# *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

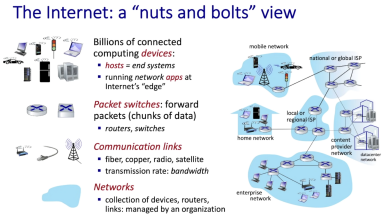



Figure 2: A nuts and bolts overview of the internet

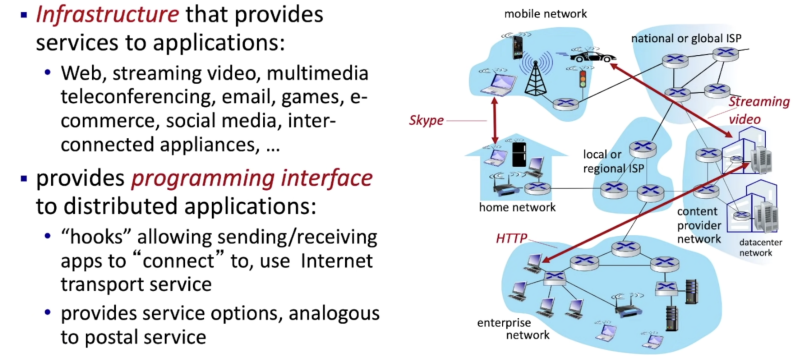



Figure 3: A "services" POV of the internet

*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.

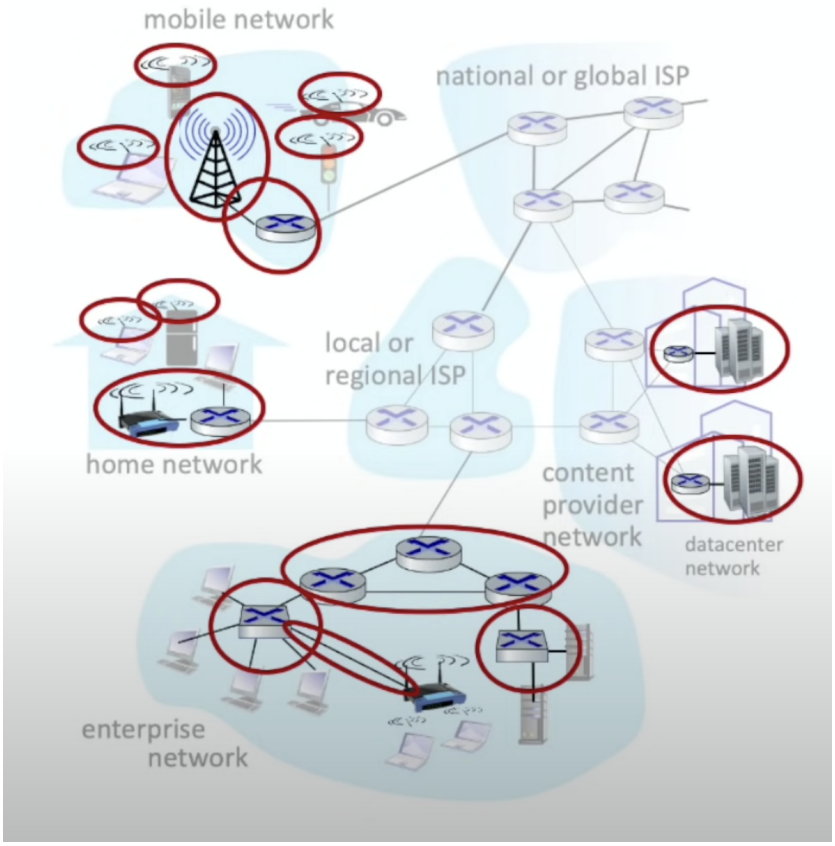


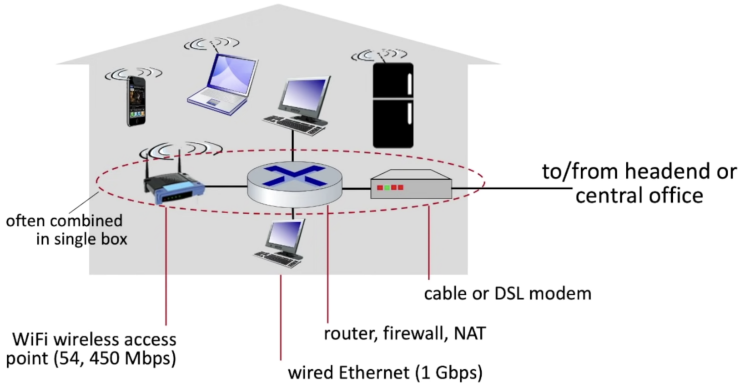

Figure 4: Access networks connect end-systems to the internet

*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:**



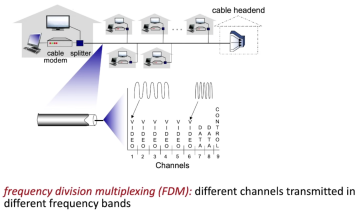




Figure 5: There only so many channels, so some users need to share the same channel (more in chapter 6)

