CPSC 317: Introduction to Computer Networking[1]

Arpit Kumar

Fall 2023/24

[1] Taught by Prof. Norman Hutchinson

These are my class notes, further research and a possibly random collection of facts from this course.

1.1 Design of the Internet

Let's start at the internet's edge and work our way in.

- There are billions of computing devices running network apps on the internet's edge (hosts/end-systems plugged into the internet / hosts called hosts since they host network apps). A few examples of such devices is provided in FIGURE 1.
- Moving deeper, we find the devices that make the network actually a network, i.e **packet switches:** devices that forward packets (chunks of data). There are two types of packet switches - **routers** and **switches**.
- Then we come across the **communication links** that interconnect the routers, hosts and end-systems.
- Finally, each of these components discussed so far are assembled into their own networks administered by some organization/individual. This is why we say that the internet is a network of networks.

Figure 1: A few examples of devices you may see connected to the internet

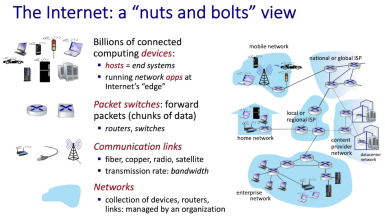


Figure 2: A nuts and bolts overview of the internet

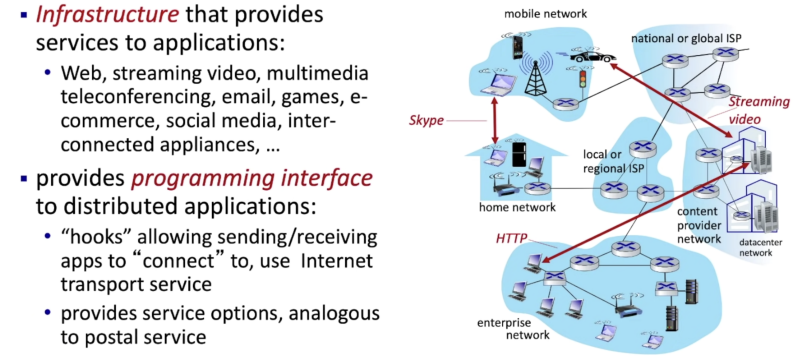


Figure 3: A "services" POV of the internet

What are network protocols?

- All communication activity in the interent is governed by protocols
- Protocols define the format, order of messages sent and recieved among network entities, and actions taken on message transmission and receipt

All communication over the internet requires a **protocol**, **source**, **destination**, **communication medium**, and of course, a **message**.

1.2 The network edge

All about the big picture here. Details further along in the course.

