Computer Networks

Lecture 1 Introduction to protocols and protocol layering

This course covers the protocols used in the internet.

What is a protocol?

It is a set of rules for transmission and formation of data.

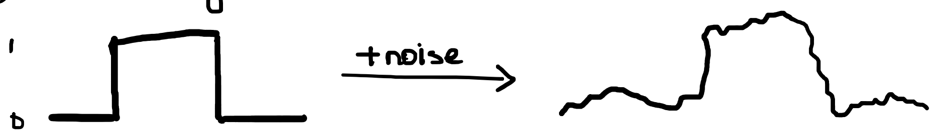
Textbook: Peterson & Davie - Computer Networks: A System's Approach (6 ed.)

The internet started out in the late 1960s as a government project ARPANET.

In different places there were different types of media (wired/wireless) and applications (telnet/rlogin/FTP). The internet is a network that works despite the massive amount of diversity. How would one even design this?

Say you have just two people A and B. You want a way to communicate data. We usually convert it into bits and send the signal across the wired or wireless medium.

However, there might be (electronic) noise in the medium.



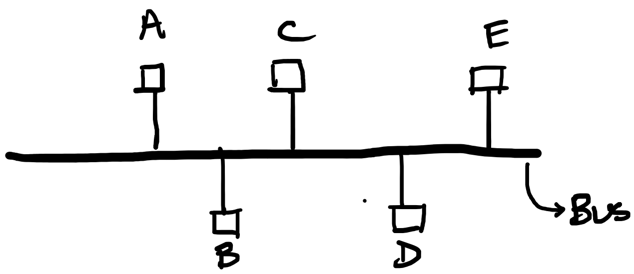
do we interpret this as a 1 or a 0?

The noise is higher in wireless media.

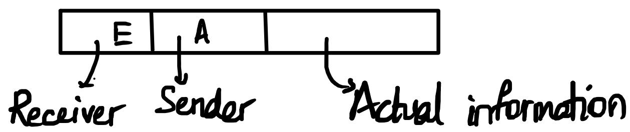
Physical Layer

Denoising, signals and other similar things come under the physical layer of the network.

Now take a slightly more complicated situation where you have 5 people.
A simple solution is to have a bus like so:



Say A wants to talk to E. Then A should be able to indicate that the message is meant for E. If two-way, then they should also be able to identify themselves.



What if two pairs, say A→E and B→C want to communicate at the same time? The signals may get superimposed and both the informations completely garbled.

Medium Access

This issue is known as medium access. see: MAC address (Medium Access Control) → unique for every device, hardcoded in hardware

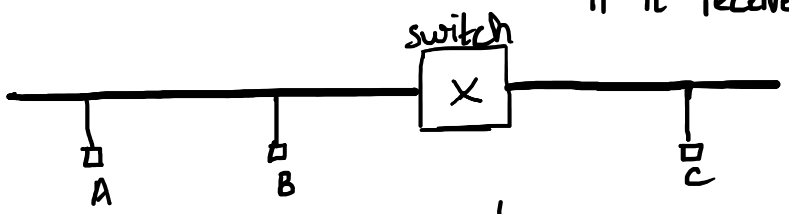
Data Link Layer

Things like medium access and framing come under the data link layer. (DLL)

framing → demarcating where the information begins/ends.

Repeater

Now say we have 1000 people. The medium access problem grows massive if we use a single bus. Since the bus will be so long, we would also need a repeater (an amplifier) to ensure the signals don't die out. There would also be a lot of interference. We could use a switch instead:

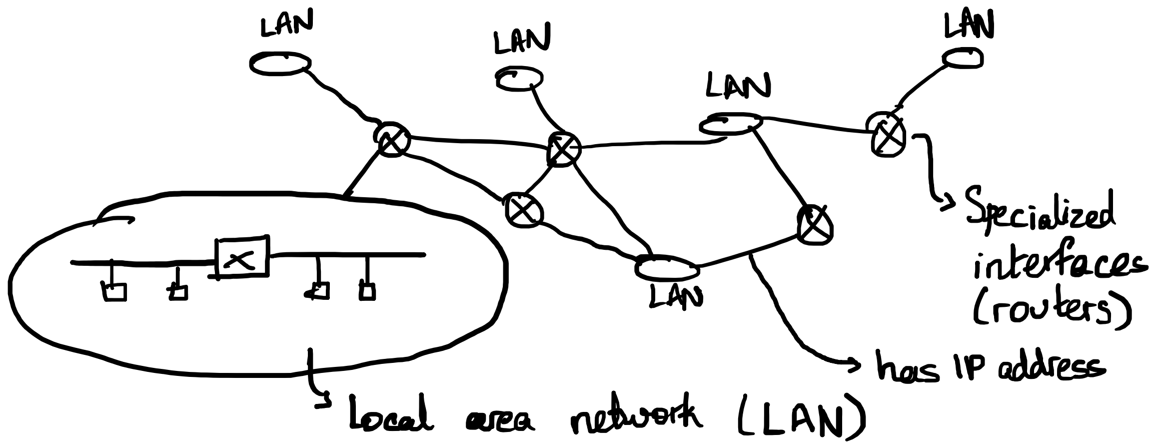


if it receives a message from A→B, then it ignores it. If say A→C, it repeats.

basically a selective repeater

This comes under the data link layer as well.
A switch used in this scenario is called a Layer-2 switch. (L2)

Now say I have a million people.



