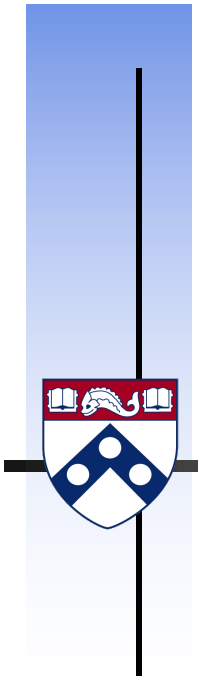


CIS 5530: Networked Systems

Network Performance

January 23, 2023

Agenda

- Network basics ✓
- Design goals of the Internet ✓
  - Taxonomy of networks ✓
  - Design goals (Clark '88) NEXT
  - Layering and the end-to-end principle
- Performance metrics

The goals of the Internet

http://ccr.sigcomm.org/archive/1995/jan95/ccr-9501-clark.pdf

"technique for multiplexed utilization of existing interconnected networks"

- Second-level goals:
  - Resilience/survivability
  - Heterogeneity
    - Different types of services
    - Different types of netwo
  - Distributed managemer
  - Cost effectiveness
  - Easy host attachment
  - Resource accountability

"This set of goals might seem to be nothing more than a checklist of all the desirable network features. It is important to understand that these goals are in order of importance, and an entirely different network architecture would result if the order were changed."

These goals were prioritized for a military network. Should priorities change as the network evolves?

The goals that started to fall by the wayside

Goal #5: Cost Effectiveness

- Many inefficiencies in the Internet
  - For small packets, headers introduce high overhead
  - End-to-end retransmission of lost packets leads to wasted bandwidth
- Arguably a good tradeoff as network speeds have grown rapidly
- Exception is wireless, where they try to correct for some of this

Goal #6: Ease of Attachment

- Anything with an IP stack can connect (hourglass model)
- Some amount of burden on end systems and programmers

- But once you get a working network stack…
  - A huge success!!

The "last" Goal #7: Resource Accountability

- Not prioritized at all
- Datagram networks make accounting tricky
  - The phone network has had an easier time figuring out billing
  - Payments/billing on the Internet is much less precise
- Convoluted business relationships
- Hot potato routing

What's Missing?

- Security
- Availability
- Accountability (the other kind)
- Support for disconnected/intermittent operation
- Mobility
- …

Agenda

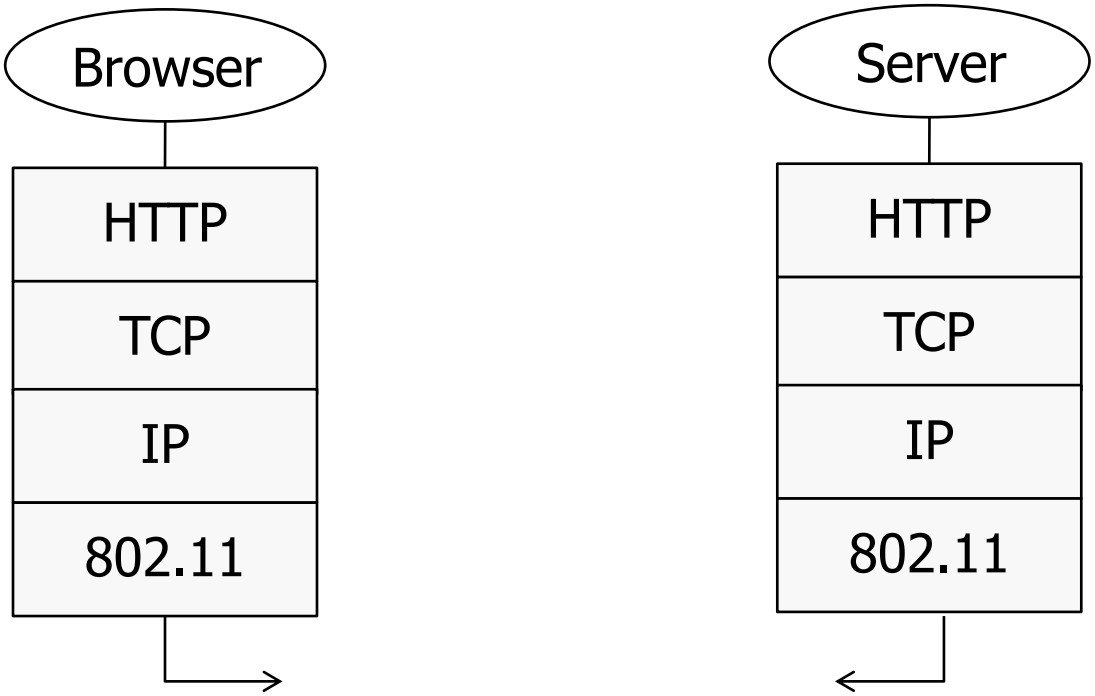
- Design goals of the Internet ✓
  - Taxonomy of networks ✓
  - Design goals (Clark '88) ✓
  - Layering and the end-to-end principle NEXT
- Performance metrics

Recap: Protocols

- An agreement between parties on how to communicate
- Defines the syntax of communication
- And semantics
  - "First a hello, then a request…"
- We will study many protocols
  - Exist at many levels (e.g., hardware and software)
  - Exist for many purposes (e.g., data transfer and routing control)
  - Defined by standards bodies like IETF, IEEE, ITU

