arp who-has 146.169.4.98 tell 146.169.4.127 (0:90:27:a3:32:5)
2 arp reply 146.169.4.98 is-at 0:c0:4f:d5:6b:58 (0:90:27:a3:32:5)
3 146.169.4.127 > 146.169.4.98: icmp: echo request
4 arp who-has 146.169.4.127 tell 146.169.4.98 (0:c0:4f:d5:6b:58)
5 arp reply 146.169.4.127 is-at 0:90:27:a3:32:5 (0:c0:4f:d5:6b:58)
6 146.169.4.98 > 146.169.4.127: icmp: echo reply
7 146.169.4.127 > 146.169.4.98: icmp: echo request
8 146.169.4.98 > 146.169.4.127: icmp: echo reply
```

ARP cache poisoning

Malicious users can send spoof **ARP Messages** to attempt to associate their **MAC Address** with a victim's **IP Address** (thus receiving their **IP Datagrams**)

This is covered in detail on the wikipedia page.

50005 - Networks and Communications - Lecture 9

Oliver Killane

21/04/22

Physical Layer

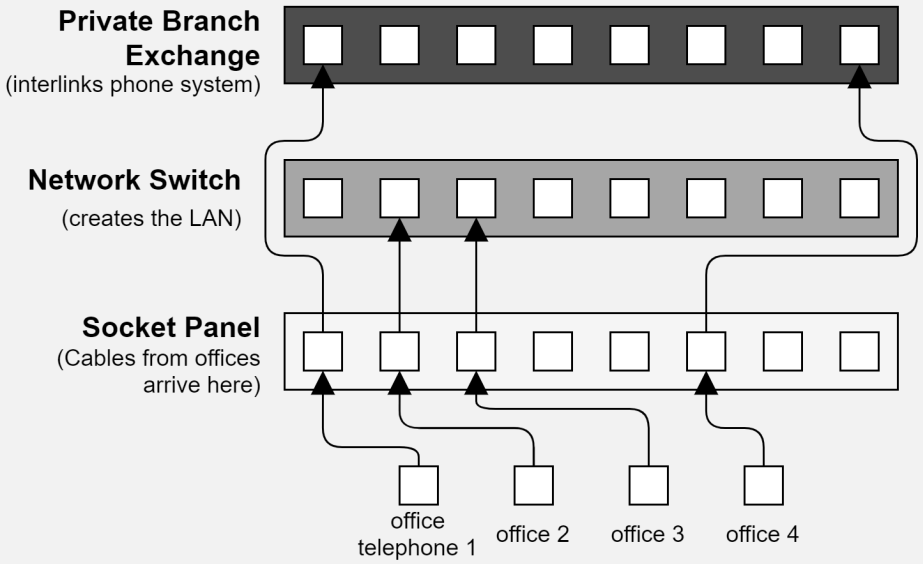
Lecture Recording

Lecture recording is available here

Network Architecture

A **network architect** designs the network (topology, standards, connections, where to put cables).
A **network engineer** installs the equipment to setup the network.

Definition: Patch Panel/Socket Panel



The **PBX** (**Private Branch Exchange**) is used for phones, however if the phone system is **IP** based, a separate **PBX** is not needed.

Wired Transmission

Definition: Unshielded Twisted Pair (UTP)

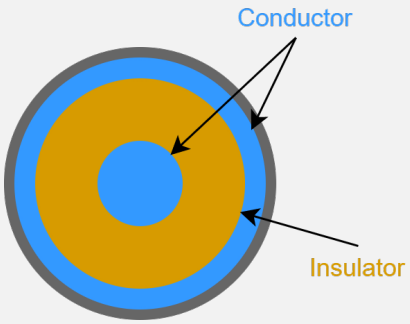
Two wires twisted together.

- Cheap & easy to mass-produce.
- Twisting reduces interference and crosstalk between cables.
- Used in telephone systems.

