Networks

Muchang Bahng

Spring 2025

Contents

1 The Internet

The **Internet** is a global network of computing devices communicating with each other in some way, whether they're sending emails, downloading files, or sharing websites. The Internet is an **open network**, which means that any computing device can join as long as they follow the **protocols** (rules that define how each device must communicate with each other). The internet is powered by many layers of protocols, and to create a global network of computing devices, we need:

1. **Wires & Wireless**: Physical connections between devices, plus protocols for converting electromagnetic signals into binary data.
2. **IP**: A protocol that uniquely identifies devices using IP addresses and provides a routing strategy to send data to a destination IP address.
3. **TCP/UDP**: Protocols that can transport packets of data from one device to another and check for errors along the way.
4. **TLS**: A secure protocol for sending encrypted data so that attackers can't view private information.
5. **HTTP & DNS**: The protocols powering the World Wide Web

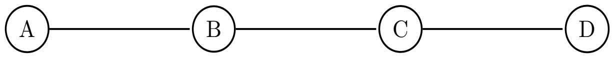
1.1 Computer Networks and Types of Networks

Definition 1.1

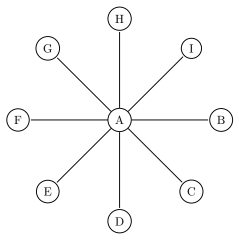
A **computer network** is a group of computers (i.e. computing devices) that use a set of common *communication protocols* over digital interconnections for the purpose of sharing resources located on or provided by the *network nodes.*
A **communication protocol** is a system of rules that allow multiple entities of a communications to transmit information via any kind variation of a physical quantity. The protocol defines the rules, syntax, semantics and synchronization of communication and possible error recovery methods.

A computer network can be visualized as a connected graph of nodes (which may include personal computers, servers, networking hardware, or other specialised or general-purpose hosts). The **network topology** is the layout, pattern, or organizational hierarchy of the interconnection of network hosts, in contrast to their physical or geographic location. Common layouts are:

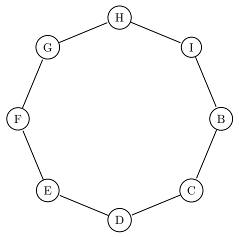
1. **Line Network**: All nodes are connected in a line.



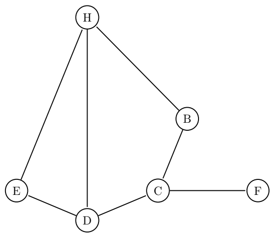
2. **Bus Network**: All nodes are connected to a common medium along this medium.
3. **Star Network**: all nodes are connected to a special central node.



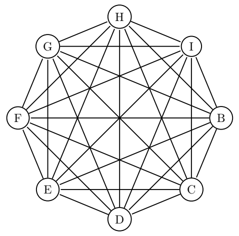
4. **Ring Network**: Each node is connected to its left and right neighbour node, such that all nodes are connected and that each node can reach each other node by traversing nodes left- or rightwards.



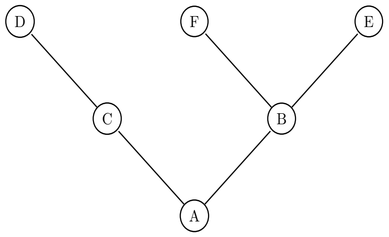
5. **Mesh Network**: each node is connected to an arbitrary number of neighbours in such a way that there is at least one traversal from any node to any other.



6. **Fully Connected Network**: each node is connected to every other node in the network.



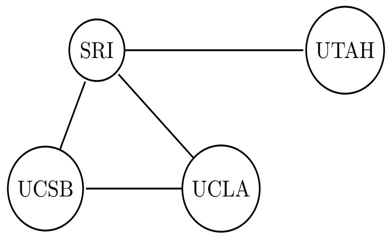
7. **Tree Network**: nodes are arranged hierarchically.



Notice how many of these networks have **redundancy**: having multiple ways to get from one node to another. That is, when a network path is no longer available, data is still able to reach its destination through another path. Usually, we would like to avoid a **single point of failure** and construct a **fault-tolerant** system that can experience failure in its components but still continue operating properly. However, building more connections may be expensive.

Example 1.1

The ARTPANET was the precursor to the Internet, the network where Internet technology was first tested out. It was started in 1969 with four computers connected to each other.



For example, even if the path between SRI and UCSB is gone, the connections between SRI and UCSB is not lost (since IP packets can travel through UCLA's router).

Because there are multiple paths that a piece of data takes to get from point X to point Y, *routing strategies* are implemented in order to determine the most optimal path.

Definition 1.2

Networks can be categorized as such:

1. A **local area network (LAN)** is a computer network that interconnects computers within a limited area. *Ethernet* and *Wi-Fi* are the two most common technologies in use for local area networks. A **wireless local area network (WLAN)** use radio frequencies to transmit and receive data.
2. A **metropolitan area network (MAN)** is a computer network that interconnects users with computer resources in a geographic region of the size of a metropolitan area.
3. In contrast, a **wide area network (WAN)** not only covers a larger geographic distance, but also generally involves *leased telecommunication circuits* or *data lines* (i.e. a private line between multiple locations provided according to a commercial contract), since no single company owns all the infrastructure across the wide geographic area. It is often composed of many LANs.
4. Another type of network is the **Data Center Network (DCN)**, a network used in data centers where data must be exchanged with very little delay.

1.2 Communication with Line Coding

Computers can connect through **physical** (e.g. cables) or **wireless** connections.

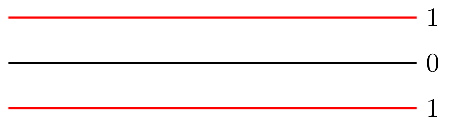
1. The **CAT5 cable** is a *twisted pair (copper) cable* that's designed for use in computer networks. It consists of four twisted pairs of copper wires. These twisted pair cables send data through a network by transmitting pulses of electricity that represent binary data. The information transmission follow the **Ethernet** standards, which is why twisted pair cables are commonly known as Ethernet cables. Use for both LANs and WANs. They can carry up to 1 Gbps across hundreds of feet, but are susceptible to interference.

2. **Fiber-optic cables** carry light instead of electricity in a fiber coated with plastic layers. The pulses of light represent binary data and also follow the Ethernet standards. They are also capable of transmitting much more data per second that copper cables, and they have the advantage of low transmission loss and immunity to electrical interference. Often used to connect networks across oceans so that data can travel quickly around the world. They can carry up to 26 Tbps acorss 50 miles (but are expensive)

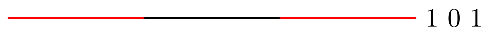
3. A wireless card inside a computer turns binary data into **radio waves** and transmits them through the air. However, they do not travel very far (100 ft in office buildings or up to 1000 ft in an open field). The waves are picked up by a *wireless access point* which converts them from radio waves back into binary data. These access points would be connected to the rest of the network using physical wiring. They can carry up to 1.3 Gbps.

4. **Infrared signals** and **microwaves** are sometimes used.

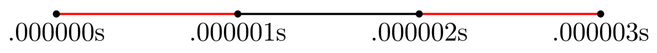
In order for the computers to send data into binary, they must convert this data into binary and send them as streams of 1s and 0s in a process called **line coding**. Furthermore, computers can raise efficiency of each wire by sending changing electric currents through a single wire. For example, rather than using three wires to encode `101` as



they send it through a single wire with intervals of $\frac{1}{3}$ seconds



or even better, at a rate of 1 megabit per second (interval of 0.000001 seconds)



As long as two computers agree on the time period in which the electricity intervals are being sent, they can communicate much more efficiently. In an electrical connection (such as Ethernet), the signal would be a voltage or current. In an optical connection (such as a fiber-optic cable), the signal would be the intensity of light.

Definition 1.3

There are many properties about line coding that are relevant:

1. The **bit rate** describes the data transfer rate of a connection. It measures the number of bit states that a channel can *transmit* per unit time. It is measured in *bits per second.* We can interpret it as the amount of water flowing through a pipe.
   Bit rate is typically seen in terms of the actual data rate. But for most transmissions, the data represents part of a more complex protocol, which includes bits representing source address, destination address, error detection/correction codes, and other information. This data is called the **overhead**, while the actual data transferred is called the **payload**. At times, the overhead may be substantial (up to 20% to 50%).
2. The **throughput** is the number of bit states of usable information, that can be successfully *received* over a channel per unit time. Without any channel noise, it is really just the payload. Note that this is an *observed, dynamic parameter* with a fixed and variable loss. It is also known as **consumed bandwidth** and is measured in *bits per second.*
3. The **bandwidth** describes the *maximum* data transfer rate of a connection; that is, the maximum throughput of a communication. It is measured in *bits per second.* We can interpret it as how thick the pipe is (i.e. how much water can flow through it at max). Note that this is different from the bandwidth used in signal processing.
   Data often flows over multiple network connections, which means the connection with the smallest bandwidth (most likely your local connection) acts as a bottleneck.
4. The **latency**, or **ping-rate**, measures the round trip time between the sending of a data message to a computer and the receiving of that message, measured in *milliseconds.* We can interpret it as the speed at which the water is flowing through a pipe. We can check latency by doing

```
>>>ping www.google.com
64 bytes: icmp_seq=0 ttl=115 time=37.868 ms

```

   which outputs a latency time of 37.868ms (to get to `www.google.com` and back) for sending a data packet of 64 bytes. Note that there is an intrinsic limiting factor to latency: the speed of light, which is approximately 1 foot per nanosecond. In addition to distance, another limiting factor is the congestion in the network and the type of connection.

Example 1.2

Given two computers connected by a wire that is configured to transfer 1000 bits per second, the bit rate would be 1 Kbps. However, if the channel has noise and demands retransmission of 10 bits out of every 1000 of the original transmission, then the throughput would be 990 bps.
Furthermore, the Ethernet frame can have as many as 1542 bytes. Say that there are 1500 bytes of payload and an overhead of 42 bytes. Then, the **protocol efficiency** would would be

$$\frac{\text{payload}}{\text{frame size}} = \frac{1500}{1542} = 0.9727 = 97.3\%$$

Typically, the actual line rate is stepped up by a factor influenced by the overhead to achieve an actual target net data rate. In One Gigabit Ethernet, the actual line rate is 1.25 Gbits/s to achieve a net payload throughput of 1 Gbit/s. In a 10-Gbit/s Ethernet system, gross data rate equals 10.3125 Gbits/s to achieve a true data rate of 10 Gbits/s. The net data rate also is referred to as the throughput, or payload rate, of effective data rate.

Some common units of measurement are:

1. Mbps, Gbps, Tbps (Mega, Giga, Terabits per second)
2. MBps, GBps, TBps (Mega, Giga, Terabytes per second)

In conclusion, speed is a combination of bandwidth and latency. Even if a computer is on a connection with high bandwidth, its speed of sending and receiving messages will still be limited by the latency of the connection.

1.3 Internet Protocol: Addresses and Routing Strategies

The **Internet Protocol (IP)** is one of the core protocols in the layers of the internet. It is used in all Internet communication to handle both addressing and routing.

