ECE 356/COMPSI 356
Computer Network Architecture

Transport Protocols. UDP.

Monday October 21st, 2019

Duke UNIVERSITY

Recap

- Last lecture: finishing up a collection of IP topics
- Readings for this lecture: **PD 5.1, 5.2.1**

2

Duke UNIVERSITY

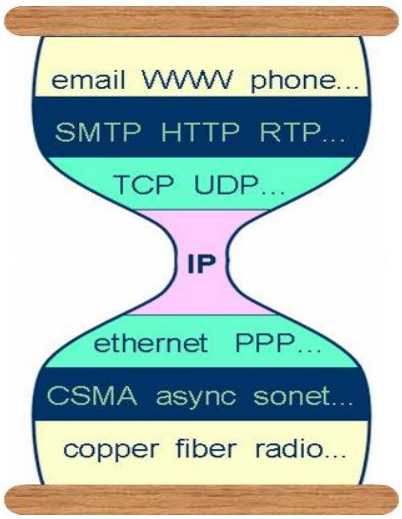
Before: How to Deliver Packets from one Host to Another

- Direct link
  - Encoding, framing, error detection, reliability
  - Multi-access control
- Multi-link network switching and forwarding
  - Datagrams, virtual circuit
  - Bridges, spanning tree algorithm
- Interconnecting multiple networks
  - IP addressing, forwarding, routing
    - ARP, distance vector, link state, path vector
  - NAT, DHCP, tunnels, …

3

Next Several Lectures

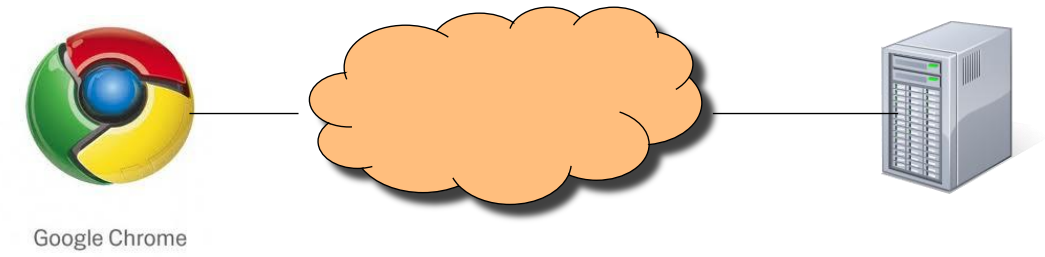
- Transport control
  - Moving up the stack
- Transport functionality
- UDP
- TCP: several lectures, covering different elements of its functionality



Lecture Outline

- **Introduction to transport control**
- User Datagram Protocol (UDP)
- Transport Control Protocol (TCP): an introduction

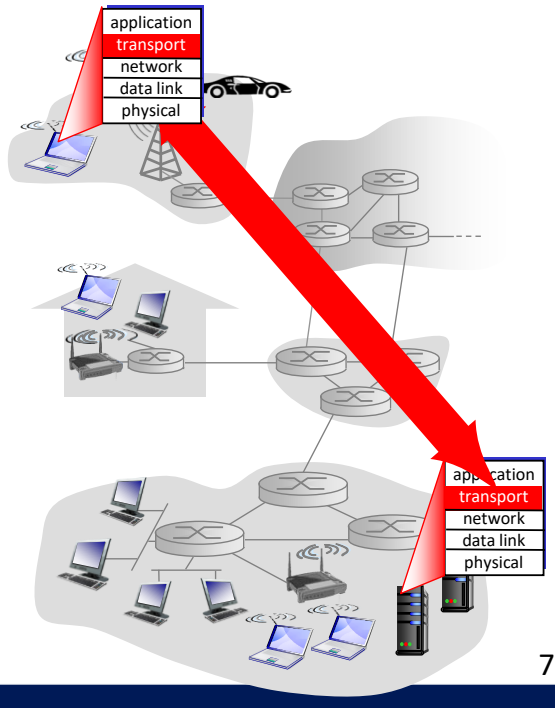
Transport Layer Design Goals



- A process-to-process communication channel
  - As if the hosts running processes were directly connected
  - Upper-layer: application
  - Lower-layer: network (IP)