Type	Speed	Description
CAT1	$1 Mbps$	Voice grade for POTS (plain old telephone service).
CAT5	$100 Mbps$	10Base-T Ethernet Cables and 100Base-TX Fast Ethernet.
CAT6	$1,000 Mbps$	1000Base-T Gigabit Ethernet.

Definition: Coaxial Cable

Conductors placed concentrically (one inside the other), separated by an insulator.



- Good sheilding, electromagnetic field mainly between inner and outer conductor.
- Large bandwidth from high range of frequencies.
- Higher cost per meter (hence **UTP** is more popular for common consumption).

Definition: Optical Fibre

Transmits data using light and refraction (explained well here).

- Single optical fibre is 2 - 125 micrometers in diameter.
- **Attenuation** (signal loss) is low, so can be used for long distances.
- Very high bandwidth.

Year	Speed	Organisation
2011	26 $Tbps$	Karlsruhe Institute of Technology
2014	43 $Tbps$	Technical University of Denmark
2014	255 $Tbps$	Eindhoven University of Technology and University of Central Florida
2021	319 $Tbps$	Japan National Institute of Information & Communications Technology
2021	1000 $Tbps$	Japan National Institute of Information & Communications Technology

	Freq Range	Attenuation	Delay	Repeater Spacing
UTP	$0 - 1\ MHz$	$0.7\ dB/km$ @ $1\ KHz$	$5\ \mu s/km$	$2\ km$
Coaxial Cable	$0 - 500\ MHz$	$7\ dB/km$ @ $10\ MHz$	$4\ \mu s/km$	$1 - 9\ km$
Optical Fibre	$186 - 370\ THz$	$0.2 - 0.5\ dB/km$	$5\ \mu s/km$	$40\ km$

Wireless Transmission

Done using electromagnetic radiation (typically radio).

- No need for wires (expensive & take time to install).
- Bidirectional communication by default.
- Typically broadcast (e.g all/most recievers can see transmissions) (works with many stations).
- Inverse square law - signal strength reduces with range.
- Environment degrades signal (interference, obstruction, reflection of signal).

Wave Types

For more look at chapter 3 of Communication Systems.

Information Representation

Definition: Digital

Discrete information, represented by a finite number of states.

e.g 0 and 1 for binary.

Definition: Analogue

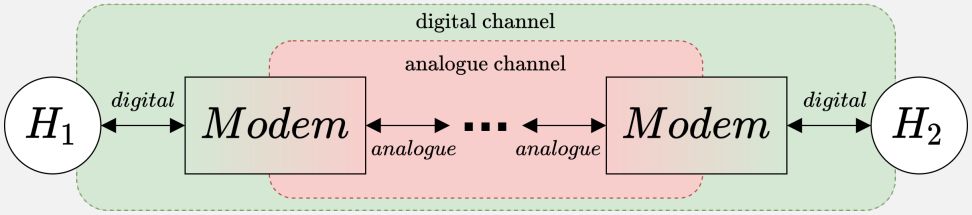
Continuous information, represented by changes in some physical state.

e.g light intensity, voltage.

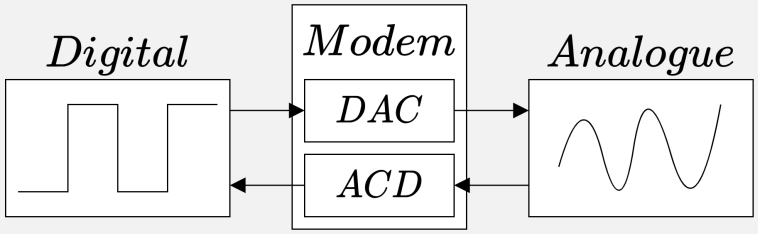
Definition: Baud Rate (Bd)

Symbol rate per second for a digital channel, where a symbol may represent more than 1 bit.