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

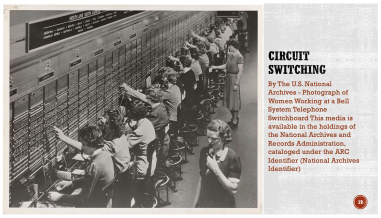



Figure 8: insane

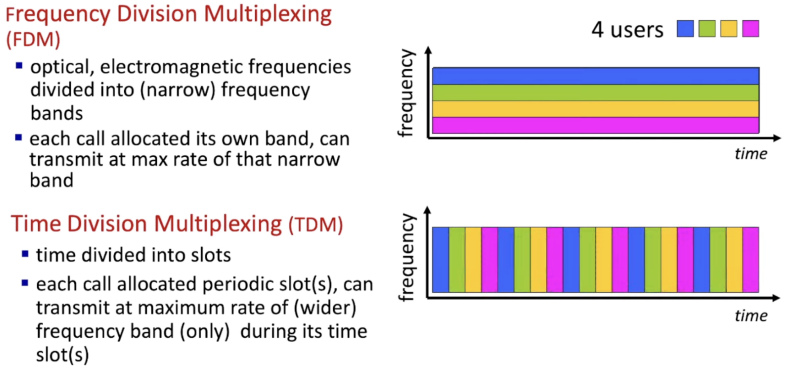



Figure 9: Circuit switching: FDM and TDM

**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.

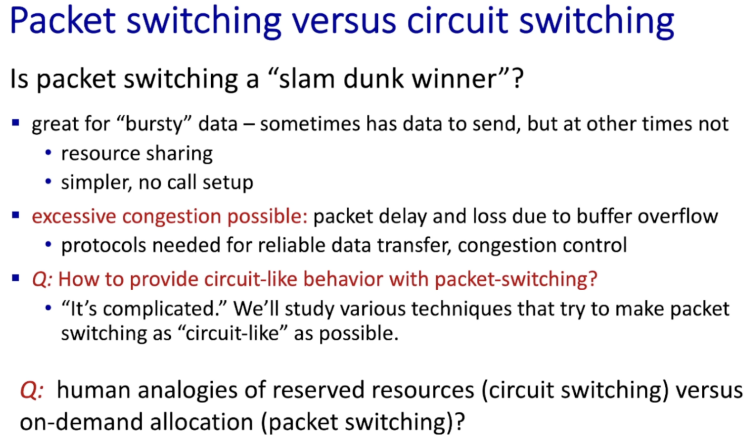



Figure 10: Is packet switching a "slam dunk winner"

## *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**

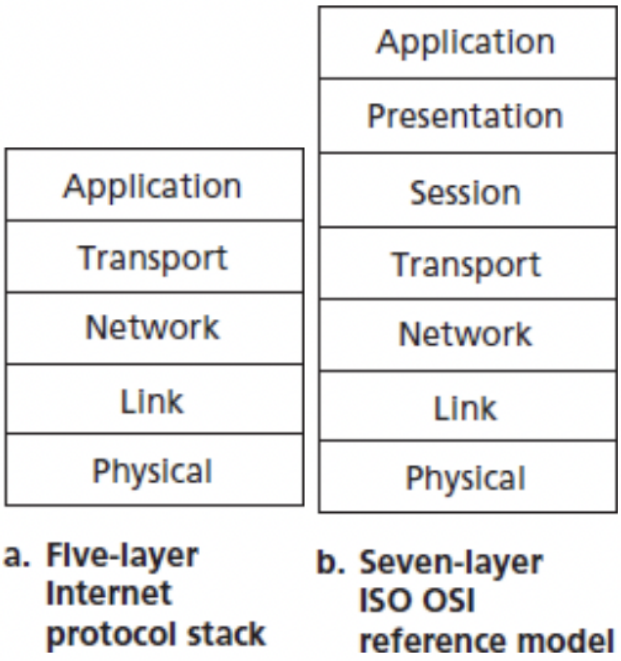



Figure 11: (a) 5-layer internet protocol stack (b) 7-layer ISO OSI reference model