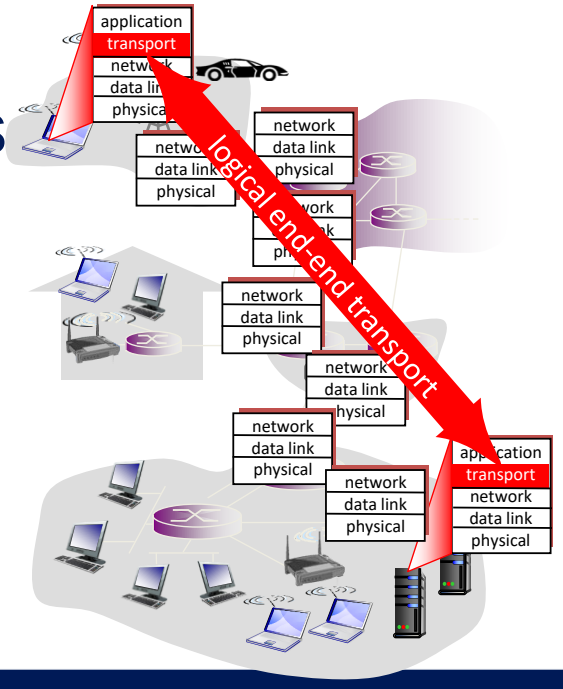
Logical Communication Between Application Processes

- Transport protocols run in end systems
  - Sender side: breaks app messages into *segments*, passes to network layer
  - Receiver side: reassembles segments into messages, passes to app layer



Transport Layer: End-to-End Protocols

- Implemented in end systems
- *Not implemented in network routers*
  - Routers act only on network-layer fields of the datagram



Network Service Model

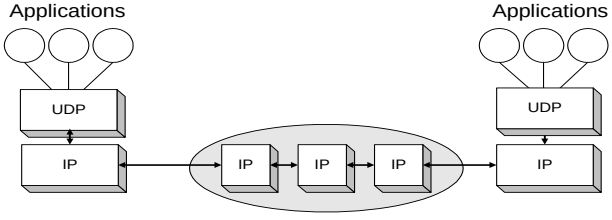
- **Best-effort**
  - Makes its "best effort", but offers no guarantees
  - May discard, reorder, duplicate messages
  - Does not guarantee integrity of data in the segments

9

Duke UNIVERSITY

Transport Control: Minimal Features

- Process-to-process delivery: transport layer multiplexing and demultiplexing
  - Supporting multiple processes on the same host
- Integrity checking
  - Error detection fields



- User Datagram Protocol (UDP) only provides these capabilities

10

Duke UNIVERSITY

Transport Control: Desirable Features

- Reliable delivery
- In-order
- No duplication
- Connection setup
- Not to send faster than a receiver can receive
- Not to send faster than the network allows
- …

Design Choices

- How to achieve the desired process-to-process service model?
  - ➢ Let applications handle it
    - Develop a set of libraries
  - ➢ Enhance the network to provide the desirable features
    - Not considered a good idea
  - ➢ Place a service layer on top of IP to handle it
    - This is chosen by the Internet design

Transport Control: Reliable Data Transfer

- Flow control
- Sequence numbers
- Acknowledgements
- Timers

- Complex: will discuss at length

13

Transport Control: Congestion Control

- Prevents any one connection from swamping the links and routers with excessive amounts of traffic

- Aims to give each connection a fair share of bandwidth

14

Transport Protocols in the Internet

The most commonly used transport protocols are UDP and TCP

UDP: User Datagram Protocol

- Unreliable, connectionless
- Simple
- Unicast and multicast
- Useful only for few applications, e.g., multimedia applications
- Used by many services
  - Network management (SNMP), routing (RIP), naming (DNS), etc.

TCP: Transmission Control Protocol

- Reliable, connection-oriented
- Complex
- Only unicast
- Used for most Internet applications:
  - Web (HTTP), email (SMTP), file transfer (FTP), terminal (telnet), etc.