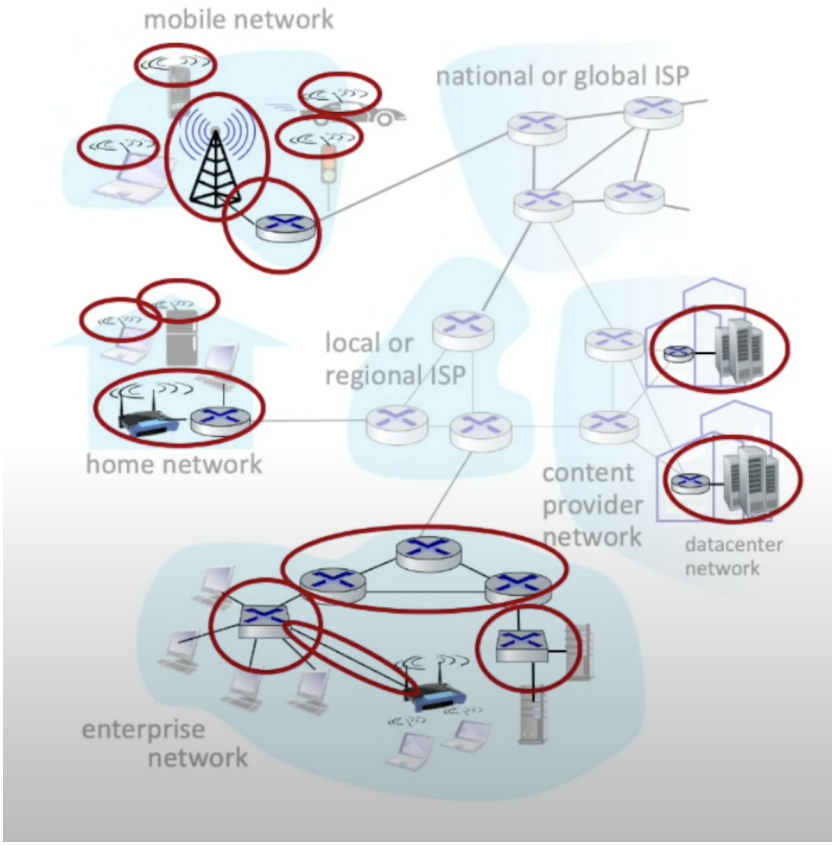


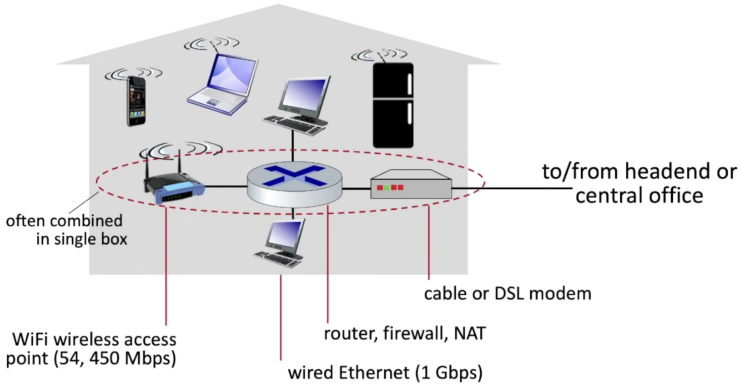
Figure 4: Access networks connect end-systems to the internet

Types of access networks

- **Cable-based access network: asymmetric** (downloads faster than uploads), homes share access network to cable headend, modem

rate limits your speeds (you get what you pay for)

- **Digital subscriber line (DSL):** use exisiting telephone line (voice, data transmitted at different frequencies) to central office DSLAM (access modem), also **asymmetric**, speeds depend on distance to central office, can't do DSL if distance over 3 miles
- **Home networks:**



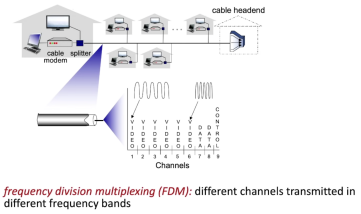


Figure 5: There only so many channels, so some users need to share the same channel (more in chapter 6)

Figure 6: Home networks setup

- **Wireless networks** Shared wirless access networks connect end-systems to router via a base station aka "access point"
  - **Wireless Local Area Networks (WLANs):** WiFi, operate within around $\sim$ 100ft
  - **Wide-area cellular access networks:** 3G/4G/5G, provided by mobile, cellular network operator
- **Enterprise network:** home network on steriods, mix of wired, wireless technologies, connecting a mix of switches and routers
- **Data center network:** connect hundreds to thousands of servers together over high bandwidth links (10s to 100s Gbps)

Links: physical media

- **Bit**: propogates between reciever/transmitter pair
- **Physical link**: what lies between transmitter and reciever
- **Guided media**: signals propogate in solid media (copper, fiber, coax)
- **Unguided media**: signals propogate freely

- **Twisted pair (TP)**: two insulated copper wires, now refers to ethernet, ADSL (Asymmetric Digital Subscriber Line)
- **Coaxial cable**: two concentric copper conductors, broadband connection (100's Mbps over multiple channels)
- **Fiber optic cable**: high speed operation (10's - 100's Gbps), low error rate
- **Wireless radio**: signals carried in various bands in the EM spectrum, broadcast (anyone can recieve - eavsdropping, interference concerns), signal fades over distance, obstruction by objects, affected by noise
  - Wireless LAN (WiFi)
  - Wide-area (4G cellular)
  - Bluetooth - short distances with limited rates
  - Terrestrial Microwave
  - Satellite

Figure 7: Fiber optics cable

1.3 Internet core

Essentially a mesh of routers connected by some connection link. The internet's core operation is based on a principle known as **packet-switching**: hosts break application layer messages into packets and the network forwards packets from one router to the next across links on path from source to destination.