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

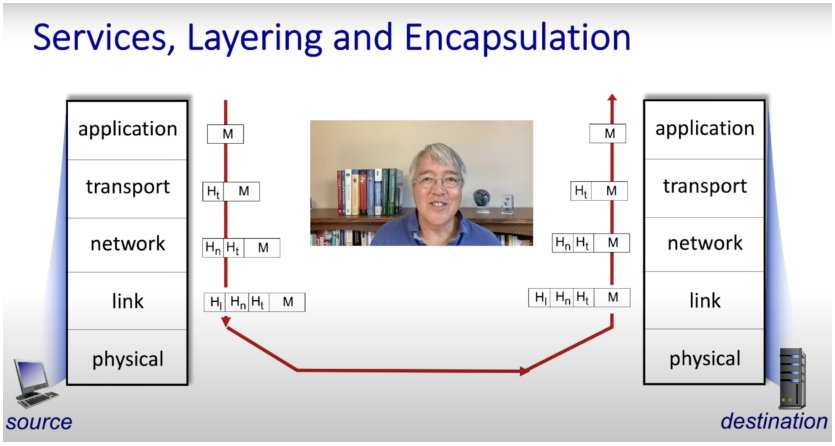



Figure 12: Encapsulation: adding header from packet of the layer above

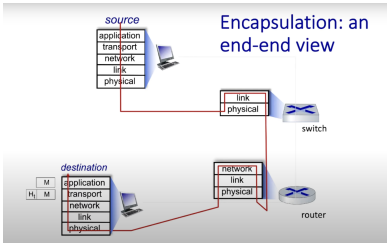



Figure 13: An example

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*

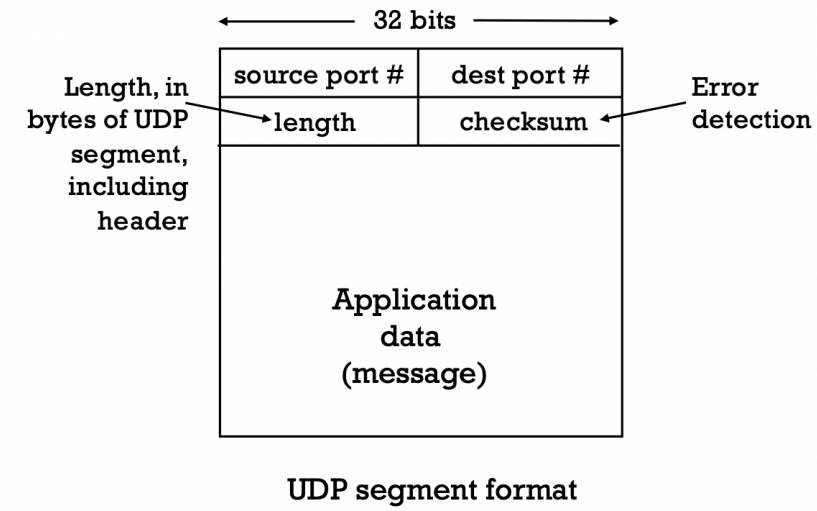



Figure 15: UDP segment format

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 0s

```
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

### *Transport Layer Multiplexing and De-multiplexing*

A host recieves IP datagrams (more on this later) - each datagram contains the source and desitnation IP addresses and one transport layer segment. Each segment contains the source and destination port numbers. The host uses IP addresses and port numbers to direct the segment to the appropriate socket. Since UDP only de-multiplexes on the basis of destination port number, it sends all datagrams from all origins to the same socket. TCP de-mutliplexes on the basis of the tuple (source IP, destination IP, source port, destination port). This is because TCP is connection oriented.

## *State Machines and Reliability*

### *Again, what is a protcol?*

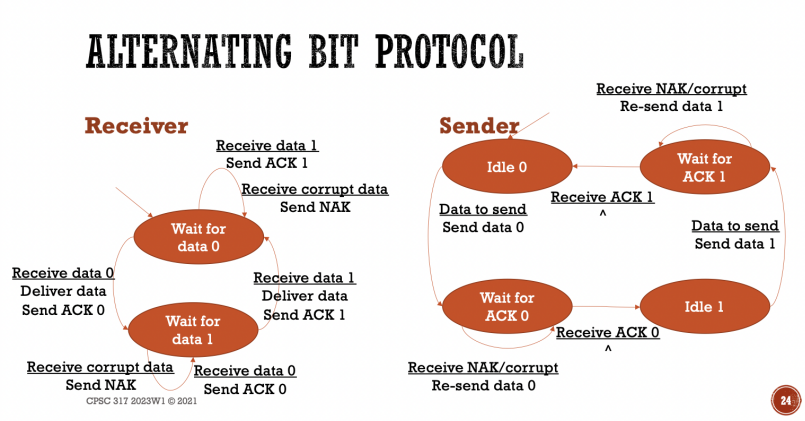
A **protocol** defines the order and format of messages between two communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other event.

### *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



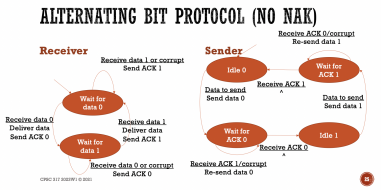




Figure 19: Can be simplified to not have any explicit NAKs

## *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.

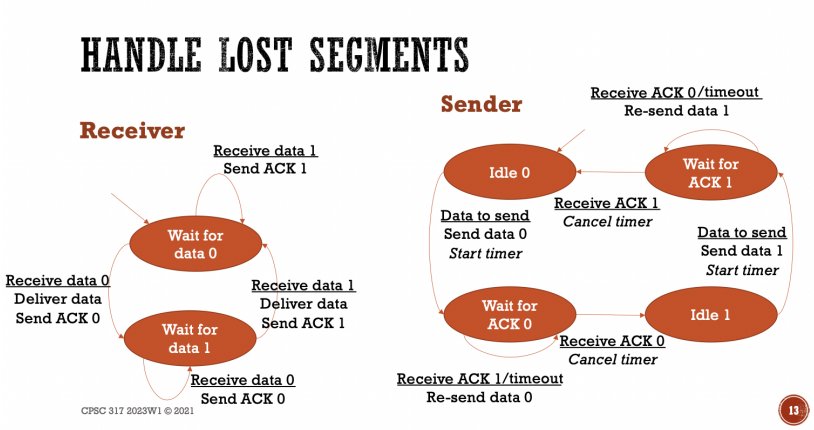



Figure 20: Handling lost data

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 $\alpha$ 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:

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

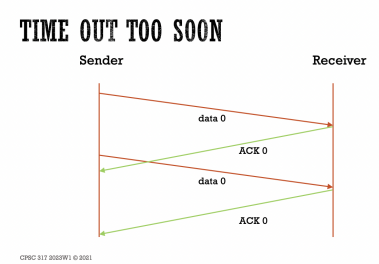



Figure 21: Timeout too soon

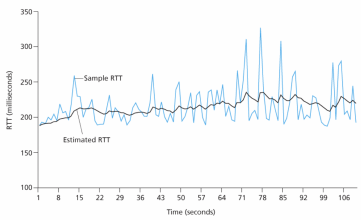



Figure 22: RTT samples and RTT estimates: our estimation formula smoothens out large fluctuations in RTT

## *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**.

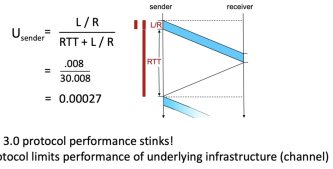



Figure 23: ABP underutilizes network bandwidth
Utilization refers to the freaction of time that a sender is busy sening data

### *Go-Back-N*

@TODO [3]

[3] Think about message reordering

### *Selective Repeat*

@TODO

### *Flow and Congestion Control*

@TODO

## *Alternate Transport Protocols*

@TODO

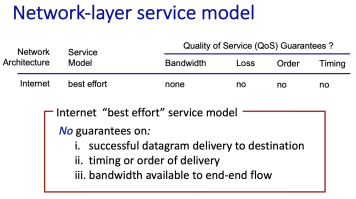



Figure 24: The network service model

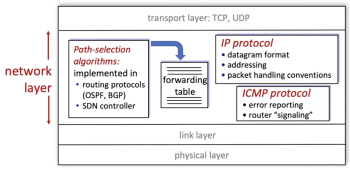



Figure 25: The network layer, notice how the IP protocol has nothing to do with the routing algorithm - that is a control plane function

## *Network Layer*

The network layer is implemented in each and every device connected to the internet and is really the glue that holds the internet together. It serves two functions:

- **Forwarding:** move packets from a router's input link to appropriate output line
- **Routing:** determine the route taken by packets from source to destination

The network layer has two sub-layers or planes:

- **Data Plane**: local, per-router function that determines how datagram arriving at router is forwarded to output port
- **Control Plane:** determines how datagram is routed across routers

Routers are the principal network devices in the network core, with a fairly simple job: examine the headaer fields in all IP datagrams passing through it and move them from input to output ports to enable transfer of the datagram along an end-to-end path.

*History and Autonomous Systems*

The internet is not really a network of end-systems - its a network of ISPs that have autonomous control over how they operate their share of the internet (i.e choosing what routing algorithms to use, etc). And so routers are organized into *autonomous systems*, with each AS consisting of routers that under the same administrative control. Each AS is identified by its globally unique autonomous system number. Routers within the same AS all run the same routing algorithm and each have information about one another.

Tier-1, Tier-2, Tier-3 networks.

*IP Addresses and Forwarding*

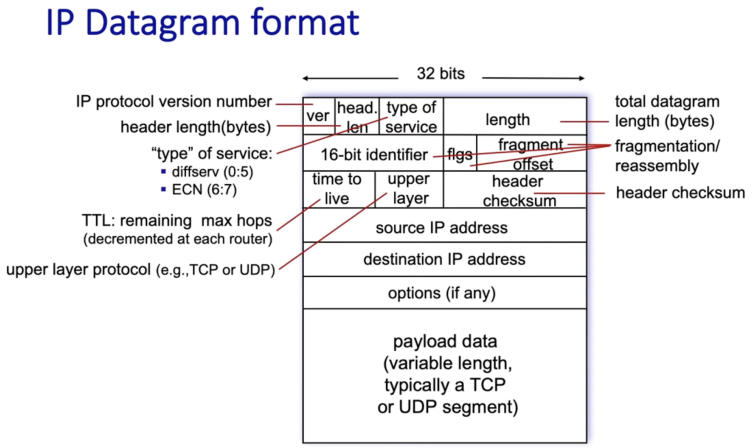



Figure 27: IPV4 Datagram, 20 bytes of IP overhead, 20 bytes of TCP overhead + application overhead

An IP address identifies **an interface** on the host, and not just the host. This is because a host connected to the internet can have multiple interfaces (WiFi, ethernet, etc).

Device interfaces that can physically reach each other without passing through an intervening router are in the same **subnet**. IP addresses are structured to identify the subnet a host belongs in, then identifies the host (with the lower order bits) in that subnet.

*A bit of IP history (pun intended)*

In the original IP addresses, first 8 bits represented the network and the remaining 24 bits identified the host. For example, 17.0.0.1 was the first host on the apple network. Note that first (i.e host address all 0s) and the last addresses (host address all 1s) is never assigned and used for different purposes. The last address is usually used to

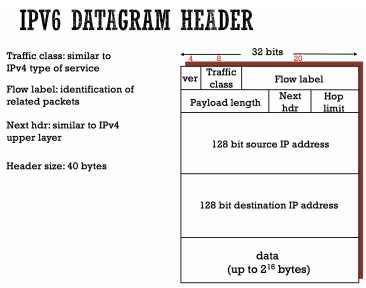



Figure 26: IPV6 Datagram

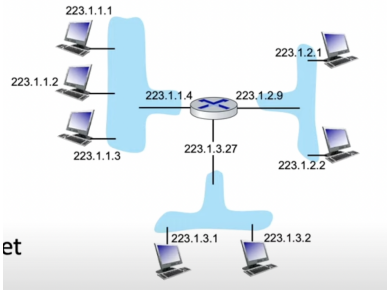
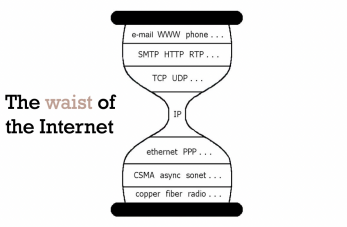



Figure 28: Subnets

An octet is RFC-speak for a byte - anciently different computers had different sized bytes, hence an octet is more precise.



broadcast messages to all hosts in a subent, while the first address represent a host connected to a network but not yet assigned an IP.[4]

Because we **underestimated** the number of networks we would need (no one expected that one day, everyone would own multiple IPs and that there would so many networks), original IP addressing (described above) quickly became obsolete - 8 bits were not enough to represent all networks. And so, a new scheme was introduced - **Class Based Addressing**

But this wasn't enough, so the scheme was complicated further still, and gave rise to **CIDR**. The subnest mask /24 or in general $/x$ identifys how many top order bits are used in the subnet part of an IP. IP addresses in format $a.b.c.d/x$ are known as Classless InterDomain Routing (or CIDR) addresses.[5]