Definition 1.4

The protocol describes the use of **IP addresses** to uniquely identify Internet-connected devices (for transmission of data). That is, when a computer sends a message to another computer, it must specify the recipient's IP address and also include its own IP address so that the second computer can reply.
There are two versions of the Internet Protocol in use today:

1. **IPv4**: The first version ever used on the Internet and having the form of 4 *octets* split by periods in between.
$$[0-255].[0-255].[0-255].[0-255]$$
Even though it presented in decimal, computers store them in binary
$$74.125.20.113 \iff 01001011.01111101.00010100.01110001$$
IPv4 addresses can take 2^{32} values, but IPv6 was created for more space.
2. **IPv6**: The newer standard (introduced in June 2012) is in the form
$$\text{FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF}$$
with hexadecimal digits (total of 3.4×10^{39} possible IPv6 values).

Definition 1.5

A **dynamic IP address** is an IP address that can change. For example, each Internet service provider (ISP) has a range of addresses that they can assign, and they might give you a different one of those addresses each time your computer pops up on the network. Therefore, switching to a different WiFi network will definitely give you a new IP address.
Computers that act like servers often have **static IP addresses**. That makes it easier for computers to quickly send requests to the servers.

Definition 1.6 (Hierarchy of IP Addresses)

The IP addresses are formatted in an *hierarchical way.* The IPv4 address hierarchy is structured as such: The first few numbers (may or may not be divided by octets) could identify a **network** administered by an Internet Service Provider. The last numbers, which can also represent **subnetworks** (subnets), identifies a home computer on that network. For example, if we represent the IP address 141.213.127.13 in binary (of 32 bits)

10001101.11010101.01111111.00001101

the first 16 bits could route to all of UMich, the next two bits could route to a specific UMich department, and the final 14 bits could route to individual computers.

1000110111010101	01	11111100001101
UMich Network	Medicine department	Lab computer

This hierarchy gives UMich the ability to differentiate between 2^2 departments and $2^{14} = 16,384$ computers within each department. In general, the ability to create hierarchical levels at any point in the IP address allows for greater flexibility in the size of each level of the hierarchy.

1.3.1 LAN vs WAN IP Addresses

In fact, your computer is not connected to the internet directly. It is actually in a **private network**, or a *LAN network*, which uses a private IP address space (supported by both IPv4 and v6). Anything on the inside of your private network is not on the Internet; it is on your LAN, an entirely separate network, with its own address space. Anything on your LAN must have a unique (within the LAN) IP address to participate properly with your local network. Therefore, anyone else who has a LAN is also not part of the internet. Even though none of your devices in your network have a public IP address, the router itself does have a public address. That is, to the outside world, all devices identify their internet activity by the one IP address assigned by your ISP.

Definition 1.7

The IP addresses that are in the private network's space are usually divided up into 3 categories. But as of now, the categories don't mean anything.

1. **Class A private range addresses**: 10.0.0.0 - 10.255.255.255 (16,777,216 IPs)
2. **Class B private range addresses**: 172.16.0.0 – 172.31.255.255 (1,048,576 IPs)
3. **Class C private range addresses**: 192.168.0.0 – 192.168.255.255 (65,536 IPs)

Since the private IPv4 address space is relatively small, many private IPv4 networks unavoidably use the same address ranges. This can create a problem when merging such networks, as some addresses may be duplicated for multiple devices. In this case, networks or hosts must be renumbered, often a time-consuming task, or a network address translator must be placed between the networks to translate or masquerade one of the address ranges.

Definition 1.8 (NAT)

In order for LAN devices to connect to the Internet, their outgoing traffic has the source address changed to match that of the internet/WAN IP address of the router. The router keeps track of this, and makes sure any response traffic gets sent to the right internal machine. This is called **Network Address Translation (NAT)**. There are generally two types of NAT:

1. **Basic, one-to-one NAT**: The simplest type of NAT provides a one-to-one translation of IP addresses. In this type of NAT, only the IP addresses, IP header checksum, and any higher-level checksums that include the IP address are changed. Basic NAT can be used to interconnect two IP networks that have incompatible addressing.
2. **One-to-many NAT**: The majority of network address translators map multiple private hosts to one publicly exposed IP address. In a typical configuration, a local network uses one of the designated private IP address subnets. A router in that network has a private address of that address pace. The router it also connected to the Internet with a *public* address assigned by the ISP. As traffic passes from the local network to the Internet, the source address in each packet is translated on the fly from a private address to the public address. The router tracks basic data about each active connection (particularly the destination address and port). When a reply returns to the router, it uses the connection tracking data it stored during the outbound phase to determine the private address on the internal network to which to forward the reply.

All IP packets have a source IP address and a destination IP address. Typically packets passing from the private network to the public network will have their source address modified, while packets passing from the public network back to the private network will have their destination address modified. To avoid ambiguity in how replies are translated, further modifications to the packets are required, such as TCP or UDP.

You can find your WAN IP address simply by googling it, since your computer sends a message to the Google computers as soon as it loads `google.com`.

Example 1.3

In my case, my computer's LAN IP address is 192.168.0.8, my phone's LAN IP address is 192.168.0.20, and the public IP address (of the router) is 211.109.203.135. Running `whois` on my computer and phone's IP addresses reveals nothing much about the LAN one (since it is private anyways and many people have the same one), while the public one (about my router) reveals that it is owned by the ISP SK Broadband Co Ltd. But note that both the LAN and WAN IP addresses can change over time depending on your ISP (and as you restart your device).
However, if the cell phone has an active data plan and connected to a local LAN at the same time, it will have a *LAN IP address* and an *IP assigned by the mobile network*. When a browser on the phone fetches a web page, the web server will see the *LAN's gateway (public) IP address*, so essentially the phone deals with three IP's at that point. The computer on the other hand will only be assigned a LAN IP address but will also be requesting web pages via the same public IP address.
Additionally, moving my computer to another network, say Coffeebay, will change its LAN IP address. As I am writing this in the cafe, the new IP address is now 192.168.0.168. The WAN IP address of the Coffeebay network is 125.132.4.126, with the ISP Korea Telecom (KT) Corp.

The three most popular ISPs in Korea are:

1. KT Corp.
2. LGU
3. SK Broadband

These ISPs are also called **broadband providers**. The most popular ones in the USA are:

1. AT&T Internet Services

2. Comcast High Speed Internet (aka Xfinity)
3. Verizon High Speed Internet
4. Charter Communications (including Spectrum formerly Time Warner Cable)

By typing in the correct IP into the browser, going into the administrator interface, and logging in, you can modify the WiFi settings.

1.3.2 Routing IP Packets

Since there are physical limitations on how large a message can be when *routing* data between computers, many networking protocols split each message into multiple small **packets**.

Definition 1.9

Due to physical limitations on how large a message can be sent, the Internet Protocol splits messages into **IP packets**. Each IP packet contains both a header (20 or 24 bytes long) and data (variable length).

1. The **header** includes the following:

Version/Length, Service Type, Packet Length	1 byte, 1 byte, 2 bytes
Identification	2 bytes
DF, MF, Fragment Offset	2 bytes
Time to Live, Transport	1 byte, 1 byte
Header Checksum	2 bytes
Source IP Address	4 bytes
Destination IP Address	4 bytes
Options, Padding (optional)	3 bytes, 1 byte
Total	24 bytes

2. The **data** is the actual content, such as a string of letters or part of a webpage.

These packets hop from router to router towards their destination.

Definition 1.10

A **router** is a type of computing device used in computer networks that helps move data packets along. The process of a router receiving and sending a packet is as such:

1. The packet gets sent to a router, and the router receives it.
2. The router looks at the packet's IP header, more specifically the destination IP address. For example, the destination IP address may be `91.198.174.192`.
3. The router must now forward the packet, but it may have multiple routers to forward it to. In order to choose the router that is the "closest" to the IP destination, it has a **forwarding table** that helps it pick the next path based on the destination IP address. The table consists of not IP addresses, but the IP address prefixes. For example,

IP address prefix	path
91.112	# 1
91.198	# 2
192.92	# 3

   Since IP addresses are by definition hierarchical, the router only needs to store the prefixes. Once the router locates the most specific row on the table for the destination IP address, it sends the packet along that path.
4. This is repeated for the next router.
5. When the final router is reached, it should have in its forwarding table the exact IP address prefix.

IP address prefix	path
91.112	# 1
91.198.174.192	Direct
192.92	#2

This router now sends the message to the destination IP address, which may be a personal computer or a server.

Definition 1.11

The **modem**, short for **modulator demodulator**, "modulates" the signals going between the LAN and Internet.

The main difference between the router and the modem is that:

1. The router crates a network between the computers in your home and routes network traffic between them (through Ethernet cables or wireless connection). Your home router has one connection to the Internet and connections to your private local network.
2. The modem serves as a bridge between your local network and the Internet.

Some ISPs offer a modem and router in a single device, which has advantages and disadvantages.

Note that there are problems with these packets:

1. A computer might send multiple messages to a destination, and the destination needs to identify which packets belong to which message.
2. Packets can arrive out of order. That can happen especially if two packets follow different paths to the destination.
3. Packets can be corrupted, which means that for some reason, the received data no longer matches the originally sent data.
4. Packets can be lost due to problems in the physical layer or in routers' forwarding tables.
5. Similarly, packets might be duplicated due to accidental retransmission of the same packet.

However, there are higher level data transport protocols in the Internet protocol stack can deal with these problems, such as the *Transmission Control Protocol (TCP)* and *User Datagram Protocol (UDP).*

1.4 UDP and TCP

The **User Datagram Protocol (UDP)** is a lightweight data transport protocol that works on top of IP. UDP provides a mechanism to detect corrupt data in packets, but it does not attempt to solve other problems that arise with packets, such as lost or out of order packets. That's why UDP is sometimes known as the *Unreliable Data Protocol.* UDP is simple but fast, at least in comparison to other protocols that work over IP. It's often used for time-sensitive applications (such as real-time video streaming) where speed is more important than accuracy.

When sending packets using UDP over IP, the data portion of each IP packet is formatted as a **UDP segment**.

$$\text{IP Packet} \implies \text{UDP Segment}$$

Each UDP segment contains an 8-byte header and variable length data. The first 4 bytes of the UDP header store the **port numbers** (identification numbers) for the source and destination. The next two bytes store the segment length and the checksum.

Source port #	2 bytes
Destination port #	2 bytes
Segment length #	2 bytes
Checksum #	2 bytes
Total	8 bytes

Definition 1.12

A networked device can receive messages on different virtual **ports**. The different ports help distinguish different types of network traffic.

For example, the following command shows the ports in use:

```
>>> sudo lsof -i -n -P | grep UDP
launchd       1            root   30u  IPv4 UDP *:137
launchd       1            root   31u  IPv4 UDP *:138
mDNSRespo   164  _mdnsresponder    6u  IPv4 UDP *:5353
mDNSRespo   164  _mdnsresponder    7u  IPv6 UDP *:5353
rapportd    356          mbahng   12u  IPv4 UDP *:3722
systemsta   665            root   17u  IPv4 UDP *:*
netbiosd  14582        _netbios    3u  IPv4 UDP *:137
netbiosd  14582        _netbios    4u  IPv4 UDP *:138
```

Each row start with the name of the process that's using the port (one for going in, another for going out) and ends with the protocol and port number. Since each port number is 2 bytes, the maximum port number can be $2^{16} = 65,535$.