Definition: Modem



A **Modulator-Demodulator** implements a digital channel using an analogue channel.

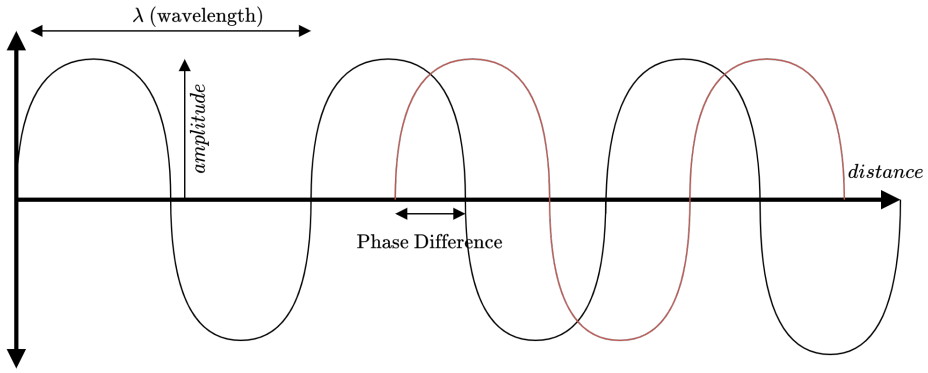


- **DAC → Digital to Analogue Converter**
- **ADC → Analogue to Digital Converter**

Definition: Codec

A **Coder-Decoder** implements an analogue channel using a digital channel.

Waves



Amplitude		Maximum displacement/strength of the signal.
Wavlength	λ	Length of a single cycle.
period	p	The time taken to complete a cycle.
Frequency	f	The number of cycles per second.

$$p = \frac{1}{f} \text{ (period and frequency)}$$

$$wavespeed = f\lambda \text{ (for radio waves} wavespeed = c = 3 \times 10^8 ms^{-1} \text{)}$$

Phase

Given two waves of the same wavelength and speed/frequency, they may be offset by some distance.

The pahse difference can be considered as a distance, or fraction of a cycle. In the latter anglke units may be used (full cycle = 360 deg = 2π).

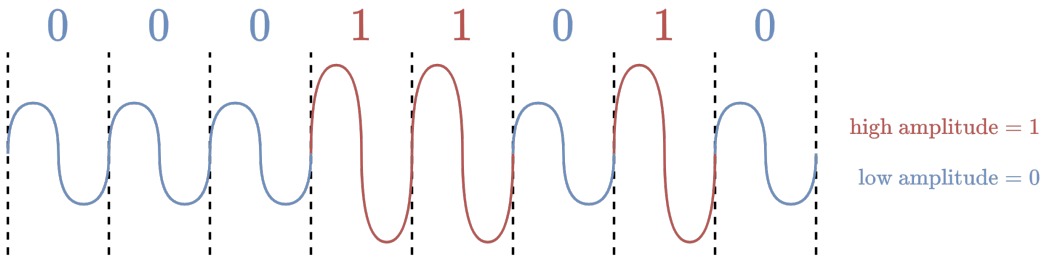
The maximum phase difference is π, where the waves are in opposite displacements for any given time during their cycle.

Modulation

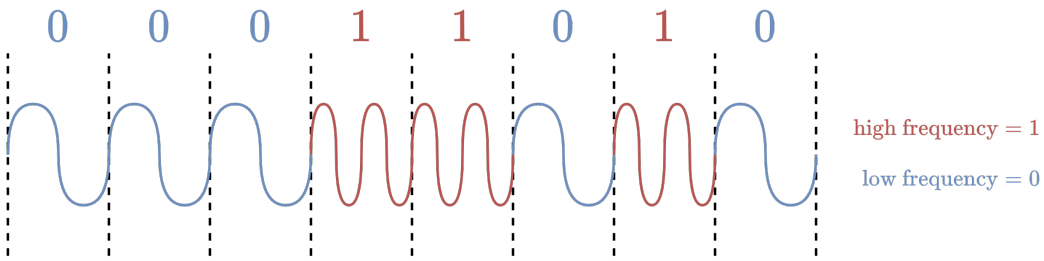
A **modulation** scheme is used to change some information signal into one more suitable for transmission.

