Computer Networking Cheatsheet

Edoardo Riggio

June 14, 2021

Computer Networking - SP. 2021
Computer Science
Università della Svizzera Italiana, Lugano

Contents

1 Computer Networking and the Internet

1.1 Internet

The Internet is both a computer network that connects together billions of computing devices around the world, and an infrastructure that provides services to applications.

All of the devices connected to the Internet are called **hosts** or **end systems**. These hosts and end systems are connected together by a network of **communication links** and **packet switches**. The transmission rate of links is measured in bits/s.

When two end systems need to exchange data, this is divided into **packets** and are sent through the network. When they arrive at destination, then they are reassembled. A **packet switch** takes a packet arriving on one of its incoming links and forwards them to one of its outgoing links. Packet switches can be of different types, such as **routers** and **link-layer switches**.

End systems can access the Internet through **ISPs**. Each ISP in itself is a network of packet switches and communication links.

1.2 Internet Services

Applications on the Internet are said to be **distributed applications**, since they involve multiple end systems that exchange data with each other. In order for a program to instruct the Internet on how to deliver data to another program on another end system, **socket interfaces** are used. These interfaces are provided by hosts attached to the Internet.

1.3 Protocols

All activity in the Internet that involves two or more communicating remote entities is governed by a protocol.

A protocol defines the format and the order of messages exchanged between two or more communicating entities, as well as actions taken on the transmission and/or receipt of a message or other event. Different protocols are used to accomplish different communication tasks.

Internet standards are developed by the **IETF** (Internet Engineering Task Force) by means of **RFCs** (Requests For Comment). These RFCs define protocols such as TCP, IP, HTTP, SMTP...

Other bodies also specify standards for network components, such as the **IEEE** (Institute of Electrical and Electronics Engineers).

1.4 Network Edge

A network edge is composed mainly of two parts:

- **Network Edge**

  Is the part of the network which is composed of applications and end systems. These end systems are also referred to as **hosts**. That is because they run application programs.

- **Access Network**

  It is the network that physically connects an end system to the fist router, on a path from an end system all the way to any other distant end system.

1.5 Physical Media

1.5.1 Guided Media

In the case of guided media, the electromagnetic/optical waves are guided along a solid medium. Some examples of this kind of medium are:

- **Twisted-Pair Copper Wire**

  Most used form is **UTP** (Unshielded Twisted Pair), and it is used for a computer network within a building.

- **Coaxial Cable**

  This medium is made out of copper (core) and has an internal concentric structure.

- **Fiber Optic Cable**

  This type of medium transfers bits as light pulses. It is immune to electromagnetic interference, have a very low signal attenuation up to 100 km and are very hard to tap.

1.5.2 Unguided Media

In the case of unguided media, electromagnetic waves are propagated in the atmosphere and/or in outer space.

- **Terrestrial Radio Channel**

  No physical wire is needed to be installed, waves can penetrate walls and have a long range of action. Terrestrial radio channel can be divided into 3 categories: short range (Bluetooth), medium range (Wi-Fi) and long range (3G, LTE, 4G, 5G ...).

- **Satellite Radio Channel**

  Two types of satellites are used: **geostationary satellites** and **LEO satellites** (Low-Earth Orbiting).

1.6 Packet Switching

In order to send a message, the source breaks a long message into smaller chunks called **packets**. Between the source and the destination, each packet travels through **communication links** and **packet switches**. Packets are transmitted at a rate equal to the full transmission rate of the link. The formula is:

$$\Delta_{TX} = \frac{L}{R}$$

1.6.1 Store-and-Forward

Most packets use the store-and-forward transmission. This means that the packet must be fully received before it can be forwarded to the next packet switch. The end-to-end delay over N routers can be measured as follows:

$$d_{end-to-end} = N \cdot \frac{L}{R}$$

1.7 Forwarding Tables and Routing Algorithms

Each router has a **forwarding table** that maps destination addresses to that router's outbound links. This is used by the router, in combination with the destination IP address in the packet's header, in order to find the next router the packet needs to be sent to.

Routing protocols may, for example, determine the shortest path the packet needs to traverse in order to reach its destination.

1.8 Circuit Switching

In circuit switched networks, the resources needed along the path to provide for communication between the end systems are reserved for the whole duration of the connection. For this duration, also a constant transmission time is reserved and it is guaranteed to be constant. A circuit in a link is implemented using either:

- **FDM (Frequency Division Multiplexing)**

  The frequency spectrum of the link is divided up among the established connections across the link. The link dedicates the frequency of the band for the entire duration of the connection. The bandwidth for telephone networks is 4 kHz, while FM radio stations share the 88-108 MHz frequency spectrum.

- **TDM (Time-Division Multiplexing)**

  In this case time is divided into frames of fixed duration, and each frame is divided into a fixed number of time slots. When the network establishes a connection, the network dedicates one time slot in every frame for this connection.

Circuit switches can be wasteful because the dedicates circuits are idle during **silent periods**. These idle periods are instead used by packet switching networks and used for other connections.

1.9 Structure of the Network

In order to connect end users to each other, a network must be built. This network is hierarchical and is composed of multiple interconnected networks. These are the networks:

- **Network Structure 1**

  This network structure interconnects all of the access ISPs with a **single global transit ISP**. This would be a network of routers and communication links that spans the globe.

- **Network Structure 2**

  This network consists of **many access ISPs** and a **few global transit ISPs**. Global transit ISPs themselves must be interconnected.

- **Network Structure 3**

  In any given region, there may be a **regional ISP** to which each **access ISP** connects. Each regional ISP then connects to **tier-1 ISPs**. These are the global transit ISPs.

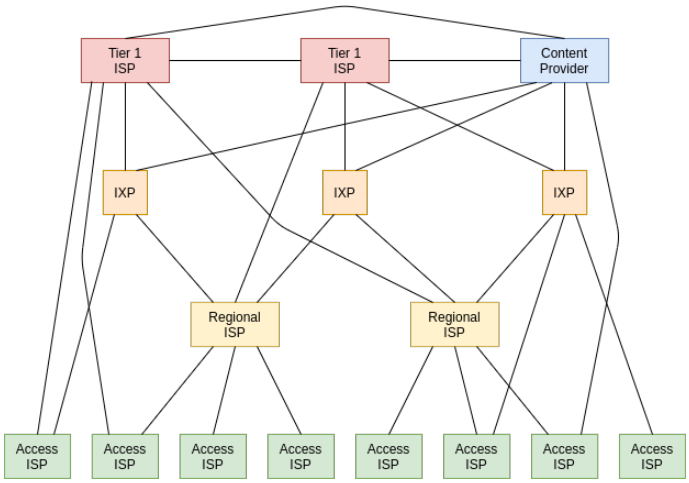
- **Network Structure 4**

  PoPs (Points of Presence) are added in order to group one or more routers at the same location such that they can be connected to provider ISPs. Any ISP, could also decide to **multi-home**. That is, to connect two or more provider ISPs. **Peering** is when two ISPs that are both near and at the same hierarchy, connect their networks together. Finally, a third-party company can create an **IXP** (Internet eXchange Point), which is a meeting point where multiple ISPs can peer together.

- **Network Structure 5**

  Content-Provider Networks are added to the network – such as Google. These networks peer with lower-level ISPs and connect to tier-1 ISPs.

This can be a schematic representation of the modern network of networks:



1.10 Packet Delay

As a packet travels from one node to the subsequent node, the packet suffers from several types of delay at each node.

1.10.1 Processing Delay

The processing delay can be caused by the time required by routers and switches to examine the packet's header and finding out where it needs to go; it can also be caused by the time the router or switch takes in order to find bit-level errors in the packet, that may happen when the packet is transferred from a node to the other.

This type of delay is in the order of μs.

1.10.2 Queuing Delay

The queuing delay is caused by the number of earlier-arriving packets that are waiting for transmission. If the traffic is heavy, then the time that the packets need to wait for them to be transmitted is longer. This kind of delay is calculated with the following formula:

$$d_{queue} = \frac{L \cdot a}{R} \cdot \frac{L}{R} \cdot \left(1 - \frac{L \cdot a}{R}\right)$$

For $\frac{L \cdot a}{R} < 1$. This type of delay is in the order of μs.

1.10.3 Transmission Delay

Assuming that we have a FIFO packet transmission, the packet can be transmitted only if all of the packets that are in front of it have been transmitted. The transmission delay can be calculated as follows:

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

This measures the time required to transmit all of the packet's bits onto the link. This type of delay is in the order of $\mu s/ms$.

1.10.4 Propagation Delay

The propagation delay depends on the physical medium of the link connecting the two nodes. The propagation speed can go from $2 \cdot 10^8 m/s$ to $3 \cdot 10^8 m/s$. The delay is calculated as follows:

$$d_{propagation} = \frac{d}{s}$$

This type of delay is in the order of ms.

1.10.5 Nodal Delay

The total nodal delay that happens at every node is:

$$d_{nodal} = d_{processing} + d_{queue} + d_{transmission} + d_{propagation}$$

1.11 Packet Loss

Because the capacity of a router/switch queue is finite, if there are more packets than the buffer can hold, then those extra packets are dropped. Performance at a node must be thus measured also in terms of packet loss other than delay.

1.12 End-to-End Delay

The total delay from source to destination is calculated as:

$$d_{end-to-end} = N \cdot (d_{processing} + d_{transmission} + d_{propagation})$$

This is true if all nodes have the same processing, transmission and propagation queues. If that were not the case, then all the single nodal delays must be added up together.

1.13 End-to-End Throughput

End-to-end throughput is the instantaneous rate at which a host can receive dat from another host. The formula is:

$$t_{end-to-end} = \frac{F}{T}$$

Throughput not only depends on the transmission rates of the links along the path, but also on the intervening traffic.

1.14 Protocol Layers

Each of the 5 layers that compose the **TCP/IP** protocol stack provides its service by performing certain actions within that layer and by using the services of the layer directly below it.

1.14.1 Application Layer

The application layer is where the network applications and their protocols reside.

Some of the most important protocols of this layer are **HTTP**, **SMTP**, **FTP**, **DNS**...

The packets at this layer are referred to as **messages**.

1.14.2 Transport Layer

The transport layer transports application-layer messages between application endpoints.

The two layers of this protocol are **TCP** and **UDP**. TCP provides a connection-oriented service. It guarantees the delivery of messages to the destination and flow control. UDP, on the other hand provides no reliability, no flow control and no congestion control.

The packets at this layer are referred to as **segments**.

1.14.3 Network Layer

The network layer moves packets from one host to the other. It also provides the service of delivering the segment to the transport layer in the destination host.

The protocol provided at this level is the **IP protocol**, which defines how routers and end systems act on the packet's fields. All Internet components that have a network layer must run the IP protocol.

The packets at this layer are referred to as **datagrams**.

1.14.4 Link Layer

The link layer moves a packet from one node to the next node in the route.

The protocols used in this layer are for example **Ethernet**, **Wi-Fi** and **DOCSIS**. A datagram might be handled by a different protocol for each node it traverses.

The packets at this layer are referred to as **frames**.

1.14.5 Physical Layer

The physical layer moves the individual bits within the frame from one node to the next.

The protocols in this layer depend on the link and on the transmission medium.

1.15 Encapsulation

Routers and link switches do not incorporate all of the layers in the TCP/IP protocol stack.

Each time the packet goes from a layer to another, a header or trailer is added to it. These headers and trailers contain information about the layer, such as the IP address and MAC address of the destination end system.

From layers 1 to 3 (application, transport and network layers) headers are added at each layer. Instead in layer 4 (link layer), both a header and a trailer are added to the packet. The trailer is used to verify the integrity of the packet.

In more complex cases, a large message may be divided into multiple transport-layer segments. These segments would have to be reconstructed by the destination host in order to get the full message.

2 Application Layer

2.1 Architectures

The application architecture is designed by the application developer and dictates how the application is structured over the various end systems.

2.1.1 Client-Server Architecture

In this architecture there is an always-on host which has a permanent IP address. This host (server) receives requests from other hosts (clients).

