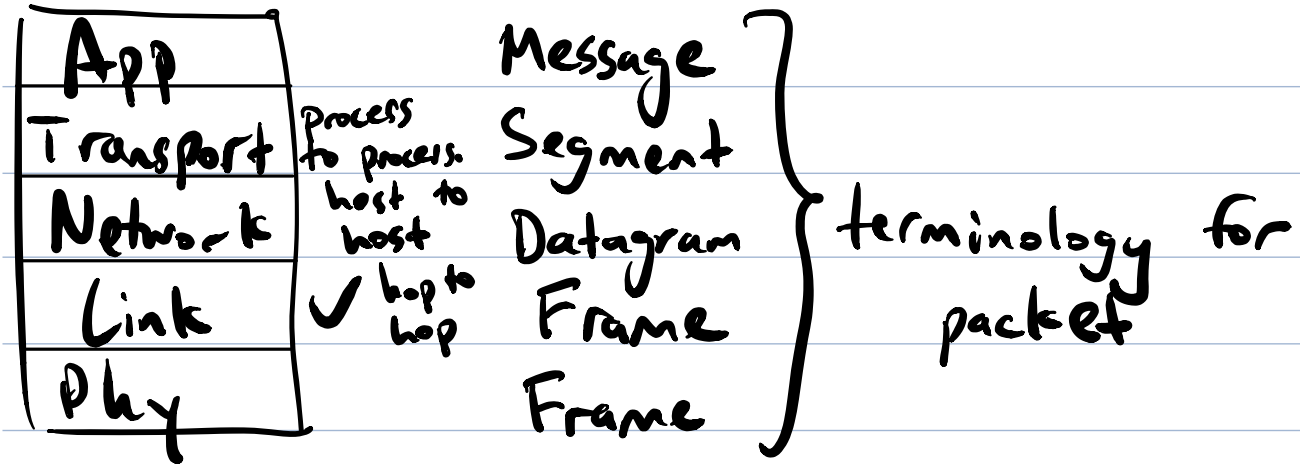
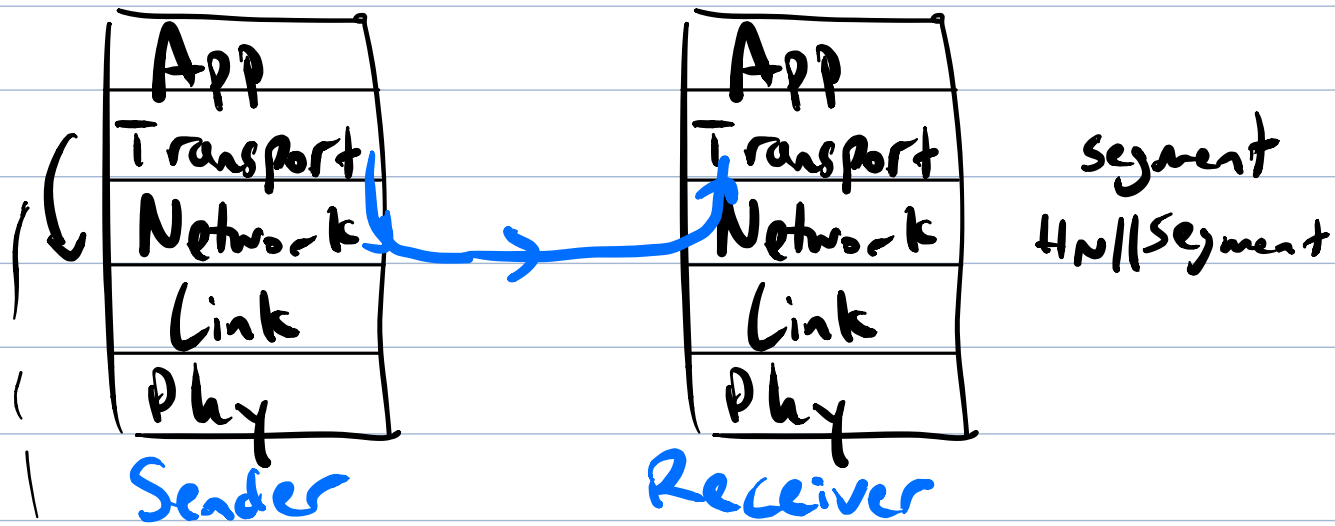
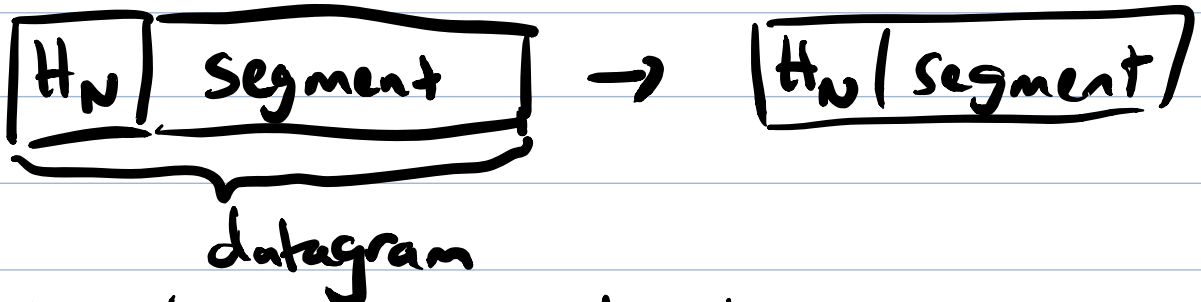