Baseband Modulation	Transmit unmodified (dedicated line sending in full).
Broadband Modulation	Uses a basic carrier signal to encode information. The carrier signal has modifications added to encode information (e.g changing amplitude, frequency or phase).

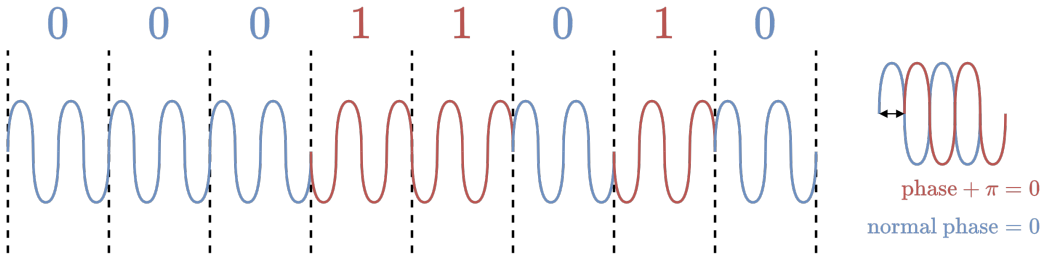
Amplitude Modulation / Amplitude Shift Keying (ASK)



Frequency Modulation / Frequency Shift Keying (FSK)



Phase Modulation

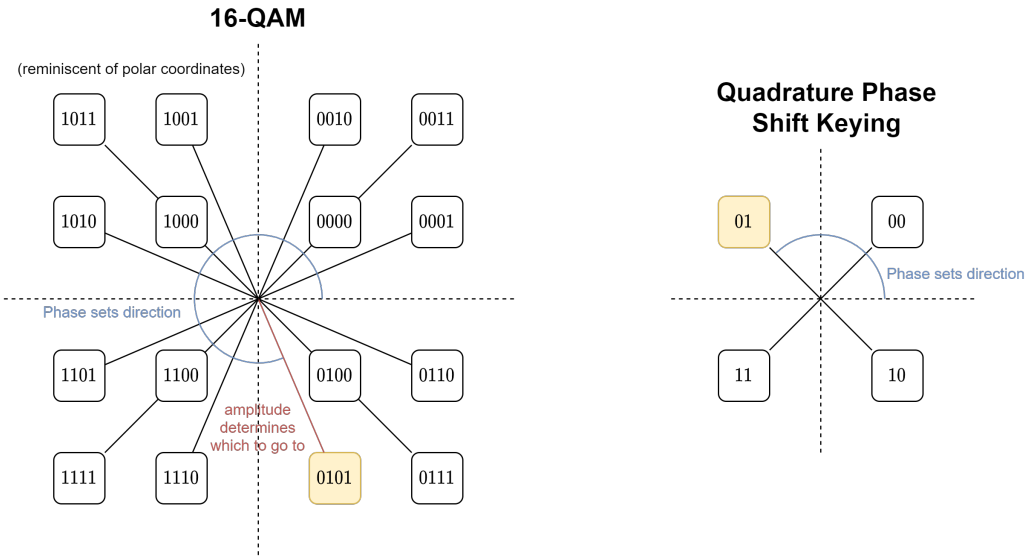


Better Modulation

To improve the data rate we can transmit multiple bits per symbol (in modulation scheme).

- Use more phase differences, amplitudes.
- Use a combination of phase, amplitude to determine symbol.

For example we can use phase (interpreted as an angle) in combination with amplitude in a scheme such as **QAM**.