But this still wasn't enough, and so introducing IP$V6$ addresses. @TODO

[4] /31 address are exempt from this convention

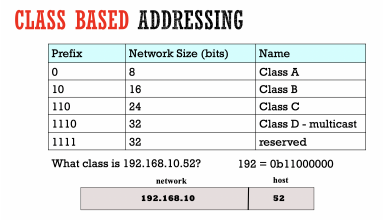



Figure 30: Class Based Addressing

[5] CIDR addresses are in dominant use now

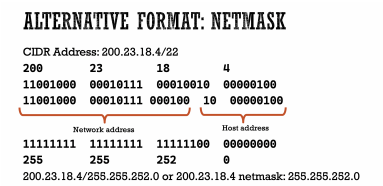



Figure 31: Alternate format of representing CIDR addresses

## *IP Address Ranges*

Now, we answer two questions:

- How does a host within a network get assigned an IP address?
- How does a network itself get assigned an IP address?

Let's tackle the first question. A host is either hardcoded an IP by the sys-admin (for eg. in .config for UNIX machines) or through the Dynamic Host Configuration Protocol (DHCP). A DHCP servers typically serves one router and all the subnets attached to that router.

Ok, now for the second question, how does a network get its subnet address? Well, it gets allocated a portion of its providers ISP's address space.

Notice that the rest of the internet really only needs the ISP's IP to access each of the 8 networks above. Route/Address Aggregation!

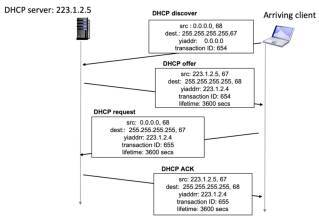



Figure 32: A simple DHCP interaction

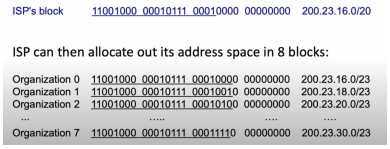



Figure 33: ISP dividing its address space for different networks

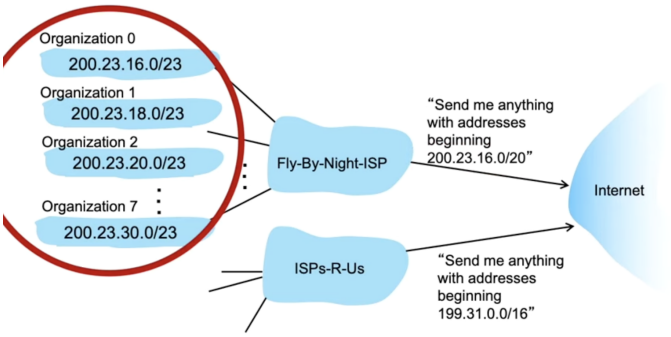



Figure 34: Route or Address Aggregation

Its never this organized in the real world. Organizations might jump ship and choose a different ISP, while still keeping their original IP address ranges. This is why the internet forwards packets based on the **longest prefix match**.

Ok, so how do ISPs get their IP address? Well, its a complicated process of going through registrars and so forth.

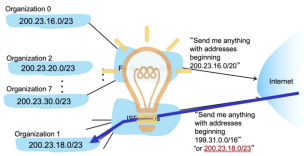



Figure 35: Longest Prefix Matching

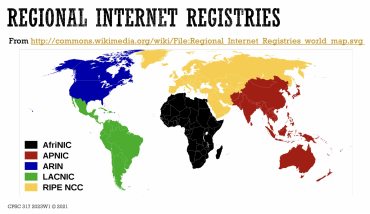



Figure 36: Regional Internet Registeries

## *Routing*

IGP vs EGP This is the problem that falls into the control-plane of the network layer. Routing algorithms can be clasified into the following.

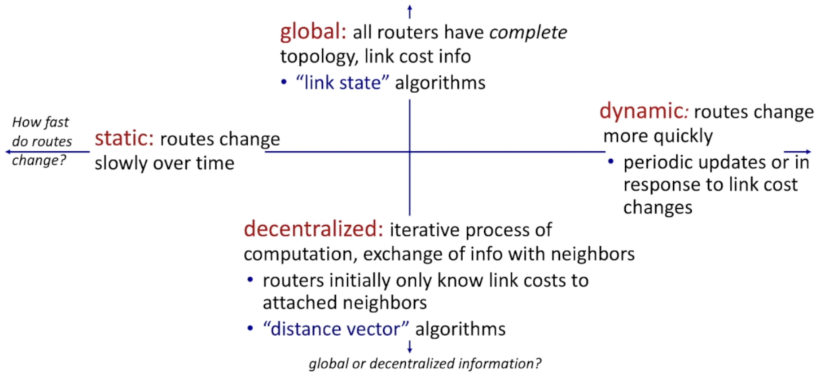



Figure 37: Classification of routing algorithms

## *Link State Routing*

Each router has the link information for every other router. Then just uses Dijikstra's algorithm to determine shortest path from it to every other router.

## *Distance Vector Routing*

From time to time, each node sends its own distance distance vector estimates to its neighbours.

## *BGP*

Along with IP, makes up the two most important algorithms in all of computer networking. Its the "glue" that holds internet together. BGP allows a network to advertise its existence to the rest of the internet. It provides an AS

- a means to obtain network reachability information from neighbouring ASes (eBGP)

- determine routes to networks outside AS based on reachability information and policy
- propogate reachability information to all internal AS routers (iBGP)
- advertise to neighbouring networks destination reachability information, again based on policy

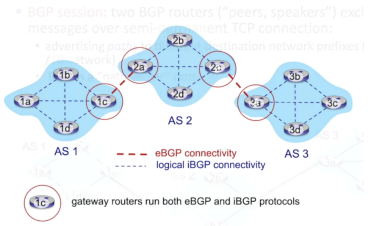



Figure 38: eBGP, iBGP