4.1 Network layer services

- transport segment from sending to receiving host

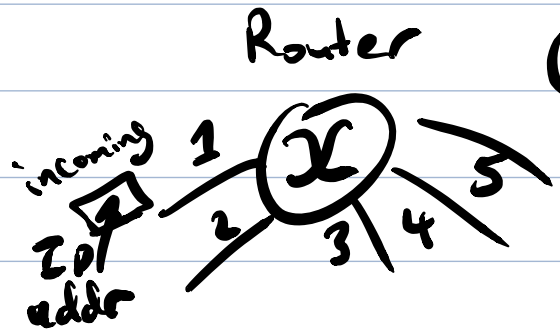


encapsulate segment into datagram

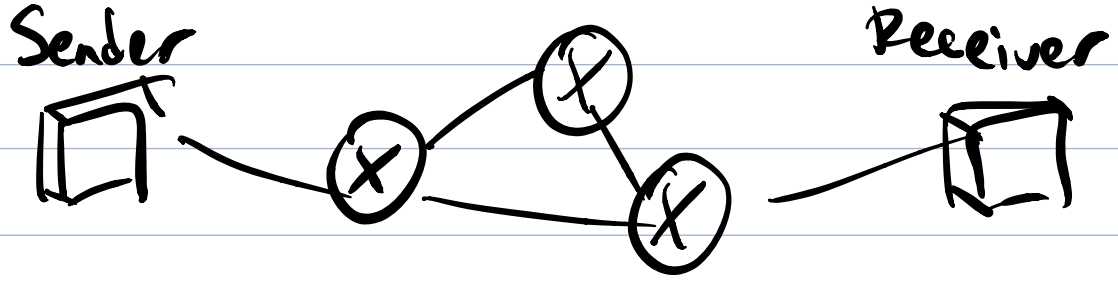


- network layer protocols in every host and router

- router examines header fields in all IP datagrams passing through it



① forwarding: which intf to pass to?



②

routing: determine route packets take from source to dest.