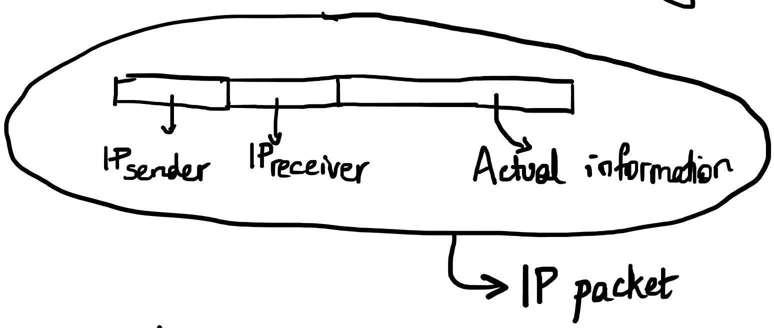
We need some common protocol so that the diversity does not cause issues

network protocol

IP protocol is the common one

Network Layer (IP)

Each of these interfaces will have IP addresses (one for each link)
If someone wants to communicate, they can use the IP address.



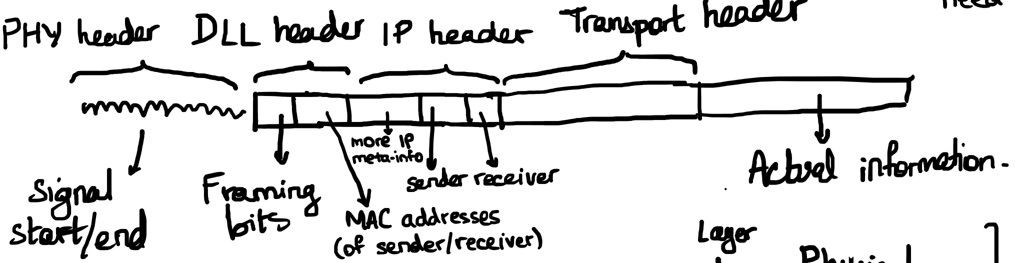
Both the sender/receiver device and the router interfaces have IP addresses.

see: NAT

Originally, not enough bits were allocated for IP, so they need not be unique.

see: private and public IP address

So overall,



Layer	
1	Physical
2	Data Link
3	Network (IP)
4	Transport
5	Application

5 layer model of protocol stack

Routers

Routers are Layer-3 switches.

The router's job is to see the IP header and decide where to forward it (if at all). How do they work together to figure this out?

→ There are many protocols for this.
BGP, ISIS, OSPF
autonomous system level routing

Layer-3 protocols take care of routing.

Routers have queues to temporarily buffer packets when the input rate is higher than the maximal output rate.

Packet loss

If the queue gets filled, then packet loss occurs (packets are just thrown away).

We do retransmission of lost packets. In ARPANET then, the retransmission starts from the original sender.* This isn't always the best solution, but ARPANET was a defense network so it makes sense. (This relic is even in today's internet?)

*as opposed to the most recent router somehow (some more stuff would need to be implemented for this)

So loss must be detected and the message must be retransmitted until the receiver receives it.

→ To know this, there should be a returned acknowledgement packet from the receiver. If it doesn't come for a long time, package lost?

when we send a lot of packets, we needn't worry if a few acknowledgements are lost

Transport Layer (TCP)

This layer is known as the transport layer. The common protocol used is TCP. (In ARPANET, the rudimentary UDP was used)

Some other models use more than 5 layers. (Eg. OSI-7 layer)

layer		handled by
Application	→	app
Transport	} →	OS level - TCP, IP
Network		
Data Link	} →	NIC - Network Interface Card (stuff like 2G/3G/...)
Physical		

Headers are added on as the message moves down.

The message is transmitted to the base station
↓
only looks at network/DLL/PHY in general

Repeater only looks at PHY.
L2 switches only look at DLL/PHY.

Why is protocol layering done?

+ The complicated networking problem is decomposed into smaller, more manageable ones.
+ Modular design so you can change individual protocols easily.
- It may help if TCP knew where the lost information was dropped. The lack of interaction between distinct layers makes this difficult.

How well does it perform?

Throughput
Latency

The throughput is the average data rate from the sender to the receiver. (~ 64 kbps for voice call)

The latency is the delay of a message across the network. (~100 ms of one-way latency)

The round trip time (RTT) is the time for a message to go from sender → receiver → sender.

→ Grouped together as Quality of Service (QoS).

Quality of Service (QoS)

Say video streaming → throughput ~1 Mbps needed
latency doesn't really matter, couple of seconds would also be fine.

When we build a network, we specify the (desired) QoS and then design the protocols. We don't have this freedom when using the internet however, we can only interact with the application layer meaningfully.

