Chapter 1: Computer Networks and the Internet

Section 1

1.1.0

> **Important**
>
> When talking about what networking is, this book will teach it using the internet as a reference to it.

When it comes to describing what a network is, there are two ways to do this:

1. *Hardware and Software*: The hardware aspect of this will be things like routers, switches, cabling, network interface cards, etc. When it comes to the software aspect of it this will be things like the operating system, network management software, security software, and protocols, etc.
2. *Network Infrastructure*: This will be things like firewalls, the path of cabling, servers, internet service provider towers, DNS, etc.

1.1.1

Each device connected to a network is called a end system (or host). These words are synonymous of each other.

The way that two different end systems communicate with each other is through communication links and packet switches. When it comes to the communication links, there are different types that change how the devices communicate with each other and they are: *coaxial cable*, *copper wire*, *optical cable*, and *radio spectrum*.

When it comes to the different types of communication links, each one will transfer data at different rates to the other end system. The speed at which the data is sent is called transmission rate. There is a way to calculate this with the equation being:

$$\text{Transmission Rate} = \frac{\text{Total data to send (bits)}}{\text{Maximum data transfer per second (bits/s)}}$$

When the data is actually now being sent to the receiving end system, this will not all be sent in one giant chunk. Instead, the total data to send will be sent in multiple smaller sections. Each of the small sections will contain extra non-related data called *header data*. The *header data* combined with the actual data is called a packet.

> **Note**
>
> For example, a total of 50 bits needs to be sent from system A to system B. The data is sent the communication link will only be able to send 15 bits per second. However, since data is split up into multiple parts, each packet will be 11 bits of actual data and 4 bits of *header information* so each packet is 15 bits. This means four packets (15, 15, 15, and 9) bits each one. It will take ~ 3.6 seconds to send all 4 packets to the desired end system. Once they reach end system B they will all be reassembled to make the original 50 bit thing of data.

A packet switch and is what takes a packet and sends it to its next destination in the transfer of that packet to the receiving end system. This works by getting received by its *incoming communication link* and being sent out the *outgoing communication link*. The two most common types of packet switches are: *routers* and *link layer switches*. When it comes to the placement of the two types, the *router* is more used in the network core while the *link layer switch* is more used in access networks.

All the packet switches that a packet had to take to get the the desired end system is called a route (or path).

An ISP (internet service provider) is a company that is providing internet related services to other end systems and deal with the routing of how to send and receive data from one end system to the other. An ISP can be from telephone companies, Universities, corporates, etc.

When it comes to how the ISP will communicate with each other, these will communicate with each other from upper tier and national ISP providers.

A network protocol defines the complete set of mandatory conventions and operating procedures that determine reliable communication between independent end systems. These standardized rules dictate every aspect of data exchange to ensure mutual understanding and successful interaction:

- Syntax and Semantics: Specifies the necessary format and structure for data to be correctly packaged and interpreted, alongside the type and scope of permissible content within the transmission.
- Connection Management: Outlines the precise mechanisms for establishing, maintaining, and terminating a connection between two end systems.
- Transmission Control: Establishes the parameters for acceptable transmission speed, data sequencing, and flow control to manage network capacity and prevent congestion.
- Data Integrity: Protocols embed procedures for error detection and recovery. These mechanisms are essential for validating data integrity

and ensuring that any corruption or loss during transit is identified and addressed.

> ⓘ Note
>
> A good example of a protocol is the *TCP/IP* protocol. This is the common one used for transferring data over the internet.

When it comes to who makes the protocols standards, this is done by the *IETF* (*Internet Engineering Task Force*). The actual documents they make to describe how the protocol would work is called *RFC* (*Request For Commits*).

1.1.2

A distributed system is comprised of multiple independent computer end systems that are physically separate but are interconnected through a network. These systems function cohesively to accomplish a unified, common objective like working a multiplayer game.