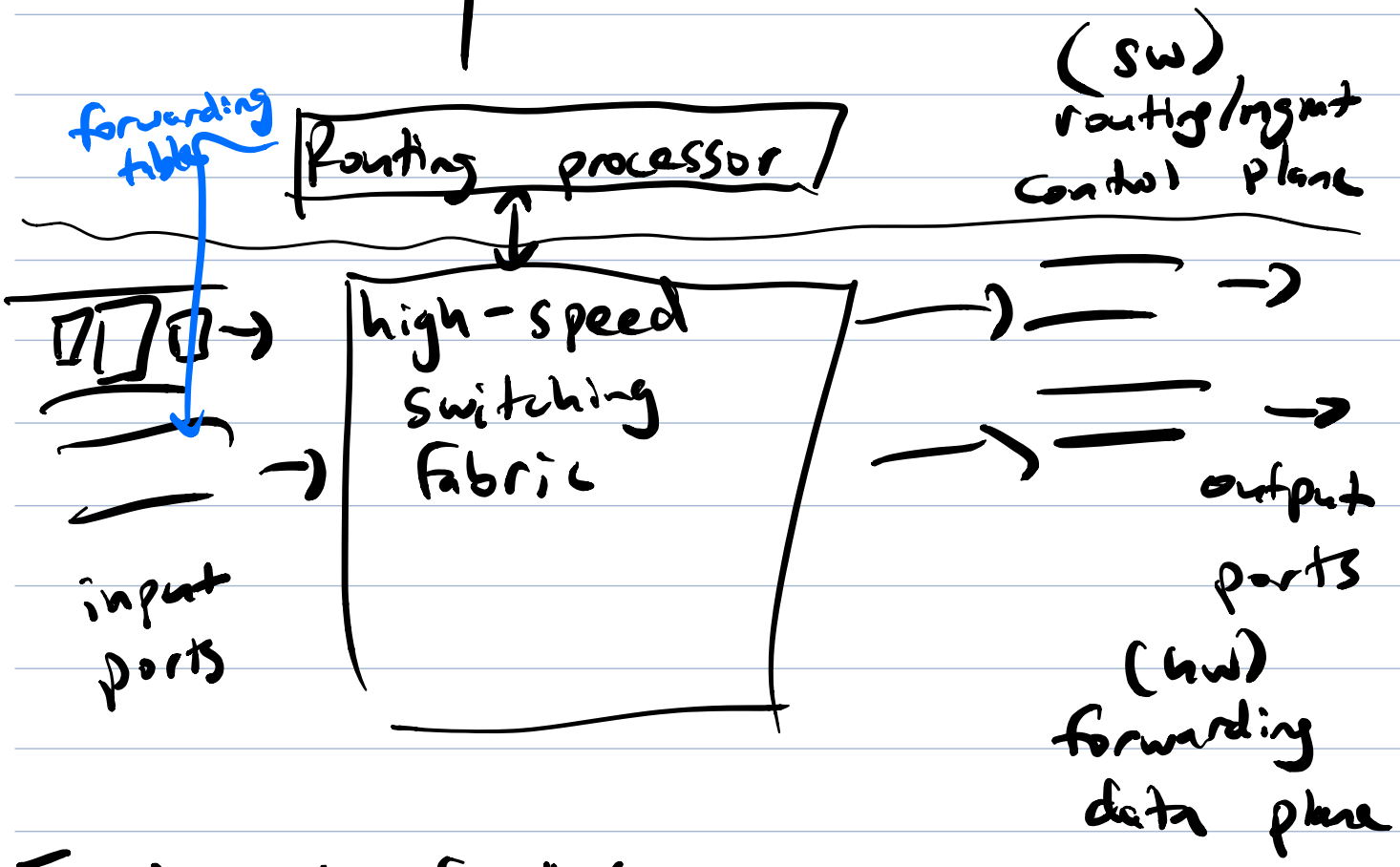
4.3 Router overview

2 key functions:

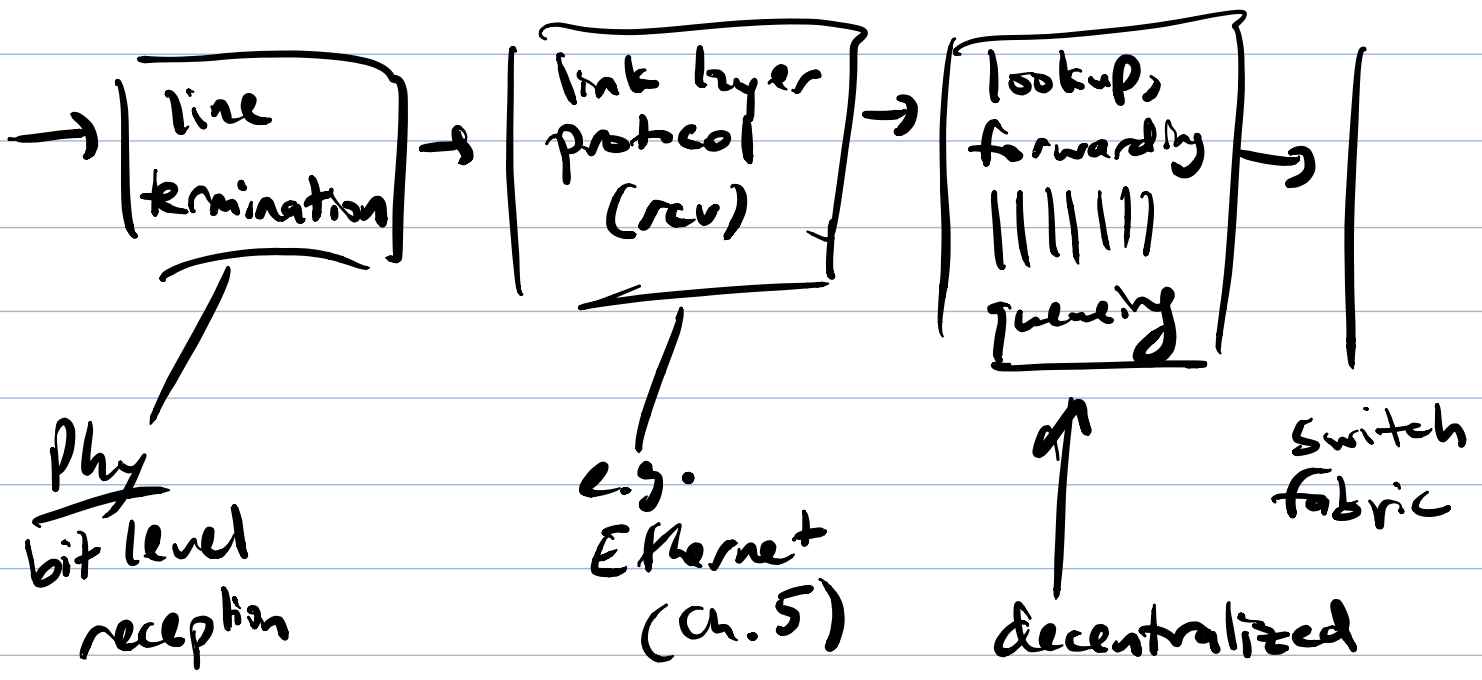
- run routing algorithms/protocol (RIP, OSPF, BGP)
- forwarding datagrams from incoming to outgoing link

The outcome of running a routing algorithm will be a forwarding table:

IP addr.	Interface
...	...

