

# Transport Layer / UDP

14-740: Fundamentals of Computer Networks  
Bill Nace

# Administrivia

- HW #1 will be posted shortly
- Mission: Learn to use network tools to gather information and for debugging
- Due in 2.5 weeks (18 Oct)
- Report due to Canvas

# Administrivia

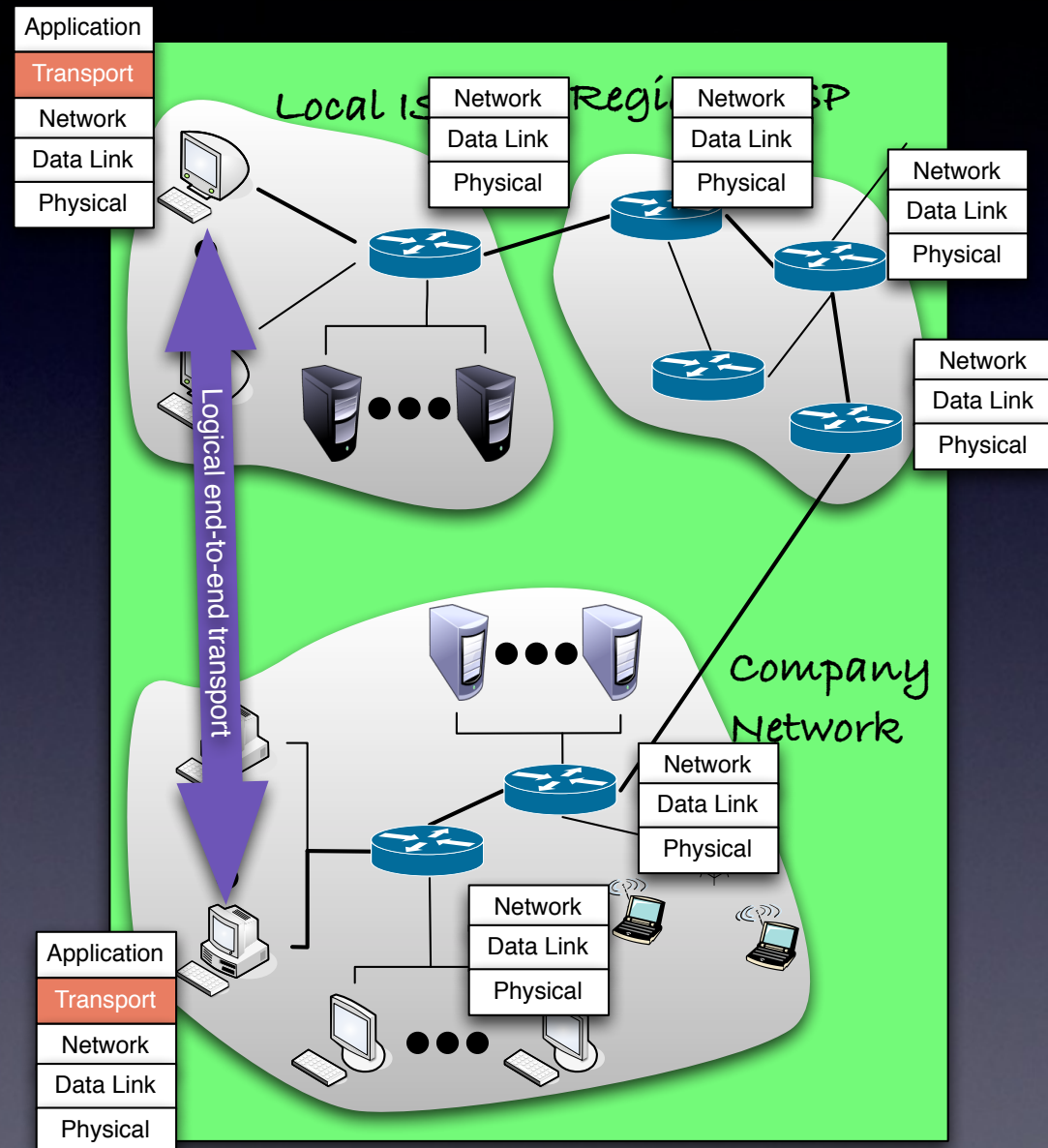
- Quiz next time!
- In class
  - Entire 1:20 available
- Closed book, no calculators
  - Equation sheet will be given if needed