15

Transport Protocols: Key Points to Remember

- Create a process-to-process communication channel
- End to end protocols
  - Not implemented in network routers
- At minimum, provide process demultiplexing and error checking
  - UDP
- Can provide reliable in-order delivery, congestion control, …

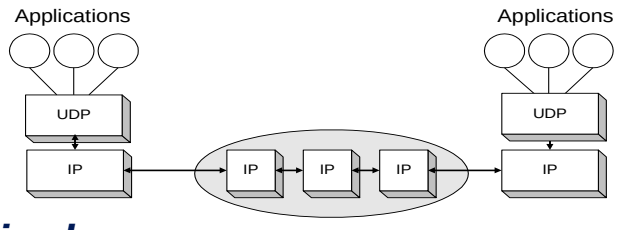
16

Lecture Outline

- Introduction to transport control protocols
- **User Datagram Protocol (UDP)**
- Transport Control Protocol (TCP): an introduction

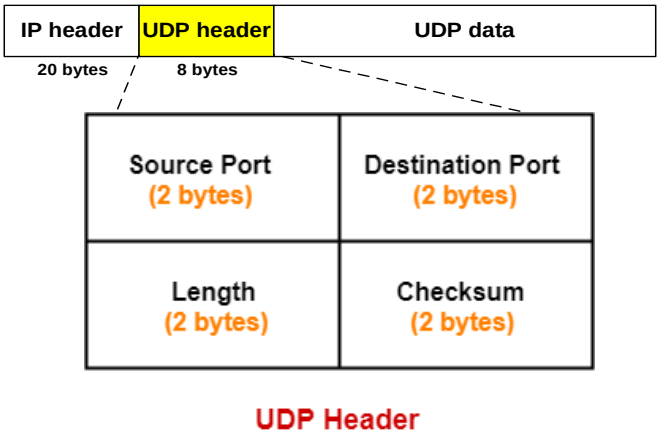
User Datagram Protocol (UDP)

- Designed in 1980
- Minimal transport service: non-guaranteed datagram delivery
- *"A no-frills, bare-bones transport protocol"*
- "*Almost a null protocol*"
- Only provides:
  - Multiplexing by port number
  - Checksumming of data
- No connection setup: ***connectionless***



UDP Header Format (1/2)

- Quite simple
- 8 bytes total

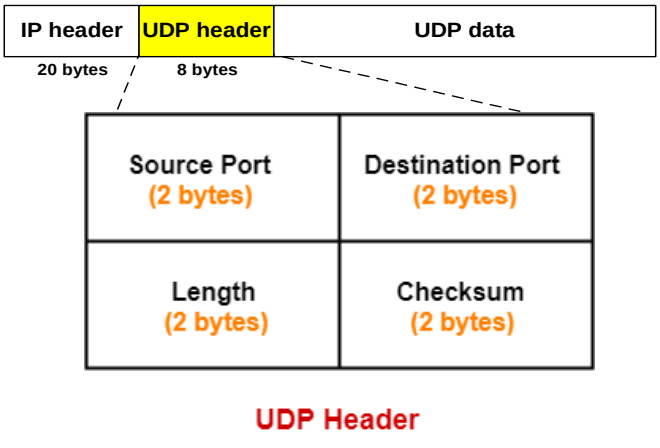


Port numbers: identify sending and receiving applications (processes) Maximum port number is 2^{16}-1= 65,535

Message Length: at least 8 bytes (i.e., data field can be empty) and at most 65,535

19

UDP Header Format (1/2)



Checksum: calculated over UDP header and data, and a "*pseudo-header*" (protocol number, IP source/dst)

- 1s complement of the sum of all the 16-bit words in the segment

20

UDP Pseudo-Header

IPv4 pseudo header format

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source IPv4 Address																															
4	32	Destination IPv4 Address																															
8	64	Zeroes								Protocol								UDP Length															
12	96	Source Port																Destination Port															
16	128	Length																Checksum															
20	160+	Data																															