2.1.2 Peer-to-Peer Architecture

In this architecture there is no always-on host. Here the direct communication between peers is exploited. P2P architectures are self-scalable and cost effective. There are also problems with security, performance and reliability.

2.2 Process Communication

Processes that are executing in different hosts communicate by exchanging messages. These processes are called the **client process** (which initiates the communication) and the **server process** (which awaits to be connected).

2.2.1 Sockets

A socket is the interface between the application and the transport layer within a host (i.e. the API between the application and the network). The only control the application developer has on the transport-layer side is on:

- Choice of the transport protocol
- Change the buffer size or maximum segment size

2.2.2 Sever Process Identification

In order to identify a server process, two pieces of information are needed:

- **IP Address**

  It is a **32-bit** or **64-bit** (respectively IPv4 and IPv6) quantity that can uniquely identify a host.

- **Port Number**

  Since many different processes could run on a host, the IP is not sufficient to determine a specific process. To identify a process in a host, a port number is needed. Every port number is associated with a specific process.

2.2.3 Services Needed by the Layer

The following services are needed by the application layer:

- **Reliable Data Transfer**

  The transport layer may need to provide a reliable data transfer rate. There exist applications that are **loss-tolerant** and application that require **100% reliable** data transfer rates. In the first case, applications can tolerate some loss of packets.

- **Throughput**

  A transport service may need to provide guaranteed available throughput at some specified rate. There exist applications that are **bandwidth-sensitive**, and other that are **elastic**. In the first case, applications require a minimum amount of throughput to be effective. In the second case, applications make use of whatever throughput they get.

- **Timing**

  A transport service may need to provide timing guarantees available throughput at some specified rate. Low delays are preferred to high delays.

- **Security**

  A transport service may need to provide the application with one or more security services, such as encryption, data integrity or end-point authentication.

2.3 Transport Services

The two protocols offered when creating a network application are **UDP** and **TCP**.

2.3.1 TCP

It includes a connection-oriented service and a reliable data transfer service. Both services are received by the application when TCP is invoked.

- **Connection-Oriented Service**

  The client and server exchange transport-layer control information with each other before the application-level messages begin to flow. This is known as **handshaking**. After a successful handshake, a connection is said to exist between the sockets of the two processes. Both processes can now send messages at the same time.

  The connection is **full-duplex**, and the connection is teared down after the application has finished sending messages.

- **Reliable Data Transfer Service**

  The communication process can rely on TCP to deliver all of the data sent, without errors and in the proper order. It also has a congestion-control mechanism, which throttles a sending process when the network is congested between sender and receiver. It also attempts to limit each TCP connection to its fair share of network bandwidth.

2.3.2 UDP

It is a connectionless and lightweight protocol. It provides an unreliable data transfer service, no guarantee that the message will even reach the destination, the message might not arrive in order and there is no congestion-control mechanism.

2.4 Application-Level Protocols

An application-level protocol defines how an application's processes – running on different end systems – pass messages to each other. In particular they define the following:

- The type of messages exchanged
- The syntax of messages
- The semantics of the fields
- Rules of determining when and how a process sends and receives messages

There are two types of protocols:

- **Open Protocols**

  HTTP, SMTP, FTP, Telnet...

- **Proprietary Protocols**

  Skype, BitTorrent...

2.5 The Web and HTTP

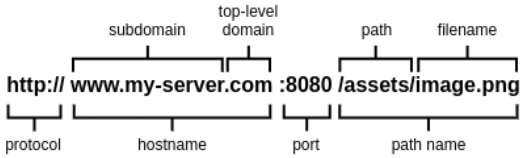
The **HTTP** (HyperText Transfer Protocol) protocol is at the heart of the Web. HTTP is implemented in two separate programs: a client program and a server program. These two programs communicate with each other by means of HTTP messages. HTTP defines how clients request Web pages from servers, and how servers transfer Web pages to clients.

A **Web page** is a document stored inside of a Web server, and contains objects. An **object** is a file that is addressable by a single URL.

When a user sends an **HTTP request**, the server receives it and responds with an **HTTP response** containing the requested objects.

2.5.1 URL

The **URL** (Uniform Resource Locator) of a web page is composed of the following parts:



2.5.2 HTTP Connection

HTTP uses **TCP** as its underlying protocol. In order to obtain an HTTP connection, the following steps must be followed:

1. The client initiates a TCP connection (by creating a socket) to the server on port 80;
2. The server accept the TCP connection request made by the client;
3. HTTP messages are now exchanged through the TCP link, from the client's to the server's socket;
4. The TCP connection is then dismantled when it is not needed anymore.

2.5.3 Persistent and Non-Persistent HTTP

HTTP is a **stateless protocol**. This means that it does not maintain any information about past client requests. There are two types of stateless HTTP connections:

- **Non-Persistent HTTP Connection**

  At most one object can be sent on this type of TCP connection. Once the response has been sent from the server to the client, the TCP connection is terminated. In total, the response time of an HTTP server is of 2 **RTTs** (Round Trip Times), plus the transmission time at the server of the HTML file.

  In their default modes, browsers open 5 to 10 parallel TCP connections, and each of these connections handles one request-response transaction.

- **Persistent HTTP Connection**

  Multiple objects can be sent over a single TCP connection between client and server. As little as one RTT is used for all the referenced objects.

  Persistent HTTP connections can also be pipelined, in which case two RTTs are needed to establish the connection, and one RTT is needed for **all** requested objects.

2.6 Round-Trip Time

The **RTT** (Round-Trip Time) is the time taken by a small packet to travel from a client to a server, and back. One RTT is used to initiate the TCP connection, and one RTT is used for the HTTP request and the first few bites of the HTTP response to arrive.

2.7 HTTP Message Format

The following is an example of HTTP request:

```
1  GET /doc/test.html HTTP/1.1 \r\n
   \_/ \____________/ \______/
    |         |          |
   Method    URL       Version

-----------------------------------------------------------------------

2  Host: my-server.com \r\n
.  ...
5  User-Agent: Mozilla/4.0 \r\n\r\n
   \_________/ \_________/
        |           |
       Name       Value

-----------------------------------------------------------------------

7  userName=Edoardo&userSurname=Riggio
```

Line 1: Request Line;
Lines 2-5: Header Lines;
Line 7: Body.

The following is an example of HTTP response:

```
1  HTTP/1.1 200 OK \r\n
   \______/ \_/ \/
       |     |   Phrase
    Version Status

-----------------------------------------------------------------

2  Date: Sat, 27 March 2021 15:51:20 GMT \r\n
.  ...
5  Content-Type: text/html \r\n\r\n
   \___________/ \_______/
         |           |
        Name       Value

-----------------------------------------------------------------

7  <h1> Edoardo Riggio </h1>
```

Line 1: Response Line;
Lines 2-5: Header Lines;
Line 7: Body.

The HTTP status codes are:

- **1xx** - Informational
- **2xx** - Success
- **3xx** - Redirection
- **4xx** - Client Error
- **5xx** - Server Error

2.8 Cookies

Since HTTP is a stateless protocol, in order to save info about the session we use cookies. In order to send and receive cookies, the following is done by the client and server:

1. The client sends an HTTP request to the server;
2. The server responds with a `Set-Cookie: xxxx` header;
3. The client then does another HTTP request, this time including the `Cookie: xxxx` header;

4. The server now responds with a normal HTTP response.

By using cookies, a server knows exactly which pages the user visited, in which order, and at what time.

2.9 Web Caching

A **web cache** is a network entity that satisfies HTTP requests on behalf of an origin web server. The goal of this type of cache is to satisfy requests without involving the origin server, The cache acts as both client (for the main server) and server (for the clients). It is installed by the ISP.

Web caching is used in order to reduce the response time for the client requests, reduce the traffic on an institution's access link, and substantially reduce Web traffic in the Internet as a whole – improving the performances for all applications.

Through the use of **CDNs** (Content Distribution Networks), Web caches are playing an important role in the Internet. The goal of CDNs is to localize traffic as much as possible.

2.9.1 Conditional GET

One problem that caches introduce are possible **stale objects**. HTTP can verify if it's actually stale with a **conditional GET**. This request includes the `If-Modified-Since: xxxx` header, so that the cache can always know if an object if stale or not.

2.10 SMTP

SMTP (Simple Mail Transfer Protocol) is a protocol that uses TCP to reliably transfer mail from client to server. Typically, a SMTP process is located on port 25 of a server.

An email server is directly sent from one SMTP server to another, no in-between server are used. SMTP is mainly a push protocol, and multiple objects in SMTP are sent in multipart messages (MIME – Multipurpose Internet Mail Extensions).

SMTP uses TCP as its underlying transport service. The following is an example of an SMTP exchange:

```
S: 220 server.com
C: HELO usi.ch
S: 250 Hello, pleased to meet you
C: MAIL FROM: <riggie@usi.ch>
S: 250 riggie@usi.ch ... Sender ok
C: RCPT TO: <mariosam@usi.ch>
S: 250 mariosam@usi.ch ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Long time no see!
C: Would you like to hang out sometime?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 server.com closing connection
```

2.11 Mail Access Protocols

Once the email has been delivered using SMTP, the user needs to access its mailbox in order to read emails. To do so, one of the following protocols is used:

- **POP3**

  The **POP3** (Postal Office Protocol version 3) protocol allows a user to download emails from a server to its computer. It is a stateless protocol. There are two modes available in POP3:

  - **Download-and-Delete Mode**

    The messages are downloaded to the computer and are later (when the user issues the `QUIT` command) deleted from the mailbox on the server.

  - **Download-and-Keep Mode**

    The messages are kept in the mailbox even after the user has logged out.

- **IMAP**

  The **IMAP** (Internet Mail Access Protocol) protocol allows a user to consult their emails from a remote server. By doing so, it can be accessed simultaneously by multiple clients.

- **HTTP**

  Mails are passed to the client using the **HTTP** protocol, rather than POP3 or IMAP. In this case emails are sent to the source server also using HTTP (transfer to destination service always happens through SMTP).

2.12 DNS

The **DNS** (Domain Name System) is a distributed database implemented in a hierarchy of DNS servers, and it is an application level protocol that allows hosts to query the distributed database.

The DNS protocol runs over UDP and uses port 53. In addition to translating hostnames into IP addresses, DNS provides a few additional services, such as:

- **Host Aliasing**

  A host with a complicated name could have one or more alias names. In that case, one of them is said to be the **canonical name**.

- **Mail Server Aliasing**

  DNS can be invoked by a mail application in order to obtain the canonical hostname for a supplied alias hostname as well as the IP address of the host. Moreover, the MX record of the DNS allows a company to use the same hostname for both their Web server and mail server.

- **Load Distribution**

  A set of IP addresses could be associated with the canonical hostname (for example when dealing with websites with lots of incoming traffic). The DNS in this case rotates the address every time a request is made.

2.12.1 DNS Servers Hierarchy

DNS uses a large number of servers organized in a hierarchical fashion. There are four classes of DNS servers:

- **Root DNS Server**

  These servers provide the IP addresses of the TLD servers. There are over 400 root servers around the world.

- **TLD Server**

  A TLD (Top-Level Domain) sever manages a top-level domain (e.g. .it, .com, .net...). TLD servers provide the IP address for authoritative DNS servers.

- **Authoritative DNS Servers**

  Every organisation with publicly accessible hosts on the Internet must provide publicly accessible DNS records. These organisations can either create a personal authoritative DNS server, or use some service provider.

- **Local DNS Server**

  The local DNS server has a local cache of recent name-to-address translation pairs. This is the server where the user query first arrives.

DNS queries usually follow a recursive pattern from the requesting host to the local DNS, and an iterative pattern for the remaining steps.

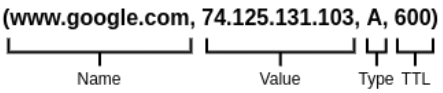
2.12.2 DNS Caching

When a DNS server receives a DNS reply, this can be cached in the server's local memory. This is mainly done to improve the delay performance and to reduce the number of DNS messages.

DNS servers will discard the cached information after a certain period of time (usually set to two days).

