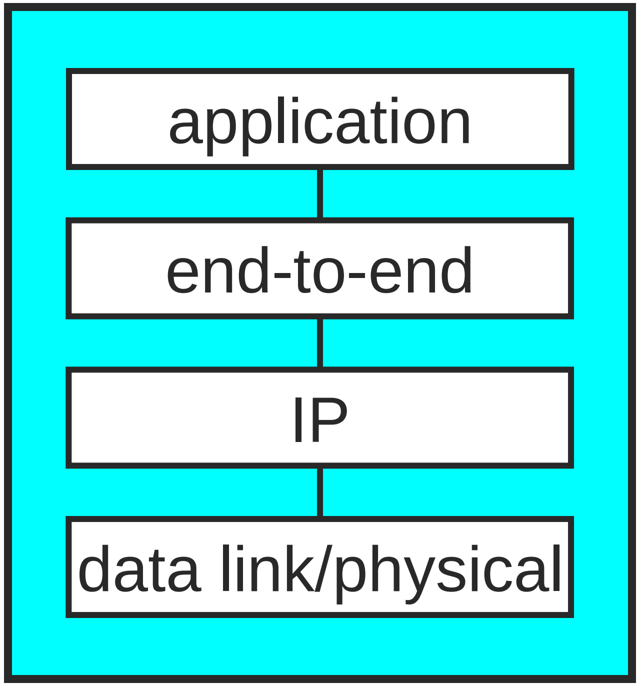
The Big(ger) Picture



application

end-to-end

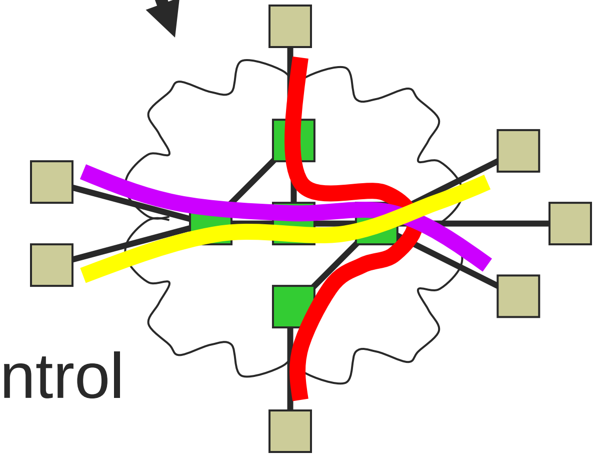
IP

data link/physical

428/598 topics

detailed description of issues here

most coverage until now



performance ← congestion control

Where are you?

- Understand how to
  - Build a network on one physical medium
  - Connect networks
  - Address network heterogeneity
  - Address global scale
- Final part of class
  - End-to-end issues and common protocols
  - Implement a reliable byte stream
  - Congestion control: TCP heuristics, switch/router approaches to fairness
  - Performance analysis

End-to-End Protocols

End-to-end Service Model

Protocol Examples

User Datagram Protocol (UDP)

Transmission Control Protocol (TCP)

End-to-End Service Model

- User perspective of network
  - Knowledge of required functionality
  - Implementation is hidden
- Focus
  - Enable communication between applications
  - Translate from host-to-host protocols
- Services
  - Services that cannot be implemented in lower layers (hop-by-hop basis)
  - Avoid duplicate effort
  - Services not needed by all applications

End-to-End Service Model

- Build on “best effort” service provided by network layer (IP)
  - Messages sent from a host are delivered to another host
    - May be lost
    - May be reordered
    - May be delivered multiple times
    - May be limited to a finite size
    - May be delivered after a long delay

End-to-End Service Model

- Support services needed by the application
  - Multiple connections per host
  - Guaranteed delivery
  - Messages delivered in the order they were sent
  - Messages delivered at most once
  - No limit on message size
  - Synchronization between sender and receiver
  - Flow control

End-to-End Service Model

- Challenge
  - Given
    - Less than desirable properties of the underlying network
  - Create
    - High-level services required by applications
- Services
  - Asynchronous demultiplexing service
  - Reliable byte-stream service

User Datagram Protocol (UDP)

