

# 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.

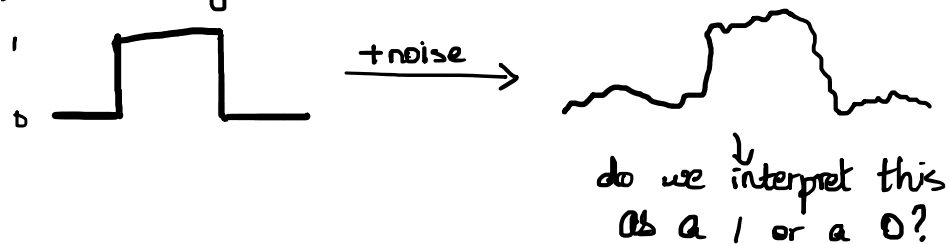
Textbook: Peterson & Davie - Computer Networks: A System's Approach (6 ed.)

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.

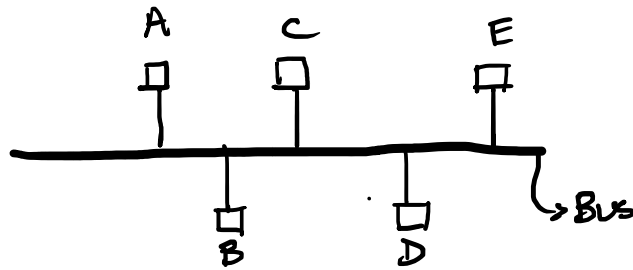
However, there might be (electronic) noise in the medium.



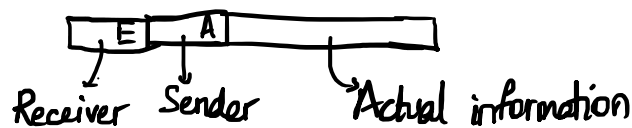
The noise is higher in wireless media.

Denoising, signals and other similar things are called 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  $A \rightarrow E$  and  $B \rightarrow C$  want to communicate at the same time? The signals may get superimposed and both the informations completely garbled.

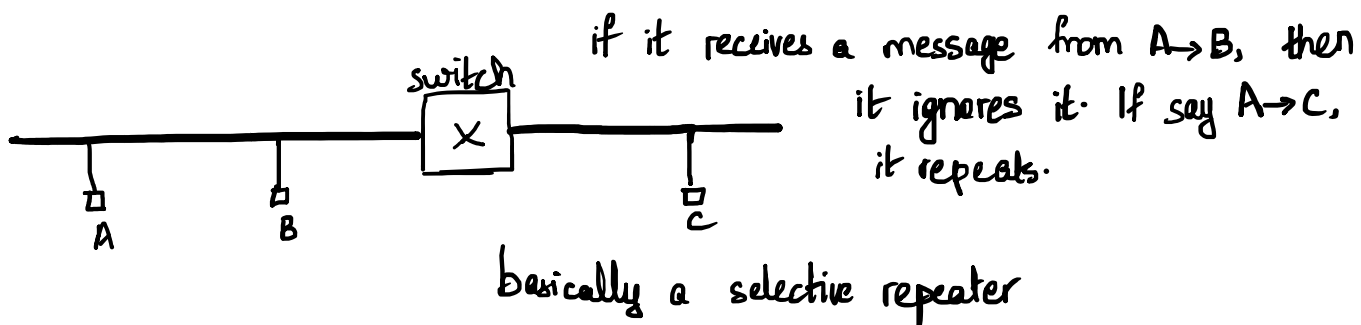
This issue is known as **medium access**.

Things like medium access and framing come under the **data link layer**.

(DLL)

↓  
demarcating where the information begins/ends.

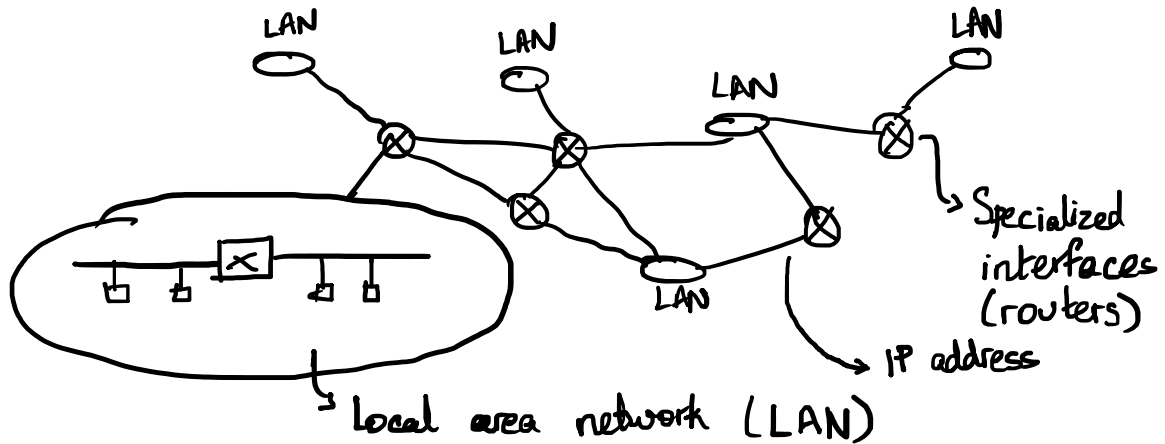
Now say we have 1000 people. The medium access problem grows massive if we use a single bus. Since the bus will be so long, we would also need a **repeater** (an amplifier) to ensure the signals don't die out. There would also be a lot of interference. We could use a switch instead:



This comes under the data link layer as well.

A switch used in this scenario is called a Layer-2 switch. (L2)

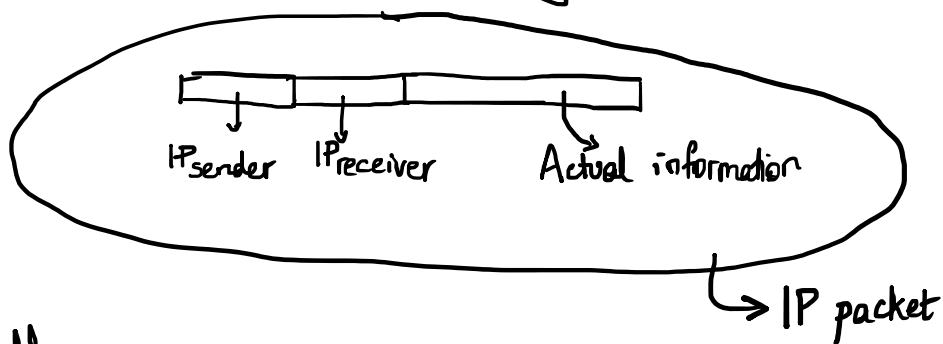
Now say I have a million people.



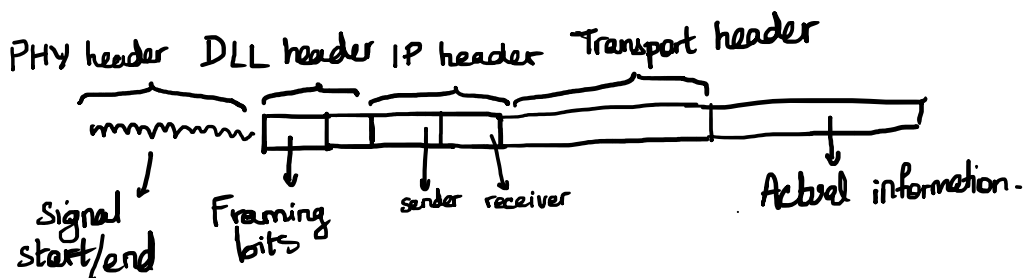
We need some common protocol so that the diversity does not cause issues

↗ network protocol  
↓  
IP protocol is the common one

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



So overall,



Layer		
1	Physical	} 5 layer model of protocol stack
2	Data Link	
3	Network (IP)	
4	Transport	
5	Application	

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's a lot of protocols for this.  
BGP, ISIS, OSPF.

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.

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

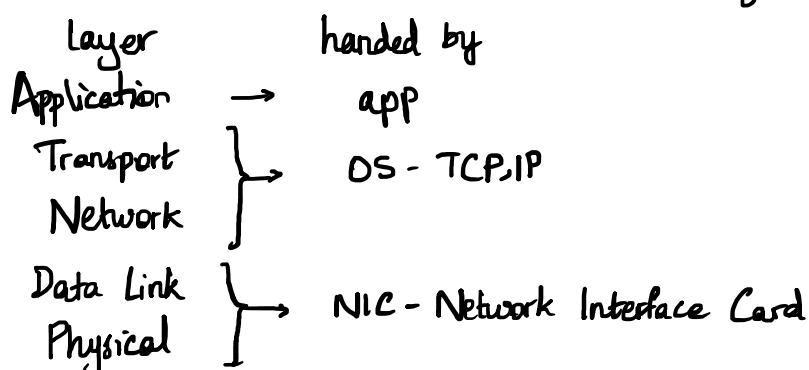
We do retransmission of lost packets. In ARPANET, then the retransmission starts from the original sender. This isn't always the best solution, but ARPANET was a defense network so it makes sense.

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?

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)



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 last information was dropped.  
The lack of interaction between distinct layers makes this difficult.

How well does it perform?

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)**.

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 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.