CSc 450/550
Computer Networks
Transmission Control Protocol

Jianping Pan

Summer 2007

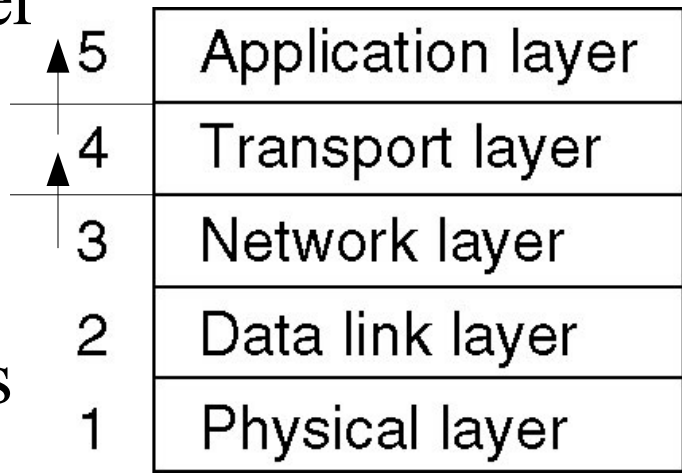
Review: application layer

- HTTP: hypertext transfer protocol
  - client-server model
  - request-reply transaction
  - normally based on TCP
- DNS: domain name system
  - DNS hierarchy
  - DNS queries
  - normally based on UDP

Q: why TCP for HTTP, UDP for DNS?

Today's topics

- Transport-layer protocol elements
  - services provided to application layer
    - to support HTTP, DNS, etc
  - services provided by network layer
    - e.g., by IP
  - transport-layer protocol mechanisms
    - i.e., how to fill the gap



Transport layer services

- Services provided by transport layer
  - endpoint-to-endpoint communication
    - *endpoint*: an application process in end-hosts
  - connection-oriented vs connectionless
  - data transfer: reliable vs unreliable
- Example: Internet transport-layer services
  - connection-oriented, reliable by TCP
  - connectionless, unreliable by UDP

Apps:
HTTP
DNS

P2: reliable datagram protocol

Network layer services

- Services provided by network layer
  - move packets from one end-host to another
  - possibly through many intermediate systems
- Example: Internet network-layer services
  - IP: store-and-forwarding packet switching
  - packets may get
    - lost at communication link, router or receiver buffer
    - duplicated
    - corrupted
    - reordered

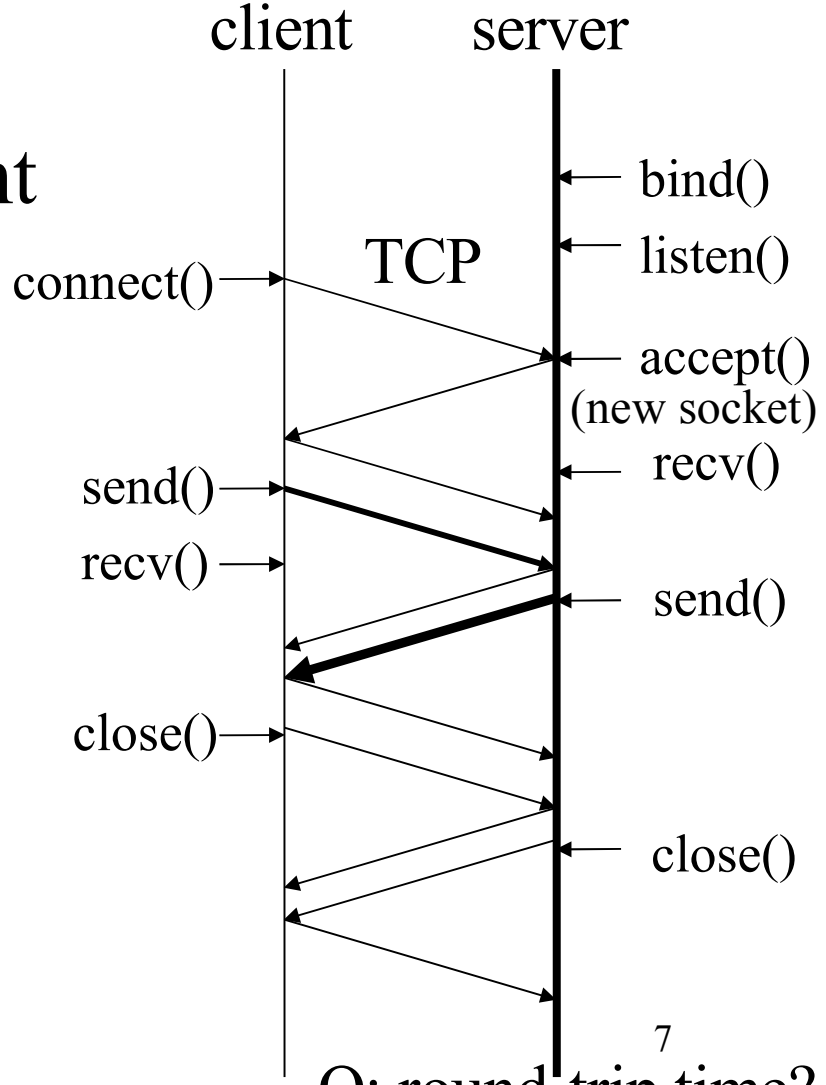
Q: possible causes?