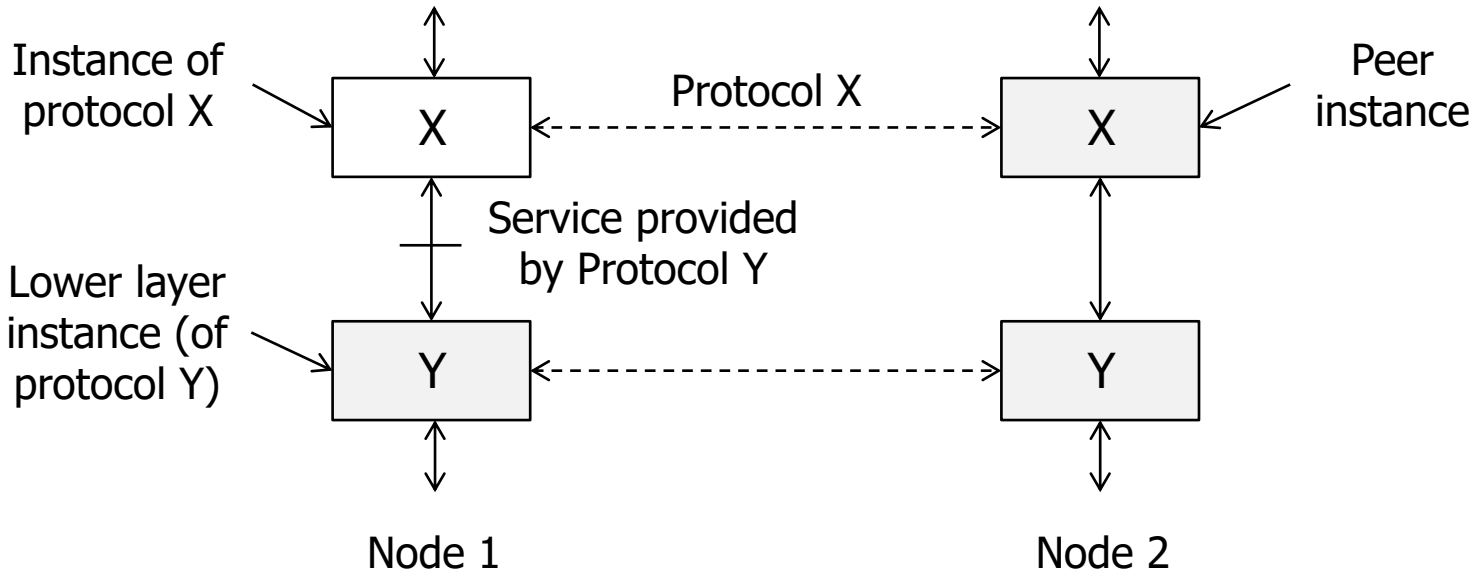
Layering

- The ability to mix and match protocols
- Set of protocols in use is called a protocol stack
- Ex:



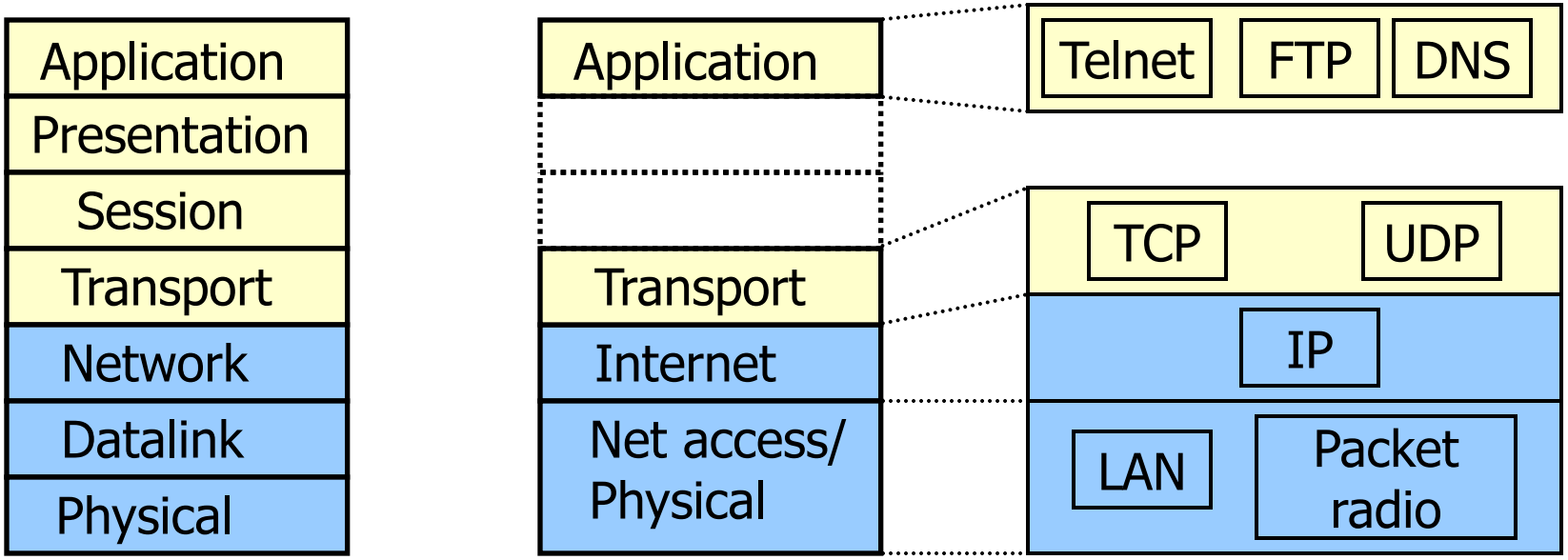
Protocols and Layers

- Protocols are horizontal, layers are vertical



OSI vs. Internet

- OSI: conceptually define services, interfaces, protocols
- Internet: provide a successful implementation