# traceroute

- Transport-layer services
- Connectionless transport: UDP

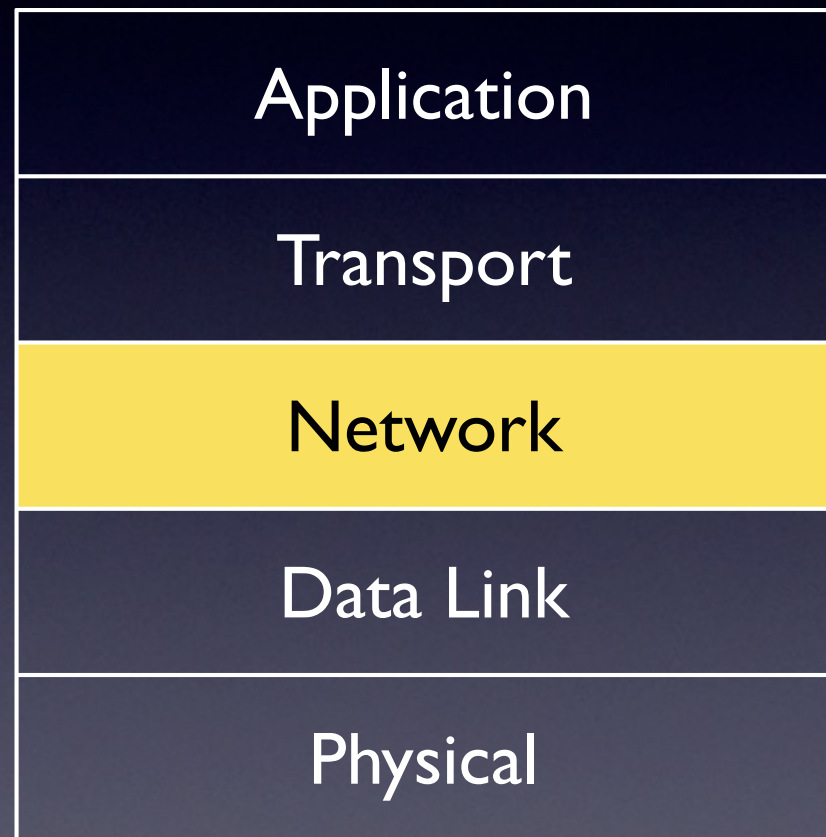
# Transport services and protocols

- Mission: provide **logical** communication between applications running on different hosts
- Logical communication = as if the applications were directly connected