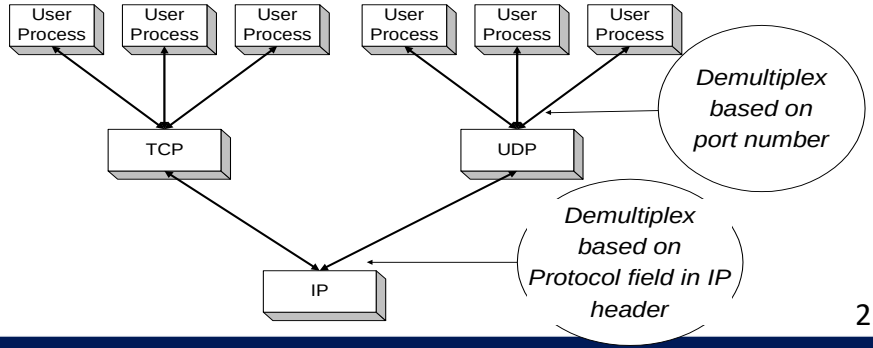
- Make sure IP does not make a mistake and delivers a wrong packet to the UDP module
- If UDP length is odd, one pad byte of zero will be added to the end for a 16-bit checksum computation
- Jumping ahead: same mechanism used in TCP as well

21

Duke UNIVERSITY

Demultiplexing: Port Numbers

- UDP and TCP use *port numbers* to identify applications
- A globally unique address at the transport layer (for both UDP and TCP) is a tuple **<IP address, port number>**
- There are 65,535 UDP ports per host



22

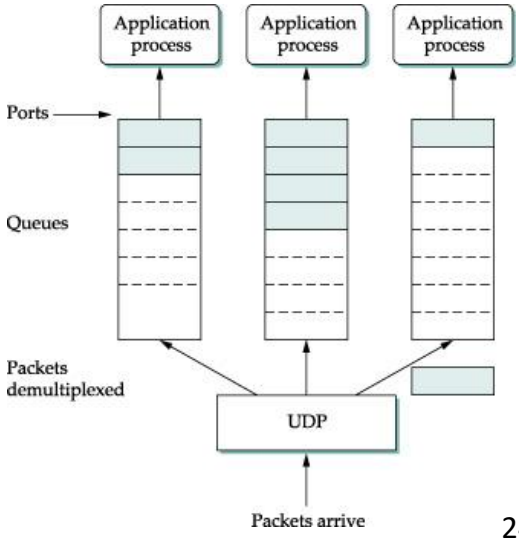
Duke UNIVERSITY

Demultiplexing: How to Find Out Application Ports

- Servers use well-known ports
  - DNS: 53
  - RIP: 520
  - OLSR: 698
- A server learns a client's port from its packets

Implementing Ports

- A "port" is an abstraction
- Implementation may differ from OS to OS
- Ex: port implemented using a message queue
  - Packets discarded when queues are full



Domain Name Service (DNS) Runs on UDP (1/2)

- Map an easy-to-remember name to an IP address
  - Without DNS, to send an IP packet, we'd have to remember
    - 66.102.7.99
    - 64.236.24.28
  - With DNS
    - **www.google.com**→ 66.102.7.99
    - **www.cnn.com**→ 64.236.24.28
- Response is needed quickly

25

Duke UNIVERSITY

Domain Name Service (DNS) Runs on UDP (2/2)

- Client constructs a DNS query
- UDP adds a header and sends the query to the name server
  - Sends immediately: no connection setup
- Waits for a reply. If no reply:
  - Resend the query
  - Try sending the query to another name server
  - Inform the invoking application that it can't get a reply

26

Duke UNIVERSITY

Isn't TCP Always Preferable?

- We already saw applications that run over UDP
  - DHCP, ICMP traceroute
- UDP has its advantages
- Do we always need reliability?
  - For file transfers, yes
  - For media streaming and real-time communications, no
    - Have no use for a video frame of a minute ago

27

Other Advantages of UDP

- No connection establishment
  - No connection establishment *delay*
- No connection state
  - Server can support many more active clients
- Supports broadcast and multicast
- Small packet overhead
  - UDP: 8 bytes
  - TCP: 20 bytes

28

Reliable Data Transfer When Using UDP?