## *Network Address Translation (NAT)*

We are out of IPv4 addresses. For a few years now. Yet, instead of switching over to IPv6, we play tricks and pretend we have more addresses than we actually do. We can do this becase:

- Do we really need globally unique IP addresses?
- Other hosts *rarely* initiate a direct connection with you - most connections target servers "out there" in the internet
- So, maybe several hosts could share a single IP address

The idea is simple. Hosts on a private network each get assigned an internal IP non-routable address. These hosts are then serviced by a NAT router, whose public IP address is used to reference every host in this network. The NAT router takes care of the mappings by maintaining a forwarding table.

| Internal IP | Internal Port | Remote IP | Remote Port | Protocol | NAT IP | NAT Port |
|---|---|---|---|---|---|---|
| 192.168.0.2 | 9999 | 123.4.5.6 | 80 | TCP | 200.1.2.3 | 12345 |
| 192.168.0.1 | 6445 | 12.34.5.6 | 53 | TCP | 200.1.2.3 | 12346 |
| 192.168.0.1 | 6553 | 1.23.45.6 | 2628 | UDP | 200.1.2.3 | 12347 |
| ... | ... | ... | ... | | ... | ... |

Figure 39: NAT forwarding table

But...what if a connection comes from the outside[6], or if the remote host too is behind a NAT (its reply will have the NAT address as the sender) - each of these cases would not match any entry in the forwarding table. We could resolve this issue by manually adding a fixed rule - say if a connection comes to address X at port Y, forward to internal host Z. Another approach is to employ UPnP (Universal Plug and Play). This is a protocol that allows host to communicate to the NAT router what traffic to forward to them[7].

NAT has been controversial - a network-level device mucking around with IP addresses and port numbers disobeys the layered

[6] known as the NAT traversal problem

[7] Nasser Hussein's video on UPnP shows you can example of doing so

responsibilities of networking, and a purist might say, let's just bite the bullet and switch to IPv6.

## *Link Layer*

Hosts and routers are nodes in the graph that is the internet. The edges connecting these nodes are the "links" we now talk about. These can be wired or wireless or connections over a LAN (switches). Link layer packets are known as frames and encapsulate a datagram. The primary resposibility of the link layer is to transfer a packet from one host (more precisely, interface) to another over a phyical medium. To do so, they also need to manage shared access to a medium - media access control.

There are two broad categories of links:

- Point-to-point: direct link between two interfaces
- Broadcast links: shared between multiple links

The link layer makes use of MAC addresses to identify interfaces.

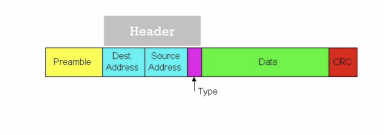



Figure 40: Link layer frame

### *MAC vs IP*

- IP is 32-bit or 128-bit, MAC is 48-bit
- IP addresses must be unique globally, while MAC must be unique in the network
- MAC addresses are burned into the ROM of the adapater - doesn't change while IP addresses depends on which subnet the host is attached to
- Notice that MAC addresses share no relationship with another - unlike IP where addresses on the same network share their prefix, this means we cannot use any from longest prefix matching when forwarding frames using MAC addresses