The primary benefit of this architecture is workload distribution (or load balancing), which allows the overall processing demands to be shared across various end systems. This parallel execution effectively mitigates the burden on any end system, thereby enhancing performance, scalability, and efficiency. Another thing this provides is fault tolerance, meaning if one component fails, the other independent systems can often continue processing, ensuring greater system reliability and availability.

> ⓘ Note
>
> An example of a distributed system is purchasing stuff from target. The purchase service might go down, but can still look at the items target has and add stuff to the cart. Even though one part of the overall system failed, the whole thing did not.

A socket is a way that two different end systems are able to know which specific *application service* running on the end system to send data to. More about this will be talked about later.

> Important
>
> An *application service* is just a process that is running on an end system. An example of this is outlook. The outlook is a running process that has a specific socket that is tells other end systems that wanna send mail to it so send it at that specific socket.

Section 2

1.2.0

When it comes to talking about end systems (or host), these can be divided into two smaller categories: client and server. A client system are things like phones, tablets, TVs, computers, etc. These devices will be the ones that request data from another device. On the other hand, a server does not request data and instead is only in charge of sending data to requesting end systems. An example of a server is a *web server*.

A server will typically be more powerful compared to a normal client machine as well.

Today, most things like servers are not a one of and there is a large amount of them together that live in a place called a data center that is a massive building to support the needs to contain that many machines at once.

1.2.1

The *access network* (also known as the *edge network*) defines the segment of the network topology that directly connects an end system to the larger core internet infrastructure. Conceptually, this path is divided into two distinct components:

1. *Originating Segment:* This is the segment where the source end system transmits its data to the first router that provides connectivity to the network core (Internet Service Provider network). This is often the consumer's or organization's *edge router*.
2. *Terminating Segment:* This is the symmetrical segment where the information is finally delivered from the last router within the core network path to the destination end system.

The three most widespread technologies currently employed for *residential broadband access* are Digital Subscriber Line (DSL), cable internet, and Fiber to the Home (FTTH).

These technologies differ primarily based on the *physical transmission medium* utilized to connect the residence to the Internet Service Provider (ISP):

- Digital Subscriber Line (DSL): Leverages existing traditional *copper* telephone lines to transmit high-speed data.
- Cable Internet: Utilizes the *coaxial cables* and infrastructure originally designed for community antenna television (CATV) systems.
- Fiber to the Home (FTTH): Employs *optical fiber cables* run directly to the customer's premise, which offers the highest capacity and fastest speeds available commercially today.

1.2.2

When it comes to physical media, this is the actual physical thing that send the data between the devices. There are differnt types that change how mant bits and how fast can be sent per second. Some populat examples are twisted-pair copper wire, coaxial cable, multimode fiber-optic cable, terrestrial radio spectrum, and satellite radio spectrum.

When it comes to the many devices that are connected to each other, they all do not need to use the same physical media type. For example, if devices A to B used twisted-pair cable and devices B - C used fiber-optic that is ok.

There is another way to even more divide the types of physical media:

1. *Guided Media* are the physical components that can be touched and are physically connected from one device to the next
2. *Unguided Media* are the devices that do not need to be connected together to work, but can send data to each other still. An example of this is a cellphone tower.

Section 1.3

1.3.1

Remember that when two end system devices want to communicate with each other, they need connect with each other. The sending of data can be to establish a connection with the other end system or to send the actual data to the end system.

When the data is send, it is not all sent at once. Instead it is broken down into smaller sub-chunks with added *header data* to form a packet.

When it comes to the actual sending of the data (packets), this is done with packet switches. This is just a collection of *routers* and *link layer switch* that send the data across each the last one is able to send its data to the destination end system.

While the packet switch is just the thing that determines where the sent data (packet) should go next (another packet switch or destination end system), the communication link is what actually sends the data from one device to the next. Remember, there can be multiple differnt communication link types. Each of the types determine the speed at which data is sent across the network.

When it comes to the rules on how the data should travel, there are a few differnt methods to this:

1. *store and forward transmission*: This is the technique that packet switches use to accept data on their *incoming communication links*. This means that ALL the data from the current packet on the incoming

link must be processed before ANY other can be processed next. Once that is done THEN is can start to send the data on its *outbound link*.