Two key network core function:

- Forwarding (aka switching): a **local** action that maps packets at input link to appropriate output link
- Routing: how does the router know where to forward packets to? Routing is a **global** action to determine the path between a source and destination

Store and forward: entire packet must arrive at a router before it can be forwarded

Queue: packets waiting at the router when the arival rate is faster than the transmission rate

- packets can be dropped if memory (buffer) in router fills up (packet loss)

Alternate to packet switching: circuit switching

- Dedicated resources, no sharing
- Circuit segment remains idle if not used by call (no sharing)

Multiplexing describes multiple input streams sharing the same medium

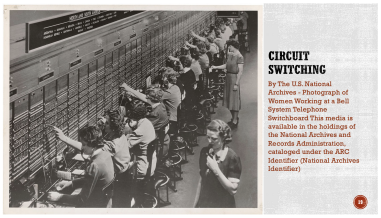


Figure 8: insane

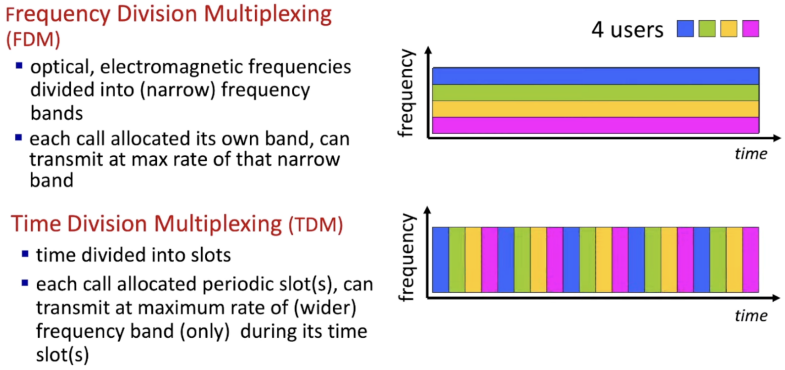


Figure 9: Circuit switching: FDM and TDM

Pakcet switching is a more viable option than **circuit switching** due to **statistical multiplexing**, i.e not all users are active at the same time and hence the probability that packets are going to begin queueing up is low.

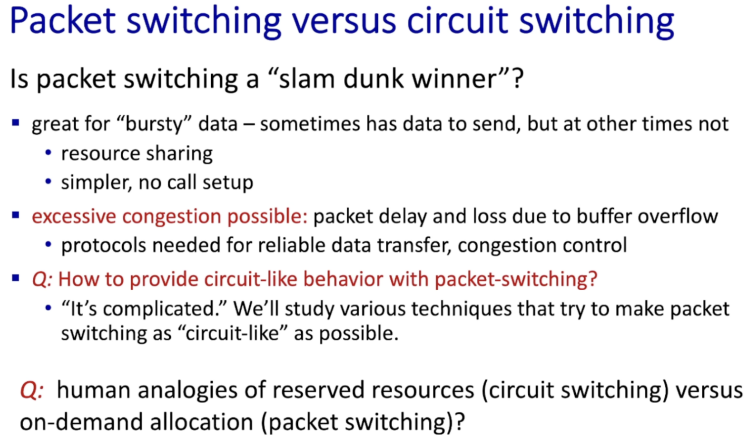


Figure 10: Is packet switching a "slam dunk winner"

1.5 Layering, Encapsulation

Why have a layered architecture? Because **explicit structure allows identification, relationship of system's pieces** and **modularization**