### *Error detection and correction*

- Single bit parity check: add a parity bit to maintain even parity
- 2-d partiy check: add a parity bit to each row, column and an overall parity bit, can correct single bit errors and detect upto 3-bit errors
- Checksums, such as the internet checksum
- Cyclic redundancy check (CRC): used to detect burst errors, which are rather common in practice...

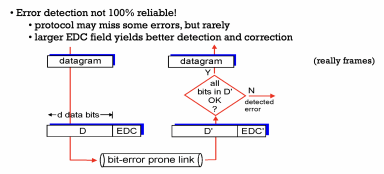



Figure 41: EDC pipeline

- parameterized by constants G, r - where G is the generator, an arbitrary bit pattern and r+1 is the length of the generator
- Some generators are better than others - given in the CRC standard
- When the sender wants to send D, they choose r CRC bits, R, such that <D,R> is exactly divisible by G modulo 2
- The receiver knows G, divides <D,R> by G, if non-zero remainder an error has been detected
- The computation of R and derivation of G can be implmented quite quickly in hardware

*Access control and ARP*

Half-duplex links are not uncommon at the link layer. If a link is half-duplex, how does a potential sender know when the link is busy? They listen. Listening is also used to detect "collision" (when two or more senders try to use the link at once). If a collision is detected or the link is busy, the senders wait a random amount of time before trying again. This kind of a protocol for access control is referred to as CSMA (Carrier sense multiple access), and it matches our human conventions for conversation. CSMA/CD adds collision detection and aborts sending if a collision is detected (again matching the human convention of conversation). The wait time before trying again is usually "binary exponential backoff" - where we choose a random number in the range $1 - 2^n$ where $n$ is the number of attempts to send that have failed due the link being busy or a collision.

Another approach to access control is "turn-based" access control where the senders take turn using the link and pass their turn if they don't need to use the link at the time. This type of a system could either be implemented in a centralized or decentralized fashion. In the centralized control version, the single control node polls every sender to see if they want a turn - there is a way for senders to signal that they would want to use the link. This is used in WiFi connections. In the decentralized version, nodes pass around a token and only the sender with the token may use the link. This introduces extra complexity - what happens if the node with the token breaks. Or what if a new token is generated even though the old one is still valid?

In general, CSMA/CD will use the network resources efficiently when there is just one host sending data - there will be no collisions and the host can send data at full speed. In the case with many senders, turn-based access control is a lot more efficient - there will rarely be a sender who wouldn't have data to send on their turn.

Now let's talk about switches - switches provide a full duplex connection between nodes in a LAN and also provide broadcast