> ⓘ Note
>
> The most common technique that packet switches use to get data is *store and forward transmission*.

> An example of store and forward
>
> There are three devices on the network. A (souce end system), B (packet switch), and C (destination end system). There is nine megabytes of data to send. The data needs to be sent in smaller chunks so the data is parsed into three megabyte chunks with a one megabyte chunk of header data added to that to form a packet. Now device A will send those three packets over to device C. To do that it needs the help of device B. Device B uses the store and forward method, so when A sends the first packet over to B, it has to fully process on data on the incoming communication link and once that is done then it can start to send the data on the outgoing communication link to device C. At the same time it is sending the data, since it was all fully processed, now it can accept another packet on the inbound communication link while the other packet is being sent on its outing communicaion link.

When a packet is being sent across a network, there could be multiple packet switches it has to go across, the communicaion links could be slower at some places compared to others, it could be a queue of packets to process, etc. All these factors being considered and calculated for a single packet is called end-to-end delay. This means the total time it took for a single data packet to get from the source end system to the destination end system.

In reality, each packet switch has multiple incoming and outgoing communicaion links. Because of this, it makes it easier to of course send and recieve data at once. However, each outbound link does not go to the same place and multiple packets could need to go out to the same outbound link at once. Because of that, it results in a queue called the output buffer (output queue). Each output link has its own buffer space to store packets until the current working packet is fully sent out of the device and then the next one can start. The time that a single packet spends waiting in the queue is called queing delay.

> Important
>
> If the output buffer for a communication link is full, then there will not be a place to store the next packet and there could be a risk of that non-stored packet of being lost; this is called packet loss. Meaning the destination end system will never get it, so it results in corrupted data because the destination end system will not have all the original data needed to correctly process the information.

When it comes to how packet switches determine what outgoing communication link to send the packet on, this is different depending on the type of network, but will be describing how this works in terms of the internet.

All end system devices have something called an IP address. This IP address is how a device location is identified on a network. A packet switch identifies which outgoing communication link to send the packet on using the IP address. The way a packet switch even gets the IP address of the destination end system is from the source end system adding that data in its *header data* section.

A packet switch has something called a routing table that it uses as a reference when determining what outgoing communication link to send the packet on. The packet switch will look at a portion of the given destination IP address in the reciving packets header data along with the routing table and then use that to determine what outbound link to choose.

When it comes to how a routing table is set, this is done with special routing protocols. This will be talked about more later on.

> Important
>
> Make sure to remember the difference between packet switching and packet switches.

1.3.2

Another way to send data across a network is called circuit switching. Unlike packet switching, a circuit switch network will reserve a set of outbound and inbound communication links across all packet switches that connect from the source to destination end system until the source end system no longer wants to send data; unlike packet switching where a packet temporarily resurves the current communicaion link it is using and nothing else so everyone can use the resources.

The way a circuit switch network works is using multiplexing. Multiplexing is a technique that allows multiple distinct signals or data streams to share a single common communication channel (like a wire, cable, or frequency spectrum) and the types are:

1. Frequency-division multiplexing (FDM): Analogous to having separate

lanes on a highway for different cars. It divides the total bandwidth of the communication medium into multiple, non-overlapping frequency bands, and each signal is allocated its own unique frequency band for the entire duration of the transmission.

2. Time-division multiplexing (TDM): TDM is like a time-share system where all signals share the same, full channel, but only one signal is permitted to transmit at any given instant. It divides the available *time* on the channel into extremely short, recurring time slots, and each signal is allocated an exclusive slot in a fixed, repeating sequence.

> Tip
>
> There are some pros and cons to using a circuit switch and packet switch infrastructure

> Tip
>
> Bandwidth just means the data transfer rate

```php
<?php
  $x = 59;
  $y = 59;

  if ($x === "59")
    echo "Was a string number";

  if ($x === 59)
    echo "This is correct";
```

Keywords

Keyword	Description

Keyword	Description

Keyword	Description

Keyword	Description