ICS 351: Today's plan

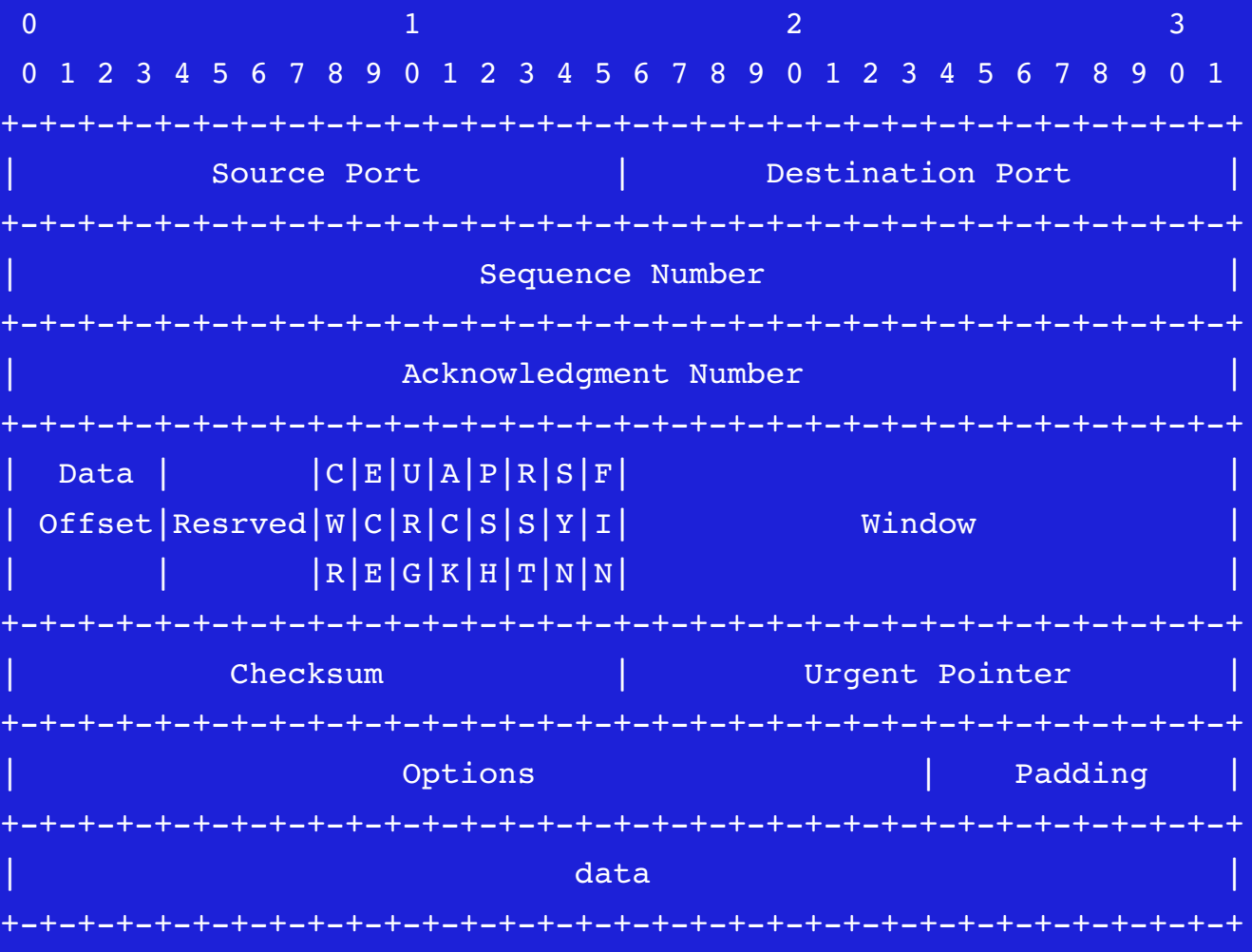
- TCP and UDP
- port numbers
- congestion control

TCP and UDP

- layered above IP (header follows IP header)
- provide another layer of addressing: port numbers, which let us identify applications (sockets) within hosts
- UDP (RFC 768) does not provide much else:

TCP Header

- RFC 793 and RFC 1122



TCP Behavior

- TCP control bits (SYN, FIN, ACK, RST) help maintain TCP connections
- three-way handshake is SYN, with SYN-ACK in answer, and a final ACK to confirm receipt of the second packet
- 32-bit sequence number, ack number count bytes rather than packets
- an ack is sent, almost for free (piggyback) in every packet except the first
- window tells the recipient how many more bytes (past the ack) the sender of this packet is willing to receive -- flow control, slowing down the sender to avoid overwhelming a slow receiver
- this is the flow control window
- setting the window to zero forces the sender to stop
- in general, TCP can send one window every RTT (round-trip time)

port numbers

- an IP address identifies an interface, and by extension a machine
- a port number identifies an application within a machine
- servers *listen* on specific, *well-known* ports
- each local port can be used for multiple *sockets*, as long as (at least) one of these is different: local/remote IP, local/remote port, protocol
- note:
- o a socket has a **local** and a **remote** port (and likewise for IP addresses)
- o a packet has a **source** and a **destination** port (and likewise for IP addresses)
- local and remote make sense on a host, whereas source and destination make sense for a packet

Congestion Collapse

- reminder: the network hardware might be working fine, but if the software fails, the network goes down
- e.g. if the routing tables include loops, packets will not get delivered
- imagine a retransmission mechanism where, when a packet is lost, I resend the lost packet and also a new one
- if a packet is lost due to congestion, the first little congestion experienced will likely lead to more congestion
- this happened a few times in the 1970's -- the network hardware was working fine, but almost no data would get through

TCP Congestion Control

- to control congestion, TCP slows down substantially (half the speed) whenever packets are lost
- TCP then slowly speeds up its transmission rate when no packets are lost.
- this is controlled by a window that (unlike the flow control window above) is maintained on each sender, and never communicated: the congestion window
- when packets are lost, the congestion window shrinks to about half its previous size (the details are complicated!)
- every RTT when no packets are lost, the congestion window grows by one packet
- the effective window is the smaller of the flow control window and the congestion window
- since each TCP can send one window every RTT, shrinking the window slows down sending
- TCP also has other mechanisms to lessen congestion, including binary exponential backoff on retransmissions, and adaptive timers to more reliably detect packet loss