2.12.3 DNS Records

The DNS servers that together implement the DNS distributed database store **RRs** (Resource Records). An RR is a four-tuple. For example:



There are four types of RRs:

- **Type = A**

  `Name` is the hostname and `Value` is the hostname's IP address. This is a standard hostname-to-IP address mapping.

- **Type = NS**

  `Name` is the domain and `Value` is the domain's hostname of an authoritative DNS server that knows how to obtain the domain. This RR is used to route DNS queries further along in the query chain.

- **Type = CNAME**

  `Name` is the alias hostname and `Value` is the hostname's canonical hostname.

- **Type = MX**

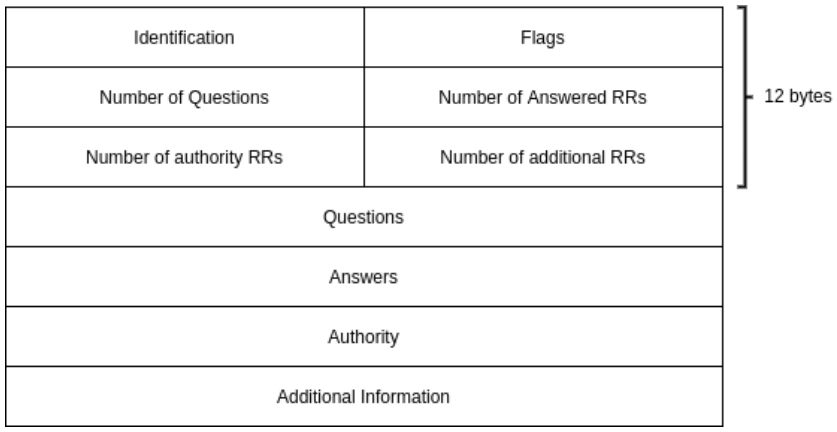
  `Name` is the alias hostname and `Value` is the mail server hostname's canonical hostname.

If a DNS server is authoritative for a particular hostname, then the DNS server will contain **Type A** records for that hostname.

If a DNS server is not authoritative for a particular hostname, then the server will contain a **Type NS** record for the domain that includes the hostname. Furthermore, it will also contain a **Type A** record that provides the IP address of the DNS server contained in the `Value` field of the NS record.

2.12.4 DNS Messages

The DNS query and reply messages are structured as follows:



The following are the fields of a DNS message:

- **Identification**

  It is a 16-bit number that identifies the query. The same identifier is set to the corresponding answer.

- **Flags**

  There are a series of flags that identify the message. For example we have 1 bit describing if the message is a query or a response, 1 bit to identify if the server is an authoritative server...

- **Questions**

  Contains information about the query that is being made. This section includes: a **name field** containing the name that is being queried, and a **type field** which indicates the type of question being asked about the name.

- **Answers**

  Resource records for the name that was queried. It can contain multiple RRs.

- **Authority**

  It contains records of other authoritative servers.

- **Additional Information**

  It contains additional information, for example info about the canonical hostname of a server in response to an MX or CNAME query.

2.13 Peer-to-Peer File Distribution

In the P2P paradigm, **peers** are pairs of intermittently connected hosts. These peers communicate directly with each other, and are not owned by any service provider. The following are the two approaches for distributing a file.

2.13.1 Client-Server Approach

- **Sending the File**

$$\frac{F}{u_s} \quad \text{(For 1 copy)}$$

$$N \cdot \frac{F}{u_s} \quad \text{(For N copies)}$$

- **Downloading a file**

$$d_{min} = \min\{d_1, d_2, ..., d_N\} \quad \text{(min download rate)}$$

$$\frac{F}{d_{min}} \quad \text{(min download time)}$$

- **Time to Distribute F to N Peers**

$$D_{CS} = \max\left\{\frac{NF}{u_s}, \frac{F}{d_{min}}\right\}$$

2.13.2 Peer-to-Peer Approach

- **Sending the File**

$$\frac{F}{u_s}$$

- **Downloading a file**

$$u_s + \sum_{i=1}^{N} u_i \quad \text{(min download rate)}$$

$$\frac{F}{d_{min}} \quad \text{(min download time)}$$

- **Time to Distribute F to N Peers**

$$D_{P2P} = \max \left\{ \frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i} \right\}$$

2.14 BitTorrent

BitTorrent is a popular P2P protocol for file distribution. In BitTorrent, a collection of peers participating in the distribution of a particular file is called a **torrent**. Peers in a torrent download equal-sized chunks of size 256 kbytes. Once a peer has acquired the entire file, it can choose to either leave the torrent, or stay in the torrent. Once a torrent is left, it can later be rejoined.

Each torrent has an infrastructure node called a **tracker**. This tracker will provide the user with a subset of peers from the participating peers on the list. The new peer now tries to establish a concurrent TCP connection with all of its peers.

As time goes on, some peers might leave the torrent, and some others might join the torrent. The **rarest chunks** are requested first, this is done in order to equalize the numbers of copies of each chunk in a torrent.

The peer gives priority to the neighbors that are supplying data at the highest rate. The peer reciprocates by sending data to the best four peers, any other peer is said to be **choked**, meaning that the user does not send chunks to them.

Every 10 seconds, the peer recalculates the rates of its neighbors and possibly modifies its set of four peers – which are said to be **unchoked**. Every 30 seconds, the user also picks up a new peer – which is said to be **optimistically unchoked**, since it may become one of the user's top four uploaders.

2.15 Video Streaming

Videos are pre-recorded and saved onto a server. Users can request these videos on-demand from their end systems. These videos are compressed to a desired bit-rate. The higher the bit-rate, the better the image quality. There are two main methods to encode videos:

- **Spacial Coding**

  This is when instead of sending N values of the same color, only two values are sent: the color value and the times it is repeated.

- **Temporal Coding**

  This is when instead of sending a complete frame $i + 1$ (where i is the current frame), only the differences from frame i are sent.

Here are some of the bitrates of different encodings:

- **MPEG 1** - 1.5 Mbps
- **MPEG 2** - 3-6 Mbps
- **MPEG 3** - 64 Kbps-12 Mbps

2.16 DASH

DASH (Dynamic Adaptive Streaming over HTTP) is an HTTP-based streaming protocol. In DASH, the video is encoded into several different formats, everyone with different bit-rates. The servers keeps each version, and offers a **manifest file**. In this file there are all the bit-rates offered by the server and the location of the corresponding video.

The client can choose the bit-rate which is suitable to its needs, namely the current bandwidth of the client.

2.17 CDN

A **CDN** (Content Distribution Network) manages servers located in multiple geographically distributed locations. CDNs replicates content across its clusters. Many CDNs may not want to push videos to their clusters, but instead use a simple pull strategy. If a client is requesting a video that is not on that cluster, then the cluster retrieves the video and stores a local copy. There are two main server placement strategies for CDNs:

- **Enter Deep**

  Deploy server clusters in access ISPs all over the world. The goal is to get as close as possible to the user. In this approach, maintaining and managing the server clusters is challenging and expensive.

- **Bring Home**

  Build larger server clusters at a smaller number of sites. These CDNs are usually placed in IXPs. This approach has a lower maintenance overhead.

A CDN must intercept the video request made by a client and perform two operations: determine a suitable CDN server cluster for that client at that time, and redirect the client's request to a server in that cluster. To do so most CDNs take advantage of DNS to intercept and redirect requests.

In order to select the best cluster for a user a particular time, two methods ca be used:

- **Geographically Closest Cluster**

  Simply assign the cluster that is geographically closer to the client. This

is not always a good idea, because the closest client might be the one with the greatest number of hops, thus the client might experience big delays.

- **Current Traffic Conditions**

  In this case the CDN periodically computes the current traffic conditions and, based on these calculations, decides which server cluster the client must be connected to.

3 Transport Layer

A transport-layer protocol provides for logical communication between the application processes running on different hosts. From an application's perspective, it is as if the two hosts were directly connected.

There are two types of protocol actions in end systems:

- **Sender**

  Breaks the application message into segments, which are then passed to the Network Layer.

- **Receiver**

  Reassembles the segments into messages, which are then passed to the Application Layer.

3.1 Constraints from the Network Layer

The **Network Layer** uses the **IP** (Internet Protocol) protocol. This protocol provides for the logical communication between hosts, where each host has a unique IP address.

The IP protocol is what's known as a best-effort delivery service. This means that there is no guarantees of in-order delivery or even delivery per se.

3.2 Transport Layer Protocol

The transport layer has two protocols:

- **TCP**

  TCP (Transmission Control Protocol) offers a reliable, in-order delivery. It also offers connection setup, flow control and congestion control.

- **UDP**

  UDP (User Datagram Protocol) offers an unreliable and unordered delivery.

Neither TCP nor UDP provide either delay or bandwidth guarantees. Instead, what they both provide is integrity checking and multiplexing-demultiplexing.

3.3 Multiplexing and Demultiplexing

Multiplexing and demultiplexing are used in order to extend the host-to-host delivery service provided by the network layer to a process-to-process delivery service for applications running on the hosts. The following are the ways multiplexing and demultiplexing are implemented:

- **Multiplexing at Sender**

  It is the job of gathering data chunks at the source host from different sockets, encapsulating each chunk of data with header information in order to create segments, and passing the segments to the Network Layer.

- **Demultiplexing at Receiver**

  It is the job of delivering the data in a transport-layer segment to the correct socket.

Sockets have unique identifiers and each segment has a special field that indicates the socket to which the segment is to be delivered. These fields are the **source port number field** and the **destination port number field**.

Port numbers ranging from 0-1023 are called **well-known ports**. These ports are restricted, and reserved to well-known application protocols (HTTP, FTP...).

3.3.1 Connectionless Multiplexing and Demultiplexing

When creating a segment to send into a UDP socket, the sender must specify the following:

- **Destination IP Address**
- **Destination Port Number**

When the receiving host receives the UDP segment, it does two things:

- Checks the destination port number in the segment
- Directs the UDP segment to the socket with that port number

All UDP datagrams with the same destination port number will be directed to the same socket at the receiving host.

3.3.2 Connection-Oriented Multiplexing and Demultiplexing

A TCP connection is defined by a 4-tuple:

- **Source IP Address**
- **Source Port Number**
- **Destination IP Address**
- **Destination Port Number**

In the demultiplexing process, the receiver uses all four values of the tuple to direct the segment to the appropriate socket.

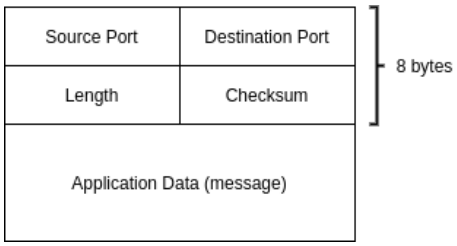
Servers may also support may simultaneous TCP connections. Each connection is identified by its own 4-tuple, and to each of these connections is associated a different connecting client.

3.4 UDP

There are several peculiarities of UDP, such as:

- Finer application-level control over what data is sent and when
- No connection is established
- There is no connection state neither at the sender nor at the receiver
- It has only 8 bytes of overhead
- No congestion control

The following is the structure of a UDP segment:



The fields are:

- **Length**

  This field specifies the total number of bytes in the UDP header (header + data)

- **Checksum**

  It is used to verify the integrity of the UDP segment by revealing bit-flips.

3.4.1 UDP Checksum

The checksum is treated in different ways by both server and receiver.

- **Server**

  It treats the UDP segment fields contents as sequences of 16-bit integers. These integers are all added up, and the final checksum is calculated as the one's complement of the actual sum. Finally the result is stored inside of the checksum field of the UDP segment.

- **Receiver**

  All of the segment's fields are summed up and it is added to the sender's checksum. If any bit with value of 0 is detected in the final sum, then there is an error in the packet. The packet is thus discarded by the receiver.

```
- Sender

  Source Port:       0110011001100000 +
  Destination Port:  0101010101010101 +
  Length:            1000111100001100 =
                     ------------------
                     0100011011000001
                             |
                             | 1's complement
                             V
                     1011100100111110

- Receiver

  Source Port:       0110011001100000 +
  Destination Port:  0101010101010101 +
  Length:            1000111100001100 =
                     ------------------
                     0100011011000001 +
  Sender Checksum:   1011100100111110 =
                     ------------------
                     1111111111111111   <- The checksum is valid
```

3.5 Reliable Data Transfer

Reliable data transfer is when no transferred data bits are corrupted or lost, and all transferred data bits are delivered in the order they where sent in. **TCP**

offers reliable data transfer.

