Queue Management + Middleboxes

Soheil Abbasloo
Department of Computer Science
University of Toronto

Fall 2022

Outline

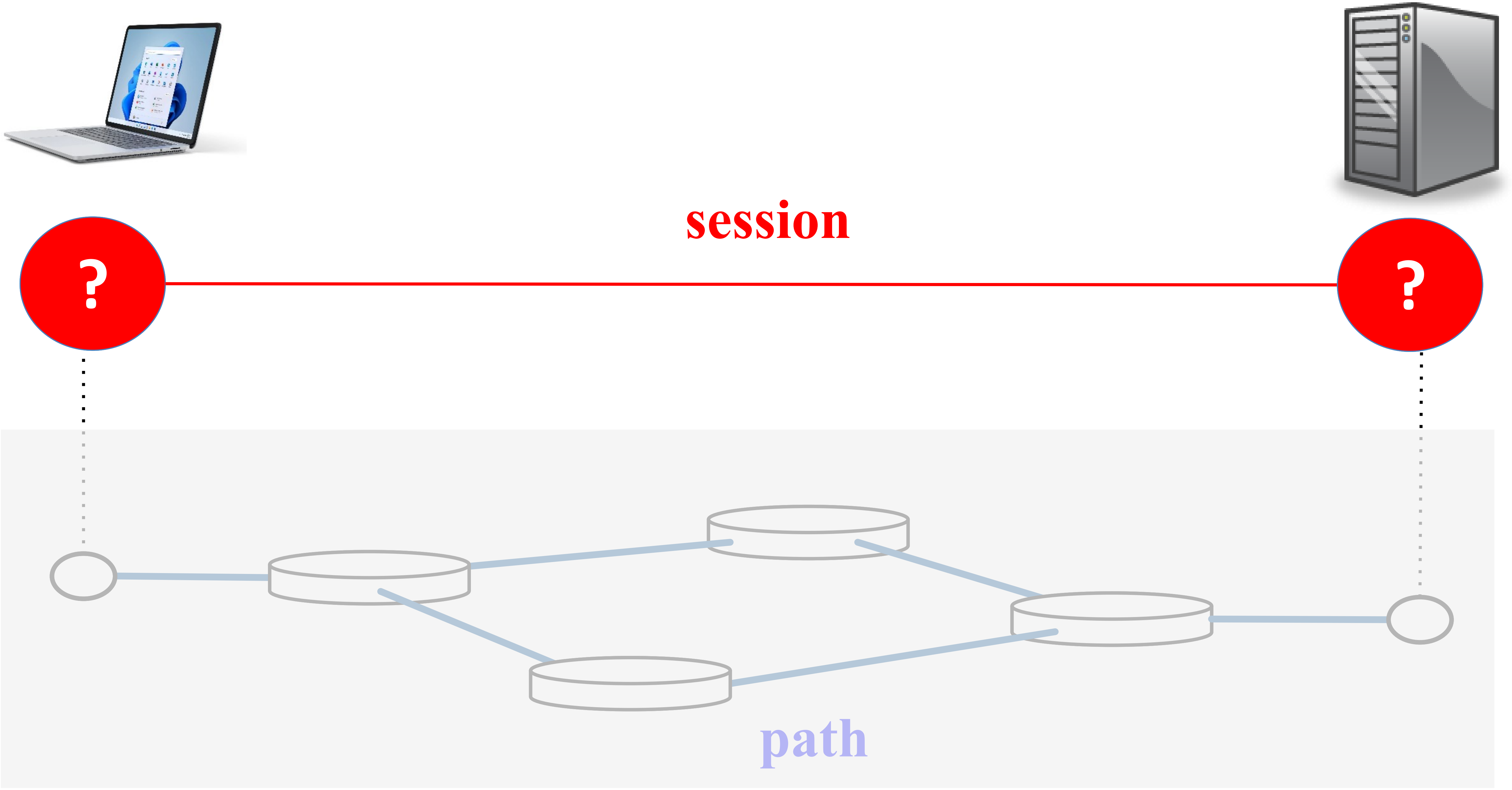
- Queue Management
  - Queues
  - Early Congestion Detection
  - Link Scheduling
  - QoS
- Middlebox
  - Firewall
  - NAT
  - Load Balancer
  - Tunneling

