UDP (The fire and forget protocol)

* UDP is defined in RFC768 to be a minimal message-oriented transport layer protocol, as it provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism.
* The properties of UDP are listed below:
* Provides no guarantee to the upper layers on message delivery
* Is stateless – retains no state of UDP messages once sent
* Suitable for large numbers of clients, such as in streaming
* Good for unidirectional communication (like in the stock market distribution)
* No error detection and correction.
* No connection-oriented links
* No verification of delivery order.
* Basic datagram delivery service.
* Easy to implement and minimal overhead.
* Low intensity tasks performed.
* Graphical user interface, application, Word

  Description automatically generatedIn UDP data is sent across the network in finite packets. Each Datagram contains a header and a payload (contains the data). The datagram is either a request or reply message and provides minimal transport layer functions, the header looks like this:
* In order to send the datagrams, the client needs the source and destination port numbers, but how are they acquired? Well, for example, a client may try connecting to a server that provides some form of service like DNS, and every DNS server running on the internet uses the port number 53. There can be different DNS instances such as the Google’s DNS server or QUB’s DNS server, but they both use port number 53.
* Every DNS server program that starts executing on a computer must tell the OS that it wants to use port number 53. This is known as Binding a port.
* Here are some well-known port assignments:

Graphical user interface, table

Description automatically generated

* Seeing as how each port number is an unsigned integer and the port number part of the UDP header is 16 bits long, that allows for 65535 different port numbers.
* However, it was decided that ports 0-1023 would be reserved so that only programs run by the Root/Admin could use them. (Services are included in this)
* IANA standardises port use and so manages the list of registered port numbers.
* RFC 6335 defines the process for registering the use of a port.
* Graphical user interface

  Description automatically generatedThe table below shows how the port numbers are divided:
* When a client sends a data packet to the server, software on Layer 4 in the OS will, at the time of sending, choose a free port from the EPHEMERAL PORT RANGE and put this value into the source port of the packet. A table is kept that links this source port to the client program so that then the server can use that source port as the destination for reply packets.
* Therefore, Ephemeral ports allow each client session to run on a unique port.
* Although the server program always binds itself the same port number, the client will get a different port number from the Ephemeral pool each time it’s run. This is due to a lot of factors like:
* Other clients code will also be running
* It depends on the order in which these clients were started
* After a client is shutdown, the port number it was using is frozen for a short period of time before being put back in the pool of port numbers.
* The OS can use any policy to allocate the port number from the Ephemeral pool
* Important Note: The port + protocol is unique. Therefore, UDP port 53 is completely different from TCP port 53. These can both run at the same time as the OS can distinguish between them.

Diagram

Description automatically generated with medium confidence

UDP Multicast

* In order to overcome the unreliability of UDP, the data can be sent over two different channels. Although, it’s critical that these are routed differently through the network (Geographically diverse routing (GDR)), so that if an error occurs, it is less likely to affect both, but also for security purposes.
* However, now the client software, may receive the same message twice or not receive the message at all. Therefore, the software has to be programmed to handle these situations. So, every message is assigned a unique number so that the software can check:
* Receiving the same message twice
* Gaps in the sequence of messages
* Never receiving the message.
* Line Arbitrage is the algorithm used to only receive one message out of the two if the first message is successful
* There’s also a fallback TCP connection to the exchange is order to request messages that have been missed.