3.5.1 Rdt 2.0

The underlying channel in charge of sending packets may flip bits while transmitting. Reliable data transfer protocols based on retransmission are known as **ARQ** (Automatic Repeat reQest). In order to see if there are bit-flips or not, we use:

- **ACKs (Acknowledgments)**

  The receiver specifically tells the sender that the packet arrived correctly.

- **NAKs (Negative AcKnowledgments)**

  The receiver specifically tells the sender that the packet arrived with errors. The sender needs to retransmit the package.

This version of the protocol uses a **Stop & Wait** technique, meaning that the sender sends the packet and waits for a response to arrive.

3.5.2 Rdt 2.1

In this version of the protocol a sequence number is added to the packet. By doing so, the infrastructure has to remember whether the expected packet should have a sequence number of 1 or 0. Furthermore, it needs to check is the packet that was sent is a duplicate or not.

This protocol still uses ACKs and NAKs in order to check if the packets are corrupted.

3.5.3 Rdt 2.2

This protocol has the same features that version 2.1 has. In this case, though, the protocol only uses ACKs. Thus the receiver sends back another ACK in the case that everything is OK, and the packet arrived correctly.

3.5.4 Rdt 3.0

By using this protocol we assume that the channel can also lose packets. In such cases retransmission is not enough. A **timeout** is introduced.

The sender waits a predetermined amount of time (the timeout) in order for it to receive the ACK. If after this amount of time no ACK has been received, then the packet is retransmitted.

The performance of this version is poor. This is because it spends more time waiting than working.

3.5.5 Rdt 3.0 with Pipelining

In pipelining operations, the senders allows for multiple yet-to-be-acknowledged packets to be sent.

This operation can increase by a significant amount the usage and performance of the infrastructure. This is where the **Go-Back-N** protocols come to play.

3.5.6 Go-Back-N

The **Go-Back-N** protocol allows the sender to transmit multiple packets without waiting for an acknowledgment.

The sender can have at most N unacknowledged packets in the pipeline. The sender holds a "window" of up to N consecutive transmitted but unACKed packets.

The following function is to indicate that n packets (n included) have to be all ACKed.

```
ACK(n)
```

The following function indicates that all packets with a higher sequence number than n in the window have to be retransmitted.

```
Timeout(n)
```

If a packet has been received out-of-order, it can either discard it or buffer it. The out-of-order packet is then reACKed based on which packet has the highest sequence number.

The main problem with GBN is that a packet error would cause the retransmission of many other packets.

3.5.7 Selective Repeat

The receiver individually acknowledges all correctly received packets. Furthermore, the sender retransmits individually for unACKed packets – the sender needs to maintain a timer for each packet.

3.6 TCP

TCP (Transmission Control Protocol) implements a reliable data transfer service. This includes services such as:

- **Error Detection**
- **Retransmission**

- **Cumulative Acknowledgments**
- **Timeouts**
- **Header Fields for Sequence Numbers**
- **Header Fields for Acknowledgment Numbers**

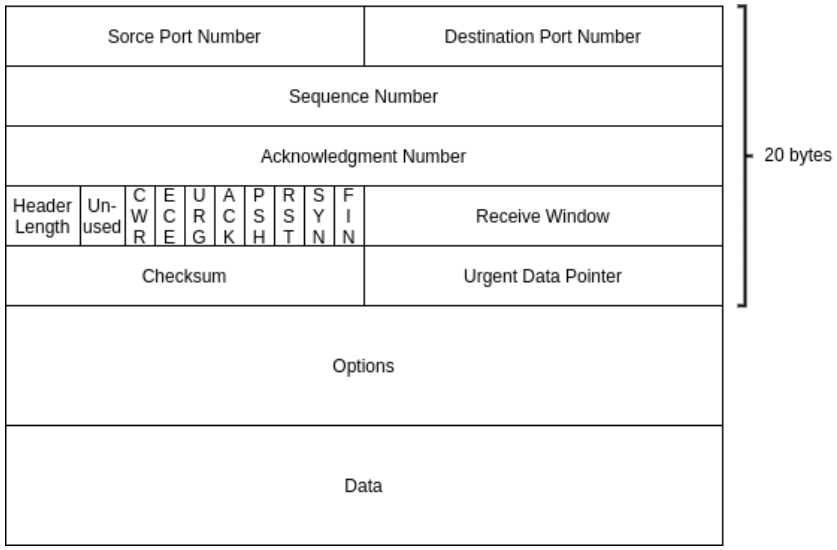
TCP is a **connection-oriented** protocol. This connection is full-duplex and point-to-point.

3.6.1 MSS and MTU

The **MSS** (Maximum Segment Size) is the maximum amount of application-layer data in the segment. The **MTU** is the Maximum Transmission Unit.

3.6.2 TCP Segment Structure

The following is the structure of a TCP segment:



Some of the main sections are:

- **Sequence Number**

  It is the byte-stream number of the first byte in the segment.

- **Acknowledgment Number**

  It is put by the host and is the sequence number of the next byte the host is expecting from the other host in the connection.

TCP only acknowledges bytes up to the first missing byte in the stream. For this reason, TCP is said to provide **cumulative acknowledgments**.

TCP also uses timeouts in order to determine if a packet has to be retransmitted or not. The timeout value can be difficult to set. If it is too short, there is the risk of premature timeouts and unnecessary retransmissions. On the other hand, if the timeout is too long, there is a slow reaction to segment loss.

3.6.3 Estimating the RTT

In order to estimate the round trip time and find a suitable timeout value, we use the following formula:

$$RTT_{Est.} = (1 - \alpha) \cdot RTT_{Est.} + \alpha \cdot RTT_{Sample}$$

This is also known as the **EWMA** (Exponential Weighted Moving Average). Typically we have $\alpha = 0.125$.

3.6.4 Variability of RTT

In order to set the proper timeout value, we have to need a "safety margin". Mathematically, this is defined as:

$$Ti = RTT_{Est.} + 4 \cdot RTT_{Dev}$$

Where:

$$RTT_{Dev} = (1 - \beta) \cdot RTT_{Dev} + \beta \cdot |RTT_{Sample} - RTT_{Est.}|$$

Here typically we have $\beta = 0.25$.

3.7 TCP Flow Control

The **receive window** field inside of the TCP segment controls the sender. This means that the sender will never overflow the receiver's buffer by transmitting too much, too quickly. The size of the window is computed as:

```
rwnd = RcvBuffer - [LastByteRcv - LastByteRead]
```

3.8 TCP Connection Management

Before sending and receiving data, the sender and receiver handshake. This is done in order to agree to establish connection, and on the connection parameters.

Two-way handshakes do not always work in networks. There are several reasons why. For example there can be variable delays, some messages could be re-transmitted or there could be message reordering. In such cases an half-opened connection could be established. This is why a three-way handshake is instead needed.

3.9 Congestion Control

A congestion happens when too many senders are sending too much data at once. This could cause long delays and packet loss.

These are some congestion insights:

- Throughput can never exceed capacity
- Delay increases as the maximum capacity is approached
- Packet loss and re-transmission decreases the effective throughout
- Unneeded duplicates further decreases the effective throughput
- Upstream transmission capacity and buffering are wasted for packets lost downstream

3.9.1 End-to-End Congestion Delay

In order to mitigate congestion, TCP provides an **end-to-end congestion control**. In this case there is no explicit feedback from the network, and the congestion is inferred from the amount of packets that are lost and from the delay.

3.9.2 Network-Assisted Congestion Control

In the case of **network-assisted congestion control**, the routers provide direct feedback to sending and receiving hosts with flows passing through the congested router. This could indicate the congestion level or explicitly set the sending rate.

3.10 TCP Congestion Control

TCP implement what is known as **AIMD** (Additive Increase/Multiplicative Decrease). In this approach senders can increase the sending rate until packet loss occurs. When this happens, the sender will decrease the sending rate on loss events.

Additive increase will increase the sending rate by 1 maximum segment size every RTT until loss is detected. When loss is detected, we have a **multiplicative decrease**, where the sending rate is cut in half.

Loss is detected by the arrival of a triple ACK, and the rate is cut in half. When loss detection by timeout is detected, instead, will make sure that the rate is cut to 1 MSS (Maximum Segment Size).

A TCP connection is initially slow, with an initial cwnd of 1 MSS. For every RTT where there is no loss, the cwnd is doubled. This is done by incrementing the cwnd by one for every ACK the sender receives.

4 Network Layer

The network layer's goal is to transport the segment from the sending to the receiving host. The network layer is present in every Internet device.

A **router**'s job is to examine the header files in all IP datagrams passing through it. After being examined, the datagrams are moved from input ports to output ports, in order to transfer them along their end-to-end path.

There are two key functions of the network layer, these are:

- **Forwarding**

  To move packets from a router input link to the appropriate output link.

- **Routing**

  To determine the route taken by packets from source to destination.

There are also two main elements in the network layer:

- **Data Plane**

  It is a local, per-router function. It determines how datagrams arriving in the router are forwarded to router output ports.

- **Control Plane**

  It is a network-wide logic. This determines how datagrams are routed among end-to-end paths from source to destination.

  There are two control plane approaches. The first involves **traditional routing algorithms**, and are directly implemented in routers. Another approach involves **SDN** (Software-Defined Networking), which is implemented in remote servers. Thus there are two kinds of control planes:

- **Per-Router Control Plane**

  The individual routing algorithm components in each router interact inside of the control plane.

- **SDN Control Plane**

  The remote servers compute and install the forwarding tables directly on the routers.

4.1 Network Service Model

The Internet Network Layer does not guarantee the following:

- Successful packet delivery to destination
- Timing or order of delivery
- Bandwidth available to end-to-end flow

It simply offers a best-effort service. The mechanism is fairly simple, sufficient provisioning of bandwidth allows performance of real-time applications to be "good enough". The replication of application-layer distributed services allow these services to be provided from multiple locations.

4.2 Architecture of a Router

The four core components of a router are:

- **Input Ports**
- **Output Ports**
- **Switching Fabric**
- **Routing Processor**

While the first three components are part of the **Forwarding Data Plane**, the last component is part of the **Control Plane**.

4.2.1 Input Ports

Input ports are composed of three layers:

- **Line Termination**

  It is the physical layer of the router.

- **Link Layer Protocol**

- **Lookup, Forwarding and Queuing**

  If the datagrams arrive faster than anticipated, then they will enter a queue. The header of the packet is used in order to lookup the output port – which is the second function of this layer.

  When looking in the forwarding table for a given destination address, the router uses the **longest prefix** that matches the destination address.

  If a control packet is received, then the switch forwards it directly to the routing processor.

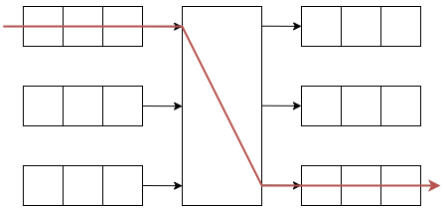
4.2.2 Switching Fabric

The goal of the switching fabric is to transfer the packet from the input port to the appropriate output port.

The **switching rate** is the rate at which packets can be transferred from input to output ports.

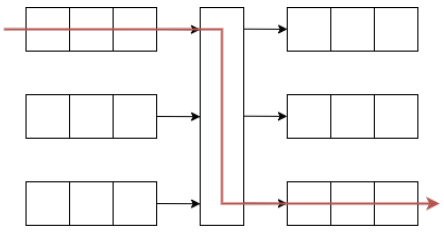
There are three main types of switching fabrics:

- **Memory**



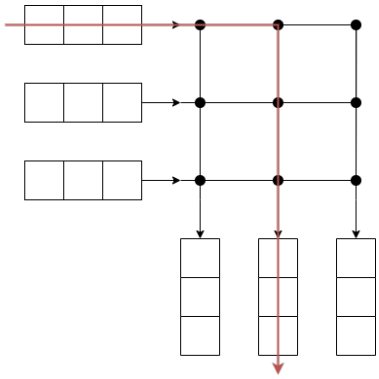
  The packet is copied to the system's memory, and then transferred to the correct output port. The speed is limited by the memory bandwidth and by its physical space.