Announcement …

- Final Exam
  - December 14th
  - For exact location and time, check this:
    - https://www.artsci.utoronto.ca/current/faculty-registrar/exams-assessments/exam-assessment-schedule#exam-assessments-schedule-accordion-1

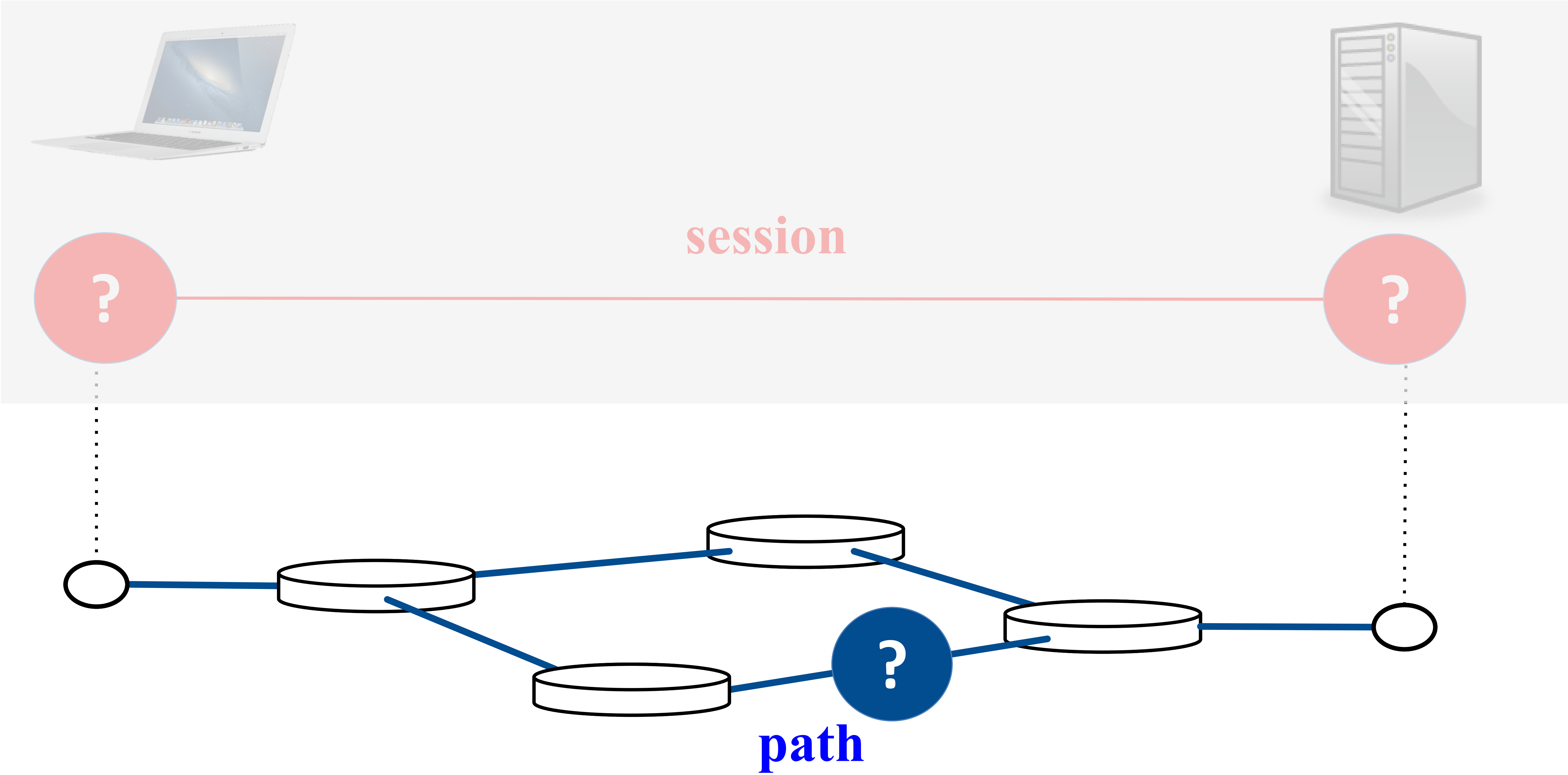
Last Time: Congestion Control

What can the **end-points** do to collectively make good use of shared underlying resources?