The **segment length** stores the length of the the entire UDP segment (including the header). Since the segment length can be represented in 16 bits, the maximum length of the UDP segment can be 65,635 bytes.

The **checksum** is used check for data corruption. It is computed as such. Before sending off the segment, the sender:

1. Computes the checksum based on the data in the segment (by literally summing up sections of the the binary representation of the data.
2. It stores the computed checksum in the field.

Upon receiving the segment, the recipient:

1. Computes the checksum based on the received segment.
2. Compares the checksums to each other. If the checksums aren't equal, it knows that the data was corrupted.

The **Transmission Control Protocol (TCP)** is a transport protocol that is used on top of IP to ensure reliable transmission of packets. TCP includes mechanisms to solve many of the problems that arise from packet-based messaging, such as lost packets, out of order packets, duplicate packets, and corrupted packets. Since TCP is the protocol used most commonly on top of IP, the Internet protocol stack is sometimes referred to as **TCP/IP**.

When sending packets using TCP/IP, the data portion of each IP packet is formatted as a **TCP segment**.

$$\text{IP Packet (4 bytes)} \implies \text{TCP Segment (variable)}$$

The TCP header can contain many more fields than the UDP header and can range in size from 20 to 60 bytes, depending no the size of the options field. It does contain the source port number, destination port number, and checksum.

Source port #	2 bytes
Destination port #	2 bytes
Sequence number	4 bytes
Acknowledgement #	4 bytes
Offset and Reserved	10 bits
URG, AFK, PSH, RST, SYN, FIN bits	6 bits
Window Size	2 bytes
Checksum	2 bytes
Urgent pointer	2 bytes
Options/Padding	4 bytes
Total	39 bytes

The process of transmitting a packet with TCP/IP is as such:

1. Two computers first establish a connection through a **three-way handshake**. Computer 1 sends a packet with the SYN bit set to 1. Then computer 2 sends back a packet with the ACK bit set to 1 plus the SYN bit set to 1. The first computer replies back with ACK=1. (Note that the SYN and ACK bits are part of the TCP header) The three packets involved in the three-way handshake do not typically include any data. Once the computers are done with the handshake, they're ready to receive packets containing actual data.
2. When a packet of data is sent over TCP, the recipient must always acknowledge what they received in the following way: Computer 1 sends a packet with data and a sequence number. Computer 2 acknowledges it by setting the ACK bit and increasing the acknowledgement number by the length of the received data (note that both numbers are also part of the TCP header). It is easy to see how these two numbers help the computers keep track of which data was successfully received, which data was lost, and which data was accidentally sent twice.
3. To close the connection, computer 1 initiates it by sending a packet with the FIN bit set to 1. Computer 2 replies with an ACK and another FIN. After one more ACK from computer 1, the connection is closed.

However, some problems can occur. TCP connections can detect lost packets using a timeout. After sending off a packet, the sender starts a timer and puts the packet in a retransmission queue. If the timer runs out and the sender has not yet received an ACK from the recipient, it sends the packet again. The retransmission may lead to the recipient receiving duplicate packets, if a packet was not actually lost but just very slow to arrive or be acknowledged (which happens when the packet takes a slower route through the Internet). If so, the recipient can simply discard duplicate packets.

TCP connections can also detect out of order packets by using the sequence and acknowledgement numbers. When the recipient sees a higher sequence number than what they have acknowledged so far, they will know that they are missing at least one packet in between.

1.5 Domain Name System (DNS), Hypertext Transfer Protocol (HTTP)

Definition 1.13

The **world wide web**, or **the web**, is a network of webpages, programs, and files that are accessible via URLs. It is a subsection of the Internet. A web browser loads a webpage using various protocols:

1. **Domain Name System (DNS) protocol** for converting domain names into IP addresses.
2. **HyperText Transfer Protocol (HTTP)** to request the webpage contents from that IP address.
3. **Transport Layer Security (TLS) protocol** to serve the website over a secure, encrypted connection.

Note that the web browser uses these protocols on top of the Internet protocols. The Web is just one of the applications built on top of the Internet protocols, but it is by far the most popular.

1.5.1 DNS

Definition 1.14

Computers are identified by their IP addresses, but to make these addresses more readable by humans, we identify them through the **domain name system**. For example, `www.wikipedia.org` connects us to the computers powering Wikipedia. Each domain name is made up of parts:

```
third-level-domain.second-level-domain.top-level-domain
```

1. There are a limited set of top level domains (**TLD**s), and many websites use the most common TLDs, such as `.com`, `.org`, and `.edu`.
2. The second level domain is unique to the company or organizaiton that registers it, like `wikipedia` or `facebook`.
3. The third level domain, also called a subdomain, is also owned by the same group of the second level domain. It often just directs you to a subset of the website.

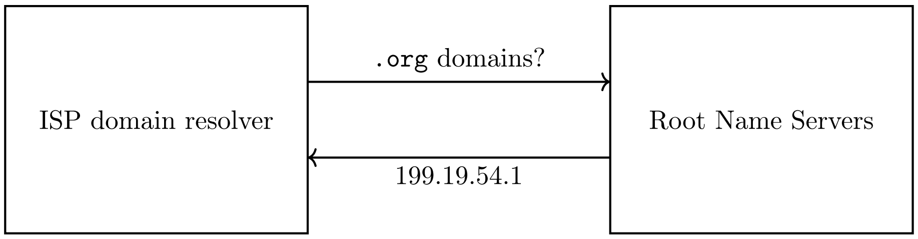
$$\texttt{m.wikipedia.org} \implies \text{mobile-optimized Wikipedia}$$
$$\texttt{es.khanacademy.org} \implies \text{Spanish-language Khan Academy}$$

In reality, these domain names are only for humans, and each domain name maps to an IP address. But since the computer can't store the 300 million domain names locally, it goes through a multi-step process to find out the IP address.

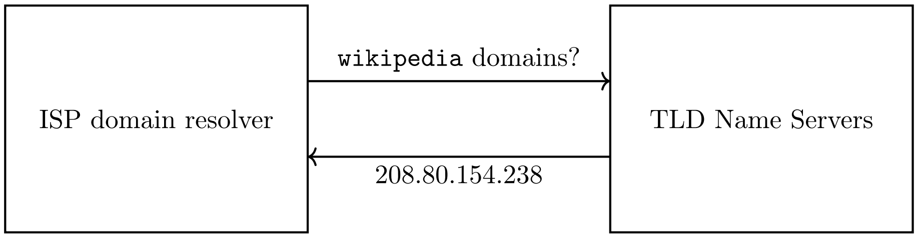
1. Check the local cache. Since people often visit the same website multiple times, they keep their own local cache of domain name to IP mappings. The cache stays small and is constantly updated. Each browser can keep their own cache.
2. Ask the ISP (Internet Service Provider) cache. Every ISP provides a domain name resolving service and keeps its own cache so the cache may contain the websites accessed by people with the same ISP (such as neighbors).
3. Ask the name servers. There are domain name servers scattered around the globe that are responsible for keeping track of a subset of the millions of domain names. There are three types of servers, which are ordered in a hierarchy:

   $$\text{Root name servers} \rightarrow \text{TLD name servers} \rightarrow \text{Host name servers}$$

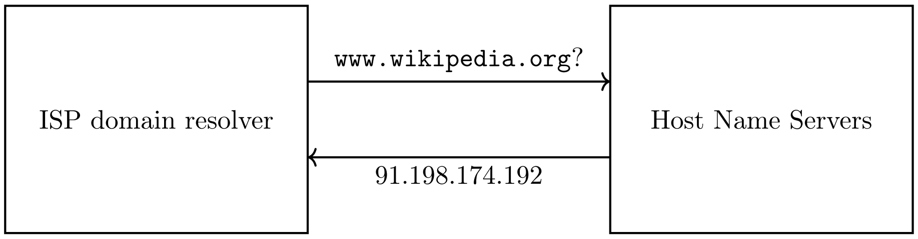
   The ISP starts by going to the root name servers and sending a request for the IP address of, say the name server of the `.org` domains. The root name server responds with the IP address of a TLD name server that tracks `.org` domains.

   Then, the ISP asks the TLD name server for, say the `wikipedia` domains. The TLD name server responds with the IP address of the host name server that contains the wikipedia records.



Finally, the ISP asks the host name server for the specific domain `www.wikipedia.org`. The host name server responds with an exact IP address, and now the computer can successfully connect with the computer powering that domain.



However, lots of information is cached, so it is rare that a **DNS lookup** has to go through all the steps.

However, the domain name system is not always secure. For example, if a cyber-criminal manages to take control of a name server or redirect requests to its own server, then it can have the server give out false IP addresses that can lead to websites filled with malware. This act is called **DNS spoofing**, or **DNS cache poisoning**. Fortunately, the recent **DNSSEC protocol** extends the original DNS protocol and specifies the best way for DNS resolvers to authenticate the information sent to them, which can prevent DNS spoofing.

1.5.2 HTTP

Whenever a pageon the web is visited, the computer uses the **Hypertext Transfer Protocol** to download that page from another computer somewhere on the Internet. The steps of this process are as such:

1. We access the web with a **browser application**. The user either types a **Uniform Resource Locator (URL)** in the browser or follows a link from an already opened page.
2. The domain names of these URLs map to IP addresses, the true location of the domain's computers. The browser uses a DNS resolver to map the domain to an IP address.
3. Browser sends HTTP request. Once the browser identifies the IP address of the computer hosting the requested URL (i.e. the **host computer**), it sends an **HTTP request**. An example HTTP request can be:

```
GET /index.html HTTP/1.1
Host: www.example.com
```

   (a) The word `GET` is the request. There are other verbs for other actions, such as `POST` for submitting form data.
   (b) The next part specifies the path (`/index.html`). The host computer stores the content of the entire website, so the browser needs to be specific about which page to load.
   (c) The final part of the first line specifies the protocol and the version of the protocol: `HTTP/1.1`.
   (d) The second line specifies the domain of the requested URL. That's helpful in case a host computer stores the content for multiple websites.

4. Once the host computer receives the HTTP request, it sends back a **HTTP response** with both the content and metadata about it.