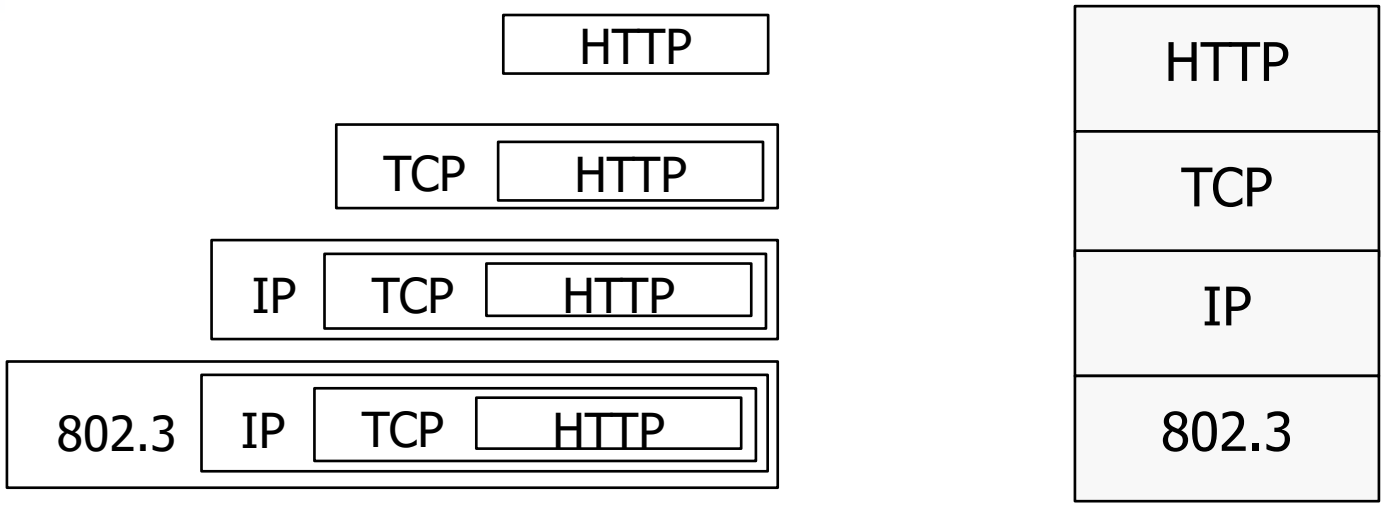
OSI (formal)

Internet (informal)

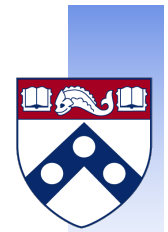
Encapsulation and (decapsulation)

- Encapsulation is the mechanism used to effect protocol layering
  - Lower layer wraps higher layer content, adding its own information to make a new message for delivery
  - Like sending a letter in an envelope; postal service doesn't look inside

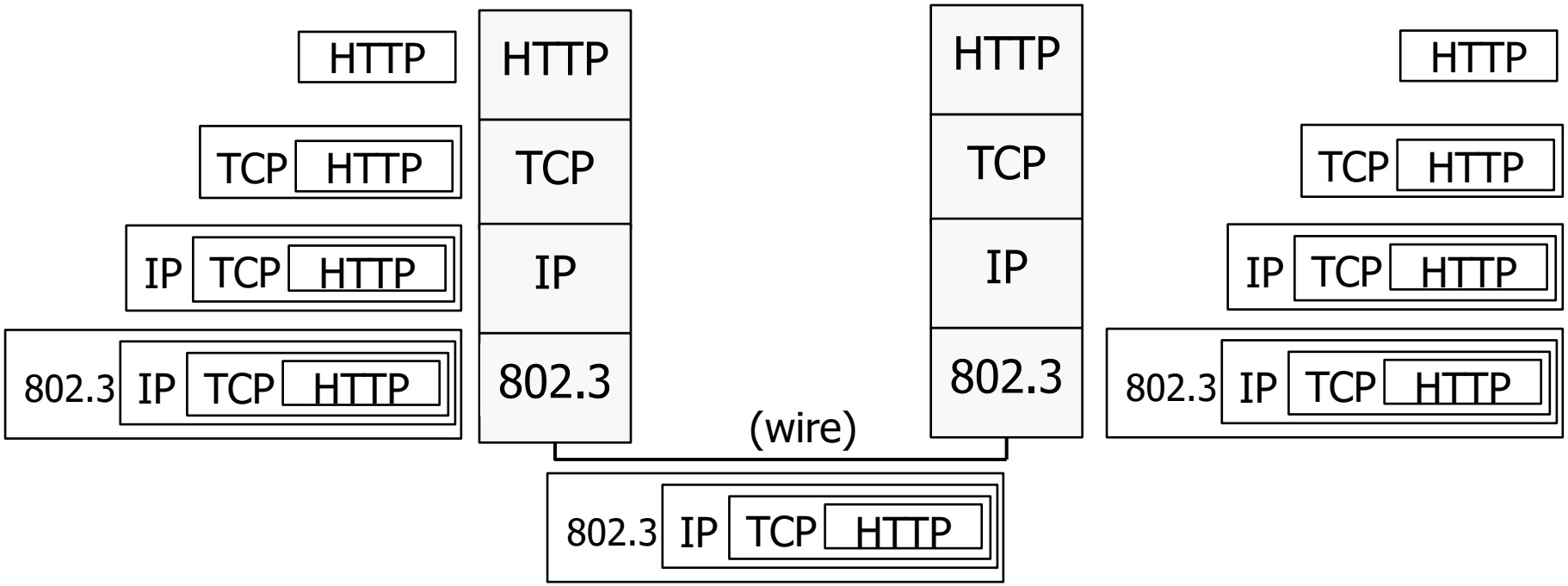
Encapsulation (Cont.)



- Message "on the wire" begins to look like an onion
  - Lower layers are outermost

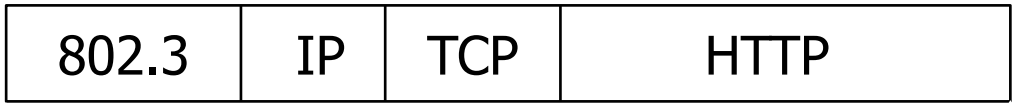


Encapsulation (Cont.)



Encapsulation (Cont.)