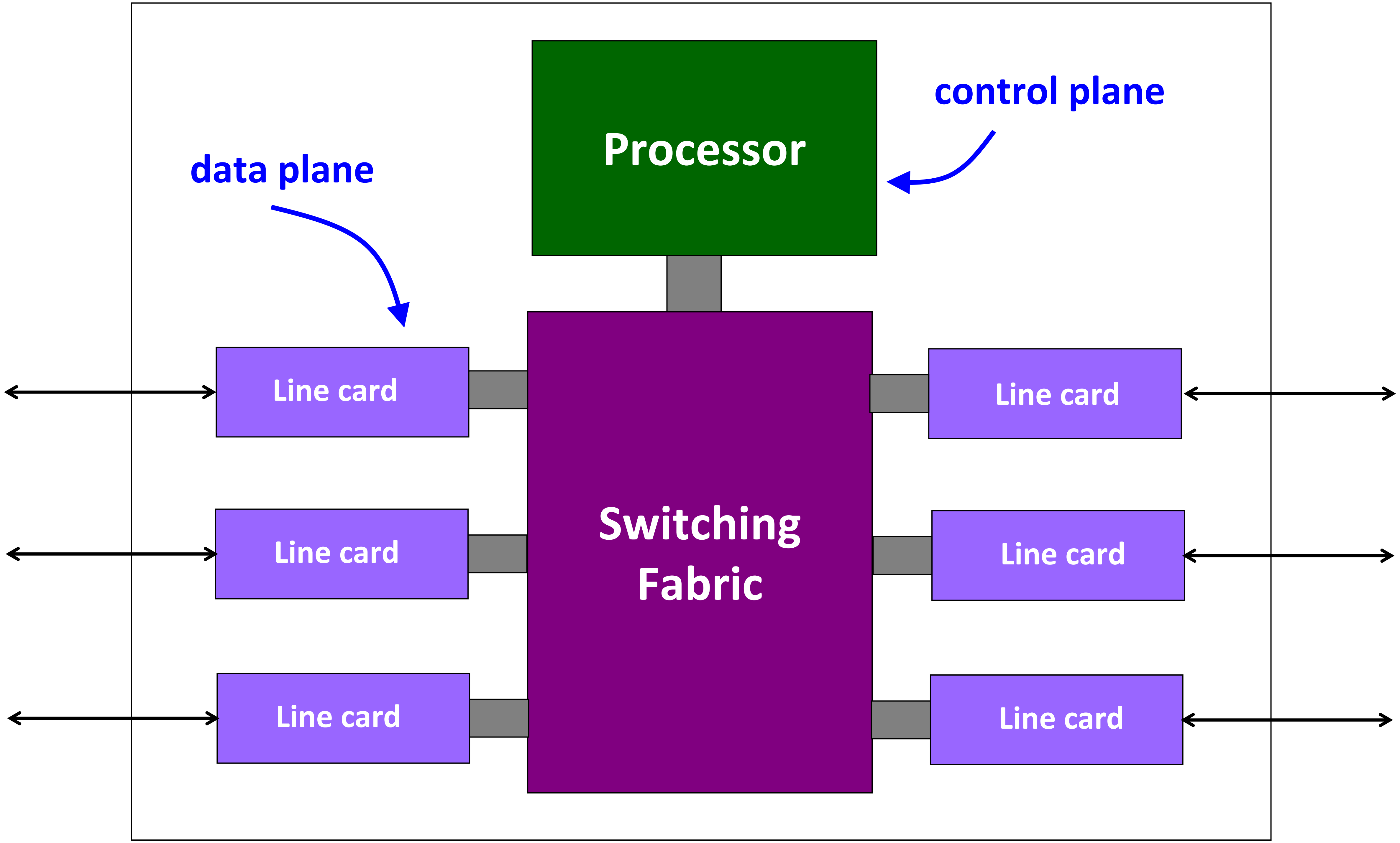
Today: Queue Management

What can the individual **links** do to make good use of shared underlying resources?