- **Bus**



The datagram is passed from the input port to the output port via a shared bus. The switching speed is limited by the bus' bandwidth.

- **Interconnection Network**



It is an interconnection network consisting of $2N$ buses that connect N input ports to N output ports. The crosspoints at which the buses intersect with each other can be opened or closed by the switching fabric.

4.2.3 Input Port Queuing

If a switch fabric is slower than the input and output port rate combined, queuing may occur at the input ports.

We have **HOL** (Head-Of-the-Line) blocking when queued datagrams at the front of the queue prevent others in the queue from moving forward. There are two steps in which the queues are handled:

- **Output Port Contention**

  Only the first in the queue is transferred, while the others are blocked.

- **One Packet Time Later**

  The next packet in the queue will suffer from HOL blocking.

4.2.4 Output Port Queuing

Buffering is required when datagrams arrive from the fabric faster than the link transmission rate. This could cause datagrams to be lost – congestion or lack of buffers could be the causes.
The scheduling discipline chooses among queued datagrams those to be transmitted – this is called **priority scheduling**. In order to calculate the buffering,

we can use the following formula:

$$Buf = RTT \cdot \frac{C}{N}$$

The router buffers drop packets whenever they are full. There are two ways packets can be dropped:

- **Tail Drop**

  Drop the incoming packet.

- **Priority Drop**

  Drop or remove a packet on a priority basis.

4.3 Packet Scheduling

The scheduling algorithm decides which packet to send next on the link. There are several scheduling algorithms, such as:

- **FCFS/FIFO**

  FCFS (First Come, First Served) is when packets are transmitted in order of arrival, to the output port.

- **Priority**

  The arriving packets are classified by any header field of the packet. The packets with the highest priority are sent first.

- **Round Robin**

  The arriving packets are classified by any header field of the packet. The server now cyclically scans the queues, sending on complete packets from each input in turn.

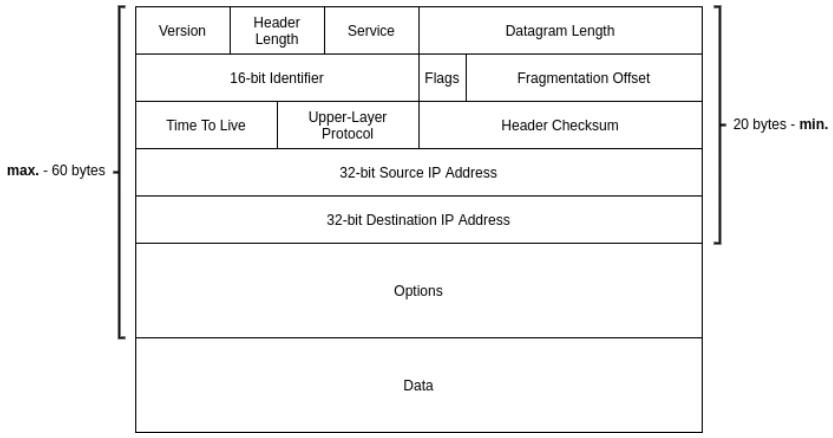
- **WFQ**

  WFQ (Weighted Fair Queuing) is a generalized Round Robin algorithm. Each input has a weight. This weight is calculated using the following formula:

$$W_a = \frac{W_i}{\sum_i i}$$

4.4 IPv4 Datagram Format

The IPv4 datagram is composed by several field. The following is its visual representation:



Some of the fields are:

- **Version**

  Indicates the version of the IP protocol.

- **Type of Service**

  It indicates the Internet service quality selection (real-time or non-real-time)

- **Identifier**

  Uniquely identifies the fragments of a particular datagram.

- **Flags**

  It is composed of 3 bits. **Bit 1** must be 0, **Bit 2** indicates if the datagram may fragment (0 - Yes, 1 - No), and **Bit 3** indicates if this is the last fragment (0 - Yes, 1 - No).

- **Time To Live**

  It is decremented by one for every time that the datagram is processed by a router. If `TTL == 0`, the datagram is dropped.

- **Upper-Layer Protocol**

  Indicates the protocol to which to pass the datagram at the final destination (e.g. 6 → TCP, 17 → UDP...).

- **Header Checksum**

  The Internet checksum of the datagram is computed by treating each 2 bytes in the header as a number. This must be recomputed and stored again at each router.

- **Destination IP Address**

  It is inserted by the sender of the datagram, and determined by the DNS.

4.5 IPv4 Addresses

There are 2^{32} possible addresses in this format. To create such addresses, we use the dotted-decimal notation. The following is an example:

```
11000001 | 00100000 | 11011000 | 00001001
\______/   \______/   \______/   \______/
   |          |          |          |
  193    .    32    .   216    .    9
```

4.6 Network Interface

The boundary between the host and the physical link is called an **interface**. An IP address is technically associated with an interface, rather than the host containing that interface.

Each interface on every router and host must have an IP address that is globally unique.

4.7 Subnet Mask

A **subnet mask** is indicated by the leftmost x bits (e.g. 223.1.1.0/24, it would indicate the 24 leftmost digits) of an IP address.

In order to determine the subnets, detach each interface from its host/router, creating islands of isolated networks. Each of these isolated networks is called a **subnet**.

4.8 IP Assignment

IP can be assigned using classful addressing – which is obsolete, or by **CIDR** (Classless InterDomain Routing).

When a host sends a datagram with destination address 255.255.255.255, the message is delivered to all hosts on the same subnet. This address is also commonly referred to as the **broadcast address**.

4.8.1 Internet Number Registry System

The **ICANN** (Internet Corporation for Assigned Names and Numbers) distributes IP addresses. It also manages DNS root servers, assigns domains...

While router addresses are manually configured by network administrators, host

addresses are configured using **DHCP** (Dynamic Host Configuration Protocol). The goal of DHCP is to dynamically assign an IP address to the host, the moment it joins the network. The following are the steps of the DHCP:

1. The host broadcasts a **DHCP discover message**.
2. The DHCP server responds with a **DHCP offer message**.
3. The host requests the IP address by means of a **DHCP request message**.
4. The DHCP server sends back the address by means of a **DHCP ACK message**.

A DHCP server can return much more than just the allocated IP address on the subnet. DHCP can also return:

- The address of the first-hop router for the client.
- The name and IP address of the DNS.
- The network mask.

4.8.2 Network Address Translation

Thanks to the NAT, as far as the rest of the Internet is concerned, only one IPv4 address is used by a local network.

All of the devices in the local network have a 32-bit address in a private IP space. Some of the possible masks are:

- 10/8
- 172.16/12
- 192.168/16

The NAT has several jobs, such as:

- **Replace Outgoing Datagrams**

  The source IP address of every outgoing datagram is repplaced by the NAT IP address. Same thing goes for port numbers.

- **Remember**

  It has to internally store a table containing every translation pair.

- **Replace Incoming Datagrams**

  The NAT IP address of every incoming datagram is replaced by the source IP address stored in the NAT. Same thing goes for port numbers.

4.9 Internet Protocol Version 6

IPv6 was created because of the exhaustion of 32-bit IPv4 addresses. It also introduces some new features, such as:

- **Expanded Addressing Capabilities**

  The **anycast address** has been introduced with IPv6, and allows a datagram to be delivered to any one of a group of hosts (for example, an HTTP GET request can be sent to a number of mirror sites that contain a given document).

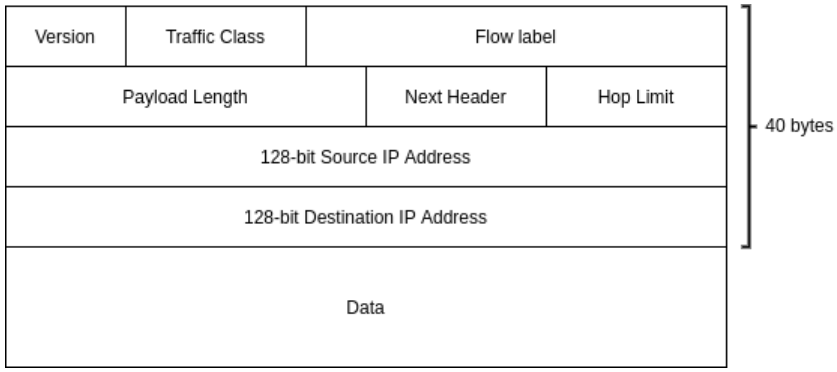
- **40-byte Header**

  This allows for a faster processing by routers.

- **Flow Labeling**

  A **flow** allows the labeling of packets belonging to particular flows for which the sender requests special handling (e.g. audio or video transmission, or traffic carried out by a high-priority user).

The following is a visual representation of an IPv6 datagram:



Some of the fields are:

- **Version**

  Version of the IP protocol.

- **Traffic Class**

  It is used to give priority to certain datagrams.

- **Flow Label**

  It identifies a flow of datagrams.

- **Next Header**

  Indicates the protocol to which to pass the datagram at the final destination.

- **Hop Limit**

  Same as TTL in IPv4 datagrams. In this case, the counter decrements with each hop the datagram makes (i.e. with each intermediate router it goes through).

4.10 Transition from IPv4 to IPv6

The Internet can operate with mixed IPv4 and IPv6 router thanks to **tunneling**. Tunneling enables for IPv6 datagrams to be transmitted as the payload of IPv4 datagrams among IPv4 routers.

4.11 Generalized Forwarding

In order to forward packets, each router has a **forwarding table**. Routers also follow a **match plus action** abstraction, meaning that they match the bits in the arriving packet, and then take action.

There exist two main types of forwarding:

- **Destination-Based Forwarding**

  The match and action is solely based on the destination IP address.

- **Generalized Forwarding**

  Many header fields can determine the action to be taken by the router. Many actions are possible, such as: drop, copy, modify, log the packet.

4.12 OpenFlow

OpenFlow is a communications protocol that gives access to the forwarding plane of a network switch or router over the network.

4.12.1 Flow Tables

The flow of the datagram is defined by the header field value of the datagram.

The actions of flow tables determine the processing that needs to be applied to a packet that matches a flow table entry. Some of the most important actions are: forwarding, dropping and modify fields.

4.13 Middleboxes

A middlebox is any intermediary box performing functions apart from normal, standard functions of an IP router on the data path between a source host and a destination host.

Examples of middleboxes are:

- NAT
- Firewall and IDS (Intrusion Detection System)
- Load balancers
- Caches

The services provided by the middleboxes can also be divided into three main categories:

- **NAT Translation**
- **Security Services**
- **Performance Enhancers**

Although middleboxes were initially proprietary, they have now moved towards **"whitebox" hardware** implementing an open API.

NFVs (Network Functions Virtualization) are programmable services that operate over whitebox networking.

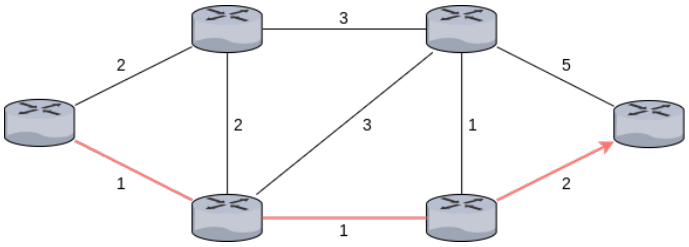
5 Control Plane

The main function of the control plane is **routing**. This means that it needs to determine the route that packets need to take to arrive to their destination.

5.1 Routing Protocols

Routing protocols are used to determine the best path, from sending to receiving host, through the network of routers.

A **path** is a sequence of routers, packets need to traverse from a given initial source host to a final destination host.



There are four types of routing algorithms:

- **Centralized**

  All routers have a complete topology and link cost information. These are the **link state** algorithms.

- **Decentralized**

  Iterative process of computation, where there is an exchanging of info with the neighbors. The routers initially only know the link cost to the attached neighbors. These are the **distance vector** algorithms.