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 208
<!DOCTYPE html>
<html>
    <head>
        <title>Example Domain</title>
    </head>
    <body>
        <h1>Example Domain</h1>
        <p>This domain is to be used for illustrative examples in documents.</p>
    </body>
</html>
```

   (a) The `HTTP/1.1` is the protocol and version. The next number is the **HTTP status code**. In this case, a `200` represents a successful retrieval of the document, "OK." Another code is the `404` code, which represents "file not found."

   (b) The 2nd and 3rd lines are the **headers**, which provides additional details. The content-type tells the browser what type of document is being sent back. `text/html` represent HTML text files; `image/png` represent images; `video/mpeg` are videos; `application/javascript` are scripts; and so on. The content length gives the length of the document in bytes.

   (c) The rest of the HTTP response writes our the actual document requested.

5. The browser renders the response, and you see the regular webpage.

Note that HTTP is a protocol that is built on top of the TCP/IP protocols. That is, each HTTP request/response is inside an IP packet (or more often, in multiple packets). There are many other protocols built on top of TCP/IP, like protocols for sending email (SMTP, POP, IMAP) and uploading files (FTP).

Note that the protocols powering the Internet and the Web were designed for scalability. Any computing device can send data around the Internet if it follows the protocols, and routing is dynamic, so new routers can join a network at any time and help to move data packets around the internet.

Definition 1.15

A **scalable system** is one that can continue functioning well even as it experiences higher usage.

However, there are limitations to the scalability of the internet. For example,

1. Network connections have limited bandwidth, so huge amounts of data could easily overwhelm low bandwidth connections, leading to delays or dropped packets.

2. Routers have limited throughput (the amount of data they can forward per second). A modern consumer router has a throughput around 1 Gbps while much more expensive enterprise routers can forward up to 10 Gbps.

3. Wireless routers often have a limitation in the number of devices that can be connected to them, typically up to 250 devices. If everyone tried to use a shared WiFi network at the same time (like in a university or library), they might find themselves simply unable to join.

Definition 1.16

Engineering teams can prepare for spikes in usage by doing **load testing**: simulating high amounts of traffic in a short period of time to stress test the system. Load testing can uncover bottlenecks or hard-coded limits in the system.

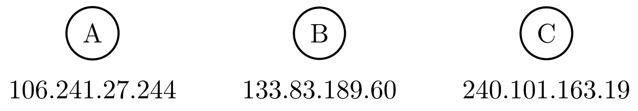
1.6 The Internet Protocol Suite

There are many protocols that power the Internet. Each protocol operates at a different layer, building functionality on top of the layer below it. We can visualize it in the following diagram.

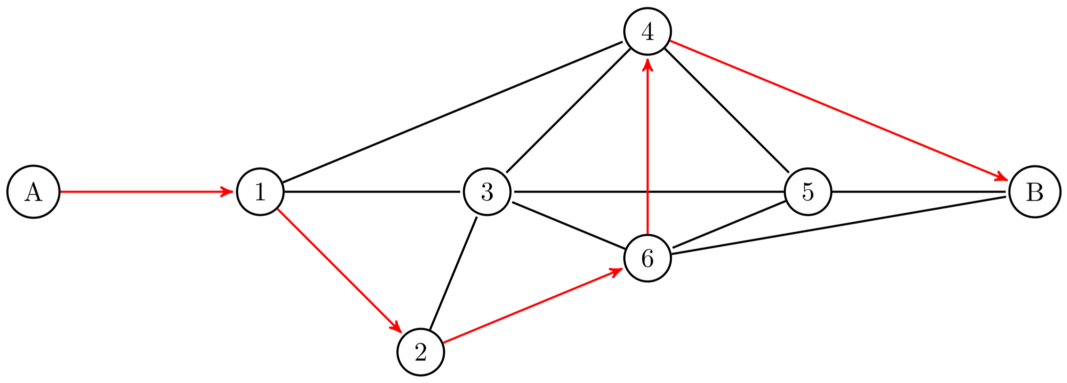
Application Layer	HTTP, TLS, DNS
Transport Layer	TCP, UDP
Network Layer	IP (v4, v6)
Link Layer	Ethernet, Wireless LAN

1. At the link layer, 2 computing devices need a physical mechanism to send digital data to each other. They send electromagnetic signals either over a wired or wireless connection and interpret the signal as bits. This type of physical connection affects the bit rate and bandwidth.

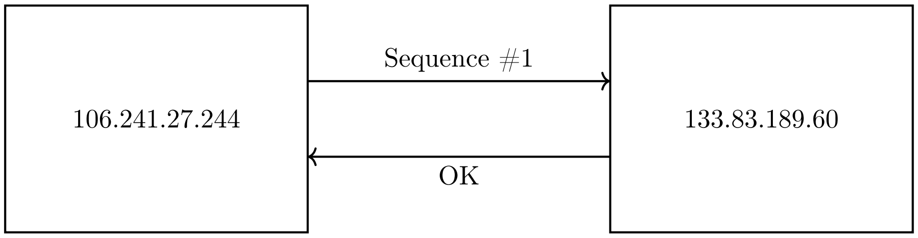
2. Once a network is bigger than two computers, we need addressing protocols to uniquely identify who is sending data and who should receive the data. Every node on the Internet is identified with an IP address.



3. The route between any two computers on the Internet isn't just a straight path from A to B. The data must pass from router to router until it finally reaches its destination, a strategy that comes from the Internet routing protocol.



4. Data needs to be broken up into small packets, which are then reassembled at the destination. The Transmission Control Protocol (TCP) is used to ensure reliable transport of those packets, with sequencing, acknowledgement, and retries. A faster but less reliable transport protocol is the User Datagram Protocol (UDP).



The Transport Layer Security (TLS) protocol uses algorithms to encrypt the data (cryptography). *Certificate authorities* help users trust the encryption.

Example 1.4

When loading a webpage from a domain your browser has never visited before, your browser may need to make a DNS request, which is represented by the following stack of protocols when the request is sent through the internet.

Application Layer	DNS
Transport Layer	UDP
Network Layer	IP
Link Layer	Wireless LAN

Then, your browser will make an HTTP request to fetch the webpage. This prototol stack is used when an HTTP request is sent:

Application Layer	HTTP
Transport Layer	TCP
Network Layer	IP
Link Layer	Wireless LAN

If the webpage is served over HTTPS, then the stack includes multiple protocols at the application layer (both HTTP and TLS):

Application Layer	HTTP, TLS
Transport Layer	TCP
Network Layer	IP
Link Layer	Wireless LAN

In a network, we must also make sure that the computers follow the *same* protocol, i.e. there is *standardization* within the network. Fortunately, the protocols of the Internet are **open** (not owned by any particular company and not limited to a particular company's products). For every protocol that is both standardized and open, there is a publicly viewable document describing the protocol, often called the **specification**. These specifications are maintained by the **Internet Engineering Task Force (IEFT)**. Some of them are:

1. the HTTP specification
2. the TCP specification

The languages of the web are also open standards with online specifications. They include the

1. HTML living standard
2. many specifications for CSS
3. ECMAScript standard for JavaScript.

1.6.1 Checking Network Processes

To check all open network connections (by process) and their usage of network bandwidth, use the `nettop` keyword on the command line. This can be used to find out which processes are taking up all of your local network bandwidth.

```
>>>nettop
                            interface       state    bytes_in  bytes_out
systemstats.64                                            0 B        0 B
udp4 *:*<->*:*

config.66                                                 0 B        0 B
udp4 *:*<->*:*

remoted.71                                            489 KiB    680 KiB
tcp6 IPv6\%en3.49160<->*.*        en3      Listen
tcp6 IPv6\%en3.49160<->IPv6       en3 Established     494 KiB    687 KiB

apsd.97                                                18 KiB     21 KiB
tcp4 IPv4:52737<->IPv4:5223       en0 Established      18 KiB     21 KiB

timed.99                                                144 B      144 B

usbmuxd.100                                           104 KiB     75 KiB
tcp4 IPv4:52695<->IPv4:54848      en0 Established     104 KiB     75 KiB

bluetooth.119                                             0 B        0 B
udp4 *:*<->*:*

AirPlayXPCHelpe.123                                       0 B        0 B
udp4 *:*<->*:*
udp4 *:*<->*:*

loginwindow.131                                           0 B        0 B
udp4 *:*<->*:*

mDNSResponder.164                                    6996 KiB   2606 KiB
udp6 *.5353<->*.*                 lo0                1717 KiB    790 KiB
udp4 *.5353<->*:*                 en0                5235 KiB   1808 KiB

symptomsd.189                                             0 B        0 B
udp4 *:*<->*:*

findmydeviced.190                                      1349 B     3061 B
tcp6 IPv6\%en3.49161<->IPv6.49248 en3 Established      1349 B     3061 B

airportd.198                                              0 B        0 B
udp4 *:*<->*:*
udp4 *:*<->*:*
udp4 *:*<->*:*
udp6 *:*<->*:*
udp4 *:*<->*:*

corekdld.203                                            293 B     31 KiB
tcp6 IPv6\%en3.49163<->IPv6.49256 en3 Established       293 B     31 KiB

bosUpdateProxy.204                                      406 B     2903 B
tcp6 IPv6\%en3.51336<->IPv6.49260 en3 Established       406 B     2903 B

#Only the processes will be shown from now on.

SubmitDiagInfo.205                                   65 KiB        9210 B
mobileactivatio.206                                  9238 B        11 KiB
locationd.270                                           0 B           0 B
```

```
biometrickitd.290                                       509 KiB        95 KiB
accountsd.338                                          5518 B        1247 B
trustd.342                                             4363 B         610 B
Simplenote.355                                         6316 B        6661 B
rapportd.356                                            10 KiB       4965 B
ControlCenter.359                                         0 B           0 B
Finer.361                                                 0 B           0 B
identityservice.369                                    4600 B        2961 B
itunescloudd.374                                        30 KiB       2533 B
com.apple.geod.392                                     3512 B        1353 B
WirelessRadioMa.414                                       0 B           0 B
sharingd.417                                              0 B           0 B
nsurlsessiond.420                                         6 KiB      9246 B
CalendarAgent.424                                        95 KiB        39 KiB
wifivelocityd.426                                         0 B           0 B
NewsToday2.437                                          392 KiB        27 KiB
AMPDeviceDiscov.441                                    2602 B        3725 B
assistantd.453                                           22 KiB      8764 B
ScreenTimeWidge.463                                    8029 B         936 B
WeatherWidget.464                                      7471 B        1168 B
corespeechd.471                                        7226 B          25 MiB
com.apple.Safar.530                                      11 KiB      2478 B
AdobeDesktop S.616                                        0 B           0 B
node.640                                                  0 B           0 B
comapple.WebKi.1274                                      17 MiB       627 KiB
com.appleSafar.1335                                    9956 B        2303 B
PowerChime.1353                                        2853 B          19 KiB
adprivacyd.1388                                          24 KiB      5253 B
lskdd.1768                                             1805 B        5851 B
PerfPowerServic.15327                                     0 B           0 B
netbiosd.19203                                          277 KiB       142 KiB
bluetoothaudiod.19268                                     0 B           0 B
Notify.20116                                              0 B           0 B
```

Note that pressing `q` quits nettop; pressing `p` renders the traffic numbers as bytes or in human-readable formats; pressing `c` collapses the display, showing only the network apps (no sockets); pressing `e` expands the display to show sockets.

Under each process (network application) there is a list of **sockets**, which are endpoints in a two-way communication between two programs on a network; for example, between a web server and your web browser. Looking at the columns,

1. The leftmost column contains the list of all the names of the processes and sockets, followed by a dot and their process ID (PID); so in the form

   `NetworkApp.PID`

   For example, `locationd.270` would be the process locationd with a PID of 270.

2. A **network interface** is the point of interconnection between a computer and a private/public network. The *network interface card* connects your computer to a local data network or the internet. The card translates computer data into compatible electrical signals it sends through the network.

   (a) `lo0` is loopback interface. A **loopback** is the routing of electronic signals, digital data streams, or flows of items back to their source without intentional processing or modification. It is primarily a means of testing the transmission tests.

   (b) `en0` is Wifi (was ethernet at one point)

   (c) `fw0` is the FireWire network interface

(d) `utun1`

(e) `stf0` is an IPv6 to IPv4 tunnel interface to support the transition from IPv4 to IPv6 standard.

(f) `gif0` is a more generic tunneling interface.

(g) `awdl0` is Apple Wireless Direct Link

3. The **state** refers to the state of the connection between sockets.

   (a) The state of a server waiting for a connection on a port is `Listen`

   (b) The state of a connection recently closed is `TimeWait`

   (c) `Established` means that the connection is active

   (d) `SynSent` occurs when a client initiates the connection to a server by sending the SYN packet (a part of the 3-way handshake in TCP) and awaits the ACK packet.

4. The `bytes_in` and `bytes_out` shows how much traffic has come in and gone out for that socket (or for the entire process).

The format of each socket is

```
TransportProtocolVersion localhostIPaddress:port<->remoteIPaddress:port
```

Notice that all of the sockets use the standardized UDP or TCP protocol (with the corresponding IP address Version: IPv4 or IPv6).

1. `udp4, udp6` - the connection is one-way since there are no SYN and ACK bits being sent to confirm the connection.

2. `tcp4, tcp6` - the connection guarantees that both ends are aware of one other, so datagrams can be sent back and forth until the FIN bit is sent and acknowledged.

A transport protocol (say, tcp) that use IPv4 is in the form

```
tcp4 192.168.0.88:52737<->17.57.145.138:5223
```

while one that uses IPv6 is in the form

```
tcp6 fe80::aede:48ff:fe00:1122%en3.4915<->fe80::aede:48ff:fe33:445%en3.5960
```

Some sockets may have asterisks rather than actual IP addresses in them.

