50005 - Networks and Communications - Lecture 3

Oliver Killane

21/01/22

The Web

Lecture Recording

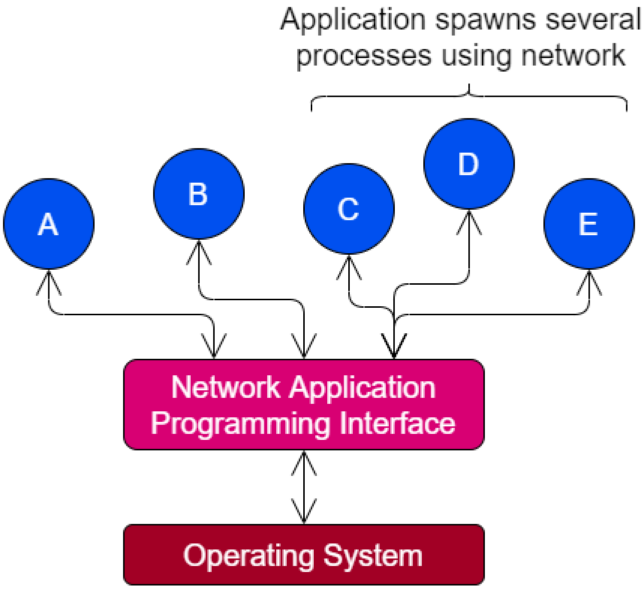
Lecture recording is available here

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.


Application spawns several processes using network
A
B
C
D
E
Network Application Programming Interface
Operating System

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

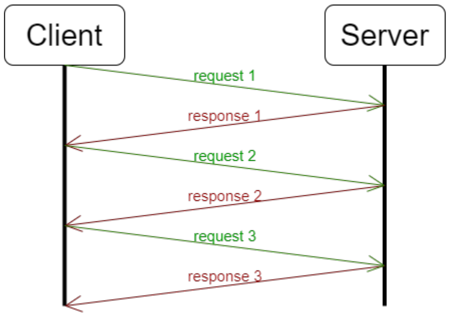
Expected to replace HTTP/1.x completely within a few years.

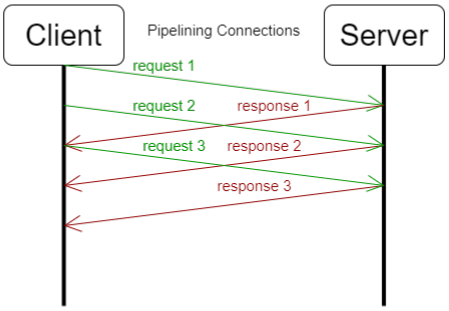
- Exchanges content in binary, allowing for more compact representation and higher speed (less data to transfer).
- Connection is fully multiplexed (not ordered or blocking)
- Can use a single **TCP** connection with requests in parallel.

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 → 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).

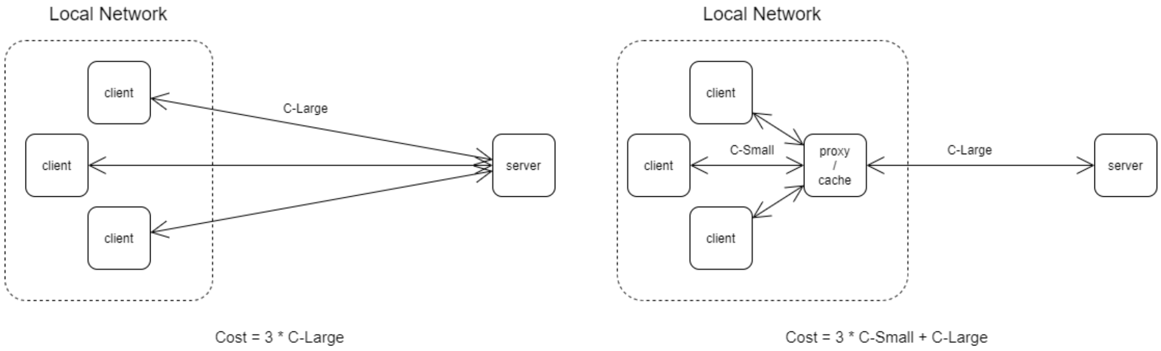
We can use **telnet** to send plain-text commands directly to a server listening on a specific port (80 for HTTP).

```
> telnet www.imperial.ac.uk 80
> GET /computing/ HTTP/1.1
> 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

```
GET /this/2122/are/of/site.html HTTP/1.1
Host: www.mywebsite.ic.ac.uk
Cache-Control: no-cache

GET /this/2122/are/of/site.html HTTP/1.1
Host: www.mywebsite.ic.ac.uk
Cache-Control: max-age=30
```

Example: Responses

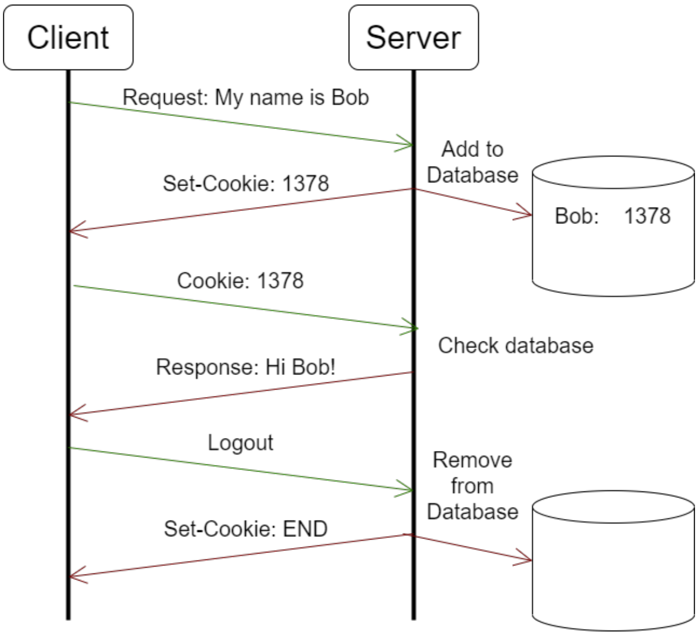
```
<!-- Cache for at most 100 seconds, then revalidate -->
HTTP/1.1 200 OK
Cache-Control: max-age=100, must-revalidate