eases maintainence, updating of system

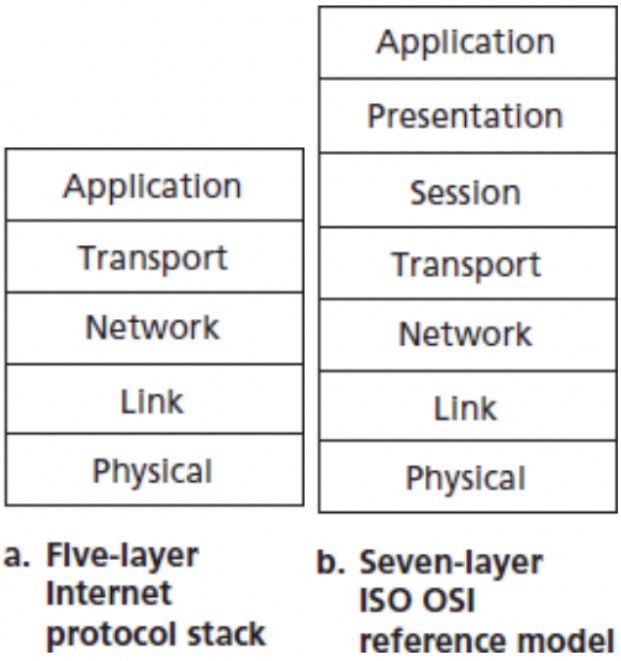


Figure 11: (a) 5-layer internet protocol stack (b) 7-layer ISO OSI reference model

Each layer has its own set of protocols to choose from.
Each layer takes data from above

- adds header information to create a new data unit
- passes new data unit into the layer below
- recieving system passes these data units up the stack
- recieveing system reads, parses then unpacks each data unit and sends it to the layer above

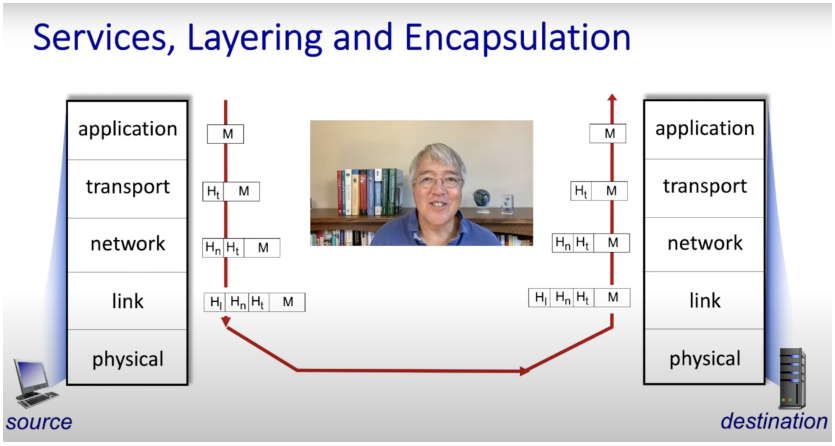


Figure 12: Encapsulation: adding header from packet of the layer above

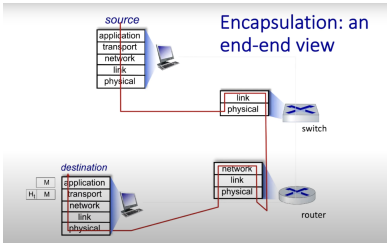


Figure 13: An example

Note that routers and switches don't operate on higher levels of the stack since they only deal with routing and forwarding data packets

Okay, now let's describe what each layer in this stack is responsible for. @TODO

Application Layer

@TODO

Transport Layer

Provides logical communication between application processes between different hosts. Handles breaking down messages into **segments** and reassembly at the reciever end. Also provides **multiplexing** of communication over the network. It enhances the services of the network layer by providing services such as congestion control, reliable delivery, etc.

Transport layer vs Network layer

Network layer terminates at the interface while the transport layer terminates once the message has delivered to the application process (a particular port on the host).

Two main transport layer protocols available to internet applications include **UDP** and **TCP**. UDP is unreliable. TCP is reliable. This distinction is important and taken into consideration by application layer protocols to decide which protocol to use.

THE TRANSPORT BIG PICTURE

Reliable stream	vs	Unreliable packet
Connection		No connection
Reliable ordered delivery		Best effort
Flow/Congestion control		Nope
Possible delays		No (transport level) delay