```
airportd.198                                              0 B        0 B
udp4 *:*<->*:*
udp4 *:*<->*:*
udp6 *:*<->*:*
```

An asterisk means that these sockets are open. The operating system creates these open sockets as placeholders of sorts, so that it can respond faster to incoming data (since incoming data will trigger the creation of a socket, which causes delay).

Definition 1.17

In addition to the IP address, the **port number** is the part of the addressing information used to determine what protocol incoming traffic should be directed to. That is, port number identifies a specific process to which an Internet or other network message is to be forwarded when it arrives at a server. They are represented by 16-bit numbers, meaning that port numbers can have values up to $2^{16} = 65,536$. However,

1. 0-1023 are restricted port numbers and are used by well-known protocol services. Some of them

include:
   (a) 80 for HTTP
   (b) 123 for NTP
   (c) 67, 68 for DHCP traffic
   (d) 443 for HTTPS (almost all ports in the browser socket will be 443).
   (e) 137, 138 for netbios. NetBIOS is a protocol used for File and Print Sharing under all current versions of Windows.
2. 1024-49,151 are registered port numbers; they can be registered to specific protocols by software corporations.
3. 49,152-65,536 are used as dynamic/private ports and can be used by anybody.

The differences between the IP address and port number are:

1. The IP address is used to identify a host in the network, while a port number is used to identify a process/service on your system.
2. IP address is the address of the layer-3 Internet protocol suite, while the port number is the address of the layer-4 protocols.
3. IP address is provided by admin of system or network administrator, while the port number is provided by the kernel of the operating system.

1.7 Online Data Security

Definition 1.18

Personally identifiable information (PII) refers to data that can directly or indirectly identify individuals. Some of the most common PIIs are:
1. Name
2. Social Security Number
3. Biometric Data (DNA, fingerprints, etc.)

Another weaker form of PII are **linkable PII**, which refers to data that can be combined from separate sources to identify individuals.

It is hard to classify certain data as PII or not since the capabilities and the creative use of them is changing. An instance of when attackers steal PII from companies is known as a **data breach**. You can check whether you are a victim of a data breach with services like `haveibeenpwned.com`.

The web is not private by default; websites often use **cookies** to track user action on their site and even across other sites (to improve their services).

Definition 1.19

An **HTTP cookie** is a small amount of text that helps a website track information about a user across multiple pages of the website and personalize the user's experience on the website.

If you've ever logged into a website, a cookies kept you logged in across multiple pages. A cookie is set in the following steps:

1. When a user navigates to a website for the first time (in a particular browser), the browser sends an HTTP request to the server that hosts the website.

```
GET /index.html HTTP/1.1
Host: www.shoopshop.com
```

2. The server sends back an HTTP response and includes a `Set-cookie` header in that response.

```
HTTP/1.0 200 OK
Content-type: text/html
Set-Cookie: sessionId=abc123; Expires=Wed, 09 Jun 2021 10:18:14 GMT
...
```

   The cookie contains a name (`sessionId`) and a value (`abc123`), plus an expiration date for the browser to clear this cookie from its memory. If it wants to set multiple cookies, it adds more `Set-cookie` headers to the response.

3. The browser saves the cookie information, storing it on the user's hard drive. That way, the data will persist even after restarting the browser or computer, which is why this type of cookie is called a **persistent cookie**. There are also **session cookies** which have no expiration date and are always deleted when the browser is shut down.

4. When the user navigates to a different page on the website, the browser sends along the stored cookies with each HTTP request.