- **Dynamic**

  Routes change more quickly. There are periodic updates in response to link cost changes.

- **Static**

  The routes change slowly over time.

- **Load-Sensitive**

  The link costs vary dynamically to reflect the current level of congestion of the underlying link.

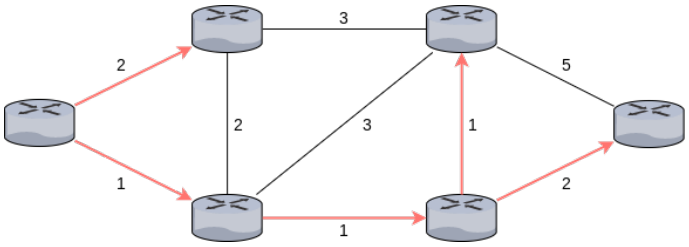
- **Load-Insensitive**

  A link's cost does not explicitly reflect its level of congestion.

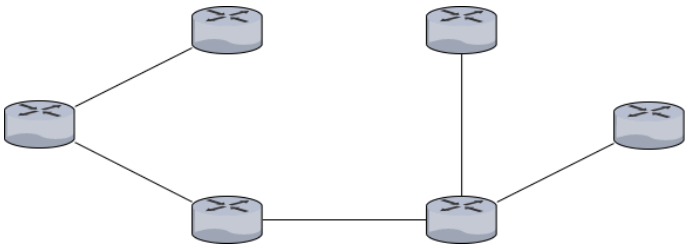
5.1.1 Dijkstra's Link-State Routing Algorithm

This is a **centralized** algorithm, which computes the least cost paths from one node to all other nodes in the network. This algorithm is iterative, and after the kth iteration of it the least-cost path are known up to k destination nodes.

For example, the following graph:



Would generate the following least-cost-path tree – from the left-most router:



5.1.2 Distance-Vector Algorithm

This algorithm is based on the Bellman-Ford equation. The equation is the following:

$$D_x(y) = \min_v \{c_{x,v} + D_v(y)\}$$

The cost of the least-cost path from x to y is calculated as the minimum – taken over all neighbors v of x – of the sum the direct cost of the link from x to v, plus the least-cost path from v to y.

This algorithm is **distributed**, as well as self-stopping, iterative and asynchronous.

Whenever a link cost changes, the node detects the change and recalculates the local DV. In the case that the DV changes, then the neighbors are notified.

5.2 Scalable Routing

Both LS and DV assume all routers are identical and that the network is flat. This is unrealistic in the real world, thus the solution is to group routers into **Autonomous Systems**.

An Autonomous System is a group of routers under the same administrative

control. An algorithm running inside of an AS is called an **intra-autonomous system routing protocol**.

There are two types of routing – regarding scalable routing:

- **Intra-AS**

  Routing within the same AS.

- **Inter-AS**

  Routing between ASs.

5.2.1 Interconnected ASs

Being ASs interconnected, the forwarding table of each router is configured accordingly. **Intra-AS routing** determines the entries for destinations within the AS, while **Intra- and Inter-AS** both determine the entries for external destinations.

5.3 Intra-AS Routing Protocols

5.3.1 RIP (Routing Information Protocol)

RIP is DV based, where DVs are exchanged between routers every 30 seconds. This protocol is no longer used.

5.3.2 EIGRP (Enhanced Interior Gateway Routing Protocol)

EIGRP is also DV based. This protocol was formerly CISCO-proprietary software.

5.3.3 OSPF (Open Shortest Path First)

OSPF is a link-state protocol that uses **flooding of link state information** and **Dijkstra's least-cost path algorithm**. Each router constructs an entire topological map of the entire AS. Each router locally runs the Dijkstra algorithm to determine the shortest-path tree to all subnets.

OSPF does not mandate a policy for how link weights are set. Whenever there is a change in a link's state, a router broadcasts this new link-state to all other routers.

Some of OSPF's advances are:

- **Security**

  The exchanges between OSPF routers can be authenticated.

- **Multiple Same-Cost Paths**

  When multiple paths to a destination have the same cost, OSPF allow multiple paths to be used.

- **Integrated Support For Unicast and Multicast Routing**

- **Support for Hierarchy within a Single AS**

  One OSPF area in the AS is configured to be the backbone area. The primary role of the backbone area is to route traffic between the other areas on the AS. The backbone area contains all area border routers in the AS and may also contain non-border routers as well.

5.4 Inter-Autonomous Routing Protocols

5.4.1 BGP

BGP (Border Gateway Protocol) is arguably the most important of all the Internet protocols, as it is the protocol that glues the thousands of ISPs in the Internet together. This protocol is decentralized and asynchronous, and uses DVs.

In BGP, packets are not routed to a specific destination address, but instead to CIDRized prefixes – each prefix represents a subnet or a collection of subnets.

BGP provides each router a means to:

- **Obtain Prefix Reachability Information from Neighboring ASs**

  This protocol allows each subnet to advertise its existence to the rest of the Internet.

- **Define the Best Routes to the Prefixes**

  A router may learn about two or more different routes to a specific prefix. In order to determine the best route, the router will locally run a BGP route-selection procedure.

Pairs of routers exchange routing information over semi-permanent **TCP** connections using port 179. Each such connections is called **BGP connection**. There are two types of BGP connections:

- **Internal BGP (iBGP)**

  These are BGP sessions between routers in the same AS.

- **External BGP (eBGP)**

  These are BGP sessions that span two ASs.

When a router advertises a prefix across a BGP connection, it includes with the prefix several **BGP attributes**. Two of the most important attributes are:

- **AS-PATH**

  It contains a list of ASs through which the advertisement has passed.

- **NEXT-HOP**

  This is the IP address of the router interface that begins the AS-PATH.

5.4.2 Hot Potato Routing

This is a routing algorithm used by BGP. The hot potato routing algorithm will choose the gateway that has the least intra-domain cost. In this case there is no need to worry about the inter-domain cost.

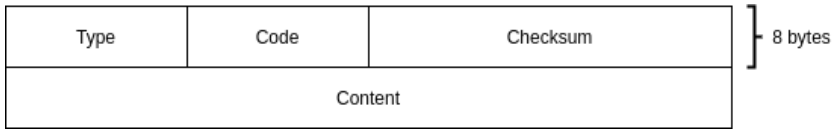
5.5 Routing Policy

ISPs, in a real-world scenario, only want to route traffic to/from its customer networks. To do so, **routing policies** are enforced. These policies can trump all other considerations when selecting a route to destination.

5.6 ICMP

ICMP (Internet Control Message Protocol) is a protocol used by hosts and routers to communicate network-level information to each other. ICMP messages are sent as the payload of an IP address (as UDP and TCP).

A visual representation of an ICMP message is the following:



Some fields are:

- **Code**

  The ICMP subtype

- **Content**

  Copy of the IP header and at least 8 bytes of IP data.

5.6.1 Ping

The ping program sends an ICMP type 8, code 0 message (echo request) to the specified host. The destination host, seeing the packet, sends back a type 0, code 0 message (echo reply) to the sender.

5.6.2 Traceroute

The traceroute program allows us to trace a route from a host, to any other host in the world. This program is implemented with the use of ICMP messages. These are the steps for a traceroute:

1. Source sends a series of ordinary IP datagrams to the destination. Each of these datagrams is carrying a UDP message with an unlikely port number. each of these datagrams have a TTL of 1, 2, 3 ... n.
2. The source starts timers for each of the datagrams.
3. When the nth datagram reaches the nth router, that router will observe that the packet TTL is expired. The receiving router thus sends back to the source an ICMP type 11, code 0 message (TTL expired). This message includes the name and IP address of the router.
4. Once the source obtains the ICMP message, it stops the timer on that datagram and returns the RTT, name and IP address of the nth router.
5. The source stops sending packets when the destination host sends back an ICMP type 3, code 3 message (destination port unreachable) back to the source.

6 The Link Layer and LAN

The basic service of the link layer is to move a datagram from one node to an adjacent node over a single communication link. Most of the link layer is implemented in hardware, while high-level functionalities may be implemented in software that runs on the host's CPU.

6.1 Definitions

A **communication network** is a system that allows two or more endpoints to be connected and to exchange data.

An **endpoint** is any kind of equipment that is able to connect to the network.

A **node** is any device that runs a link-layer protocol.

A **link** is a communication channel that connects adjacent nodes along the communication path.

6.2 Communication Media

Both **wired** and **wireless** communication links exist. **Wired** links are:

- **Fiber Optic Cable**
- **Coaxial Cable**
- **Twisted Pair Copper Cable**

Wireless links are:

- **Radio Waves**

6.3 Communication Links

Some communication link types are:

- **Point-to-Point**

  When there is a single sender at one end of the link and a single receiver at the end of the link.

- **Broadcast**

  When there are multiple sending and receiving nodes all connected to the same, single and shared broadcast channel.

- **Unidirectional**

  Also called simplex. It provides one-way communication.

- **Bidirectional**

  Can be either **full-duplex** (two one-way channels) or **half-duplex** (one two-way channel). For example Ethernet an Fiber Optics are full-duplex, while coaxial is half-duplex.

6.4 Link Access

The **MAC** (Medium Access Control) protocol specifies the rules by which a frame is transmitted onto the link.

This is easy for point-to-point links, but gets complex for broadcast links.

The types of multiple access protocol are:

- **Channel Partitioning Protocols**
- **Taking-Turns Protocols**
- **Random Access Protocols**

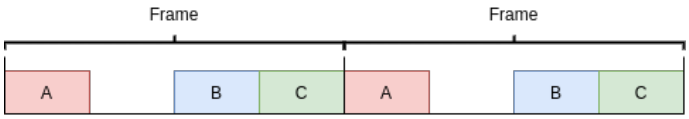
The Link Layer guarantees reliable delivery. That is, network-layer datagrams are guaranteed to be relayed across the link. This is typically achieved by means of retransmissions and acknowledgments.

6.5 Multiple Access Problem

In order to solve the issue of multiple nodes sharing the same single broadcast link, there is the need of a MAC protocol in order to coordinate the frame transmission of the nodes.

6.5.1 Channel Partitioning Protocols

In **TDM** (Time Division Multiplexing), time is divided into rounds, and each round is divided into N slots. Each node gets a fixed length slot in each round, and unused slots remain idle.



The advantages are that there are no collisions and perfect fairness.

The disadvantages are that the max rate is always $\frac{R}{N} bps$ – even when only one node has frames to send, and that there is a fixed latency before a node can send data.

In **FDM** (Frequency-Division Multiplexing), a channel is divided into N different frequency bands – each with a bandwidth of $\frac{R}{N}$. Each node is assigned to one of the N fixed frequency bands, and the unused transmission time in frequency bands go idle.

This method has the same advantages and disadvantages as TDM.

6.5.2 Taking-Turns Protocols

In **polling protocols**, the master node polls each node in a Round-Robin fashion.

The advantages are that there are no collisions and fairness.

The disadvantages are that the rate is less than R *bps* even if only one node is active, and the master node is a single point of failure.

In **token-passing protocols**, there is no master node. Instead, a small frame, called **token**, is exchanged among the nodes. Only the node that has the token is allowed to transmit.

The advantages are that this protocol is fully decentralized, and that it is efficient.

The disadvantage is that the token could be lost.

6.5.3 Random Access Protocol

The transmitting nodes always sent at the full R *bps* rate. But if there is a collision, a back-off mechanism is used in order to decide when to retransmit the packet. Some examples of random access protocols are:

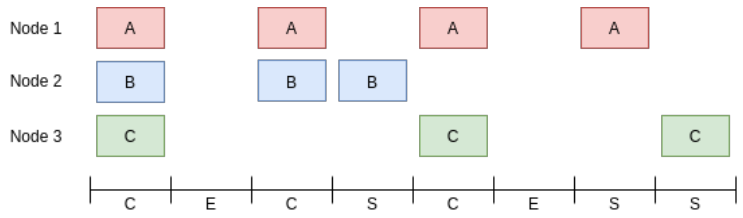
- **ALOHA**

  This was used by Hawaiians to connect all of their islands with a central computer located on the university campus of the main island.