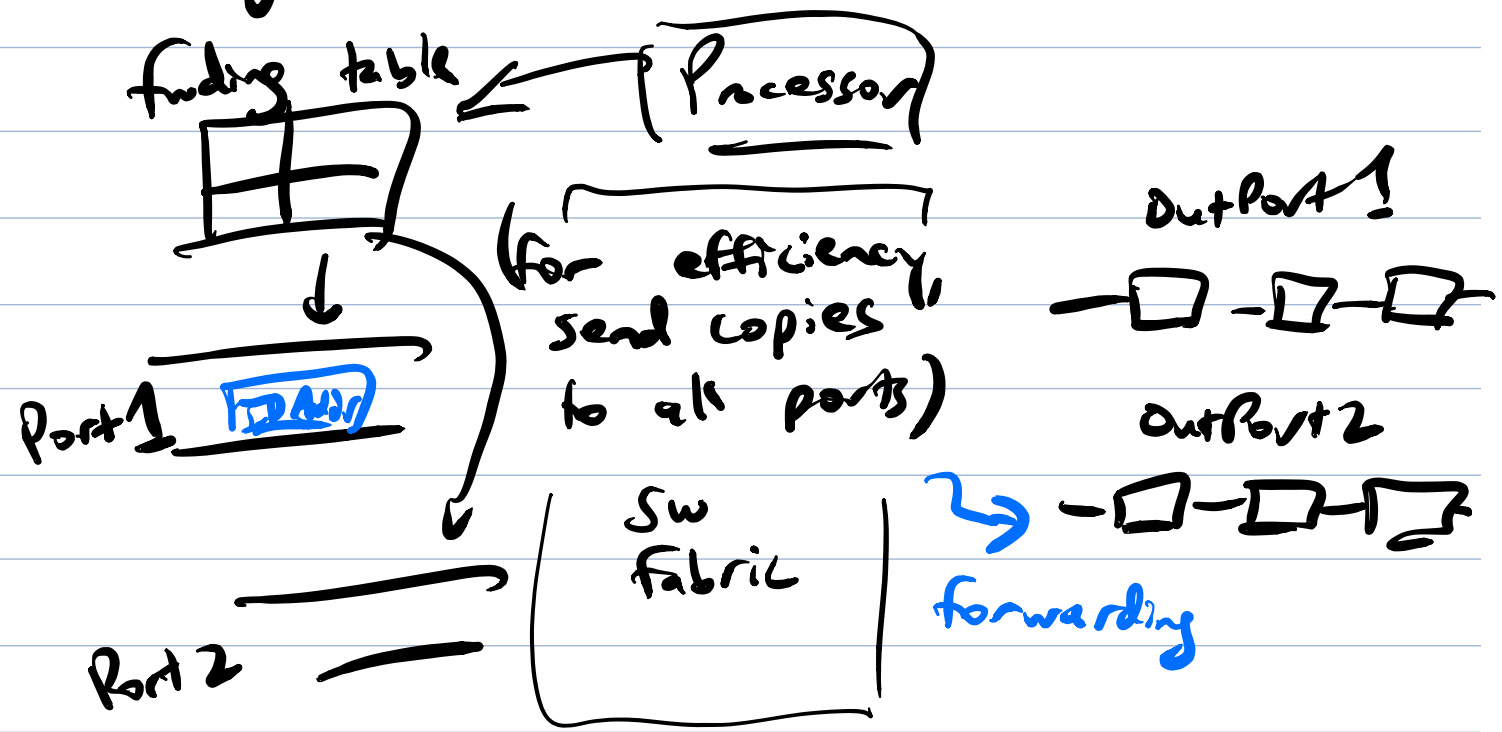


Input port functions



- given datagram dest, lookup output port using fwding table in input port memory ("match plus action")