```
GET /shop.html HTTP/1.1
Cookie: sessionId=abc123
```

5. When the server receives the HTTP request, it inspects the cookies and sees that this request is coming from a user with a known `sessionId`. It can then look up that session ID in its database and use any information about the session to personalize the response.

Cookies can have many uses, such as:

1. A search engine can use them to remember how many results a user prefers seeing per page.
2. A news site can use them to recommend headlines that are similar to the articles you've already read.
3. All sorts of websites can use cookies to track analytics, like how long you spent on a page and which buttons you clicked.
4. Any website with a log-in uses a cookie to keep you logged in on every page of the site. When you log out of that site, it clears the cookie and doesn't set it again until you login again.

It is clear that you should never share your cookies.

Definition 1.20

Each cookie stored by a browser is associated with a domain and path. When you visit a website and its server sends back an HTTP response with a cookie, the browser associates that cookie with the domain of the server. That's called a **first-party cookie**.
However, a website can also include resources from other domains, like an image, iframe, or script. When the browser requests those resources, their servers can also send back cookies, which will now be associated with their domain. These are called **third-party cookies**.

Imagine a user that visits a food blog with a recipe for gluten-free cookies. That blog includes a Facebook ad with a cookie. The user then visits `facebook.com` and notices a sudden uptick in ads about gluten-free products, which resulted from the cookies in the Facebook ad in the blog. Third-party cookies more often serve the purpose of collecting information for advertising and infringe more on the privacy of web users.

1.7.1 Search and Browsing History

Definition 1.21

A **search engine** is a service that builds an index of the World Wide Web and gives users a way to search that index.

It is important to know that in order to improve their services (like spelling correction) all search engines collect data on search queries (i.e. what was searched). A search query itself isn't private information, but some search engines can log much more than the query:

Search query	Date	Time	IP address	User agent
"jet ski"	03/11/20	11:14	49.121.111.73	Mozilla/5.0 (Windows NT 5.1)
"home depot"	03/11/20	16:00	49.121.111.73	Mozilla/5.0 (Windows NT 5.1)
"cheap pizza"	03/12/20	21:07	49.121.111.73	Mozilla/5.0 (Windows NT 5.1)
"Windsor"	03/13/20	14:32	49.121.111.73	Mozilla/5.0 (Windows NT 5.1)

A combination of these queries over a period of time can definitely be PII, and additionally, the search history can include a cookie or even a user ID if you were logged into the search engine website when you issued the query. Third party cookies can allow a website to track a user's browsing history across other websites, as long as each site loads the cookie from the same domain.

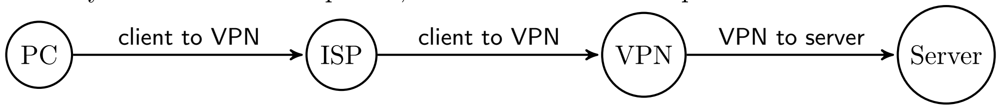
Many browsers also provide an *incognito browsing mode*, which will not store browsing history at all. Once you close the window, it will also forget any cookies generated in that session. There are certain search engines, such as DuckDuckGo, that collect only search queries and do not collect any PII.

Note that since all requests (packets of data) are forward through the router, anyone with access to the router can monitor the destinations of HTTP requests. An Internet Service Provider (ISP) administers the first routers that a packet travels through (excluding the home/office/school) router, so the ISP can see every HTTP request that's sent through those routers. Users can use HTTPS-secured websites to hide the contents of their requests, but HTTPS will still reveal the domain names. ISPs can use that information to find customers that are engaged in illegal activities, such as downloading pirated movies. Government organization such as the National Security Agency (NSA) have reportedly installed backdoor surveillance monitoring programs on routers before they were exported to foreign customers.

1.7.2 VPNs and Tor

Definition 1.22

When using a **Virtual Private Network (VPN)**, the computer sends a packet of encrypted data with a destination of the VPN server to the ISP. The VPN server decrypts the data, finds out where the user actually wants to send the packet, and then forwards the packet to that destination.



The VPN server knows the user's browsing history, but the ISP does not. Plus, other routers after the VPN will only see that the packet came from the VPN IP address, not from the user's IP address. A VPN subscription is often expensive, however, and the additional stop along the way can result in a slower browsing experience.

Another option is **Tor**, an open source program for anonymizing Internet traffic. When using Tor, the computer sends an encrypted packet through a large number of volunteer relays. The data is packaged such that each relay only knows where it came from and where it's going, and no relay knows both the sender IP address and the destination IP address.

Tor can provide truly anonymous browsing, but it also severely slows down the browsing experience, since it has to hop through volunteer relays that can be located anywhere on the Internet.

Definition 1.23

A **proxy server** is a server application or appliance that acts as an intermediary for requests from clients seeking resources from servers that provide those resources. A proxy server thus functions on behalf of the client when requesting service, potentially masking the true origin of the request to the resource server.
Instead of connecting directly to a server that can fulfill a requested resource, such as a file or web page, the client directs the request to the proxy server, which evaluates the request and performs the required network transactions. This serves as a method to simplify or control the complexity of the request, or provide additional benefits such as load balancing, privacy, or security. Proxies were devised to add structure and encapsulation to distributed systems. Some types of proxies are:

1. A **gateway** or a **tunneling proxy** is a proxy server that passes unmodified requests and responses.
2. An **open proxy** is a forwarding proxy server that is accessible by any Internet user. Hundreds of thousands of open proxies are operated on the internet.
   (a) **Anonymous proxies** reveals its identity as a proxy server, but does not disclose the originating IP address of the client.
   (b) **Transparent proxies** also identifies itself as a proxy server, but the originating IP address can be retrieved. The main benefit of using this type of server is its ability to cache a website for faster retrieval.
3. A **reverse proxy** is a proxy server that appears to clients to be an ordinary server. Reverse proxies forward requests to one or more ordinary servers that handle the request. The response from the proxy server is returned as if it came directly from the original server, leaving the client with no knowledge of the original server. Reverse proxies aer installed in the neighborhood of one or more web servers, and all traffic coming from the Internet goes through the proxy server. The use of *reverse* originates in its counterpart *forward proxy* since the reverse proxy sits closer to the web server and serves only a restricted set of websites.
   (a) Encryption/SSL acceleration: When secure websites are created, the *Secure Sockets Layer (SSL)* encryption is often not done by the web server itself, but by a reverse proxy that is equipped with SSL acceleration hardware.
   (b) Load balancing: the reverse proxy can distribute the load to several web servers, each web server serving its own application area.
   (c) Serve/cache static content: A reverse proxy can offload the web servers by caching static content like pictures and other static graphical content.
   (d) Compression: the proxy server can optimize and compress the content to speed up the load time.
   (e) Security: the proxy server is an additional layer of defense and can protect against some OS and Web Server specific attacks. However, it does not provide any protection from attacks against the web application or service itself, which is generally considered the larger threat.

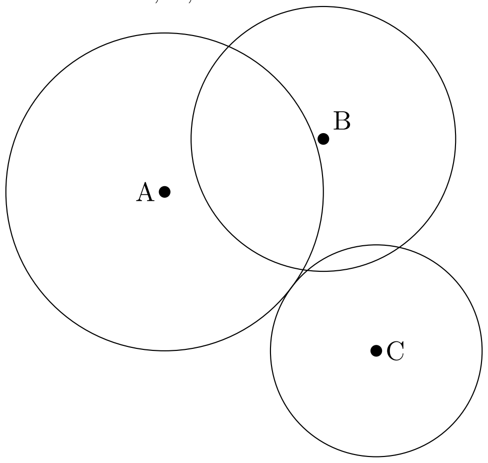
1.7.3 Geolocation

Definition 1.24

The **geolocation** of a device is an approximate latitude and longitude describing its geographic location.

Definition 1.25

The most popular method in which geolocation is determined is through **trilateration**, which is a geometric process of of determining absolute or relative locations of points by measurements of distances, using the geometry of circles, spheres ,or triangles. For example, given three points $A, B, C \in \mathbb{R}^2$, a unique point with certain distances from A, B, C can be determined.



Other methods of trilateration exist.

One way to determine geolocations is through the **Global Position System (GPS)**, a project started by the US government in the 1970s controlled by approximately 30 GPS satellites orbiting the Earth. **GPS receivers** are tiny sensors with antennas that receive radio signals from the GPS satellites orbiting in the sky above. If a sensor can receive signals from at least 4 satellites, the receiver can calculate its position using trilateraion. Since they depend on radio signals from satellites, GPS is most accurate in an outdoor environment with a clear view of the sky.

On the other hand, **WiFi positioning** is a strategy that works well in dense, urban areas filled with WiFi networks. First, a device with a WiFi antenna scans for WiFi access points and measures the signal strength to each network. (Note that signal strength is always negative, so the number closest to 0 is strongest)

BSSID	MAC address	Signal strength (RSSI)
NETGEAR09	A3:F3:5D:2A:A3:1B	-59
NETGEAR09-5G	A3:F3:5D:2A:A3:1B	-72
Sonic-b346	53:19:DA:E0:57:3A	-79
Emdutos	E3:84:14:BC:BC:FF	-84
Baskind Bunch	52:8D:5E:29:E7:5A	-85
Sonic-9472-5G	4C:4C:DB:91:1A:1A	-88
xfinitywifi	F8:59:F4:FC:C5:F1	-93

Then, the device determines the location of each access point by looking it up in a WiFi location database or in their own (smaller) cache of locations. It then estimates its own location based on the found locations and their signal strength using trilateration.

A more accurate technique is **fingerprinting**, but only possible if a fingerprint map has been made ahead of time. To make the map, a portable device computes the fingerprint for many reference points within a particular area. Each fingerprint is the list of nearby networks and their signal strength, like the table above, plus a pair of geographic coordinates. When a mobile device enters the area and needs to know its location, it can send its fingerprint to the machine with the radio map, and the machine uses an algorithm to compute the closest fingerprint and estimate the coordinates accordingly. This is basically just using WiFi positioning ahead of time.

If a cell phone is unable to use GPS to report its location, it can instead use **cell tower trilateration**. Cell towers are what makes cellular networks possible. Each cell tower includes three sets of directional antenna arrays in a triangular shape, and using trilateration, multiple cell towers can be used to determine the geolocation of a mobile device.

The least accurate of all methods is **IP-based geolocation**. IP geolocation databases contain millions of rows mapping IP addresses to locations. Companies create those databases based on a variety of sources such as regional IP address registries, user-submitted locations on websites, data from ISPs, and estimates based on network routes. They usually get the country and state correct, but often there are deviations in any more specific location data. Furthermore, if a user is accessinfg the Internet through a VPN, their true IP will be hidden and the VPN's IP could be geolocated in an entirely different continent.

Note that when a user visits a website, their browser sends an HTTP request to the web server. The HTTP request is wrapped in an IP packet, so it always includes the sender's IP address. Therefore, the web server can always use an IP geolocation service to turn the user's IP address into an approximate location, which can give better demographics for the company.

1.8 Cyber Attacks

A **phishing attack** is an attempt to trick a user into divulging their private information. Some signs of a phishing attack are:

1. suspicious email addresses. However, a legitimate email address is not a guarantee that an email is 100% safe. Attackers mught have figured out a way to spoof the legitimate email address or hacked their way into control over the actual email.
2. Suspicous URL. Attackers may
   (a) misspell the original URL (`goggle.com`)
   (b) use similar looking characters from other alphabets (the `e` and `a`) in `wikipedia.org` are actually different characters in those two domains)
   (c) subdomains that look like the domain name. (`paypal.accounts.com` vs `accounts.paypal.com`)
   (d) have a different top level domain (TLD) (`paypal.io` vs `paypal.com`). Popular companies try to buy their domain with the most common TLDs, such as `.net, .com, .org`, but there are hundreds of TLDs out there.
   (e) Have a hyperlinked text directed to a different URL.
3. Phishing websites sometimes may not use HTTPS. Any website that is asking you for sensitive information should be using HTTPS to encrypt the data sent over the Internet.

Definition 1.26

An **access point** acts as a translator between wireless and wired signals. Access points connect to the Internet via a wired connection but share it wirelessly with many devices like your computer. Most routers include access points since they are responsible for transporting packets, not for providing wireless Internet access. Most of them have an Ethernet cable in the back that connects it to the Internet and antennae that broadcase and receive wireless signals.

However, another form of cyberattacking is through **rogue access points**, which an access point installed on a network without the network owner's permission. If an attacker owns the access point, they can intercept the data (PII) flowing through the network. There are two ways rogue access points can intercept PII:

1. *Passive interception.* A rogue access point can read your data but cannot manipulate it. If you connect to a network with a rogue access point and enter your password on a site over HTTP, the rogue access point can read your password. They also have access to your Internet footprint.