Transport layer protocols

- Protocol mechanisms
  - addressing and multiplexing
    - how to identify an *endpoint* in an *end-host*
  - connection management
    - for connection-oriented transport services
  - flow control: avoid outpacing the receiver
  - error control
    - for reliable transport services
  - congestion control: avoid overloading the network

Example: Socket API

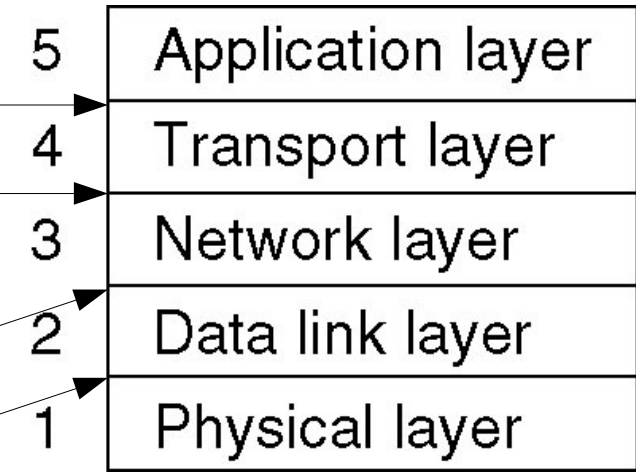
- Connection establishment
  - bind(), listen()
  - connect()
  - accept()
- Data transfer
  - send(), recv()
- Connection release
  - close()



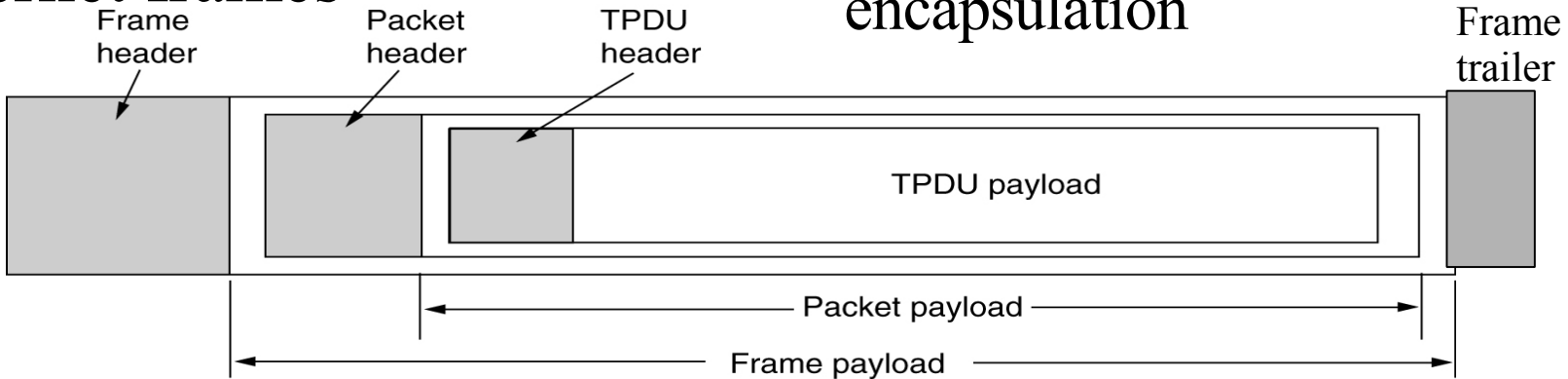
Q: round-trip time?