Digital Subscriber Line (DSL)

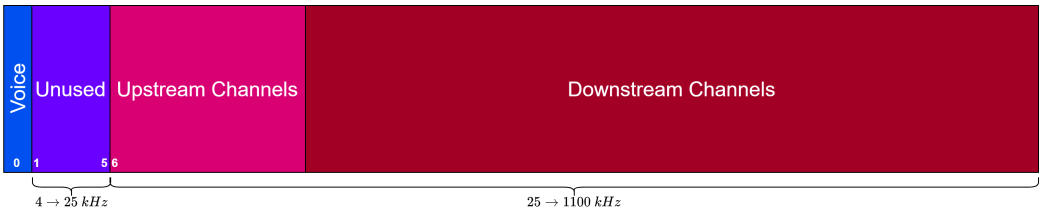
With the V.90 Modem Standard, using conventional phone lines to transfer data.

- Maximum $56,000$ *bps* downstream (download) and $33,000$ upstream (upload)
- Limited as phone lines limited to $3,000$ Hz bandwidth (human voice goes to $3,400$ Hz and was originally developed only for voice communication).
- Anything outside that range is filtered out as noise.

By removing the limitation (by removing the bandwidth filter) **DSL** allows for more bandwidth and hence a higher data rate.

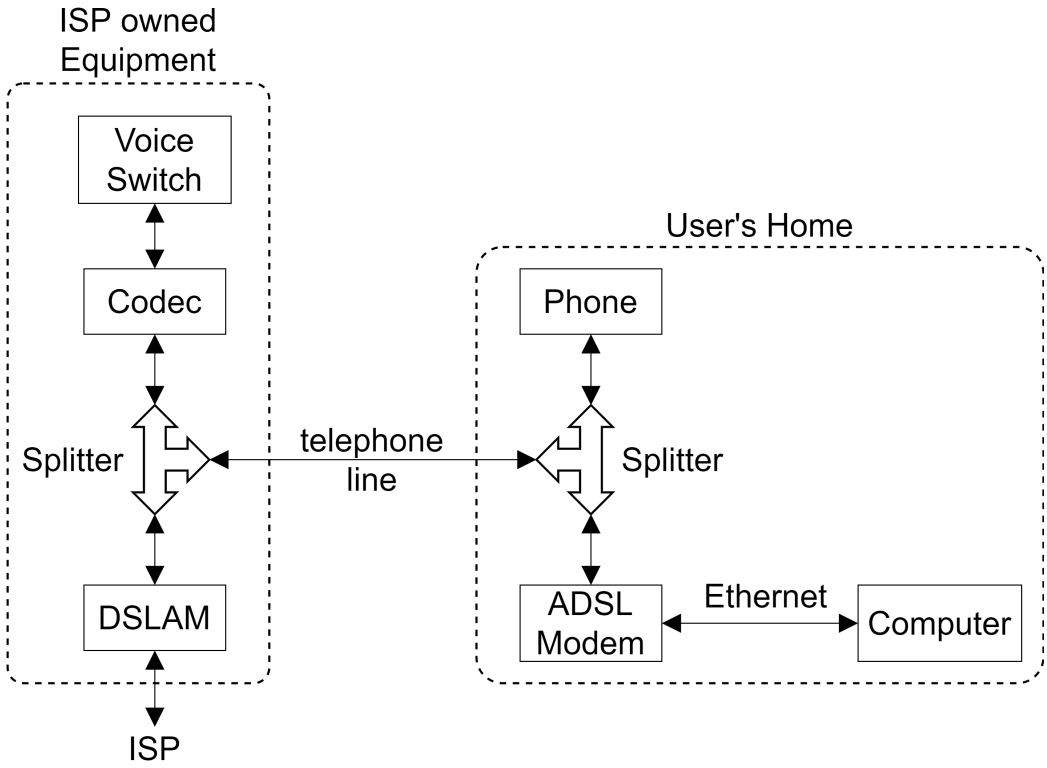
However noise now becomes a limiting factor.