- Simple connectionless demultiplexer
  - No handshaking
  - Each segment handled independently
- Service Model
  - Thin veneer over IP services
  - Unreliable unordered datagram service
  - Addresses multiplexing of multiple connections
- Multiplexing
  - 16-bit port numbers
  - Well-known ports
- Checksum
  - Validate header
  - Optional in IPv4
  - Mandatory in IPv6

User Datagram Protocol (UDP)

- Why is there a UDP?
  - No connection establishment
    - Low delay
  - Simple
    - No connection state at sender, receiver
  - Small header
  - No congestion control
    - UDP can blast away as fast as desired

- What kind of applications is UDP good for?
  - Streaming multimedia apps
  - Loss tolerant
  - Rate sensitive
- Other UDP uses
  - DNS, SNMP
- Reliable transfer over UDP
  - At application layer
  - Application-specific error recovery

UDP Header Format

0 8 16 31

Source Port	**Destination Port**
UDP Length	**UDP Checksum**

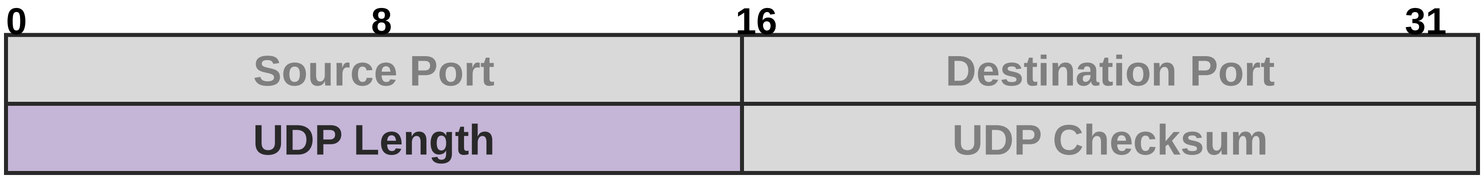
UDP Header Format

0 8 16 31

Source Port	Destination Port
UDP Length	UDP Checksum

- 16-bit source and destination ports

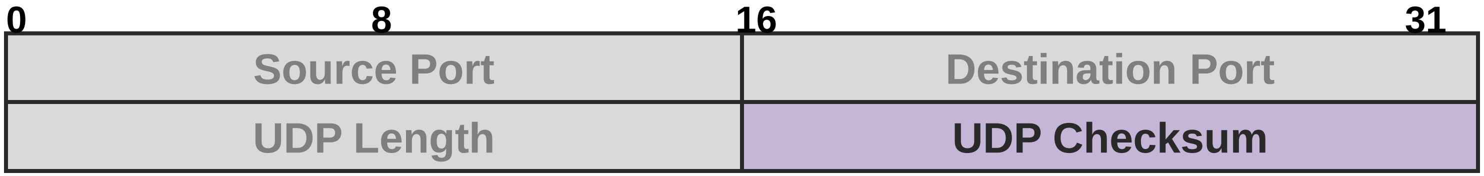
UDP Header Format



0	8	16	31
Source Port		Destination Port	
UDP Length		UDP Checksum	

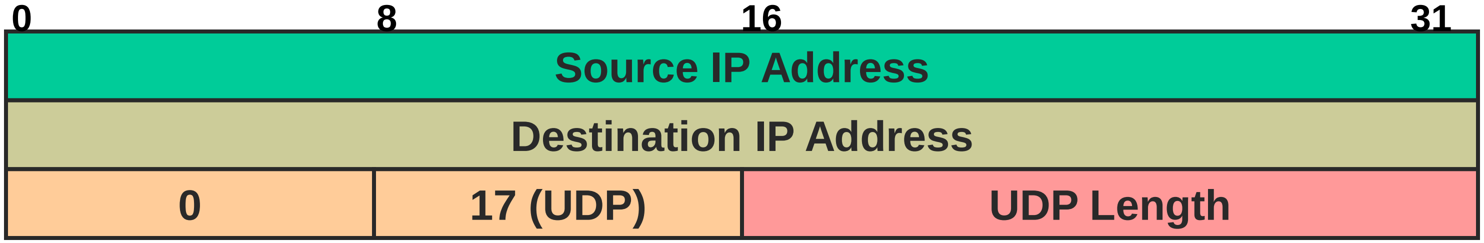
- Length includes 8-byte header and data