What's under Socket?

- Socket
  - socket messages
- TCP
  - TCP segments with TCP header
- IP
  - IP packets with IP header
- Ethernet
  - Ethernet frames



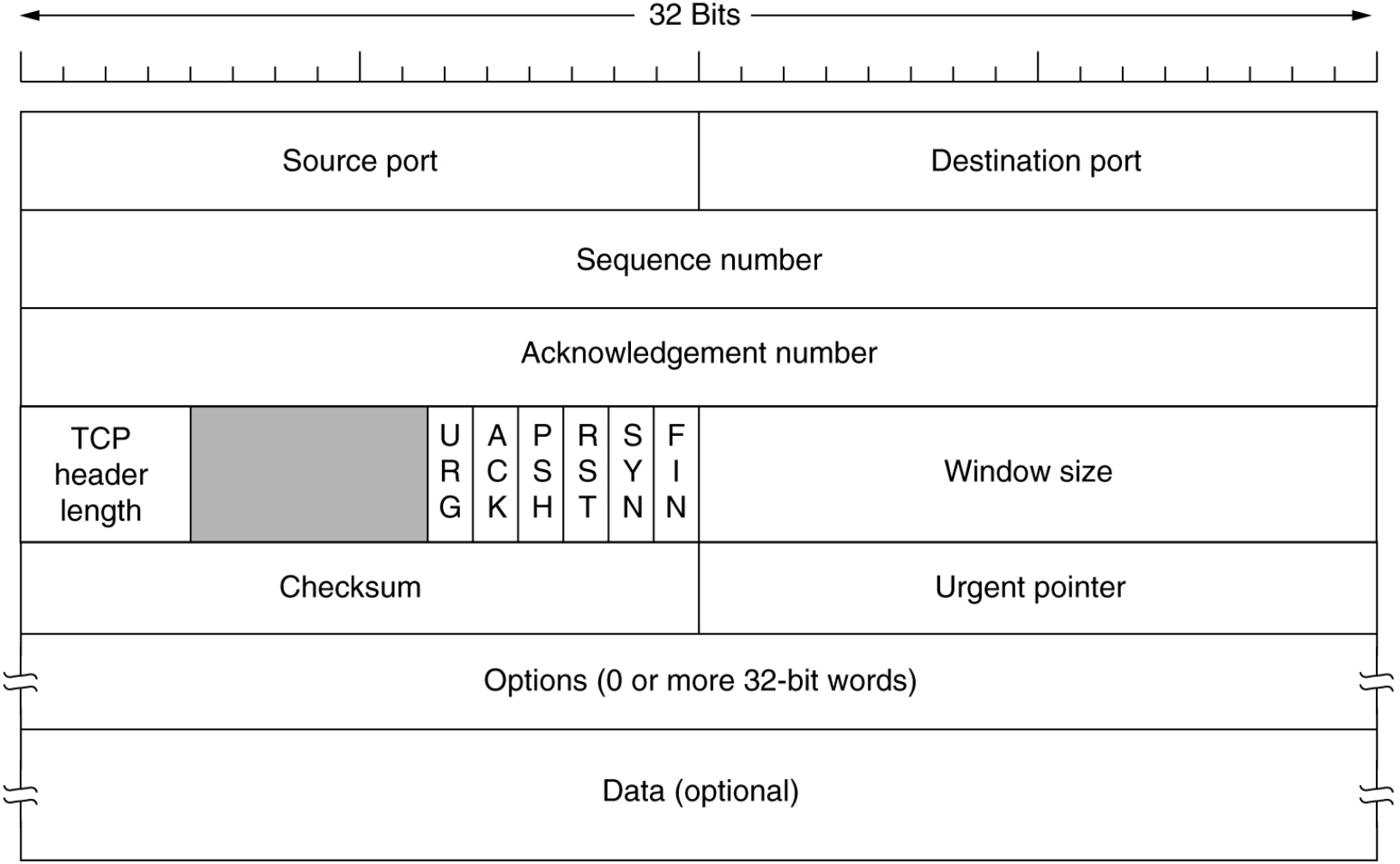
encapsulation



TCP

- Transmission control protocol [RFC793]
- Services provided by TCP
  - connection-oriented, point-to-point, bi-directional
  - reliable, in-sequence, stream-like
- Services provided by IP
  - packets: duplicated, lost, reordered, corrupted
- TCP protocol mechanisms
  - connection management
  - flow, error and congestion control

TCP packet header



Port number

- TCP port number (16-bit)
  - source, destination port numbers
    - addressing and multiplexing
- Port number allocation (ref: iana.org)
  - well-known port numbers (0~1023, privileged)
    - e.g., 80: http; 443: https
  - registered port numbers (/etc/services)
    - http-alt 8080/tcp
  - dynamically allocated port numbers

bind(), connect()

TCP connection ID

- TCP connections
  - connection: initiator, responder
    - (initiator IP, *initiator port*, responder IP, responder port)
- One connection: one flow in each direction
  - for each flow: source, destination
    - (source IP, source port, destination IP, destination port)
  - 5-tuple (or 4-tuple when protocol ID is implied)
    - (src IP, src port, protocol ID, dst IP, dst port)
- Socket, connection, flow

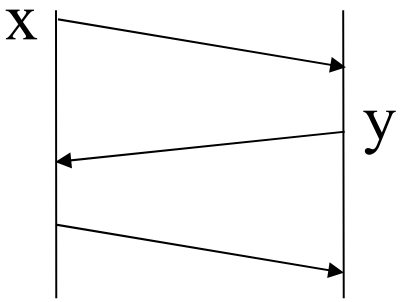
listen()
connect(), accept()

send(), recv()

Q: port reuse?