Asymmetric Digital Subscriber Line (ADSL)



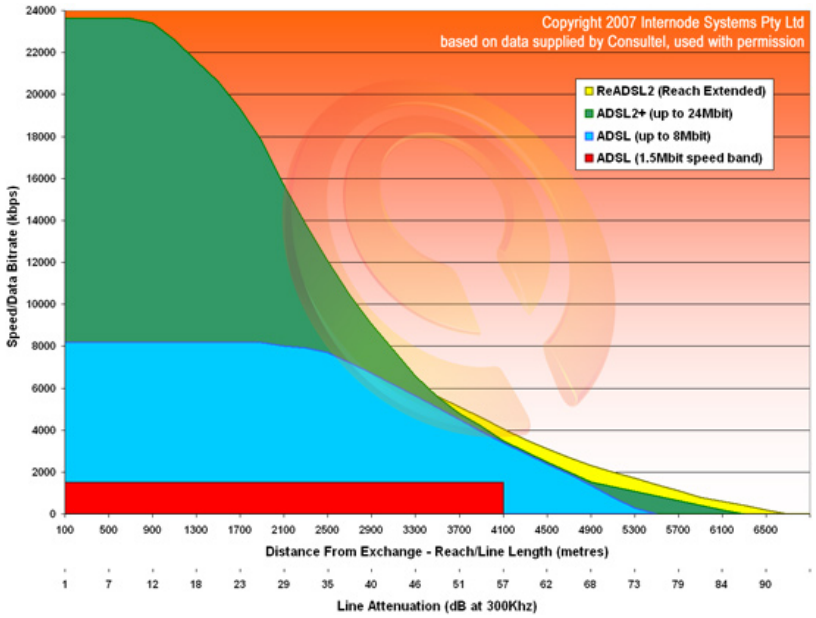
- 1.1 MHz of bandwidth divided into 256 4000 Hz channels.
- Channels $1 \rightarrow 5$ ($4 \rightarrow 25$ kHz) are unused to avoid interference between voice and data channels.
- More channels are allocated to download than upload as download used more heavily.
- Voice is cahnnel 0 ($0 \rightarrow 4$ kHz)
- V.24 modulation uses 224 downstream channels (13.44 $Mbps$)
- A **ADSL Splitter** separates the voice band from data.
- An **ADSL modem** performs modulation.

DSL Access Multiplexer (**DSLAM**) (typically owned by the **ISP**) connects local telephone cables to the **ISP**



DSL Advancement

ADSL2	12 $Mbps$	2.2 MHz	
ADSL2+	12 $Mbps$	2.2 MHz	More bits per symbol
VDSL	52 $Mbps$	12 MHz	Very-high-bit-rate DSL
VDSL2	200 $Mbps$	30 MHz	Currently popular



Network Simulation

Lecture Recording

Lecture recording is available here

Network simulation is used to design networks cheaply.

Different simulators provide different features:

- **Cisco Packet Tracer** Strong adacemic backing
- **gns3** Strong, open community backing
- **OPNET** Professional use, quite technical

Cisco packet Tracer allows code to be run inside the simulation:

- Cisco IOS commands (Cisco's proprietary Operating System)
- Terminal commands inside applications on Desktop/Laptops
- Web documents (through server's http service)
- Python & Javascript
- MQTT (Message Queue Telemetry Transport) (a lightweight machine-to-machine Publisher/-Subscriber messaging protocol)
-

Network Programming

Lecture Recording

Lecture recording is available here

Simple Echo

1. Run server, waiting for connections on a user-defined port.
2. Client connects to the port.
3. Server listens for input from the client.
4. User types into client, client sends message to server.
5. Server echos recieved data back to client.
6. Client disconnects.
7. Server closes.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.net.UnknownHostException;

class EchoClient {

    static String hostName;
    static int portNumber;

    public static void main(String args[]) {
        try (Socket echoSocket = new Socket(hostName, portNumber);

            // Communication with the server through the socket
            PrintWriter out = new PrintWriter(echoSocket.getOutputStream(), true);
            BufferedReader in = new BufferedReader(new InputStreamReader(echoSocket.