Figure 14: The big picture of transport protocols

UDP

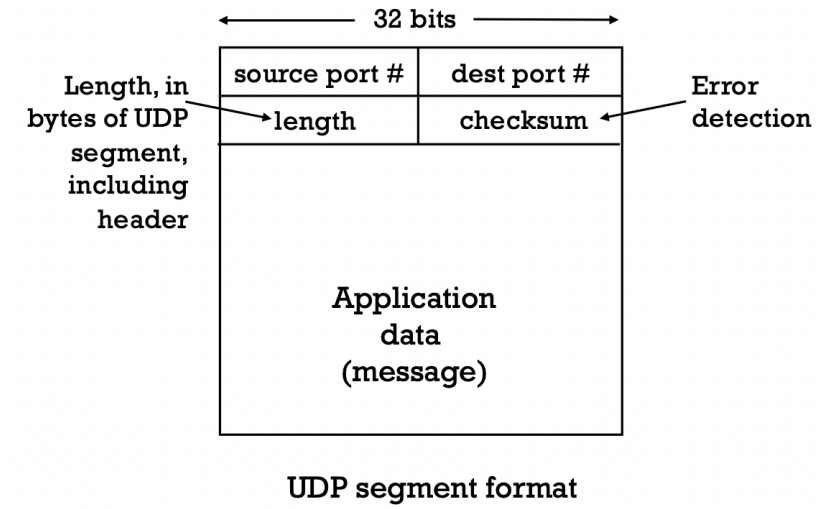


Figure 15: UDP segment format

Notice the checksum field - its purpose is to detect errors that may have occurred during transmission.[2]

[2] Checksums appear at the transport layer, network layer and link layer. They serve a different purpose at each of these layers. The same algorithm is used to compute the checksum at the transport and network layer.

Checksum Algorithm

- Treat data as a sequence of 16-bit integers
- Take the 1's complement sum of all these of 16 bit integers
- Checksum is the 1's complement of the computed value
- To verify, add this checksum to the sequence of integers under consideration and repeat the steps above - valid if you get all os

```
1 1 0 1 0
0 1 0 0 1
1 0 1 1 0
1 0 0 1 1
---------
0 1 1 0 0   Sum
      1 0
---------   Add the carry
0 1 1 1 0

1 0 0 0 1   1's complement
```

Figure 16: Compute checksum

```
      1 1 0 1 0
      0 1 0 0 1
      1 0 1 1 0
      1 0 0 1 1
      1 0 0 0 1
      ---------
1 0 1 1 1 0 1   Sum and carry
          1 0
      ---------   Add the carry
      1 1 1 1 1

      0 0 0 0 0   1's complement
```

Figure 17: Verify checksum

State Machines and Reliability

Again, what is a protcol?

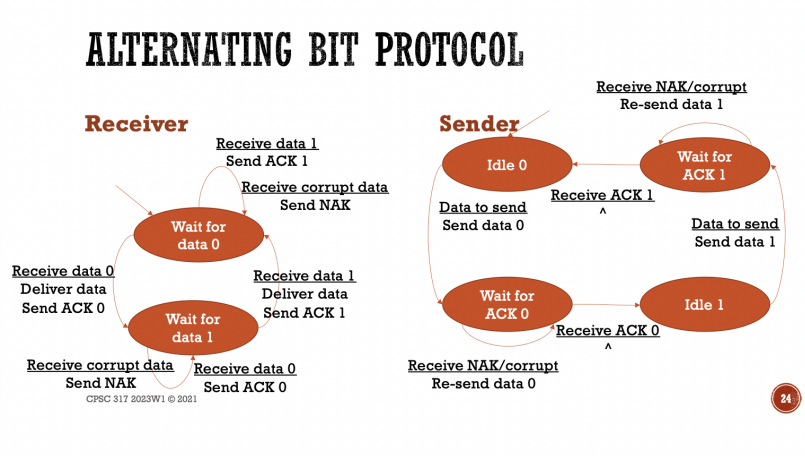
A **protocol** defines the order and format of messages between two communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other event.

Alternating Bit Protocol

Alternating bit protocol is a reliable transport layer protocol that gurantees delivery and ensures no duplication.

Services provided by the ABP are summarized below:

- Send only one segment at a time
- Identify when sending is allowable action
- Identify when re-sending is required
- Enumerate events and actions for both sender and receiver



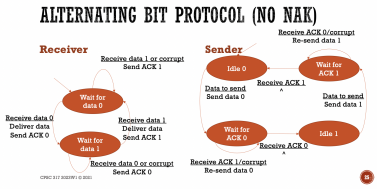


Figure 19: Can be simplified to not have any explicit NAKs

Lost Segments and Timeouts

Now it may be that data is lost in transmission. We can further simplify our protocol by treating corrupt data as lost data, i.e instead of sending a NAK, we just don't reply. The sender keeps track of the time elapsed since the message was sent, and if the an ACK is not received in time, the message is sent again.