  This setup was composed of one main host and many client machines, a downlink from the host to the clients with frequency f_1, and an uplink from the clients to the host with frequency f_2.

- **Slotted ALOHA** Time is divided into slots. The length of a slot is $\frac{L}{R}$ seconds – where L is the number of bits in the frame and R is the channel rate).

  All of the nodes are synchronized, meaning that each node knows when a slot begins. These nodes only transmit frames to the hub at the beginning of slots.



- **CSMA/CD**

  CSMA (Carrier Sensing Multiple Access) is when a node listens to the channel before transmitting. If a frame from another node is being transmitted, the node trying to send the frame waits and tries again later.

  CD (Collision Detection) is when a transmitting node listens to the channel while it is transmitting. If it detects that another node is transmitting an interfering frame, the node trying to send stops transmitting, it waits and tries again later.

  CSMA/CD are mechanisms used in the Ethernet protocol.

- **Ethernet**

6.6 Link-Layer Address

A link-layer address is the unique address of a network adapter. A **network adapter** is a piece of hardware that connects a computer to a network. Network adapters contain both the physical and link layers.

Each adapter has its own MAC address. This address is permanent and usually burned into its ROM (Read-Only Memory).

6.7 MAC Address

6.7.1 MAC Address Format

There are several different MAC address formats. One example is **EUI-48** (Extended Unique Identifiers). This type of MAC address is used in both Ethernet and Wi-Fi.

This MAC address is made up of 6 bytes – 2^48 addresses are possible. These 6 bytes are divided in two parts:

- **OUI (Organizational Unique Identifier)**

  24 bits are used for the OUI. This is assigned by IEEE and is globally unique.

- **NIC (Network Internet Controller)**

  24 bits are used for the NIC.

6.7.2 MAC Address Spoofing

This is a technique used in order to change the factory-assigned MAC address of a network interface on a network device.

MAC address randomization can be used to prevent the tracking of devices in Wi-Fi rich environments.

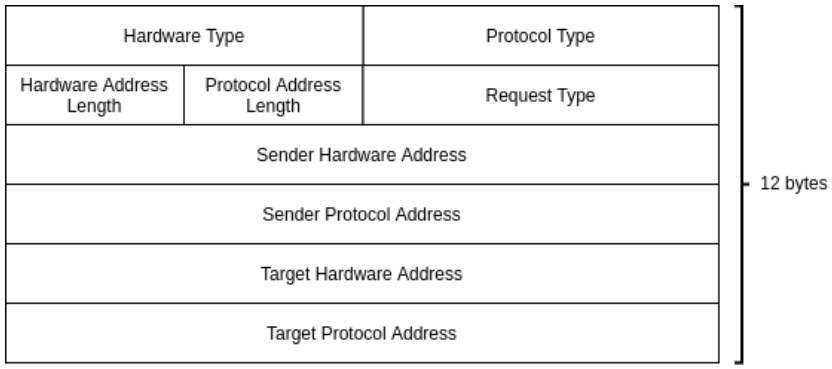
6.8 Frame Forwarding at the Link Layer

The link-level frames contain the MAC address of the intended destination of the frame. The broadcast address of MAC addresses is FF-FF-FF-FF-FF-FF.

6.9 Address Resolution Protocol

Since there are two types of both network- and link-layer addresses – IP and MAC addresses, there is the need to translate between them.

The ARP (Address Resolution Protocol) protocol allows a host to find the MAC address of a node given the node's IP address. This is done by building an ARP table with all IP - MAC mappings.



6.9.1 ARP Query

An ARP query is sent when the ARP does not find an entry in the table for a given IP address.

The ARP query packet is sent to the broadcast address of the subnet, thus being received by all hosts in the subnet.

6.9.2 ARP Response

If a host matches the **TPA** (Target Protocol Address), it replies to the query with an ARP response packet.

If the destination node is in another subnet, then the frame must be passed to the router that interconnects the two subnets.

6.9.3 ARP vs DNS

ARP is used to resolve addresses for hosts that are in the same subnet.

DNS is used to resolve host names for hosts that are anywhere in the Internet.

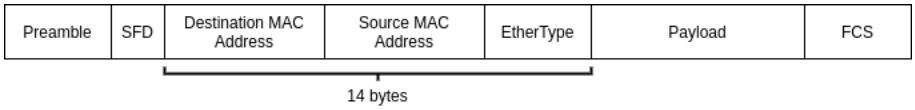
6.10 Ethernet

Ethernet is a family of computer networking technologies commonly used in LANs (Local Area Networks), MANs (Metropolitan Area Networks) and WANs (Wide Area Networks). The first Ethernet standard was the **10BASE-5** (10 Mbps rate, baseband signalling and max length of 500 m).

IEEE 802.3 is a collection of standards defining the physical layer an data-link layer's MAC of wired Ethernet.

6.10.1 Ethernet Frame

Here is a graphical representation of the Ethernet frame:



The following are its fields:

- **Preamble**

  7 bytes of alternating 0s and 1s.

- **SFD**

  The SFD (Start of Frame Delimiter) signals the end of the preamble and the start of the packet.

- **Destination a Source MAC Addresses**

- **EtherType**

  Identifies the high-level protocol to which this frame should be delivered.

- **Payload**
- **FCS**

  The FCS (Frame Check Sequence) is a checksum used to verify the integrity of the packet.

6.11 Bus Topology with 10BASE-5 Ethernet

All stations a connected to each other through a coaxial cable. The maximum length of the signal is of 500 meters. The stations connect to the cable by means of an AUI (Attachment User Interface).

Multiple segments are joined by repeaters. A maximum of 4 repeaters can be but between any pair of hosts, thus making the maximum length of the signal 2500 meters.

6.11.1 Collision Domain

Any signal placed on an Ethernet segment by a host is broadcasted over the entire segment.

All hosts that compete for access to the same link are said to be in the same collision domain.

6.11.2 Receiver Algorithm

An Ethernet 10BASE-5 adapter receives all frames sent by any host and accepts:

- Frames to its own address
- Frames to the broadcast address (all 1s)
- Frames to the multicast address (starts with 1), if the adapter was instructed to do so
- All frames, if it has been set in promiscuous mode

There are three techniques used for transmitting frames:

- **CSMA (Carrier Sensing Multiple Access)**

  The station that wants to send some data measures the level of current of the wire. If the level is over a certain threshold, this means that the carrier is busy.

- **CD (Collision Detection)**

  The sending station continuously measures the signal of the wire. If the signal is higher or lower than the transmitted signal, then there is a collision.

- **Binary Exponential Back-off**

  It measures the time to wait for retransmission after a collision has been detected.

6.12 Link-Layer Switches

6.12.1 Hub vs Switch

A **hub** has the following characteristics:

- Multiple inputs
- Multiple outputs
- **Repeats an incoming signal to all of its output ports**
- **Has no buffer**

A **switch** has the following characteristics:

- Multiple inputs
- Multiple outputs
- **Forwards an incoming signal to a specific output port**
- **Has a buffer**

While **hub-based topologies** need collision-preventing algorithms (such as CSMA and CD), **switch-based topologies** have no collisions.

6.12.2 Switch Ports

Each device connected to a switch port can transfer data to any of the other ports at any time.

Switches can also store incoming data and forward it to the target interface.

6.12.3 Operations of a Switch

Two of the main operations of a switch are:

- **Filtering**

  Determine if an incoming frame needs to be dropped or not.

- **Forwarding**

  Determine the interface to which incoming frames should be directed to, and move those frames.

All of this is done by the **switch table**. There are also some rules that apply to switch tables, namely:

- If there is no entry in the table for a particular frame, it will be forwarded to all interfaces.
- If there is an entry in the table associated to the same port the frame is coming from, the frame is dropped.
- If there is an entry in the table associated to a port, then the frame is forwarded to that interface

6.12.4 Characteristics of a Switch

Switches have many interesting characteristics, such as:

- **Self Learning**

  Each incoming frame is stored in the switch table (which is initially empty). Entries that are in the table for more than the aging time are deleted.
- **Plug & Play**
- **Full-Duplex** Any switch interface can send and receive at the same time.
- **No Collisions**
- **Heterogeneous Links** Each link can run at different speeds and over different media.
- **Easy Network Management**

6.12.5 Switch vs Router

A **switch** is a two-layered packet switch, where the upper layer is the link layer.

A **router** is a three-layered packet switch, where the upper layer is the network layer.

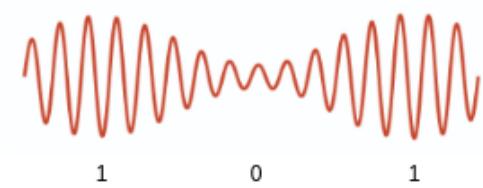
6.13 Transmission of Bits

Electromagnetic waves transport energy through empty space, stored in the propagating electric and magnetic field. The magnetic field variation is perpendicular to the electric field.

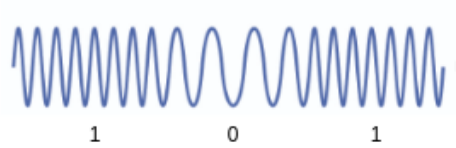
A single frequency electromagnetic wave exhibits a sinusoidal variation of electric and magnetic fields on space.

Transmission of bits may happen in two different ways:

- **Amplitude Modulation**



- **Frequency Modulation**



Amplitude and frequency may change due to, for example, reflection and refractions during propagation.

6.14 Bit Errors

The **BER** (Bit Error Ratio) can be computed by the following formula:

$$BER = \frac{N_{err}}{N_{tot}}$$

The typical value for the BER in Ethernet links is 10^{-7} to 10^{-13}, while for wireless, the BER is 10^{-3} to 10^{-6}.

6.14.1 Bit Errors Detection and Correction

There are two types of error correction:

- **FEC (Forward Error Correction)**

  Correction is done from the sender to the receiver. Additional error correcting bits are added to the sender.

- **ARQ (Automatic Repeat reQuest)**

  Correction is done from the receiver to the sender. Acknowledgments are sent by the receiver to signal a successful/unsuccessful reception.

6.14.2 FEC Techniques

The following are the FEC techniques used:

- **Parity check**

  The sender adds a 1 bit so that the number of bits equal to 1 in the sequence is even.

  The receiver checks the number of 1s in the sequence + EDC (Error Detection and Correction) bits. If the number is odd, then there is an error.

  Most of the times, if there is one bit flip, there can also be more than one. This is why there exist more advanced parity checks – such as two-dimensional parity.

- **Checksums**

  The field frames are treated as integers used as input to a checksum function.

  A checksum is needed at the UDP level, because there is no guarantee there is error checking at lower levels. It could also be that bit errors are introduced when a UDP segment is stored in the router's memory.

- **CRC (Cyclic Redundancy Check)**

  A CRC field R of r bits is added to a piece of data D.

  $$R = \text{remainder of } \frac{D \cdot 2^r}{G}$$

  G is called the **generator**, which is composed of $r + 1$ bits, and is known to both sender and receiver.

  The sender chooses r bits such that $d + r$ is exactly divisible by G. If the remainder of the division is not 0, then there is an error.

7 Wireless Networks

7.0.1 Wireless Propagation

Propagation is error prone because of three main reasons:

- **Interfaces**

  This occurs when two or more waves overlap. This could be caused by other radio devices or other electromagnetic noise. Waves can either be constructive or destructive.

The power of Wi-Fi devices is limited to 100 mW in Europe.

- **Path Loss**

  There is a difference between the transmitted power (P_{tx}) and received power (P_{rx}):

$$P_{rx}(d) \approx \beta \cdot P_{tx} \cdot \frac{1}{d^{\alpha}}$$

- **Multiplicative Propagation**

  Portions of the electromagnetic wave may reflect off objects and the ground, taking paths of different lengths between sender and receiver.

7.1 Wi-Fi

The Wi-Fi alliance is a "global non-profit association with the goal of driving the best user experience with a new wireless networking technology".

7.1.1 Wi-Fi Access Point

A Wi-Fi access point is a Wi-Fi certified device that provide wireless connectivity. These access points act as a gateway to other networks – which can either be wired or wireless.