                ↪ getInputStream()));

            // Read user input from terminal
            BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in
                ↪ ))) {


            String userInput;
                while ((userInput = stdIn.readLine()) != null) {
                out.println(userInput);
                System.out.println("echo: " + in.readLine());
            }

        } catch (UnknownHostException e) {
            System.err.println("Don't know about host " + hostName);
            System.exit(1);

        } catch (IOException e) {
            System.err.println("Couldn't get I/O for the connection to " + hostName);
            System.exit(1);

        }
    }
}
```

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
```

```java
import java.net.Socket;

public class EchoServer {

    static int portNumber;
    public static void main(String args[]) {

        // try with resources statement closes the writer, reader and sockets
        // after the statement
        try (
            ServerSocket serverSocket = new ServerSocket(portNumber);

            // wait for the client to connect
            Socket clientSocket = serverSocket.accept();
            PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
            BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket
                ↪ .getInputStream()));
            ) {

            System.out.println("Client connected on port " + portNumber + ".
                ↪ Servicing requests.");
            String inputLine;
            while ((inputLine = in.readLine()) != null) {
                System.out.println("Received message: " + inputLine + " from " +
                    ↪ clientSocket.toString());
                out.println(inputLine);
            }
        } catch (IOException e) {
            System.out.println("Exception caught either when trying to listen on port
                ↪  " + portNumber + " or while listening for a connection");
        }
    }
}
```

Concurrent Executor

We can use a thread pool to handle running tasks for many clients connecting, and sending input.

```java
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class ConcurrentServer {
    static int threads = 5;
    static int portNumber;
    public static void main(String args[]) {
        try (ServerSocket serverSocket = new ServerSocket(portNumber);) {
            ExecutorService executor = Executors.newFixedThreadPool(threads);

            System.out.println("Waiting for clients to connect...");

            while (true) {
                Socket clientSocket = serverSocket.accept();
```

```
22                  executor.execute(new RequestHandler(clientSocket));
23              }
24
25          } catch (IOException e) {
26              System.out.println("Exception caught when trying to listen on port " +
                    ↪ portNumber + " or listening for a connection");
27          }
28
29      }
30  }
31
32  class RequestHandler implements Runnable {
33      Socket clientsocket;
34
35      RequestHandler(Socket clientsocket) {
36          this.clientsocket = clientsocket;
37      }
38
39      @Override
40      public void run() {
41
42          try (
43              BufferedReader in = new BufferedReader(new InputStreamReader(clientsocket
                    ↪ .getInputStream()));
44              BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(
                    ↪ clientsocket.getOutputStream()));
45              ) {
46              System.out.println("Thread started with name:" + Thread.currentThread().
                    ↪ getName());
47
48              String userInput;
49              while ((userInput = in.readLine()) != null) {
50                  userInput = userInput.replaceAll("[^A-Za-z0-9 ]", "");
51                  System.out.println("Received message from " + Thread.currentThread().
                        ↪ getName() + " : " + userInput);
52                  writer.write("You entered : " + userInput);
53                  writer.newLine();
54                  writer.flush();
55              }
56
57          } catch (IOException e) {
58              System.out.println("Exception raised while attempting to handle request")
                    ↪ ;
59
60          }
61      }
62  }
```

Oracle Guides

The guides these examples were based on can be found here.

50005 - Networks and Communications - Lecture 10

Oliver Killane

29/04/22

Lecture Recording

Lecture recording is available here

Future of Networking

Faster Hardware

Use of ASICs (Application Specific Integrated Circuits) to make faster network switches.

One example of this is Barefoot Networks (purchased by Intel in 2019). They create high speed Ethernet ASICs with a programmable pipeline (using a language called P4). Their Tofino 2 switch can handle 12.8 $Tbps$.