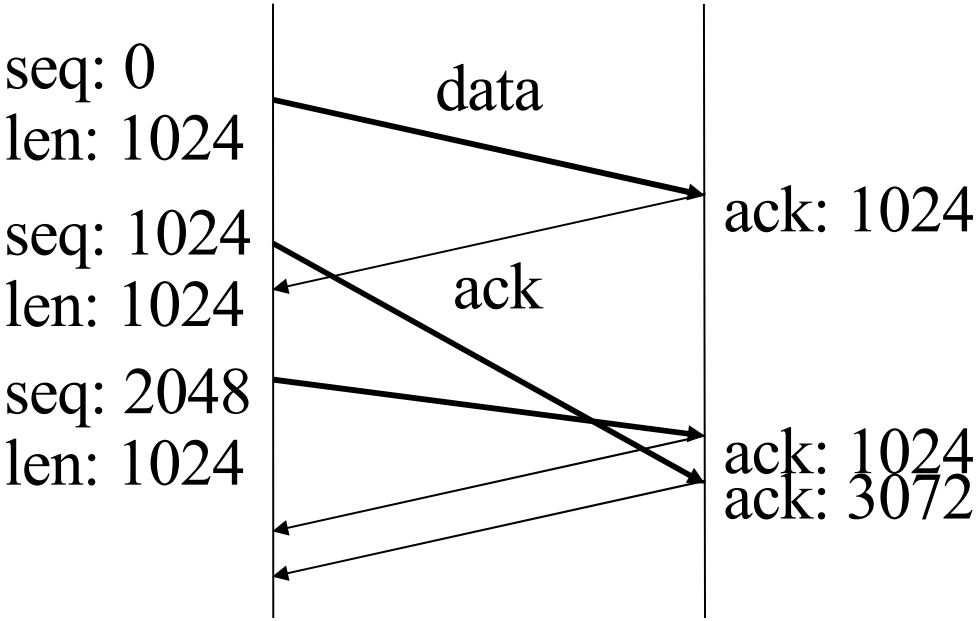
Sequence number

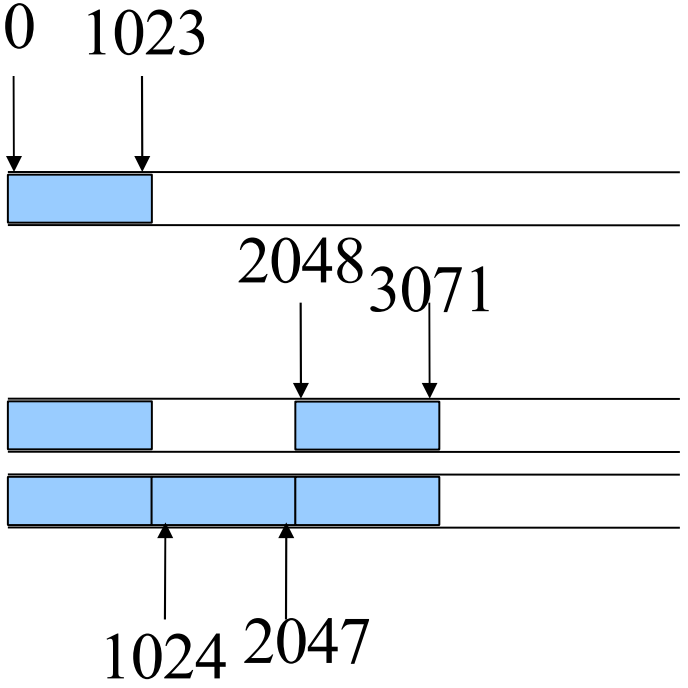
- TCP sequence number (32-bit)
  - byte sequence for the ***first*** byte in payload
    - exception: SYN/FIN sequence number
  - random initial sequence number
    - exchanged during 3-way handshake
  - sequence number rollover
- Acknowledgment number (32-bit)
  - byte sequence for the ***next*** byte to expect



Q: why random ISN?

Sequence vs acknowledgment





Q: byte vs packet sequence?

Header length/Data offset

- Variable-length header due to TCP options
- TCP header length/data offset (4-bit)
  - number of 32-bit words!
  - at least 5 for fixed TCP header fields
  - maximum: 15
    - i.e., 40 bytes in total for TCP options
- TCP design feature: bit-alignment
  - fields of x-bit length at offset n*x (n: an integer)
  - 32-bit word