- Can implement reliability at the application layer
- Example: QUIC
  - Used in Google's Chrome browser
  - Aims to be nearly equivalent to a TCP connection, but with much-reduced latency
  - Retransmissions at the level of QUIC

29

UDP: Key Points to Remember

- Bare-bones transport protocol
  - Only provides *demultiplexing by port numbers* and *checksumming*
- Has important advantages
  - No retransmissions
  - No connection establishment delay, no state

30

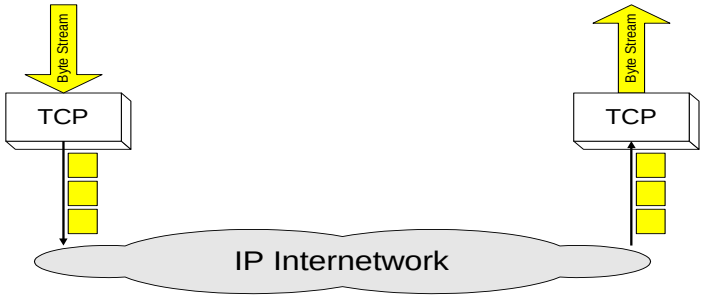
Lecture Outline

- Introduction to transport control protocols
- User Datagram Protocol (UDP)
- **Transport Control Protocol (TCP)**
  - TCP segment format
  - TCP connection establishment and termination

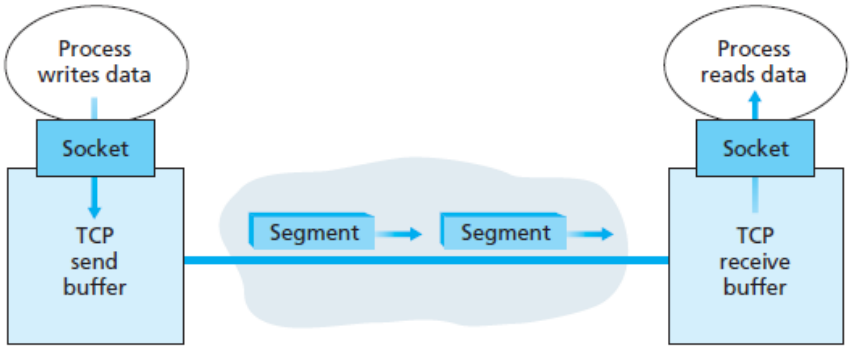
Overview

TCP = Transmission Control Protocol

- Connection-oriented protocol
- Provides a **reliable unicast** end-to-end byte stream over an unreliable internetwork



TCP Manages a Byte Stream



- One-way shown for simplicity; bi-directional in general
- Transmitting *segments*: carrying a segment of the byte stream

33

Duke UNIVERSITY

TCP is Important

- Foundational
  - Internet protocol stack is often referred to as "**TCP/IP**"
  - Existed since 1974: substantially earlier than UDP
  - Authors Vint Cerf and Bob Kahn received the 2004 Turing Award
- Dominant
  - HTTP uses TCP

34

Duke UNIVERSITY

Unique Design Challenges

- We've learned how to reliably transmit over a direct link
  - Coding/encoding, framing, sliding window
- What's new?
  1. Process-to-process communication → connection setup
  2. Heterogeneity
     - Bandwidth varies: how fast should the sender send?
     - RTT varies: when should a sender time out?
  3. Out of order
  4. Resource sharing
     - Many senders share a link in the middle of the network

35

TCP: Connection-Oriented

- Host processes must first "handshake" with each other
  - Exchange messages
  - Establish the parameters of data transfer
- Note: state is established in the ***end hosts***, <u>not the intermediate routers</u>
  - Intermediate routers are oblivious to TCP connections
  - Note the difference with circuit switching
- Full-duplex
- Point-to-point only: no broadcast, no multicast

36

TCP Features

- Reliable, in-order byte stream service
- Flow control: not to overrun a receiver
- Congestion control: not to congest the network

37

TCP: Key Points to Remember

- One of the key Internet protocols
- Connection-oriented, reliable communications

38

Next Lecture

- Transport control protocol (TCP)