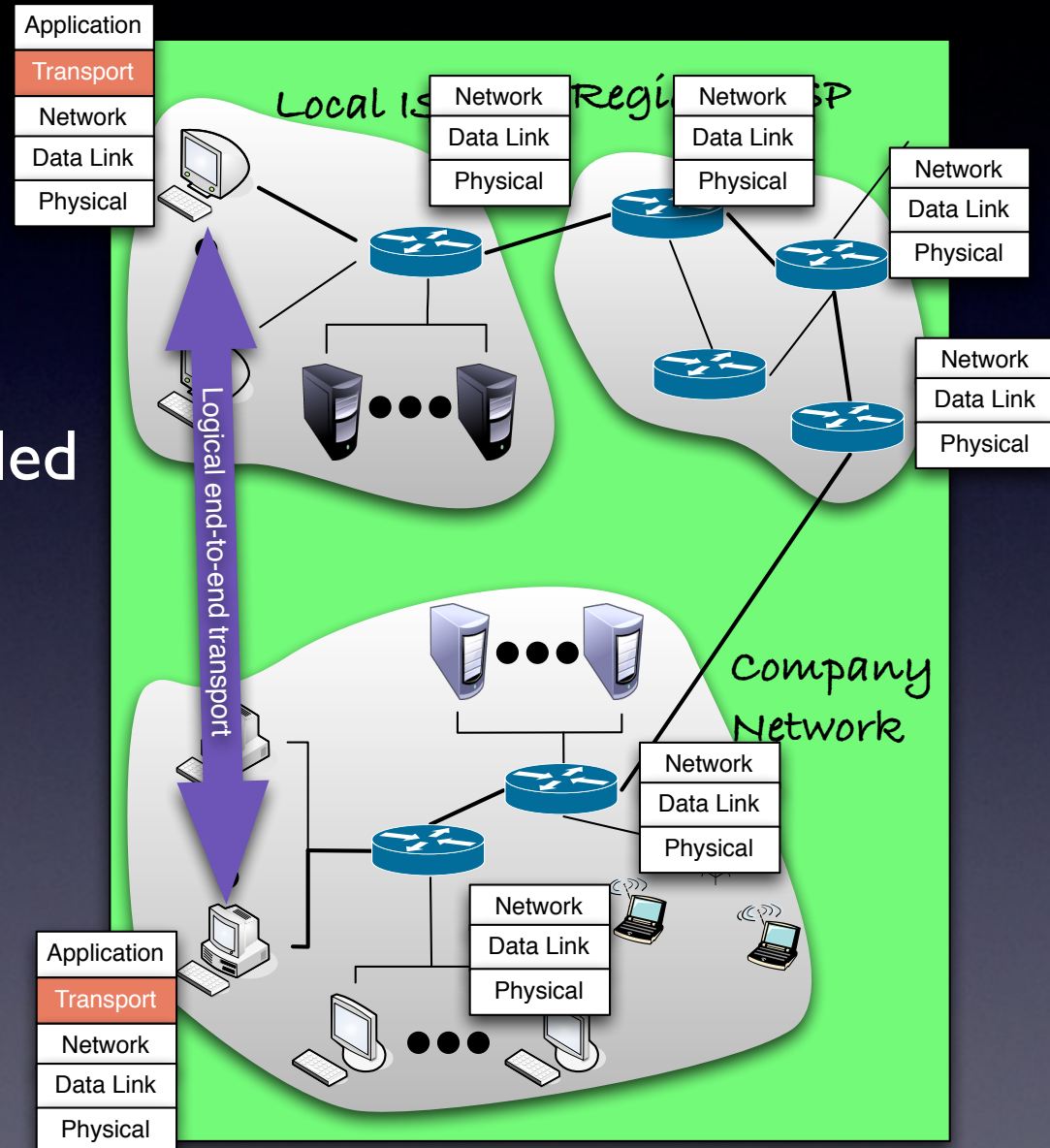
# Using Network Services

- Network layer provides Internet Protocol (IP)
- Provides best-effort logical communication between **hosts**
  - ...of data provided in packets
- Does not provide reliability guarantees
  - or many others (timeliness, security, bandwidth, etc)



# Key Functions

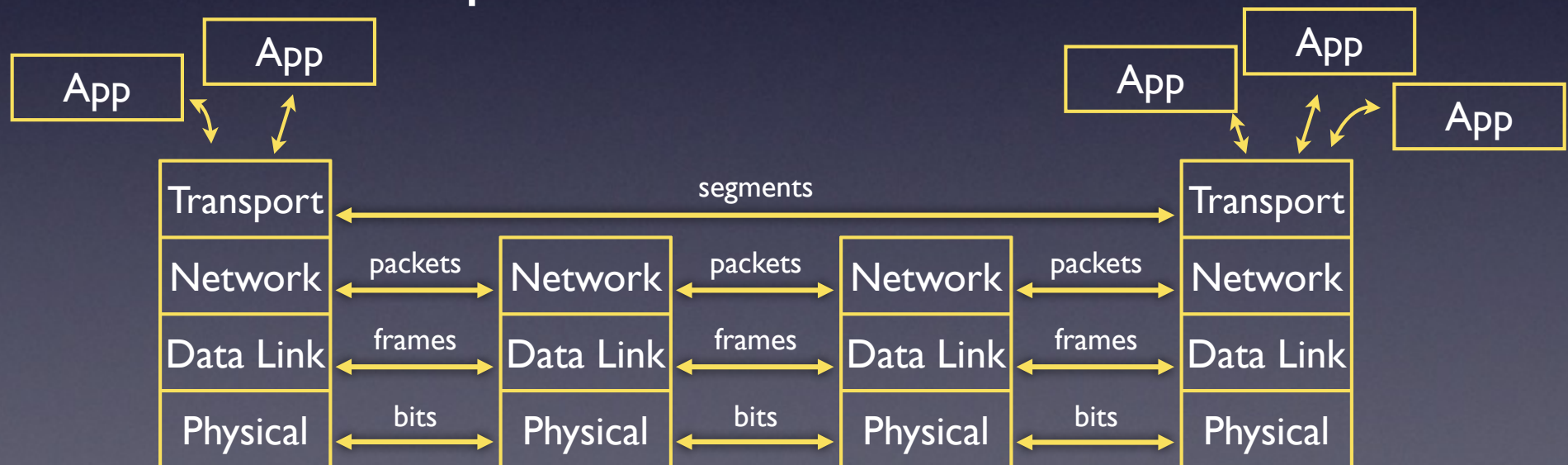
- Multiplex messages ...
- ... and De-multiplex
- Breaking data into appropriate size pieces, called **segments** ...
- ... and re-assembly
- Connection setup, state management, teardown (if necessary)
- TCP: reliability guarantees