UDP Header Format



0	8	16	31
Source Port		Destination Port	
UDP Length		UDP Checksum	

- Checksum
  - Uses IP checksum algorithm
  - Computed on header, data and pseudo header



0	8	16	31
Source IP Address			
Destination IP Address			
0	17 (UDP)	UDP Length	

UDP Header Format

0	8	16	31
Source Port		Destination Port	
UDP Length		UDP Checksum	

- Checksum
  - What purpose does the checksum serve?
  - Why is it mandatory when using IPv6?

0	8	16	31
Source IP Address			
Destination IP Address			
0	17 (UDP)	UDP Length	

Transmission Control Protocol (TCP)

- Reliable byte stream
- Service model
  - Multiple connections per host
  - Guaranteed delivery
  - Messages delivered in the order they were sent
  - Messages delivered at most once
  - No limit on message size
  - Synchronization between sender and receiver
  - Flow control

- Multiplexing
  - Equivalent to UDP
- Checksum
  - Equivalent to UDP
  - Mandatory

TCP

- Connection oriented
  - Explicit setup and teardown required
- Full duplex
  - Data flows in both directions simultaneously
  - Point-to-point connection
- Byte stream abstraction
  - No boundaries in data
  - App writes bytes, TCP send segments, App receives bytes

TCP

- Rate control
  - Flow control to restrict sender rate to something manageable by receiver
  - Congestion control to restrict sender to something manageable by network
  - Both need to handle the presence of other traffic

TCP Outline

- TCP and reliability
- Usage model
- Segment header format and options
- States and state diagram
- Sliding window implementation details
- Flow control issues
- Bit allocation limitations
- Adaptive retransmission algorithms

Proposal:
Reliable Network Layer

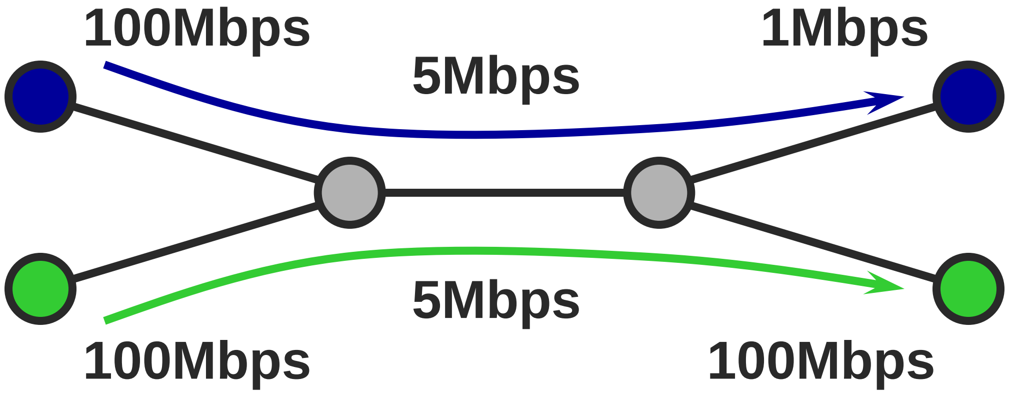
- Service
  - High probabilistic guarantee of correct, in order data transmission at the network layer
  - Hop-by hop network layer ACKs
- Is this sufficient?
- No
  - Routers may crash, buffers may overflow
- Is it beneficial?
  - Maybe, depends on link's error rate
  - Improve performance, not provide correctness

The End-to-End Argument

- Lower layer functions
  - May be redundant or of little value when compared with providing them at that low layer
- Functionality
  - Implemented at a lower layer iff it can be correctly and completely implemented there
- Real constraint
  - Implementing functionality at a lower level should have minimum performance impact on applications that do not use the functionality

End-to-End Argument

- In-order delivery
  - hop-by-hop ordering guarantee is not robust to path changes or multiple paths
- Congestion control
  - Should be stopped at source
  - But network can provide feedback



green should get 9Mbps, but gets only 5Mbps with hop-by-hop drops