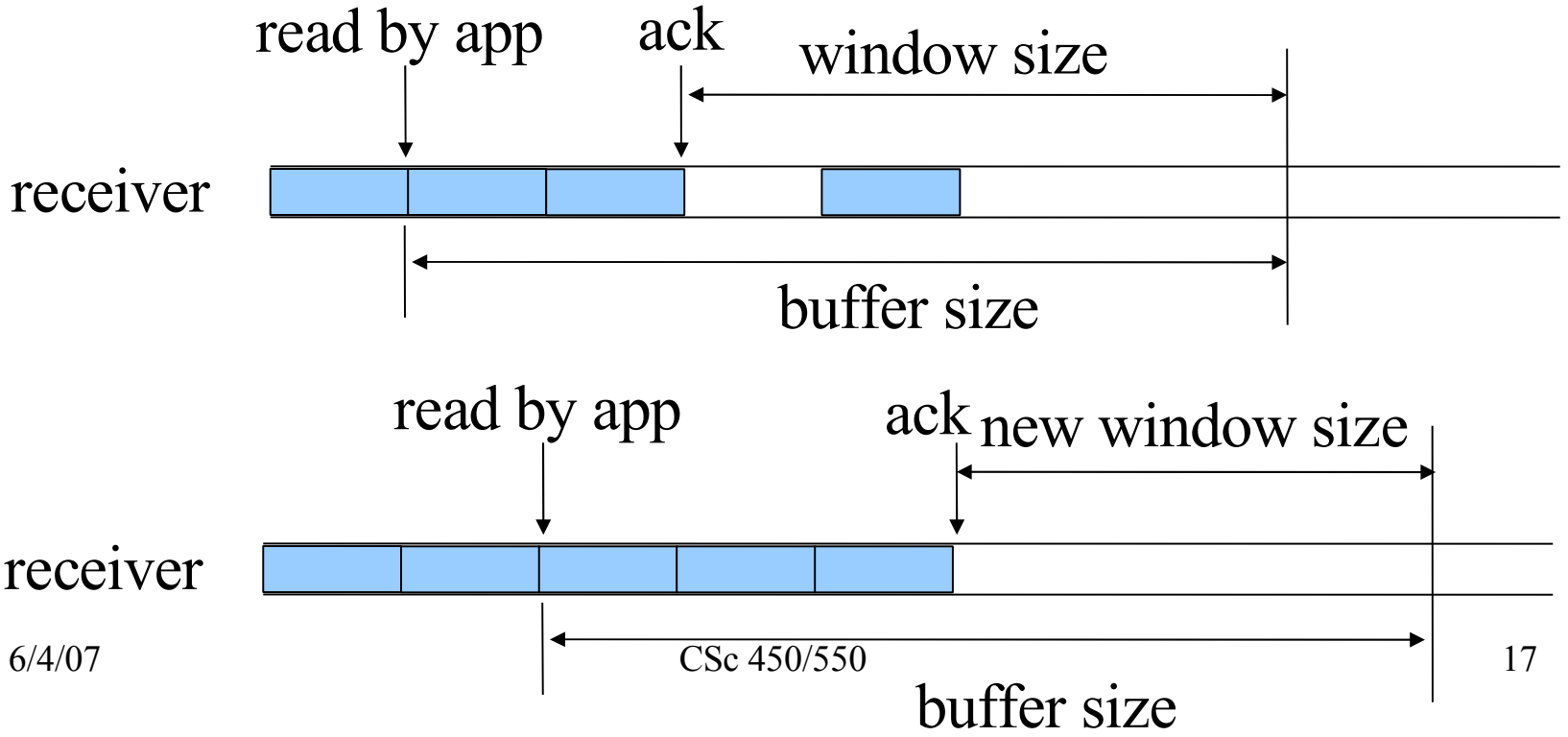
Q: why alignment?

Control flags

- URG: urgent pointer meaningful
- ACK: acknowledgment number meaningful
- PSH: logic message boundary
- RST: connection reset
- SYN: synchronization (connection establishment)
- FIN: finish (graceful connection release)
  - stay tuned: “TCP connection management”

Window size

- TCP window size (16-bit)
  - stay tuned: "TCP flow control"



Checksum

- TCP checksum (16-bit)
  - "one's complement of one's complement sum"
    - stay tuned: "TCP error control"
  - cover
    - TCP header (including options, if any)
    - TCP payload
    - TCP pseudo header
      - source and destination IP address
      - protocol ID
      - TCP segment size

Q: why TCP pseudo header?

Urgent pointer

- TCP urgent pointer (16-bit)
  - offset of the LAST byte for urgent data
    - not (LAST+1) per RFC 1122: Host requirements
  - from the current sequence number!
  - for out-of-band (OOB) control information
    - e.g., interrupt an ongoing file transfer
  - Socket interface
    - send(s, buf, len, MSG_OOB);
  - receiver should process the urgent data immediately