2. *Active interception.* In active interception, a rogue access point can also manipulate your data. They can read the incoming user data, modify the data however they want, and send the modified user data to the destination endpoint. For example, if a user visits a banking website and tries to deposit money into an account, a rogue access point can redirect the deposit to an attacker's account.

We can also protect ourselves by using VPNs (virtual private networks) or HTTPS. VPNs and HTTPS both send an encrypted form of our data across the network. Even if rogue access points intercept it, they won't be able to unscramble it.

Definition 1.27

Malware is malicious software that is unknowningly installed onto a computer and often tries to steal personal data or make money off the user.

1. A **trojan horse** is a harmful program that masquerades as a legitimate program.
2. A **virus** is self-replicating: it contains code that copies itself into other files on the system. Viruses may hide in the code of a legitimate program.
3. A **worm** is also self-replicating, but copies itself into entirely different computers within the network. It can travel along networked protocols such as email, file sharing, or instant messaging.

The effects of malware are:

1. **Spyware** steals data and sends it back to the malware creators. A common form of spyware are keyloggers, programs that monitor everything a user types including passwords.
2. **Adware** pops up advertisements to users.
3. **Ransomware** holds a computer hostage y encrypting user data or blocking access to applications, and it demands the user pay a ransom to the anonymous malware creators.
4. **Cryptomining malware** utilizes a computer's resources to mine for cryptocurrency. That allows the creators to earn cryptocurrency without needing to spend money on powering their own computers.

Some protection mechanisms against malware include:

1. A security patch is an update to the code of an application or the entire operating system, and often fixes a bug that's been exploited by malware.
2. A **firewall** is a system that monitors incoming and outgoing network traffic to a computer or internal network, and determines what traffic to allow. Firewalls can do automated detection of suspicious traffic and can also be configured manually.
3. Antivirus software protects an individual computer by constantly scanning files and identifying malware. Once an antivirus program finds a piece of malware, it can guide the user through deleting or repairing the file to be safe again.

1.8.1 Secure Internet Protocols

We assume that the reader is familiar with basic encryption techniques, including public-key encryption.

Definition 1.28

A **symmetric encryption** is any technique where the same key is used to both encrypt and decrypt the data (e.g. the Caesar Cipher, Vigenere Cipher).

The **Transport Layer Security (TLS)** adds a layer of security on top of the TCP/IP transport protocols, using both symmetric and public key encryptionfor securely sending private data. Even though this extra process increases latency in Internet communications, the security benefits are well worth it. The process is described as such:

1. TCP handshake. The client must first complete the 3-way TCP handshake with the server.
2. TLS initiation. Then, the client must notify the server that it desires a TLS connection instead of the standard insecure connection, so it sends a message describing which TLS protocol version and encryption techniques it'd like to use.
3. Server confirmation of protocol. If the server doesn't support the client's requested technologies, it

will abort the connection. That may happen if a modern client is trying to communicate with an older server. As long as the server does support the requested TLS protocol version and other options, it will respond with a confirmation, plus a digital certificate that contains its public key.

4. Certificate verification. The client can verify (or choose not to) the certificate. The client now knows the public key of the server, so it can theoretically use public key encryption to encrypt data that the server can then decrypt with its corresponding private key. For speed, they use a combination of public-key and symmetric encryption to share data.
5. With this, the client can securely send private data to the server, using symmetric encryption and the shared key.

Notice that all of this depends on the credibility of the **digital certificate** which proves the ownership of an encryption key. A server that wants to communicate securely over TLS signs up with a **certificate authority (CA)**. The certificate authority verifies their ownership of the domain, signs the certificate with their own name and public key, and provides the signed certificate back to the server. This question now boils down to whether the certificate authority can be trusted, and there are actually intermediate CAs that verifies other CAs. The CA at the "top" of this chain that verifies is called the *root CA*.

With standard HTTP, many people can see what we're reading on the Internet, which is why websites are increasingly using **HTTPS (Hypertext Transfer Protocol Secure)** to protect the privacy of their uses and prevent tampering. HTTPS is also known as HTTP over TLS, because it's implemented by encrypting HTTP requests and responses with the TLS protocol.

When the browser loads a URL that starts with `https`, it begins the process of setting up a secure connection over TLS. Early in that process, the browser must verify the digital certificate of the domain. If the browser cannot verify the certificate, then the browser may display a certificate error (e.e. the message "Your connection is not private"). If the certificate is valid, most browsers will display a lock in the address bar, which indicates a secured connection over HTTPS.

An HTTPS connection ensures that only the browser and the secured domain see the data in HTTP requests and responses. Onlookers can still see that a particular IP address is communicating with another domain/IP and they can see how long that connection lasts. But those onlookers can't see the content of the communication, which includes the full URL path, the webpage HTML, and any text submitted in forms.

2 Networking

Networking is a large field in itself, but in here I go over the most useful and practical applications of it in my everyday use. Some ways that I personally benefit from this is:

1. Connecting to WiFi and diagnosing problems.
2. Connecting to WiFi and diagnosing problems.
3. Connecting to other networks such as computing clusters or third-party blockchains.
4. Seeing how more abstract schemes such as APIs work.
5. Ethical hacking.

I introduce these concepts and how to do some basic implementation a Unix operating system.

I like to learn about networking as if I am designing it from scratch. Some big questions to ask when designing network schemes are:

1. How do we uniquely identify computers?
2. How should we establish a connection between them? Through hardware or signals?
3. What protocols should we use, like a common language, so that all computers understand what each other are saying?

4. Can we implement security measures to prevent unwanted visitors into our computer?

2.1 Computer Networks and the Internet

Let us first define a computer network, some of its architecture, and then move onto the Internet.

Definition 2.1 (Computer Network)

A **computer network** is a group of computers (i.e. computing devices) that use a set of common *communication protocols* over digital interconnections for the purpose of sharing resources located on or provided by the *network nodes*, which may include personal computers, servers, networking hardware, or other specialised or general-purpose hosts.

These network nodes may be able to communicate to certain neighboring nodes, and this graph architecture determines the **network topology**. A computer network can be visualized as a connected graph of nodes

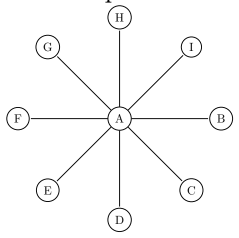
Example 2.1 (Network Topologies)

Common layouts are:

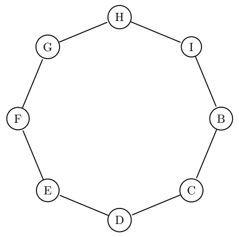
1. **Line Network**: All nodes are connected in a line.

$$\begin{tikzcd} A \arrow[r, dash] & B \arrow[r, dash] & C \arrow[r, dash] & D \end{tikzcd}$$

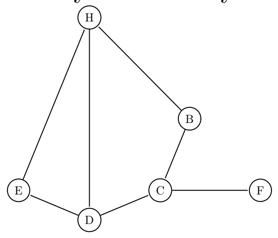
2. **Bus Network**: All nodes are connected to a common medium along this medium.
3. **Star Network**: all nodes are connected to a special central node.



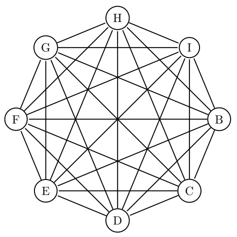
4. **Ring Network**: Each node is connected to its left and right neighbour node, such that all nodes are connected and that each node can reach each other node by traversing nodes left- or rightwards.



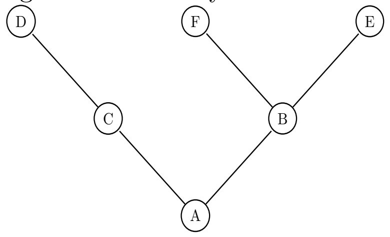
5. **Mesh Network**: each node is connected to an arbitrary number of neighbours in such a way that there is at least one traversal from any node to any other.



6. **Fully Connected Network**: each node is connected to every other node in the network.



7. **Tree Network**: nodes are arranged hierarchically.



Notice how many of these networks have **redundancy**: having multiple ways to get from one node to another. That is, when a network path is no longer available, data is still able to reach its destination through another path. Usually, we would like to avoid a **single point of failure** and construct a **fault-tolerant** system that can experience failure in its components but still continue operating properly. However, building more connections may be expensive.

Because there are multiple paths that a piece of data takes to get from point X to point Y, *routing strategies* are implemented in order to determine the most optimal path. Now in order for network nodes to communicate with each other, they should have some sort of universal method of communicating with each other.

Definition 2.2 (Communication Protocol)

A **communication protocol** is a system of rules that allow multiple entities of a communications to transmit information via any kind variation of a physical quantity. The protocol defines the rules, syntax, semantics and synchronization of communication and possible error recovery methods. A protocol can have many jobs, such as:
1. Determining how nodes will communicate with each other .
2. Making sure that these modes of communication is compatible with hardware .
3. Implementing security protocols such as encryption schemes.

Computers can connect through **physical** (e.g. cables) or **wireless** connections.

1. The **CAT5 cable** is a *twisted pair (copper) cable* that's designed for use in computer networks. It consists of four twisted pairs of copper wires. These twisted pair cables send data through a network by transmitting pulses of electricity that represent binary data. The information transmission follow the **Ethernet** standards, which is why twisted pair cables are commonly known as Ethernet cables. Use for both LANs and WANs. They can carry up to 1 Gbps across hundreds of feet, but are susceptible to interference.

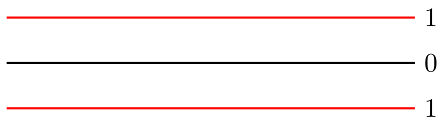
2. **Fiber-optic cables** carry light instead of electricity in a fiber coated with plastic layers. The pulses of light represent binary data and also follow the Ethernet standards. They are also capable of transmitting much more data per second that copper cables, and they have the advantage of low transmission loss and immunity to electrical interference. Often used to connect networks across oceans so that data can travel quickly around the world. They can carry up to 26 Tbps acorss 50 miles (but are expensive)

3. A wireless card inside a computer turns binary data into **radio waves** and transmits them through the air. However, they do not travel very far (100 ft in office buildings or up to 1000 ft in an open

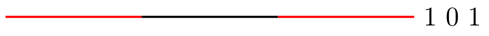
field). The waves are picked up by a *wireless access point* which converts them from radio waves back into binary data. These access points would be connected to the rest of the network using physical wiring. They can carry up to 1.3 Gbps.

4. **Infrared signals** and **microwaves** are sometimes used.

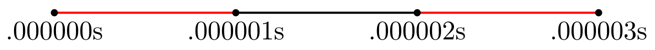
In order for the computers to send data into binary, they must convert this data into binary and send them as streams of 1s and 0s in a process called **line coding**. Furthermore, computers can raise efficiency of each wire by sending changing electric currents through a single wire. For example, rather than using three wires to encode `101` as



they send it through a single wire with intervals of $\frac{1}{3}$ seconds



or even better, at a rate of 1 megabit per second (interval of 0.000001 seconds)



As long as two computers agree on the time period in which the electricity intervals are being sent, they can communicate much more efficiently. In an electrical connection (such as Ethernet), the signal would be a voltage or current. In an optical connection (such as a fiber-optic cable), the signal would be the intensity of light.

Definition 2.3

There are many properties about line coding that are relevant, but ultimately the speed of a connection is a combination of the bandwidth and latency.

1. The **bit rate** describes the data transfer rate of a connection. It measures the number of bit states that a channel can *transmit* per unit time. It is measured in *bits per second*. We can interpret it as the amount of water flowing through a pipe.
   Bit rate is typically seen in terms of the actual data rate. But for most transmissions, the data represents part of a more complex protocol, which includes bits representing source address, destination address, error detection/correction codes, and other information. This data is called the **overhead**, while the actual data transferred is called the **payload**. At times, the overhead may be substantial (up to 20% to 50%).
2. The **throughput** is the number of bit states of usable information, that can be successfully *received* over a channel per unit time. Without any channel noise, it is really just the payload. Note that this is an *observed, dynamic parameter* with a fixed and variable loss. It is also known as **consumed bandwidth** and is measured in *bits per second*.
3. The **bandwidth** describes the *maximum* data transfer rate of a connection; that is, the maximum throughput of a communication. It is measured in *bits per second*. We can interpret it as how thick the pipe is (i.e. how much water can flow through it at max). Note that this is different from the bandwidth used in signal processing.
   Data often flows over multiple network connections, which means the connection with the smallest bandwidth (most likely your local connection) acts as a bottleneck.
4. The **latency**, or **ping-rate**, measures the round trip time between the sending of a data message to a computer and the receiving of that message, measured in *milliseconds*. We can interpret it as the speed at which the water is flowing through a pipe. We can check latency by doing