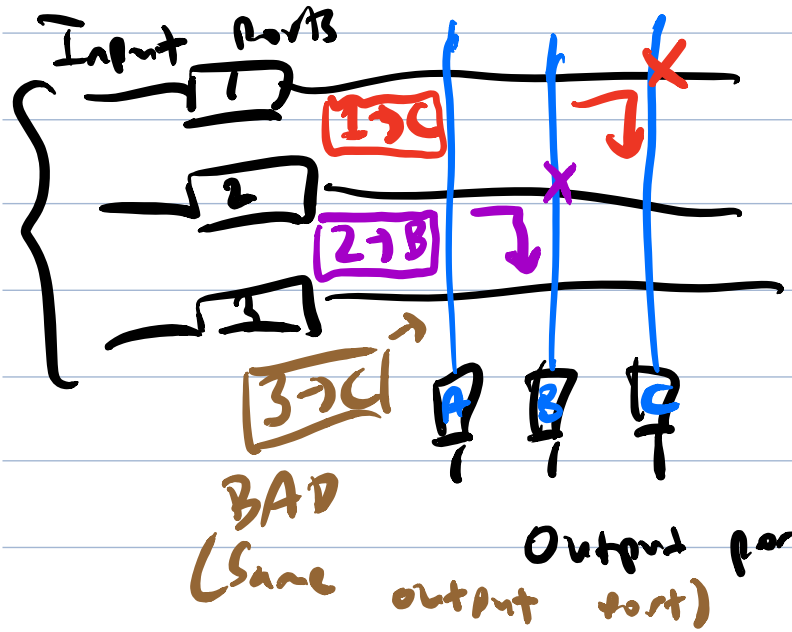
- goal: complete input port processing at 'line speed'
- queuing: if dgrams arrive faster than forwarding rate into switch fabric



Want to perform forwarding at least at the line speed (input link capacity) to avoid unbounded queue buildup

Switching fabric:
- transfer packet from input port memory to output port memory
- switching rate = rate at which packets can be txed from inputs to outputs
  - ↳ should be N * individual input line speed for N inputs

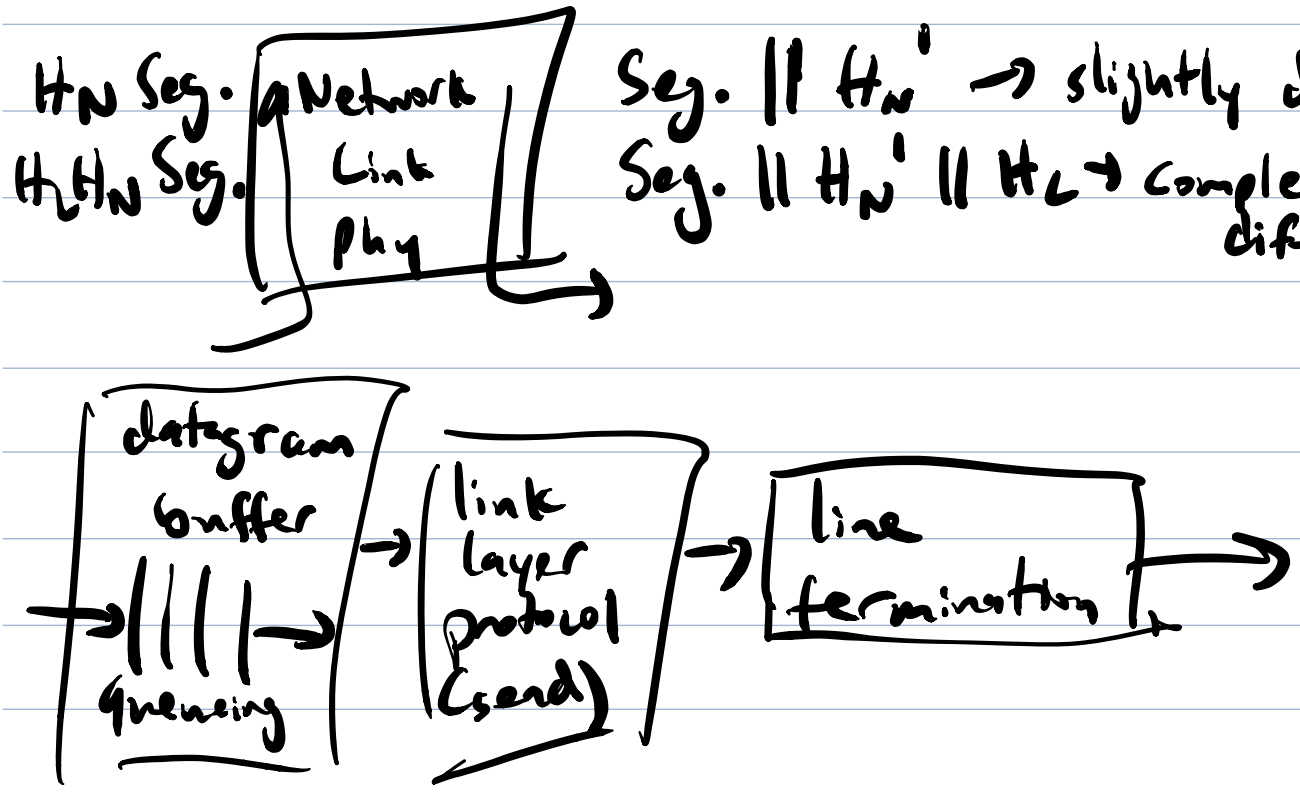
- types: memory, bus, cross bar
  - ↳ memory: 1st gen routers
  - ↳ copy packet to system's memory, then, after processing copy to output port
  - ↳ speed limited by memory b/w (2 bus crossings per datagram)
  - ↳ bus: datagram from input port memory to output port memory via shared bus
  - ↳ only 1 tx on the bus at a time ∴ (o/w collision) but sufficient speed, Cisco 5600: 32Gbps for institutional network
  - ↳ interconnection network / cross bar
  - ↳ initially for processor connection in multiprocessor
  - ↳ fragment datagram into fixed length cells, switch cells through the fabric