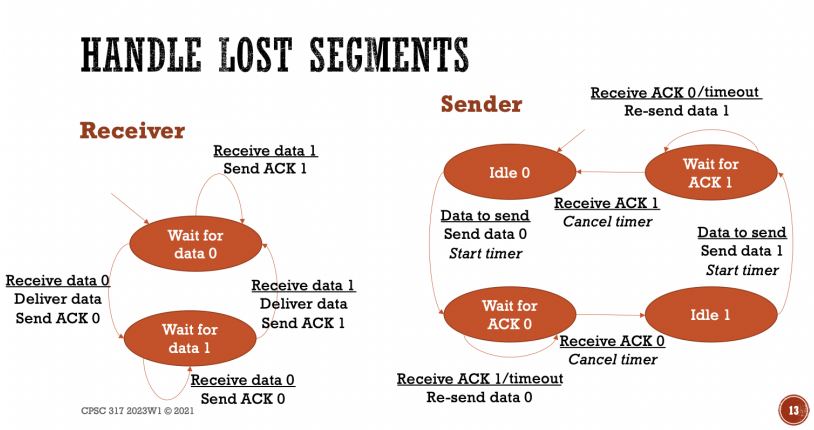


Figure 20: Handling lost data

The length of the timeout is crucial. It is clear that our timeout must be longer than the RTT of our connection. Otherwise, unescessary retransmissions will be sent. To decide on the timeout value, we need an estimate of the RTT values. TCP maintains a RTT measurement of one of the transmitted and yet to be acknowledged segments (ignoring retransmissions). This means that a new sampleRTT value is obtained once every RTT. This new value is used to update the estimated RTT by the following formula

$$EstimatedRTT = (1 - \alpha) \cdot EstimatedRTT + \alpha \cdot SampleRTT$$

Recent RTT values should be weighted more heavily. Thus, a suggested α value is 0.125.

Similarly, a measure of the deviation in the RTT is kept and updated as follows

$$DevRTT = (1 - \beta) \cdot DevRTT + \beta \cdot |SampleRTT - EstimatedRTT|$$

A recommneded value for $\beta = 0.25$. It is desirable to set the timeout equal to the EstimatedRTT plus some margin. The margin should be large when there is a lot of fluctuation in the SampleRTT values; it should be small when there is little fluctuation. The value of DevRTT should thus come into play here. All of these considerations are taken into account in TCP's method for determining the retransmission timeout interval:

$$TimeoutInterval = EstimatedRTT + 4 \cdot DevRTT$$

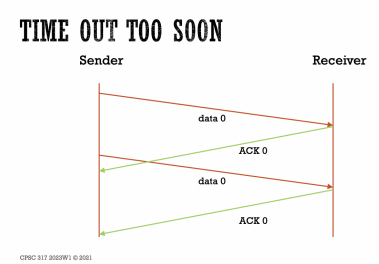


Figure 21: Timeout too soon

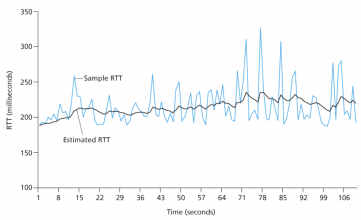


Figure 22: RTT samples and RTT estimates: our estimation formula smoothens out large fluctuations in RTT

Windowing Protocols

One major flaw with the ABP we discussed is the fact that it waits for an ACK before transmitting the next segment. This means that network resources are severly under-utilized.

Well, to solve this issue, the sender must be able to send mutliple segments without waiting for their ACK. This is known as pipelining. We will look at two pipeline reliable protocol strategies - **Go-Back-N** and **Selective Repeat**.

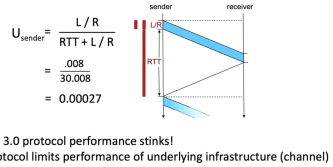


Figure 23: ABP underutilizes network bandwidth
Utilization refers to the freaction of time that a sender is busy sening data

Go-Back-N

@TODO [3]

[3] Think about message reordering

Selective Repeat

@TODO

Flow and Congestion Control

@TODO

Alternate Transport Protocols

@TODO

Network Layer

History and Autonomous Systems