Many other companies such as Cisco also vend ASIC based network gear.

Another consideration is using light as a medium for secure communications, better fibre optics.

Faster Wireless

Kumu Networks have developed programmable filters to allow wireless devices to cancel out their own transmissions. This allows full-duplex wireless as wireless devices can receive and transmit simultaneously on a single channel.

Scientists in Japan and Germany have developed on terahertz transmitters (1.1 THz) allowing for far higher data transfer speeds

Legislation

Net-Neutrality laws in the USA (though can affect the entire world as they affect internet infrastructure) allow ISPs to be selective about services provided for content on the internet (e.g slowing down a competitor's website, offering special packages allowing access to a limited number of sites).

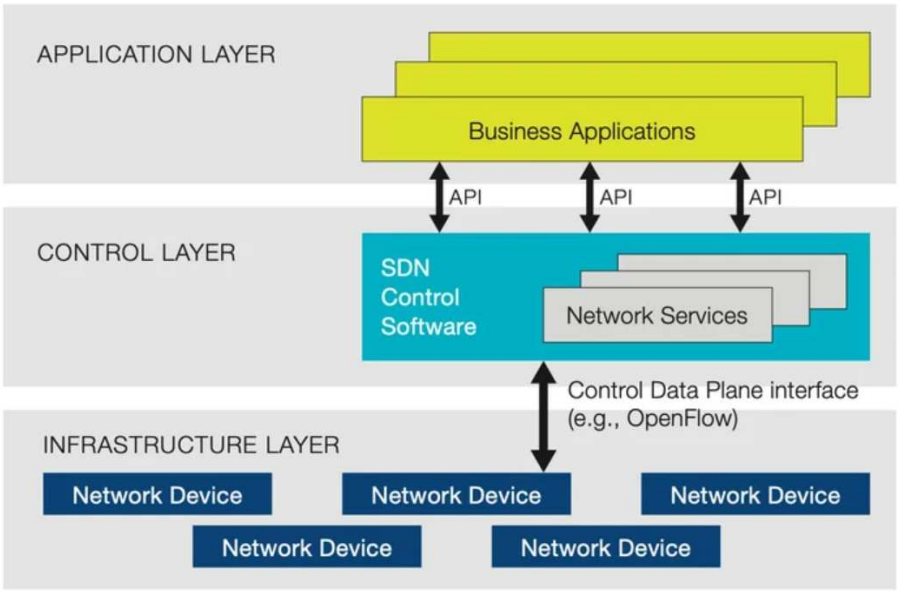
Wireless Mesh

Allowing many wireless devices to form a mesh network. For example Cisco Meraki allows for networks to self-heal when parts of the network (e.g switches) fail, by rerouting data wirelessly.

Software Defined Networking (SDN) and Network FUnctions Virtualization (NFV))

A network architecture where applications and services are abstracted from the network infrastructure & control.

Useful for containerisation, and being developed by Nicira (now owned by VMWare), Cisco and others.



Web-Decentralisation

The internet has become more centralised around large **CDNs** such as Amazon's, Google's, and around few large services (e.g facebook, youtube).

This is bad for reliability (if a few backbones go down, large services disappear).

New protocols such as IPFS intend to resolve this.

Many users can aide decentralisation by using their own domains, storage (e.g instead of google drive, dropbox) and their own hosting services.

Jobs in Networking

Definition: Network Engineer

An engineer specialising in manageing computer networks, typically with expertise in:

- Infrastructure
- Virtualisation (e.g VLANs)
- Servers
- Switches
- Firewalls
- Meraki
- WatchDog

Some certifications used include:

- **CCNP** Professional-level certification by cisco.
- **CISSP** Certified Information Systems Security Professional, a cybersecurity competency certification.
- **RHCE** Red Hat Certified Engineer