→ Cisco 12000: switches 60 Gbps through interconnection network

Output ports

- buffering required when datagrams arrive faster than the tx rate
- scheduling discipline chooses among queued datagrams for transmission

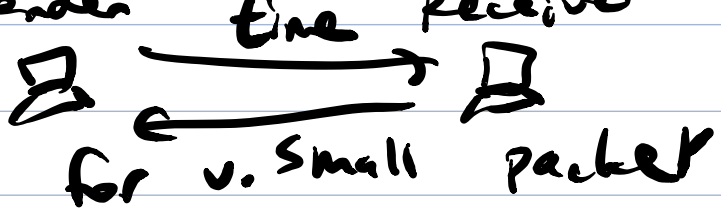


Want to keep output link utilization near 100%.

RFC 3439 rule of thumb: avg. buffering equal to typical roundtrip time (RTT), say 250 ms, times link capacity C.

e.g. C = 10 Gbps link * 0.25 s RTT
= 2.5 Gb buffer

- recent recommendation: with N flows, buffering equal to $\frac{RTT \cdot C}{\sqrt{N}}$

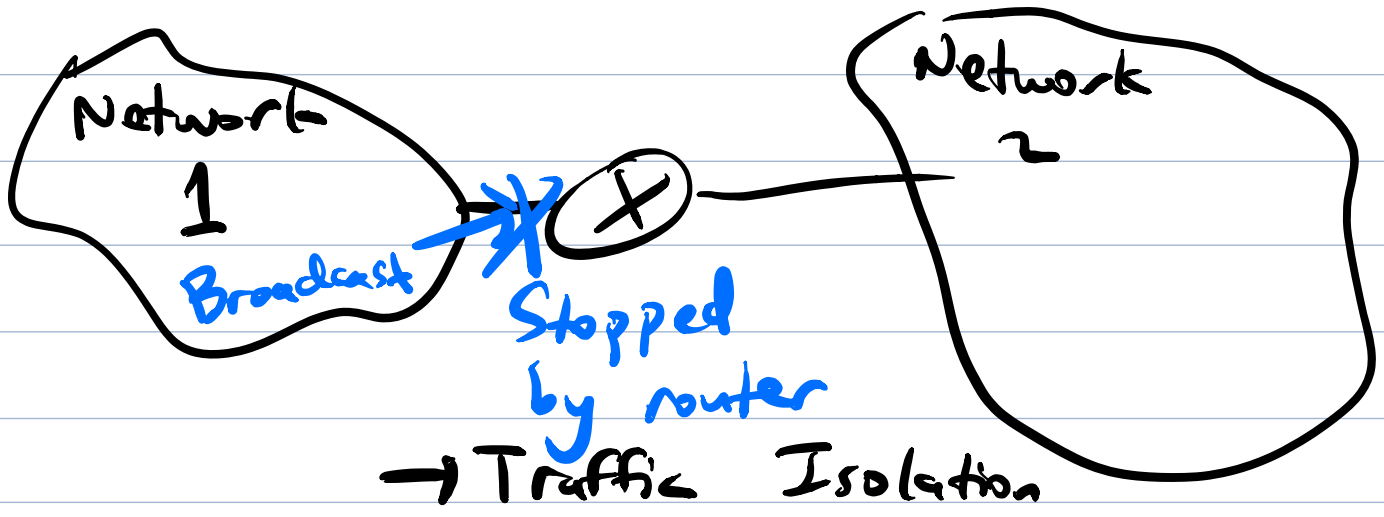
RTT: Sender  Receiver

for v. small packet

Input port queueing

Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward (due to output port contention and FIFO strict protocol)

	Router	Switch
Layer	Network	Link
Store-and-Forward	Yes	Yes
Fwd Table	Yes (algos, IPs)	Yes (learns)
Traffic Isolation	Yes	No
Processing delay	Slower	Faster
Needs config?	Yes	Plug-and-play No.
Broadcast storm?	No	Yes



→ Traffic Isolation

Broadcast at 1 intf of router not delivered to other intf.
For switches, broadcast duplicated at all other interfaces.

4.4 IP Datagram Format

Transport : TCP, UDP