```
>>>ping www.google.com
64 bytes: icmp_seq=0 ttl=115 time=37.868 ms
```

which outputs a latency time of 37.868ms (to get to `www.google.com` and back) for sending a data packet of 64 bytes. Note that there is an intrinsic limiting factor to latency: the speed of light, which is approximately 1 foot per nanosecond. In addition to distance, another limiting factor is the congestion in the network and the type of connection.

Example 2.2

Given two computers connected by a wire that is configured to transfer 1000 bits per second, the bit rate would be 1 Kbps. However, if the channel has noise and demands retransmission of 10 bits out of every 1000 of the original transmission, then the throughput would be 990 bps.
Furthermore, the Ethernet frame can have as many as 1542 bytes. Say that there are 1500 bytes of payload and an overhead of 42 bytes. Then, the **protocol efficiency** would would be

$$\frac{\text{payload}}{\text{frame size}} = \frac{1500}{1542} = 0.9727 = 97.3\%$$

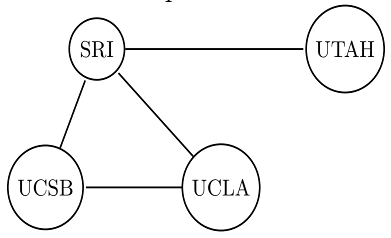
Typically, the actual line rate is stepped up by a factor influenced by the overhead to achieve an actual target net data rate. In One Gigabit Ethernet, the actual line rate is 1.25 Gbits/s to achieve a net payload throughput of 1 Gbit/s. In a 10-Gbit/s Ethernet system, gross data rate equals 10.3125 Gbits/s to achieve a true data rate of 10 Gbits/s. The net data rate also is referred to as the throughput, or payload rate, of effective data rate.

2.2 History of the Internet

IETF, ICANN, IANA, ISPs.

Example 2.3 (ARPANET)

The ARPANET was the precursor to the Internet, the network where Internet technology was first tested out. It was started in 1969 with four computers connected to each other.



For example, even if the path between SRI and UCSB is gone, the connections between SRI and UCSB is not lost (since IP packets can travel through UCLA's router).

Now we can see an implementation of these networks in the internet.

Definition 2.4 (Internet)

The **Internet** is a global network of computing devices communicating with each other in some way, whether they're sending emails, downloading files, or sharing websites. The Internet is an **open network**, which means that any computing device can join as long as they follow the protocols. The internet is powered by many layers of protocols, and to create a global network of computing devices, we need:

1. **Wires & Wireless**: Physical connections between devices, plus protocols for converting electro-

magnetic signals into binary data.
2. **IP**: A protocol that uniquely identifies devices using IP addresses and provides a routing strategy to send data to a destination IP address.
3. **TCP/UDP**: Protocols that can transport packets of data from one device to another and check for errors along the way.
4. **TLS**: A secure protocol for sending encrypted data so that attackers can't view private information.
5. **HTTP & DNS**: The protocols powering the World Wide Web

An **ISP (Internet Service Provider)** provides internet to its region. These ISPs are managed by certain continental autonomous systems (**AS**). The **Regional Internet Registry (RIR)** is divided into their regions: AFRNIC (Africa), ARIN (American), APNIC (Asia-Pacific), LACNIC (Latin America and Carribean), and RIPE NCC (European).

The main protocol suite used by the internet is **TCP/IP**, which is a collection of protocols that the internet uses. The bulk of this chapter will describe this protocol.

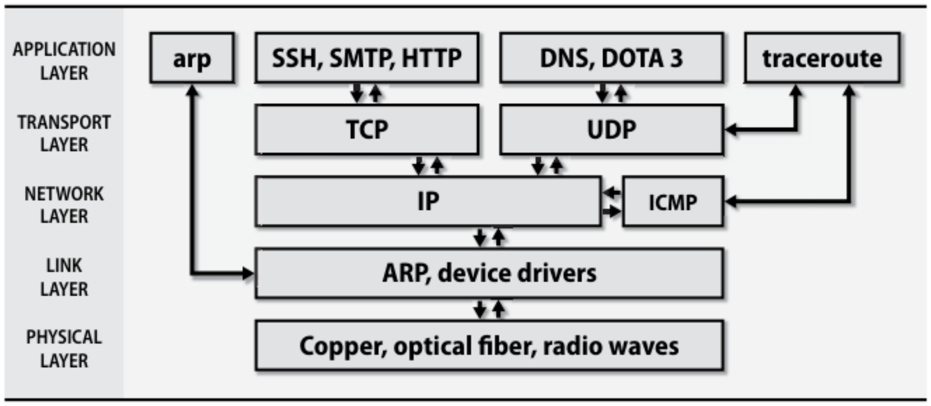


Figure 1: TCP/IP layering model.

2.3 Network Interfaces

Before we even start talking about IP addresses or protocols, we should mention that there are several interfaces from which computers can send and receive data. For example, if you are connected to both wired ethernet and WiFi, there are two paths, or interfaces, that data can travel. To see all your interfaces, use the `ip -c a` command.

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group
    link/ether 64:bc:58:11:c0:24 brd ff:ff:ff:ff:ff:ff
    inet 10.197.221.245/16 brd 10.197.255.255 scope global dynamic noprefixroute
       valid_lft 597085sec preferred_lft 597085sec
    inet6 fe80::b9e9:2f85:ded7:eaaf/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

The following lists out all the interfaces. We can see that we're connected to two interfaces, but there are a lot more. Usually, these interfaces also have a number following them that indexes different instances of the same type of interface.

1. **lo**: This is the loopback interface.
2. **wlan0**: For wireless connections
3. **tun**: When you are connected to VPN.
4. **en**:
5. **gif**:
6. **awd**:
7. **llw**:
8. **bridge**:
9. **utun**:

For each interface, there is a set of protocols that must be set for data to transfer.

2.4 Addresses

Every computer needs some address that determines its unique identity. The version of TCP/IP that has been in widespread use is IPv4, which uses 4-byte IP addresses. A modernized version, IPv6, expands the IP address space to 16 bytes and incorporates several additional features, making it faster and easier to implement.

Definition 2.5 (IP Address)

The protocol describes the use of **IP addresses** to uniquely identify Internet-connected devices (for transmission of data). That is, when a computer sends a message to another computer, it must specify the recipient's IP address and also include its own IP address so that the second computer can reply. There are two versions of the Internet Protocol in use today:

1. **IPv4**: The first version ever used on the Internet and having the form of 4 *octets* split by periods in between.
$$[0 - 255].[0 - 255].[0 - 255].[0 - 255]$$
Even though it presented in decimal, computers store them in binary
$$74.125.20.113 \iff 01001011.01111101.00010100.01110001$$
IPv4 addresses can take 2^{32} values, but IPv6 was created for more space.
2. **IPv6**: The newer standard (introduced in June 2012) is in the form
$$\text{FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF}$$
with hexadecimal digits (total of 3.4×10^{39} possible IPv6 values).

Definition 2.6 (CIDR Notation)

Sometimes, a set of IP addresses are specified using **CIDR notation**. An address of the form
$$145.201.67.4/16$$
represents all addresses of form $145.201. * .*$.

Operating systems and network devices have supported IPv6 for a long time, and the motivation behind the deployment of IPv6 was due to the concern that devices were running out of IPv4 addresses. Asia ran out first in 2011, followed by every other continent ever since then.

But we've learned to make more efficient use of the IPv4 addresses that we have. For example, **Network Address Translation** (or **NAT**) lets entire networks of machines hide behind single IPv4 addresses. **Classless Inter-Domain Routing** (**CIDR**) subdivides networks and promotes efficient backbone routing as well. Ultimately, IPv6, with better security and engineering, is going to take over, but not for a while since it's not fundamentally different from IPv4 and the drawbacks of IPv4 haven't been bad enough to spark migration.

Definition 2.7 (Hierarchy of IP Addresses)

The IP addresses are formatted in an *hierarchical way*. The IPv4 address hierarchy is structured as such: The first few numbers (may or may not be divided by octets) could identify a **network** administered by an Internet Service Provider. The last numbers, which can also represent **subnetworks** (subnets), identifies a home computer on that network.

Example 2.4 (University of Michigan)

For example, if we represent the IP address 141.213.127.13 in binary (of 32 bits)

10001101.11010101.01111111.00001101

the first 16 bits could route to all of UMich, the next two bits could route to a specific UMich department, and the final 14 bits could route to individual computers.

1000110111010101	01	11111100001101
UMich Network	Medicine department	Lab computer

This hierarchy gives UMich the ability to differentiate between 2^2 departments and $2^{14} = 16,384$ computers within each department. In general, the ability to create hierarchical levels at any point in the IP address allows for greater flexibility in the size of each level of the hierarchy.

Example 2.5 (Duke)

Duke's IP addresses are of the form 153.3._._, with the DUKE-INTERCHANGE ISP provider.

Definition 2.8 (Hostname)

IP addresses can be quite cumbersome to memorize, which is why they are often addressed with their **hostname**. Operating systems allow one or more hostnames to be associated with an IP address so that users can type `rfc-editor.org` rather than 4.31.198.49. This mapping can be set up in multiple days, e.g. with the `/etc/hosts` file or the LDAP database system to DNS the world-wide **Domain Name System**.

2.4.1 LAN Addresses and NAT

We've talked about how entire networks of machines can hide behind a single IPv4 address. Let's elaborate on this. In fact, your computer is not connected to the internet directly. It is actually in a **private network**, or a *LAN network*, which uses a private IP address space (supported by both IPv4 and v6). Anything on the inside of your private network is not on the Internet; it is on your LAN, an entirely separate network, with its own address space. Anything on your LAN must have a unique (within the LAN) IP address to participate properly with your local network. Therefore, anyone else who has a LAN is also not part of the internet. So if you are only on your LAN network, how do you actually connect to the internet?

Definition 2.9 (Router)

The **router** is a device that forms a connection between your LAN network and the internet. It has both a private local address, called a **gateway address**, and a public address. It is responsible for forwarding data between the local server computers and the internet. Therefore, to the outside world, all devices identify the network internet activity by the one public IP address assigned to the router. The gateway address can be found with `ip route` and the public address, of course, can be found with the commands previously mentioned.

Definition 2.10 (Modem)

A **modem**, short for **modulator/demodulator** is a device that converts a signal from your computer to some kind of signal to talk to other computers. The main difference between the router and the modem is that

1. The router crates a network between the computers in your home and routes network traffic between them (through Ethernet cables or wireless connection). Your home router has one connection to the Internet and connections to your private local network.
2. The modem serves as a bridge between your local network and the Internet.

To access our IP address, we can do the following:

1. To access local ip address, we can either run the command `hostname -i`, `ip -c a`, or `ifconfig`.
2. To access the public ip address, we can either google it or run `curl ifconfig.me`. Since this is public, any device connected to the same network/router should have the same IP address.

Definition 2.11 (NAT)

In order for LAN devices to connect to the Internet, their outgoing traffic has the source address changed to match that of the internet/WAN IP address of the router. The router keeps track of this, and makes sure any response traffic gets sent to the right internal machine. This is called **Network Address Translation (NAT)**. There are generally two types of NAT:

1. **Basic, one-to-one NAT**: The simplest type of NAT provides a one-to-one translation of IP addresses. In this type of NAT, only the IP addresses, IP header checksum, and any higher-level checksums that include the IP address are changed. Basic NAT can be used to interconnect two IP networks that have incompatible addressing.
2. **One-to-many NAT**: The majority of network address translators map multiple private hosts to one publicly exposed IP address. In a typical configuration, a local network uses one of the designated private IP address subnets. A router in that network has a private address of that address pace. The router it also connected to the Internet with a *public* address assigned by the ISP. As traffic passes from the local network to the Internet, the source address in each packet is translated on the fly from a private address to the public address. The router tracks basic data about each active connection (particularly the destination address and port). When a reply returns to the router, it uses the connection tracking data it stored during the outbound phase to determine the private address on the internal network to which to forward the reply.

Definition 2.12

The IP addresses that are in the private network's space are usually divided up into 3 categories. But as of now, the categories don't mean anything.

1. **Class A private range addresses**: 10.0.0.0 - 10.255.255.255 (16,777,216 IPs)
2. **Class B private range addresses**: 172.16.0.0 – 172.31.255.255 (1,048,576 IPs)

3. **Class C private range addresses**: 192.168.0.0 – 192.168.255.255 (65,536 IPs)

Since the private IPv4 address space is relatively small, many private IPv4 networks unavoidably use the same address ranges. This can create a problem when merging such networks, as some addresses may be duplicated for multiple devices. In this case, networks or hosts must be renumbered, often a time-consuming task, or a network address translator must be placed between the networks to translate or masquerade one of the address ranges.

2.4.2 Ports

IP addresses identify a machine's network interfaces, but they aren't specific enough to address individual processes or services, many of which might be actively using the network at once. TCP and UDP extend IP addresses with a concept known as a port, which is a 16-bit number that supplements an IP address to a particular communication channel. Valid ports range from 1 to $65,535$. A port, combined with an IP address, results in a **socket address** that is used to establish a connection between a client and a server.

UNIX systems restrict programs from binding to port numbers under 1024 unless they are run as root or have an appropriate Linux capability. Anyone can communicate with a server running on a low port number; the restriction only applies to the program listening on the port.

2.4.3 Hardware (MAC) Addresses

The lowest level of addressing is the network hardware. Many devices are assigned a unique 6-byte hardware address at the time of manufacture. The first 3 bytes identify the manufacturer, and the last 3 bytes are a unique serial number that the manufacturer assigns. Sysadmins can sometimes identify the brand of machine that is trashing a network by looking up the 3-byte identifier in a table of vendor IDs. In theory, ethernet hardware addresses are permanently assigned and immutable, but many network interfaces let you override the hardware address and set one of your own choosing.

2.5 TCP Packets and Encapsulation

2.6 OSI and Internet Protocols

2.7 HTTP and HTTPS

HTTP stands for hypertext transfer protocol, implemented in Layer 7, which transfers data between your computer and the server over the internet through **clear text**. This may not be the most ideal way since any interceptors can read the transferred data. This isn't a problem for regular internet browsing, but if you are inputting sensitive data, then HTTP should not be used. This is why HTTPS (which stands for secure HTTP) was invented, which is implemented in Layer 4 and encrypts the data being transferred, and every website where you input sensitive data should be using HTTPS (indicated by the `https://` prefix in the URL and a padlock symbol for modern browsers). Due to the extra security measures, HTTPS is less lightweight than HTTP, and its respective default ports are HTTP (80) and HTTPS (443).

A natural question to ask would be: which encryption scheme does HTTPS use? Both Secure Sockets Layer (SSL) and Transport Layer Security (TLS) is used in the modern web.

SSL certificate.

2.8 UDP and TCP

TCP handshake can be seen with curl.