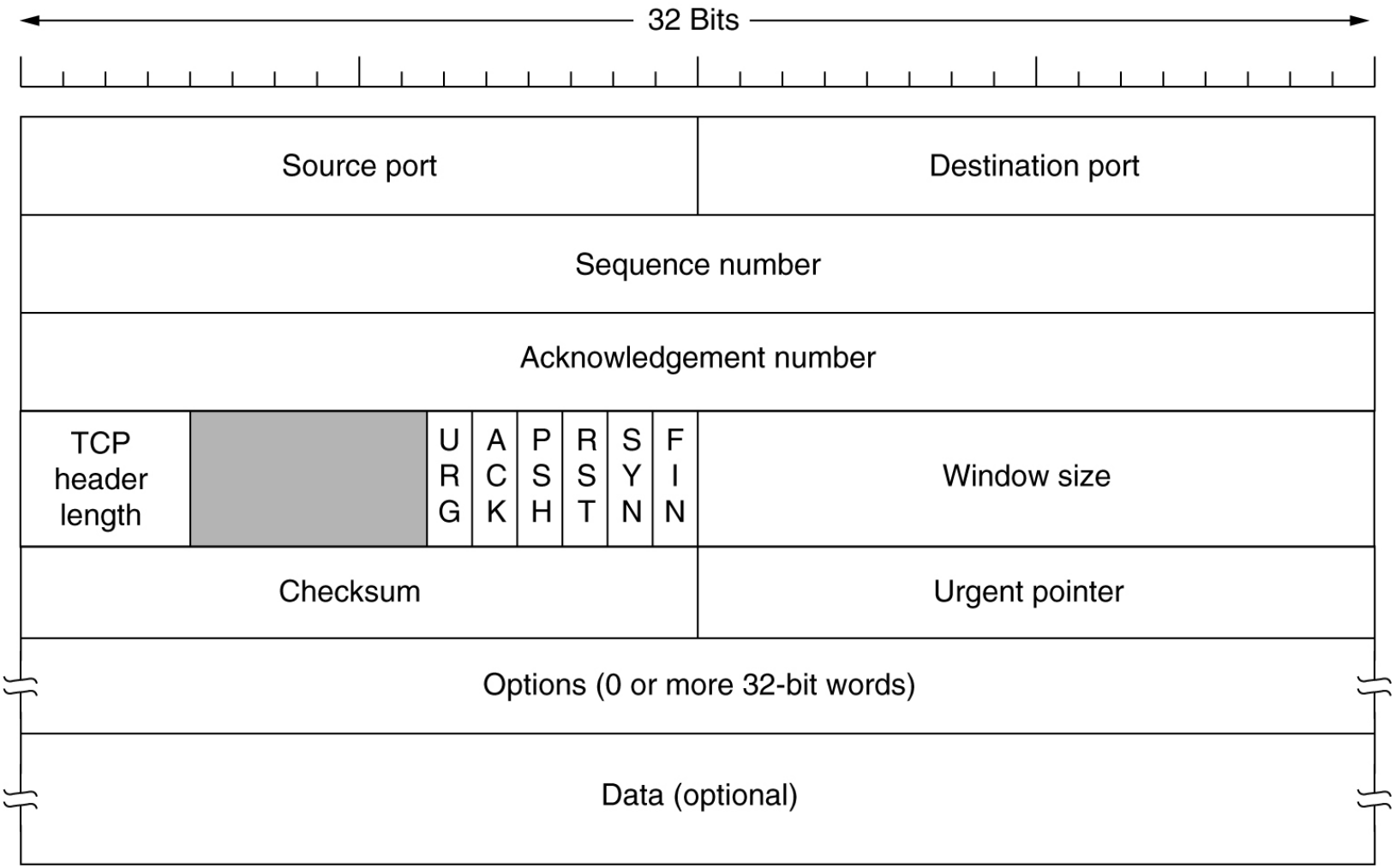
Q: URG vs PSH?

TCP options

- TLV-like options
  - option-kind: 1-byte
  - option-length: 1-byte, for the entire option
  - option-data:variable length
- E.g., Maximum Segment Size (MSS)
  - exchanged during connection establishment
  - default: 536 bytes
- E.g., Selective Acknowledgment (SACK)
  - stay tuned: “TCP congestion control”
- Zero-padding to keep 32-bit alignment

Q: big vs small segment?

TCP packet header (again)



Q: any redundant fields?

This lecture

- TCP
  - services provided by TCP
    - protocol mechanisms to support TCP services
  - TCP header fields
    - control information exchanged to support TCP
- Explore further
  - Qs on previous slides
  - the limitation of TCP and proposed changes
    - http://www.icir.org/floyd/tcp_small.html

Next lectures

- June 7: TCP connection management
- June 11: TCP flow control
- June 14: TCP error control
- June 18 and 21: TCP congestion control