# Multiplexing



- End hosts have multiple applications running
- Messages are multiplexed at the sender
- ... de-multiplexed at the receiver





# Port Numbers

- Multiplexing implies addressing
  - How do you know which application to deliver data to?
  - or which application sent the data?
- 16-bit unsigned number
  - Some applications use “well known” port numbers
  - Numbers assigned by IANA

# "Well Known" Ports

## TCP Examples

Port #	Application
22	SSH: Secure shell
25	SMTP: Email
80	HTTP
443	HTTPS

## UDP Examples

Port #	Application
53	DNS
161	SNMP: Net Mgmt
520	RIP: Routing
944	NFS: File System

# Binding

- Port numbers are “bound” to an application
- Application tells Transport layer “I will listen on this port”
- **Ephemeral** ports often used where response is sent from a different port

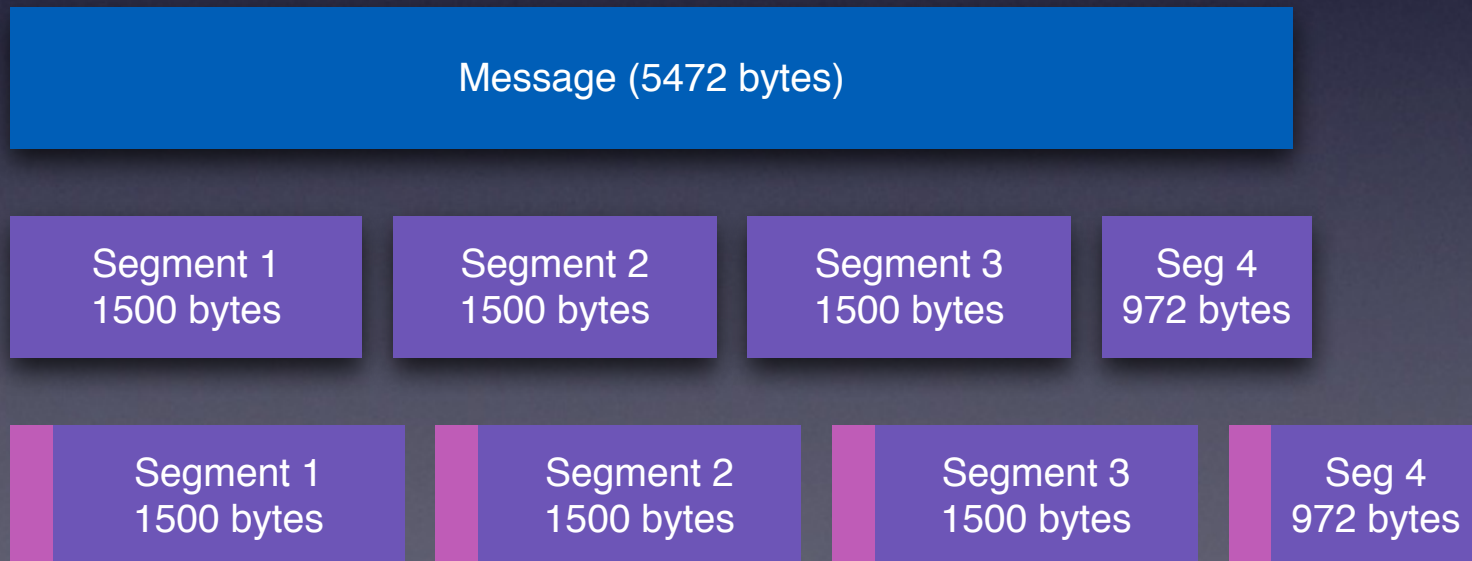
# Segmentation

- Second responsibility of Transport layer
- Sender: Application provides “messages”
  - Broken into segments, passed to Network layer one-by-one
- Receiver: collects segments, reassembles into messages, passes to application layer



# Headers

- Transport layer adds header with additional information (port numbers, ...)