functionality. A switch contains a forwarding table with 3 columns - MAC address, interface, time.

| MAC Address | Interface | Time |
|---|---|---|
| 01-02-03-04-05-06 | 1 | 8:23 |
| 02-03-04-05-06-07 | 1 | 7:56 |
| 03-04-05-06-07-08 | 2 | 7:59 |
| 04-05-06-07-08-09 | 3 | 8:01 |

Figure 42: Switch forwarding table

- Each time a fram is received on interface I, the switch looks at both the source and destination MAC addresses - if the source address appears in the table, update the interface the time, else add an entry with the MAC address, interface and time
- That was the "learning" phase. Now we need to forward the frame - if the destination MAC address doesn't appear in the table, we forward to every interface except the one on which we received the frame.
- If the destination MAC address has an entry in the table and its associated interface is I, discard the frame, else send it to I'
- If the destination address is not in the forwarding table, broadcast (except to the interface it just came on)
- In addition, we never put the broadcast address - $FF-FF-FF-FF-FF-FF$ in the table and delete entries whose time is too long ago ("too long ago" is defined by the admin)
- Again, we need to also check the CRC to ensure the frame has not been corrupted. Notice that in reality, interfaces are fixed and all that really needs to be updated is the timestamp.

Now how do you get the destination host's MAC address? Notice that a node knows it own MAC address, and can query for the destination's IP using DNS. So, given the desitination IP, we can the Address Resolution Protocol to get the MAC. There are two cases to consider:

1. The nodes are on the same LAN: in which case and ARP query is broadcast and MAC addresses are exchanged
2. The nodes are on different subnets: the sender can check the destinations IPs prefix to determine if its on the same subnet or not. If it is not, it just forwards the frame to its default gateway router (notice it knows the default gateways IP via DHCP and its MAC via ARP). The router then creates a new frame forwarding it to the next router and so on until it reaches the destination subnet and eventually the destination node.

*Physical and link layer issues*

@TODO

## *Security*

The principles of security in networking:

- Confidentiality: only relevant parties should be able to understand the message
- Authentication: parties must know if the data is being sent by a trust source
- Message integrity: data should not be altered
- Access and availability: services must be accessible and available to users

  What can Trudy do?

- Eavesdrop
- Alter messages
- Impersonate
- Hijack
- DoS

  In comes encryption. How can Trudy break an encryption scheme?

- Cipher-text only attack: Trudy only has access to the cipher text and either brute forces or does a statistical analysis
- Known-plaintext attack: Trudy has some cipher-text with its plain-text and can try to extrapolate from their
- Chosen-plaintext attack: Trudy can encrypt any plaintext, i.e has access to encryption key and can try to work her way backward

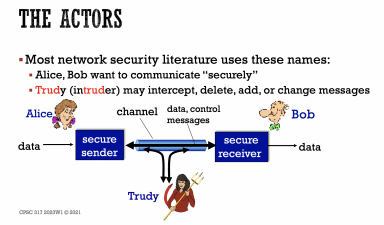



Figure 43: Network secuirty terminology; Alice, Bob and Trudy