- Normally draw message like this:
  - Each layer adds its own header



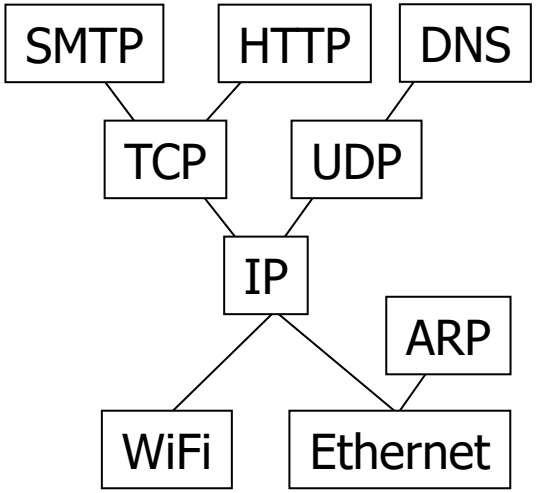
First bits on the wire

Last bits

- More involved in practice
  - Trailers as well as headers, encrypt/compress contents
  - Segmentation (divide long message) and reassembly

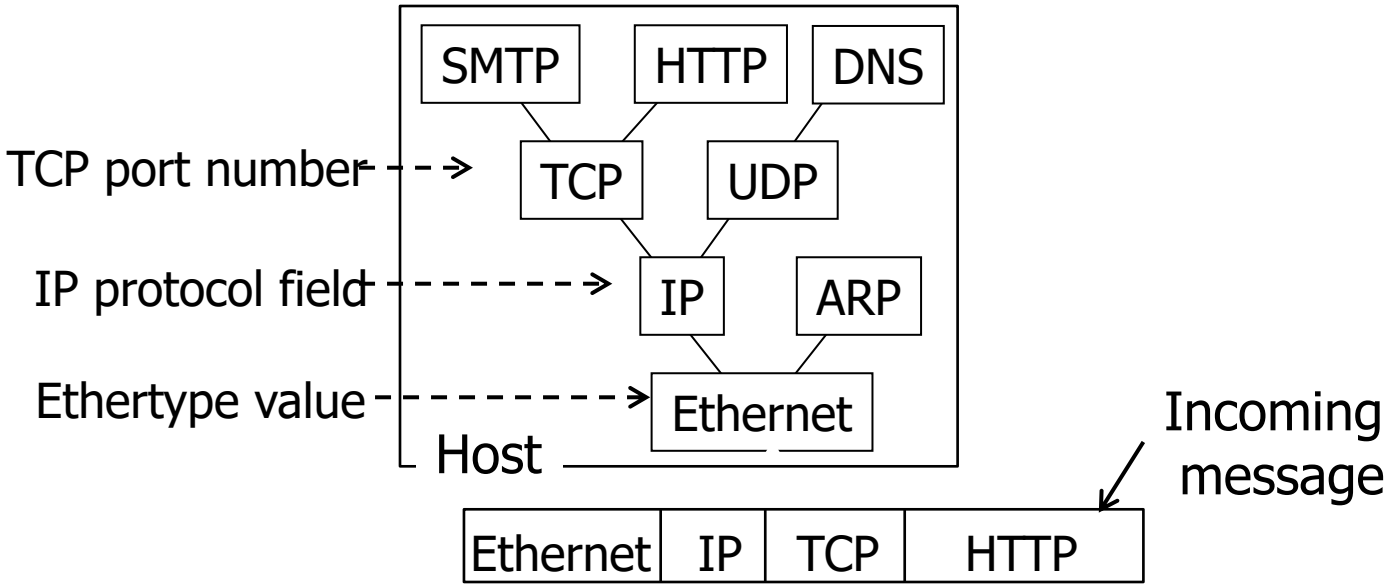
Multiplexing (and demultiplexing)

- Outgoing message must be encapsulated with the correct header
- Incoming message must be passed to the protocols that it uses



Demultiplexing (Cont.)

- Done with demultiplexing keys in the headers



Which Layer is Right for Me?

http://web.mit.edu/Saltzer/www/publications/endtoend/endtoend.txt

- Consider file transfer from A's hard drive to B's hard drive over the network
- Transient failures at many levels
  - Disk errors
  - Incorrect software (FS, FTP, network, …)
  - Hardware errors (CPU, memory, network, disk, system crash, …)

Solution Space

- Solution 1: Hop-by-hop reliability?
  - E.g. enhancing reliability of communication layer by checksums, sequence number checks, internal retry mechanisms.
- Solution 2: End-to-end reliability?
  - Store file with a checksum, transfer file, read transferred file back from disk, compute checksum, send checksum to originator to compare the two checksums.
  - If check fails, redo from beginning
- Implementing reliability at lower layers is insufficient.
  - Other stuff could go wrong (host fail, malicious router, FTP software failure) – does not reduce burden for end-to-end checks.
  - Don't over-engineer reliability if end-to-end checks are still required

When does it make sense to violate the end-to-end principle?

- Performance!
  - E.g. Lossy links (e.g. wireless). File transfer could keep retrying forever because one packet got lost!
  - Any other examples?

- Lots of gray areas:
  - Cost/engineering vs reliability tradeoff
  - Many applications may not need the functionality.
  - Reliable, in-order delivery, security, etc.
  - Why over-engineering the lower layers and make them pay?
  - Lower layers may not know best since it does not have information from higher layers

E2E rule of thumb in network design

- If hosts can implement functionality correctly, implement it a lower layer only as a performance enhancement
- But do so only if it does not impose burden on applications that do not require that functionality
- Other examples in the paper include encryption, duplicate suppression, etc.

Agenda

- Design goals of the Internet 
- Performance metrics 

Link-level Metrics

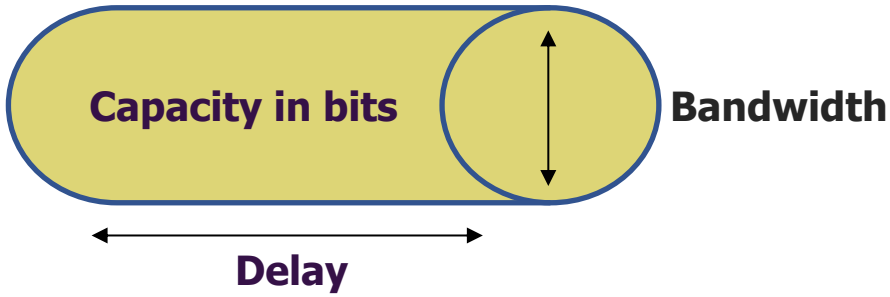
- Link bandwidth (capacity): maximum rate (in bps) at which the sender can send data along the link
- Propagation delay: time it takes the signal to travel from source to destination
- Packet transmission time: time it takes the sender to transmit all bits of the packet

Node-level Metrics

- Queuing delay: time the packet need to wait before being transmitted because the queue was not empty when it arrived
- Processing Time: time it takes a router/switch to process the packet header, manage memory, etc.