<!-- Do not cache -->
HTTP/1.1 200 OK
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	$fe80 :: 211 :: 43ff :: fecd :: 30f5/64$

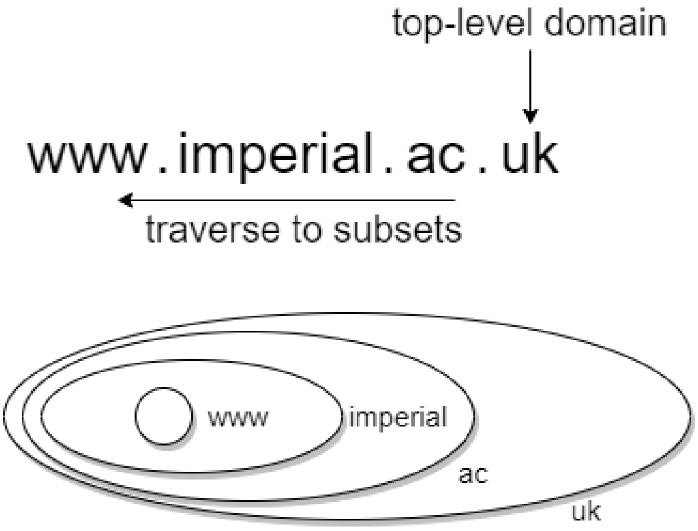
- Easy format for routers to process quickly.
- Not practical for use by people.

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

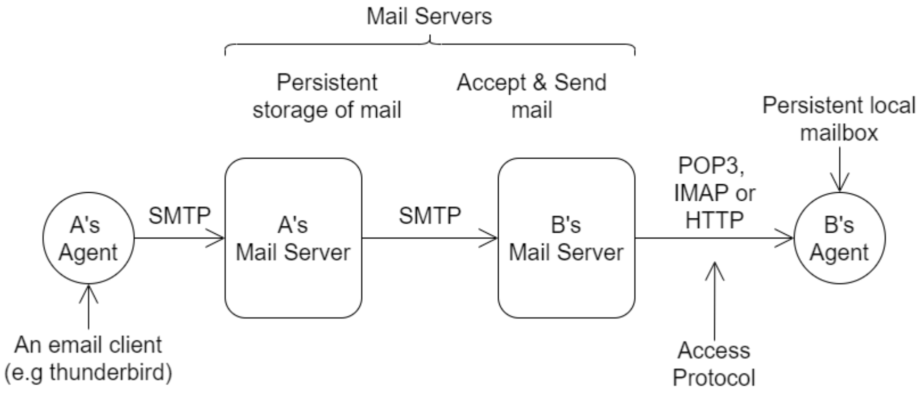
Lecture Recording

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 receipt.



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.

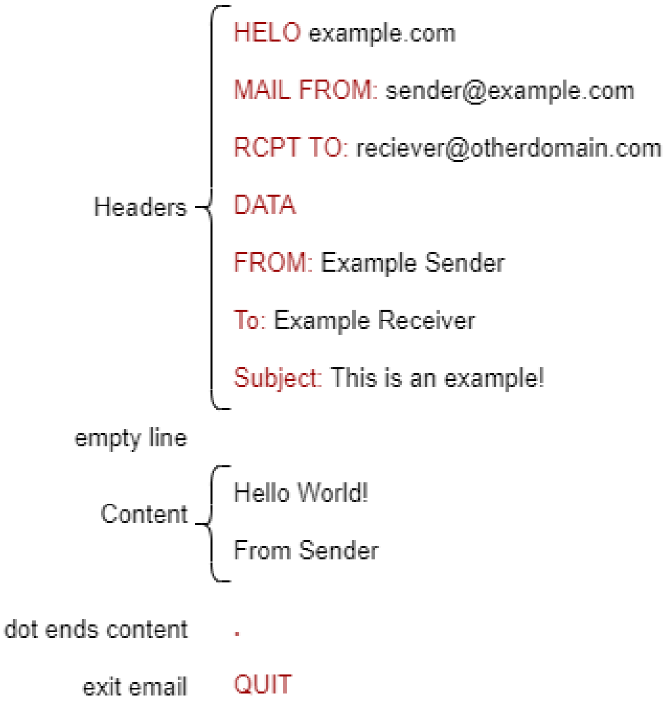
Simple Mail Transfer Protocol (SMTP)

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.



```
HELO example.com
MAIL FROM: sender@example.com
RCPT TO: reciever@otherdomain.com
DATA
FROM: Example Sender
To: Example Receiver
Subject: This is an example!

Hello World!
From Sender
.
QUIT
```

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: ”.” → $\begin{array}{c} ”..” \\ ”.” \end{array}$ → $\begin{array}{c} ”.” \\ ”.” \end{array}$ → ”.”

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**-Secure
  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** Extended **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.