Throughput is the number of bits that can be pushed from one end to another.
Latency is the time taken. (Property of network, not application)

Lecture 2 Routers and Traffic

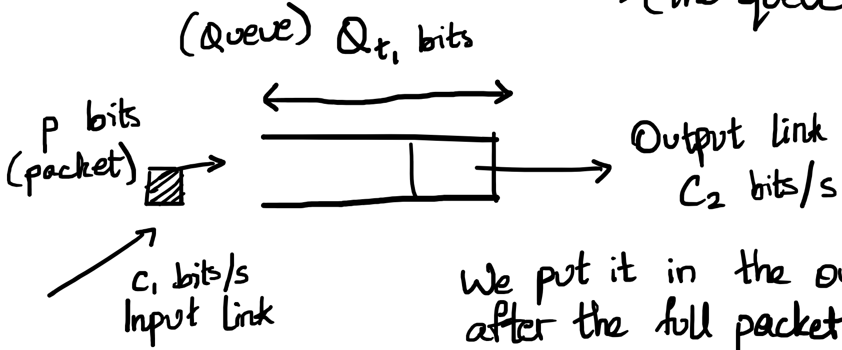
Say file transfer → the higher the throughput, the more preferred.
if there is no such "critical" required throughput, it is said to be best effort. No QoS guarantees from the network.

The internet is designed as best effort.
We are trying to move away from this and guarantee QoS, but it's difficult. (Mobile–mobile communication does guarantee QoS)

Bandwidth

In a router, the maximum data rate of transmission through a certain link is called bandwidth (link capacity).
If you have multiple people using the same link, they are competing.
The protocols don't usually guarantee/reserve any bandwidth.
(some protocols do but making every router guarantee stuff is non-trivial)

If people are competing, throughput decreases.
latency increases.
↳ (the queue is longer)



Store & Forward

We put it in the output queue only after the full packet is received.
"Store & Forward" property.

Cut-Through

In Cut-Through switching, we send it to the output queue immediately.
This may waste resources though (what if the packet is corrupted?).

Say t_0 → time of first bit at input
t_1 → time of last bit at input
Then $t_1 = t_0 + P/c_1$. (In S&F, we wait till t_1.)

$t_2 \rightarrow$ time of first bit at output.

$$t_2 = t_1 + \frac{Q_{t_1}}{c_2}$$

$t_3 \rightarrow$ time of last bit at output

$$t_3 = t_2 + \frac{P}{c_2}$$

$$\Rightarrow t_3 = t_0 + \underbrace{\frac{P}{c_1}}_{\text{Waiting due to Store \& Forward}} + \underbrace{\frac{Q_{t_1}}{c_2}}_{\text{Queuing delay}} + \underbrace{\frac{P}{c_2}}_{\text{Transmission delay}}$$

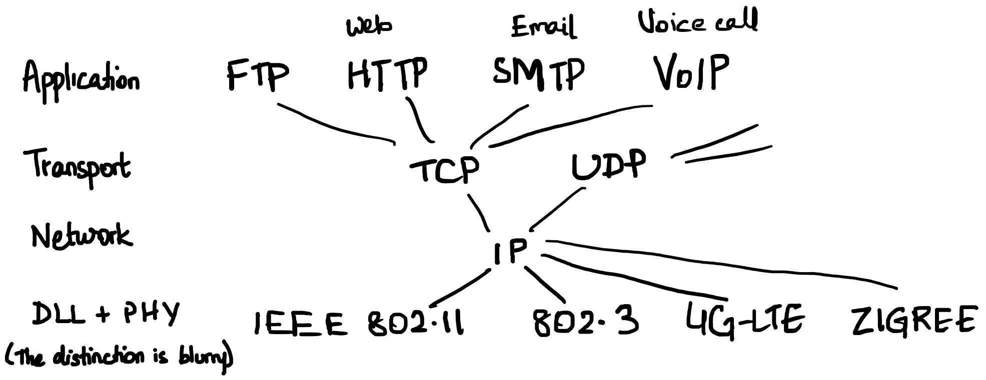
Queuing delay: Quite variable, depends on traffic.
↳ number of people competing.

Cross-Traffic

Cross - Traffic → Traffic other than your own packets.

Congestion Control

Congestion control → When the queue builds up at a router, "congestion" is said to occur. We want it to continue working somehow — the way this is handled by the protocols is congestion control. Mainly handled by Transport layer (TCP).
(if packets being dropped or RTT very high, back off a bit)
May also be handled by lower layers (DLL, PHY).*
If you want your own congestion control in the application layer, you might use UDP (which does nearly nothing special)



* if collision of signals, back off a bit and then retransmit. DLL has some stuff to do with retransmission as well

Since everyone uses IP, hourglass like figure.
Designed such that layers only need to talk to higher and immediate lower layer.

FTP HTTP
Port number ↑
TCP
Protocol field ↑
IP

We study them more in detail later

The port number decides which application to use.
The protocol field decides which application protocol to use.