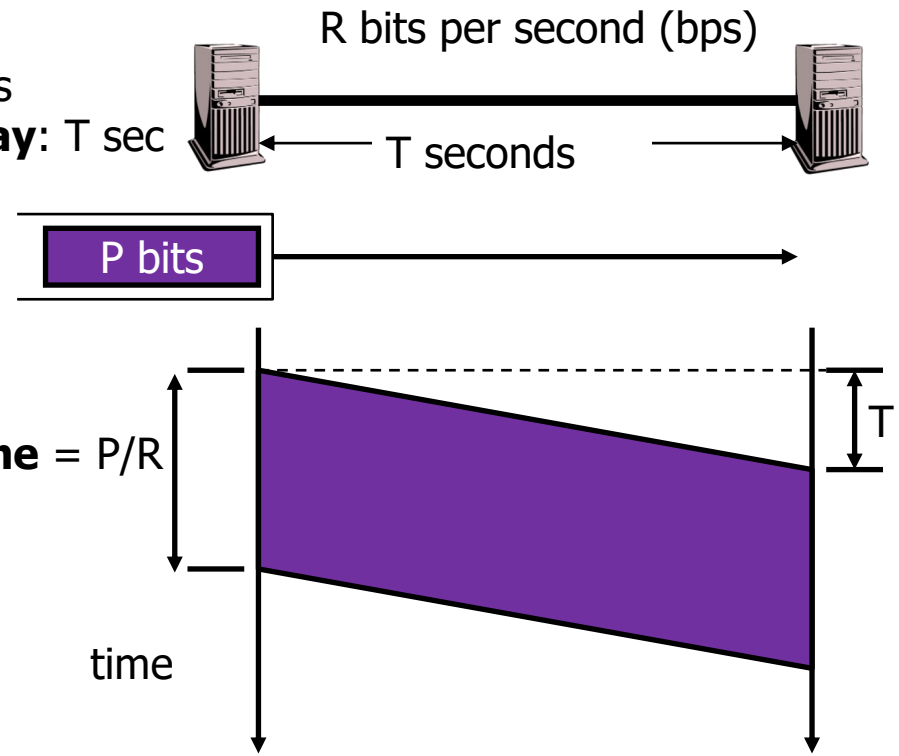
Throughput

- Throughput of a connection or link = total number of bits successfully transmitted during some period [t, t + T) divided by T
- Link utilization = (throughput of the link) / (link capacity)
- Bit rate units: 1Kbps = 103bps, 1Mbps = 106bps, 1 Gbps = 109bps [For memory: 1 Kbyte = 210 bytes = 1024 bytes]
  - Some rates are expressed in packets per second (pps)
  - Relevant for routers/switches where the bottleneck is the header processing



Latency: Sending One Packet

Bandwidth: R bps
Propagation delay: T sec



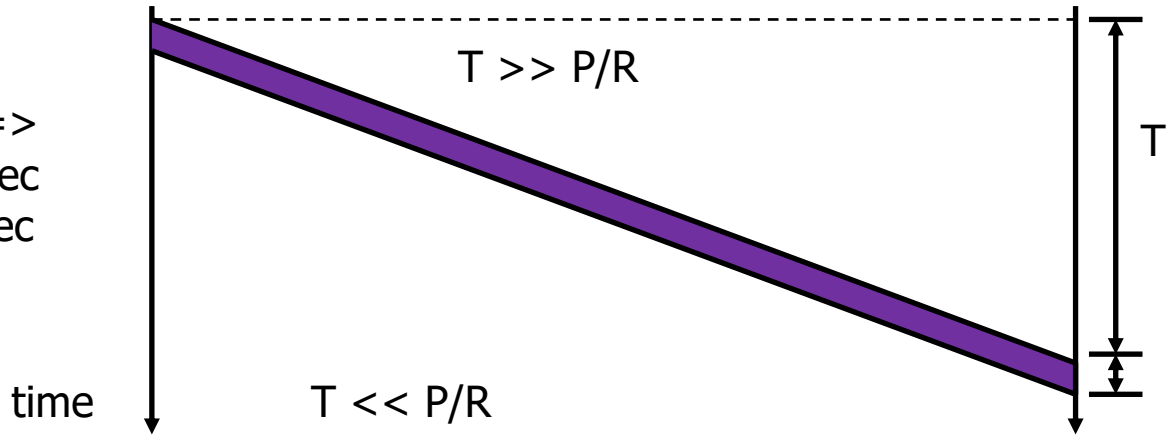
Transmission time = P/R

Propagation delay = T = Length/speed

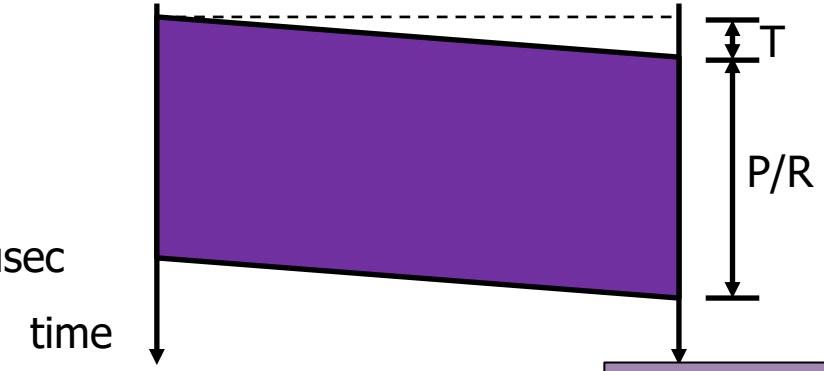
speed = 3.0 x 10^8 m/s in free space
2.3 x 10^8 m/s in cable
2.0 x 10^8 m/s in fiber