Most access points are static, but they could also be mobile. Wi-Fi devices are all those that are based on the **IEEE 802.11** standards.

7.2 IEEE 802.11

IEEE 802.11 is a MAC and physical layer specification. It is designed for use in a limited geographical area.

This protocol extends the common wired Ethernet local network into the wireless domain.

7.2.1 Channels

Depending on the specific standard, IEEE 802.11 transmits in the 900 MHz, 2.4, 3.6, 5 and 60 GHz frequency bands. It operates in the **ISM** (Industrial, Scientific and Medical) radio bands.

7.2.2 Logical Architecture

The main components of a wireless network are:

- **STA (STAtion)**

  It is a device that has the capabilities of using the 802.11 protocol.

- **AP (wireless Access Point)**

  A networking hardware device that allows STAs to connect to the wired Internet.

- **BSS (Basic Service Set)**

  Term used to describe the collection of STAs which may communicate together within an 802.11 network. An AP may or may not be part of the BSS.

- **DS (Distribution System)**

  It interconnects BSSs.

- **IBSS (Independent Basic Service Set)**

  It is comprised of one or more STAs which communicate directly with each other.

- **ESS (Extended Service Set)**

  Is one or more interconnected BSSs

7.2.3 Operating Modes

There are two possible modes of operations:

- **Infrastructure Mode** Here STAs are connected wirelessly to an AP, forming a BSS.

- **Ad Hoc Mode**

  Here STAs are directly connected to each other, forming a IBSS.

7.2.4 SSID (Service Set IDentifier)

A SSID, identifies the wireless network. The SSID is a name configured on the wireless AP or an initial wireless client that identifies the wireless network. The SSID is periodically advertised by the wireless AP or the initial wireless client using a special 802.11 MAC management frame – known as the **beacon frame**.

7.3 RSSI (Received Signal Strength Indicator)

The RSSI measures the strength of the signal that arrives at the receiver. It can be computed with the following formula:

$$RSSI = 10\ log_{10}\left(\frac{P_{rx}}{1\ mW}\right)$$

The value of the RSSI may vary depending on several factors, such as:

- **Transmission power**
- **Multipath effects**
- **Presence of obstacles**
- **Antennas not being isotropic**

  Different values are measured in different directions)

7.4 IEEE 802.11 MAC

7.4.1 Acknowledgments

When the destination host receives a frame, it checks the CRC of that frame.

If the CRC check **passes**, then the destination host waits a short time – called SIFS (Short Inter-Frame Spacing), and finally sends a link-layer acknowledgment to the source.

If the CRC check **does not pass**, then the destination discards the frame and does not send an acknowledgment. The source then re-sends the frame after a timeout.

7.4.2 CSMA/CA

Before sending a packet, the transmitter checks if the wireless channel is free or busy.

If the channel is **free**, then the transmitter waits for a short time – called DIFS (Distributed Inter-Frame Space), and finally sends the packet.

If the channel is **busy**, then the transmitter uses binary exponential back-off to calculate when to resend the packet.

There are two collision issues with wireless networks:

- **Hidden Terminal Problem**

  This happens when two hosts cannot directly hear each other, but their signal might still be able to collide.

- **Exposed Terminal Problem**

  This happens when a host can overhear another host, but the overhearing host should not hamper the host to communicate.

7.4.3 RTS/CTS Handshake

An **RTS** (Request To Send) message is sent by a STA to an AP. It includes a field that indicates the length of the packet to transmit.

To the RTS message, the AP sends back a **CTS** (Clear To Send) message. This message lets the STA know it can initialize the transfer, and any other STAs now know one STA is transmitting and for how long.

All STAs must wait for an **ACK** message sent by the AP in order to try/re-try to transmit.

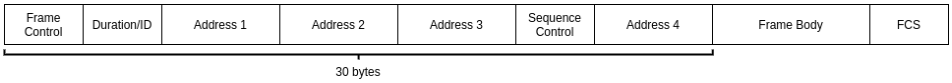
7.4.4 RTS Collisions

It might happen that an RTS packet sent from two or more STAs collide.

An RTS may also hamper the reception of a CTS packet. A solution is to make the CTS packets longer than the RTS packets – in this way, collisions are detected.

7.4.5 MAC Frame Format

The IEEE 802.11 MAC frame can be graphically represented as such:



Some fields are:

- **Duration/ID**

  Used to indicate the remaining duration needed to receive the next frame transmission.

- **Address Fields**

  Depending upon the frame type, the four address fields will contain a combination of the following address types: **BSSID** (BSS IDentifier), **DA** (Destination Address), **SA** (Source Address), **RA** (Receiver Address) and **TA** (Transmitter Address)

- **Sequence Control**

  Indicates the sequence and fragment numbers of each frame.

- **Frame Body**

  It contains information from the higher levels.

- **FCS**

  It allows for integrity check of frames.

7.5 Beaconing

Beacons are broadcast messages that contain information about a Wi-Fi access point, such as:

- **MAC Address**
- **SSID**
- **Operation Mode**
- **Active Channel**
- ...

An AP sends beacons at regular time intervals. Each AP transmits on a specific channel. The host maintains a list of all available access points and selects one to associate with.

8 APPENDIX A - Formulae Glossary

- α

  Path loss exponent. It depends on the medium in which the signal propagates.

- β

  Multiplicative attenuation factor.

- **a**

  Average rate of packets per second.

- **C**

  Link capacity.

- $\mathbf{D}_x\mathbf{(y)}$

  The cost of the least-cost path from x to y

- **d**

  Distance between the nodes. **Unit:** meters

- **F**

  File size in bits. **Unit:** bits

- **G**

  Bit pattern of $r + 1$ bits.

- **L**

  Length of a packet in bits. **Unit:** bits

- **N**

  Number of hops/peers/ports.

- $\mathbf{N}_{err}$

  Number of bit errors.

- $\mathbf{N}_{tot}$

  Total number of bits sent.

- **R**

  Transmission rate. **Unit:** bits/second

- **s**

  Propagation speed of the link. **Unit:** meters/second

- **T**

  Transfer rate. **Unit:** bits/second

- **u**

  Peer's upload rate. **Unit:** bits/second

- $\mathbf{u}_s$

  Server's upload rate. **Unit:** bits/second

9 APPENDIX B - Standard Ports

- **22** - SSH
- **23** - Telnet
- **25** - SMTP
- **53** - DNS
- **80, 8080, 8008** - HTTP
- **110** - POP3
- **143** - IMAP
- **179** - BGP
- **465** - SMTP (secure)
- **853** - DNS (secure)
- **992** - Telnet (secure)
- **993** - IMAP (secure)
- **995** - POP3 (secure)

10 APPENDIX C - Exercises

10.1 Chapter 1

- **Consider a circuit-switched network with four routers, positioned at the corner of an imaginary square. Label the four switches A, B, C, and D, starting from the top left corner and going in the clockwise direction. Assume there are five connections between any two adjacent routers. What is the maximum number of simultaneous connections that can be in progress at any one time in this network?**

  Number of nodes times the number of connections for each node → 5 * 4 = 20

- **Suppose Host A wants to send a large file to Host B. The path from Host A to Host B has three links, of rates R1 = 500 kbps, R2 = 2 Mbps; R3 = 1 Mbps. Assuming no other traffic in the network, what is the throughput R_t for the file transfer?**

  The lowest rate → 500 kbps

- **Calculate how long it would take to transfer 40 GB of data from A to B on a 1 Mbps link.**

  Convert from GB to mbit and divide by link rate. Then convert seconds to hours. → $\frac{320000}{1} = 320000\ s \rightarrow 88.8\ h$

- **Assume a packet of length 1000 bytes must be transmitted between router A and router B. The link between A and B is 2.5 km long, enables a propagation speed of 2.5 10^8 m/s, and a transmission rate of 2 Mbps. Assume the first bit of the packet leaves router A at time t_0 = 0 s. After which delay t_1 does the first bit of the packet reach router B?**

  Convert km in m, and divide it by the propagation speed (propagation delay) → $\frac{2500}{2.5 \cdot 10^8} = 0.00001\ s \rightarrow 10\ \mu s$

- **Suppose Host A wants to send a large file to Host B. The path from Host A to Host B has three links, of rates R1 = 500 kbps, R2 = 100 kbps, R3 = 1 Mbps. Suppose the file is 60 MB. Assuming all other delays are negligible with respect to the transmission delay, how long will it take to transfer the file to Host B? Please give your answer in minutes.**

  The throughput is the slowest link, thus is 100 kbps. Convert MB to kilobit and divide by throughput (transmission delay) → $\frac{480000}{100} = 4800\ s = 80\ min$

- **Assume a packet of length 1000 bytes must be transmitted between router A and router B. The link between A and B is 2.5**

km. long, enables a propagation speed of 2.5 10^8 m/s, and a transmission rate of 2 Mbps. Assume the first bit of the packet leaves router A at time $t_0 = 0$ s. After which delay $x = t_0 + t_2$ does the last bit of the packet leave router A?

Convert bytes in Mbits, then divide by the transmission rate (transmission delay) $\rightarrow \frac{0.008}{2} = 0.004\ s = 4\ \mu s$

10.2 Chapter 2

- **Consider a link over which hosts can transmit at a rate of 2048 kbps in both directions. Suppose that packets containing data are 128 kbits long, and packets containing only control (e.g., ACK or handshaking) are 256 bits long. Now suppose that a client issues an HTTP/1.1 request to a server to retrieve an object. Suppose that the downloaded object fills a single data packet. Assuming that propagation, queuing and processing delays are negligible, how much time would it take a client to obtain the object from the server?**

  Convert bits in kbits, multiply by 3 (which is the number of packets containing control) and add it to the length of the data $\rightarrow 3 * \left(\frac{0.256}{2048}\right) + \frac{128}{2048} = (3 * 0.000125) + 0.0625 = 0.000375 + 0.0625 = 0.062875 = 62.875ms$

- **Consider a link over which a sender can transmit at a rate of 2 Mibps in both directions. Assuming an object of size 256 kibit must be transmitted, how much is the transmission delay?**

  Convert kibit to Mibit and divide it by the transmission rate (transmission delay) $\rightarrow \frac{0.25}{2} = 0.125\ s = 125\ ms$

10.3 Chapter 3

- **Suppose Host A sends a TCP segments to Host B over a TCP connection. The segment has sequence number X and carries Y bytes of payload. In the acknowledgment that Host B sends to Host A, what will be the acknowledgment number?**

  Y bytes of the payload are added to the X of the sequence number $\rightarrow Y + X$

- **Suppose Host A sends a TCP segments to Host B over a TCP connection. The segment has sequence number X and carries Y bytes of payload. In the acknowledgment that Host B sends to Host A, what will be the sequence number?**

  It cannot be derived by the data

- **Assume that the current estimated round-trip-time (RTT) in a TCP connection is 250 ms. Assuming that the exponential**

weighted moving average with alpha = 0.1 is used to compute the estimate of RTT, what will be the new value of such estimate, after a new measurement of RTT equal to 200 ms is recorded?

Use the RTT estimation formula $\rightarrow (1 - 0.1) \cdot 250 + 0.1 \cdot 200 = 0.9 \cdot 250 + 0.1 \cdot 200 = 225 + 20 = 245\ ms$

- **Consider 8 TCP connections, each with a different end-to-end path, but all passing through a bottleneck link with transmission rate R = 64 MBps. If the congestion control mechanism used on these connections is fair, what will be the average transmission rate over each connection?**

  Since the TCP connection is fair, then we simply divide the transmission rate by the number of connections $\rightarrow \frac{64}{8} = 8\ MBps$

- **Assume that the slow start threshold of a TCP connection is 32 MSS when a loss event is first detected. Immediately after another loss event is detected at the sender, what will be the value of the threshold?**

  For every loss detected, the MSS is divided by two. Since we have two losses detected, we divide two times by two $\rightarrow \frac{32}{2} = 16\ MSS$ and then $\frac{16}{2} = 8\ MSS$

11 APPENDIX C - Useful Links

- **Interactive Exercises**: *interactive end-of-chapter exercises*
- **Unit converter**: *google unit converter*