# traceroute

- Transport-layer services
- Connectionless transport: UDP

# User Datagram Protocol

- “bare bones,” “no-frills” transport
- Defined in RFC 768
- “best-effort” service
  - segments may be lost
  - may be delivered out-of-order
- Connectionless
  - No handshaking
  - Each segment handled independently

# Why?

- UDP requires no connection establishment, thus no additional delay
- Very simple: no state to maintain
- Segment header is small
- No congestion control
  - UDP can blast away as fast as desired



# The Real "Why"

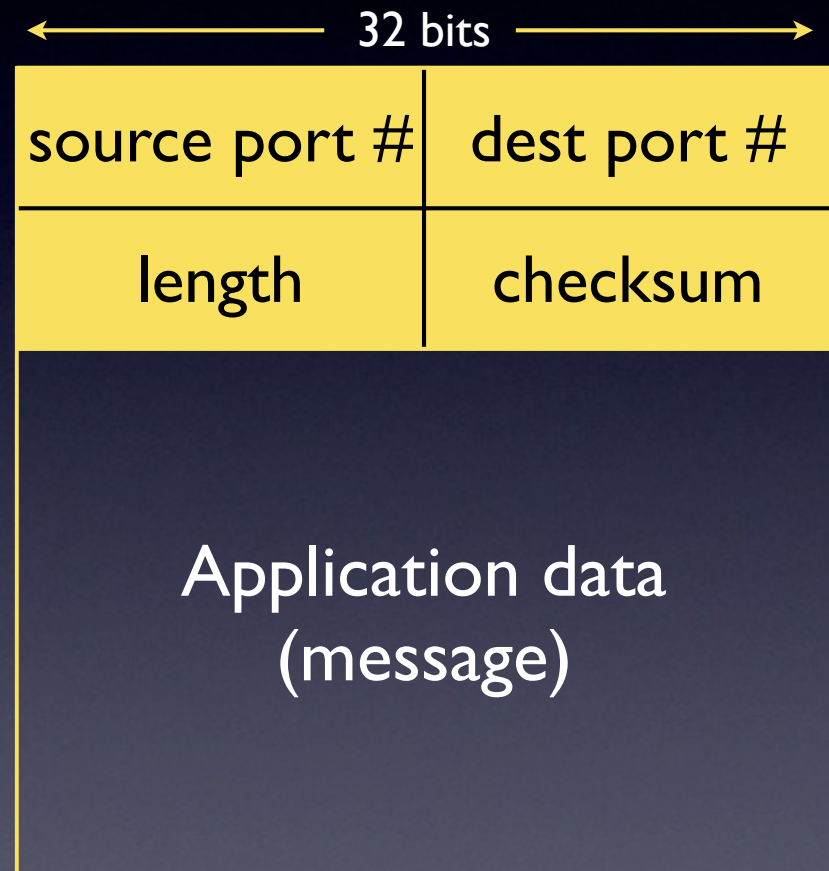
- Reliability requirements don't match TCP
  - TCP: If a segment is lost, it will be retransmitted
  - UDP: If a segment is lost, it should not be retransmitted
- Reliability can be added at app layer
  - application-specific error recovery
  - Perhaps proprietary algorithms

# Uses

- Often used for streaming multimedia apps
  - Loss tolerant
  - Rate sensitive (timeliness)
- Used in DNS, SNMP

# Segment Format

- Small header -- only 8 bytes
- Port numbers are 16 bits
  - Values of 0-65535
- Length is of entire segment, including header, measured in bytes
  - Segment could be 64KBytes



# Checksum

- Goal: detect “errors” (flipped bits) in transmitted segment
- Algorithm selected for ease of implementation in software
  - See RFC 1071 for efficient implementation details
- **End-to-end** detection of bit errors

# Checksum: Sender

- Sender:
  - Calculate the sum of all the 16-bit words in the segment
    - Wrap carry from output to input
  - Take the 1s complement (flip the bits)
  - Put the result in the UDP header

# Checksum: Receiver

- Receiver:
  - Add segment contents, *including checksum*, as 16-bit words
    - Wrapping carry
  - Error if result is not all 1s

# Checksum Example

Message

0110011001100000
0101010101010101
1000111100001100

0110011001100000
+0101010101010101
-----
1011101110110101

1011101110110101
+1000111100001100
-----
10100101011000001

Carry-out is added in again

1
+0100101011000001
-----
0100101011000010

This is the checksum

1011010100111101
------------------

Flip all the bits





# Checksum good enough?

- Detects any single-bit flip
- Does not detect all 2-bit errors
- Designers decided this is “good enough”
  - Tradeoff of
    - Ease of implementation
    - Performance overhead
    - Strong enough for common case

# Lesson Objectives

- Now, you should be able to:
  - describe the mission, scope, addressing mechanism, data types and responsibilities of the Transport Layer
  - explain UDP, including advantages/disadvantages, segment format, and reliability assumptions
  - calculate the checksum of a UDP segment