Sending One Packet: Examples

P = 1 Kbyte
R = 1 Gbps
100 Km, fiber =>
T = 500 usec
P/R = 8 usec



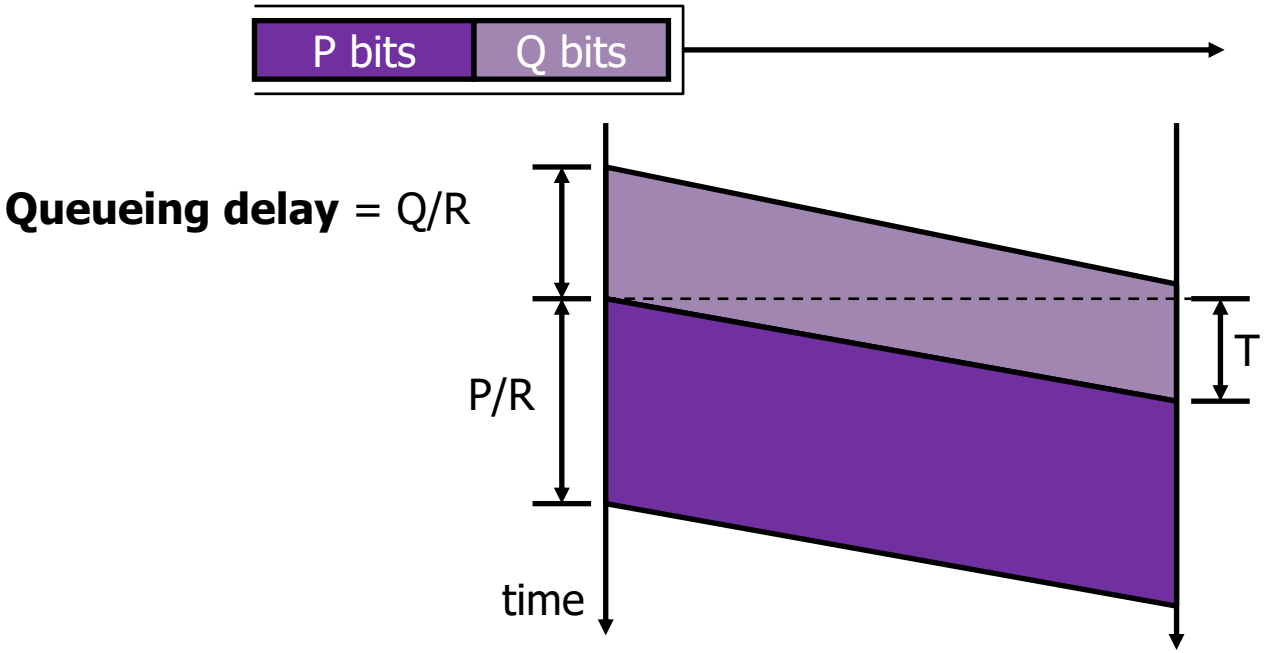
P = 1 Kbyte
R = 100 Mbps
1 Km, fiber =>
T = 5 usec
P/R = 80 usec



speed = 3.0 x 10^8 m/s in free space
2.3 x 10^8 m/s in cable
2.0 x 10^8 m/s in fiber

Queueing

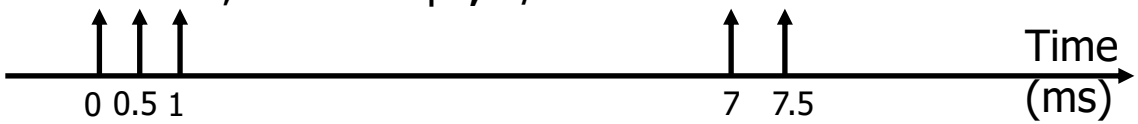
- The queue has Q bits when packet arrives
- Packet must wait for the queue to drain before being transmitted