routing protocols → find table
Network

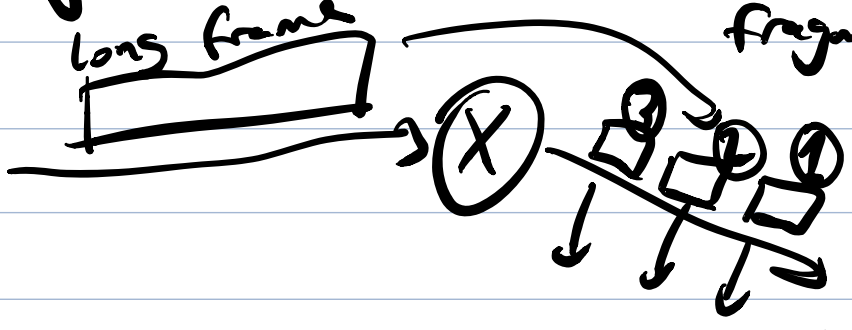
IP protocols
- addressing, datagram format, packet handling

ICMP
- error reporting

- router signaling (control)

Transport | Segment |
Network | H | segment |

Version (4 bits): 4 (=0100) or 6
Header length (4 bits): unit is 4 bytes
20 byte header = 5 (=0101) (words)
DSCP (differentiated services code point): 1 byte, type of data carried, 6-bit DSCP + 2-bits 46 → High 0 → Low
Length (16 bits): pkt length in bytes
16-bit ID: connects fragmented IP packets
3-bit flags: <Not used, Don't frag, More frags to follow>
Fragment offset x 2^3: pos of fragment in the original packet, 29 bits



TTL (time-to-live): max # remaining hops dec. by 1 at each router
↳ prevent circulating 1 byte forever

TTL = 0 => discard @ router
upper layer: protocol to deliver payload to (TCP = 6, UDP = 17) 1 byte
Header checksum: to detect bit errors in packet <u>header</u> (data errors are ignored)
source IP addr - 32 bit
dst IP - 32 bits

Checksum: Add all 16-bit blocks of header
add carry to the rest = Temp
take 1s C of Temp to get checksum
replace 0000 in checksum with ↗

add all 16 bit blocks of header + carry
1s C = 0000 => no errors