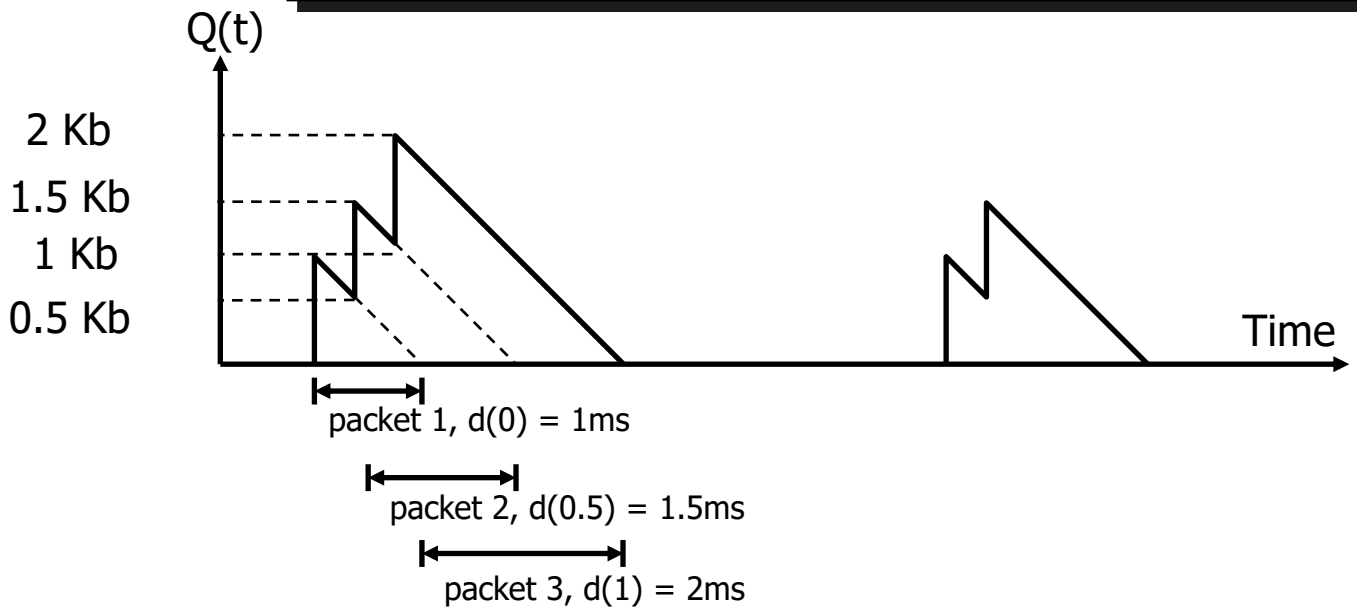
Queueing Example

P = 1 Kbit; R = 1 Mbps; P/R = 1 ms

Packet arrival



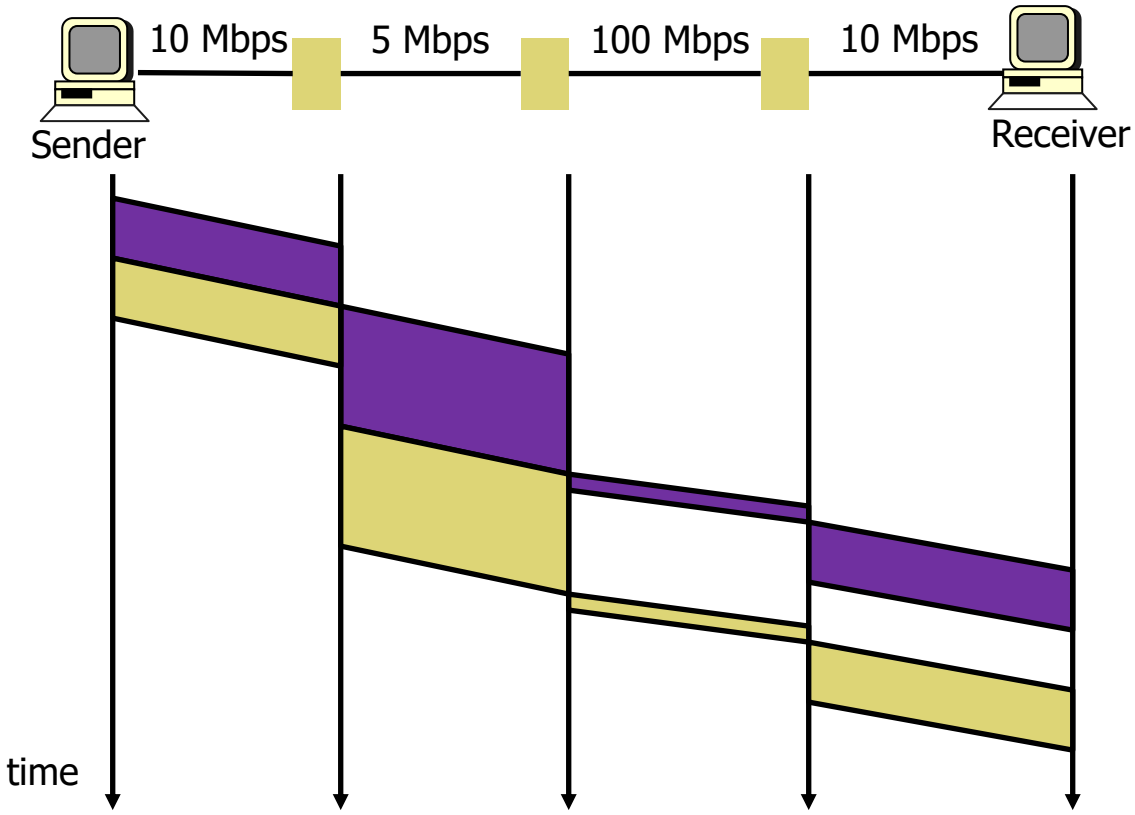
Delay for packet that arrives at time t, d(t) = Q(t)/R + P/R



Putting it all together

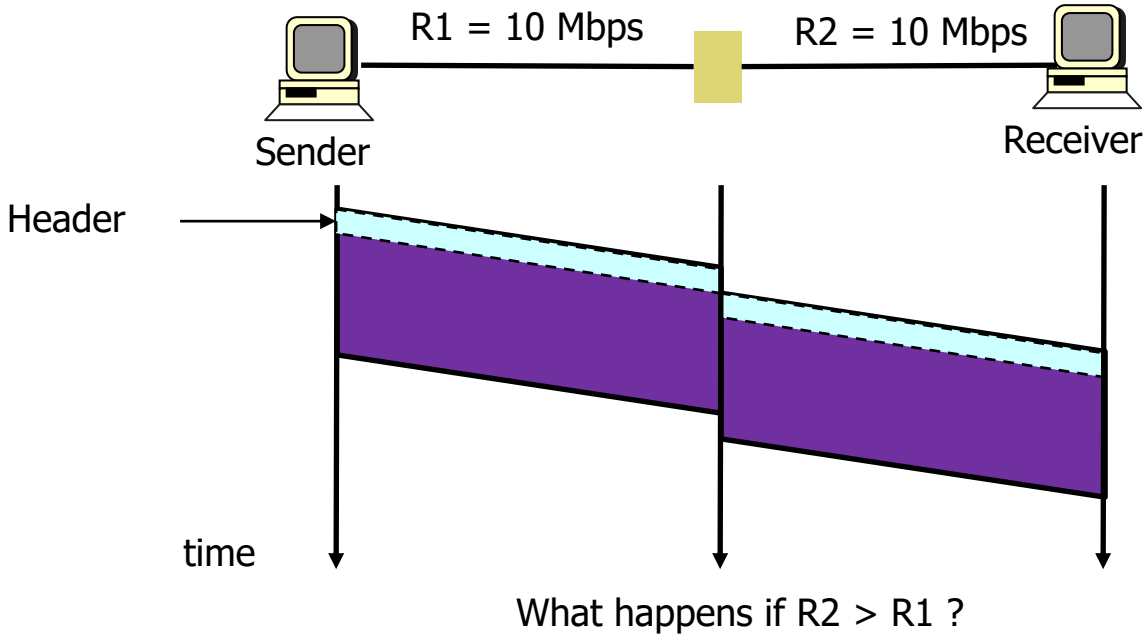
- Latency = Propagation + Transmit + Queuing Delay
- Propagation = Distance / Speed of Light
- Transmit = Packet size / Bandwidth
- Queuing Delay = Queue Length / Bandwidth

Example: Store and Forward



Example: Cut-Through Switching

- A packet starts being forwarded (sent) as soon as its header is received



Other Delay-related Metrics

- Delay (Latency) of bit (packet, file) from A to B
  - The time required for bit (packet, file) to go from A to B
- Round-Trip Time (RTT)
  - Two-way delay from sender to receiver and back

Other Delay-related Metrics

- Jitter
  - Variability in delay (more on next slides)
- Bandwidth-Delay product
  - Product of bandwidth and delay = "storage" capacity of network
  - For efficient resource usage: keep the pipe full. Affects TCP (reliability delivery) buffer size.

Jitter

- Informal definition: change in latency from packet to packet

- Why measure jitter?
  - Delay sensitive voice and video traffic over IP

Jitter

- Two aspects of temporal performance:
  - Latency: delays in real-time conversations
  - Jitter: affects service quality
    - Solution: Buffer data flow
    - Excessive jitter: buffer overflow => dropouts in audio stream or choppy video display
- What causes jitter?
  - Router: queuing delays, congestion
  - Links: failures leading to alternative paths

Measuring Jitter

- Jitter metrics:
  - Difference between the forwarding delay of two consecutive received packets belonging to the same stream (RFC 4689)
  - Average jitter is defined as the average value of the jitter of consecutive packet pairs.
  - Cisco IP-SLA: inter-packet delay variance

Measuring Jitter

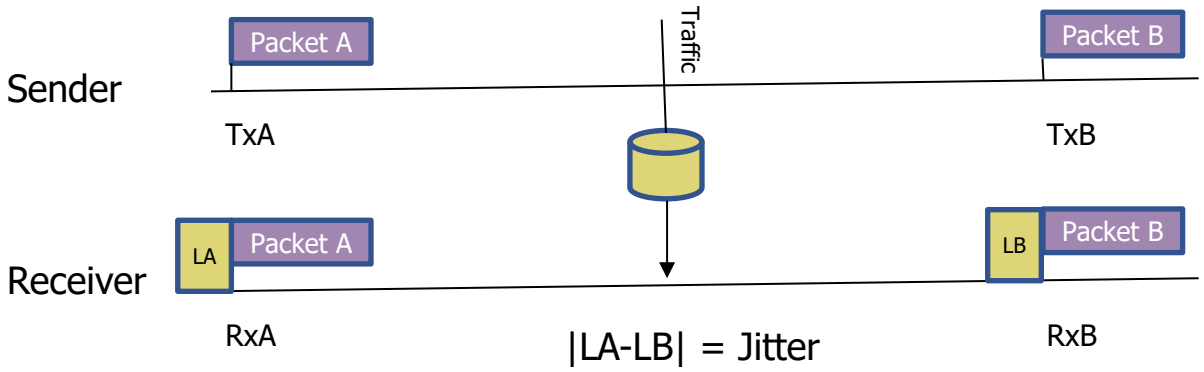
- When two packets (packets A and B) are sent through a network:
  - Packet A takes 15 ms to traverse the network.
  - Packet B takes 18 ms to traverse the network.
  - The difference in latency between the two packets in the pair is 3 ms.
  - Jitter = | 15 – 18 | = 3 ms.

Measuring Jitter

- Jitter can be expressed as:

$$|(RxA - TxA) - (RxB - TxB)|$$

  - Transmit time of the first packet in the pair (TxA)
  - Receive time of the first packet in the pair (RxA)
  - Transmit time of the second packet in the pair (TxB)
  - Receive time of the second packet in the pair (RxB)

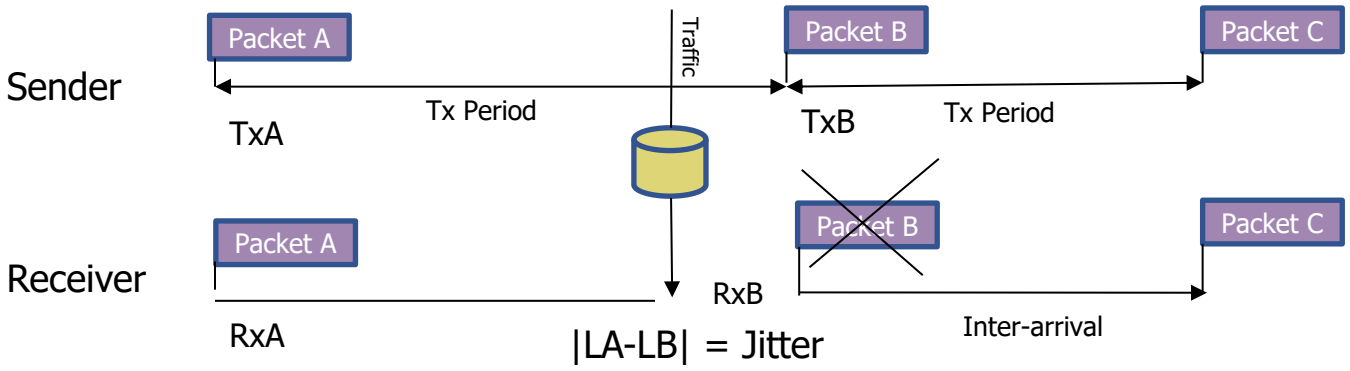


Source: Spirent Communications

Key
LA = Latency of A
LB = Latency of B

Measuring Jitter

- Transmit packets at fixed interval (TxB – TxA).
  - Measure inter-arrival time of received packets.
  - Jitter = |(RxA – TxA) – (RxB – TxB)|
    = |(RxA – RxB) + (TxB - TxA)|
  - Issues: losses, reorder of packets, realistic traffic?



Source: Spirent Communications