Packet Queues

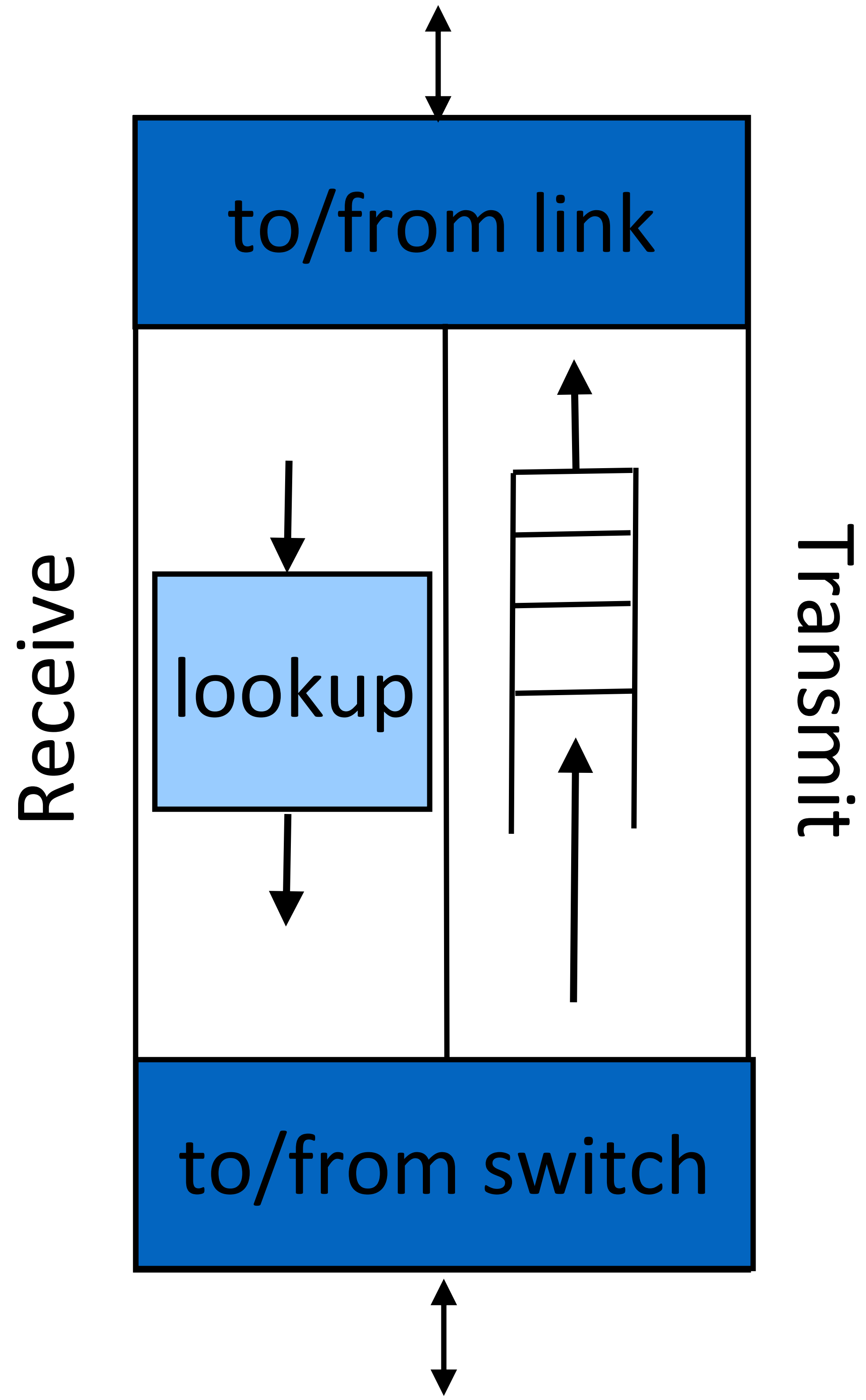
Router



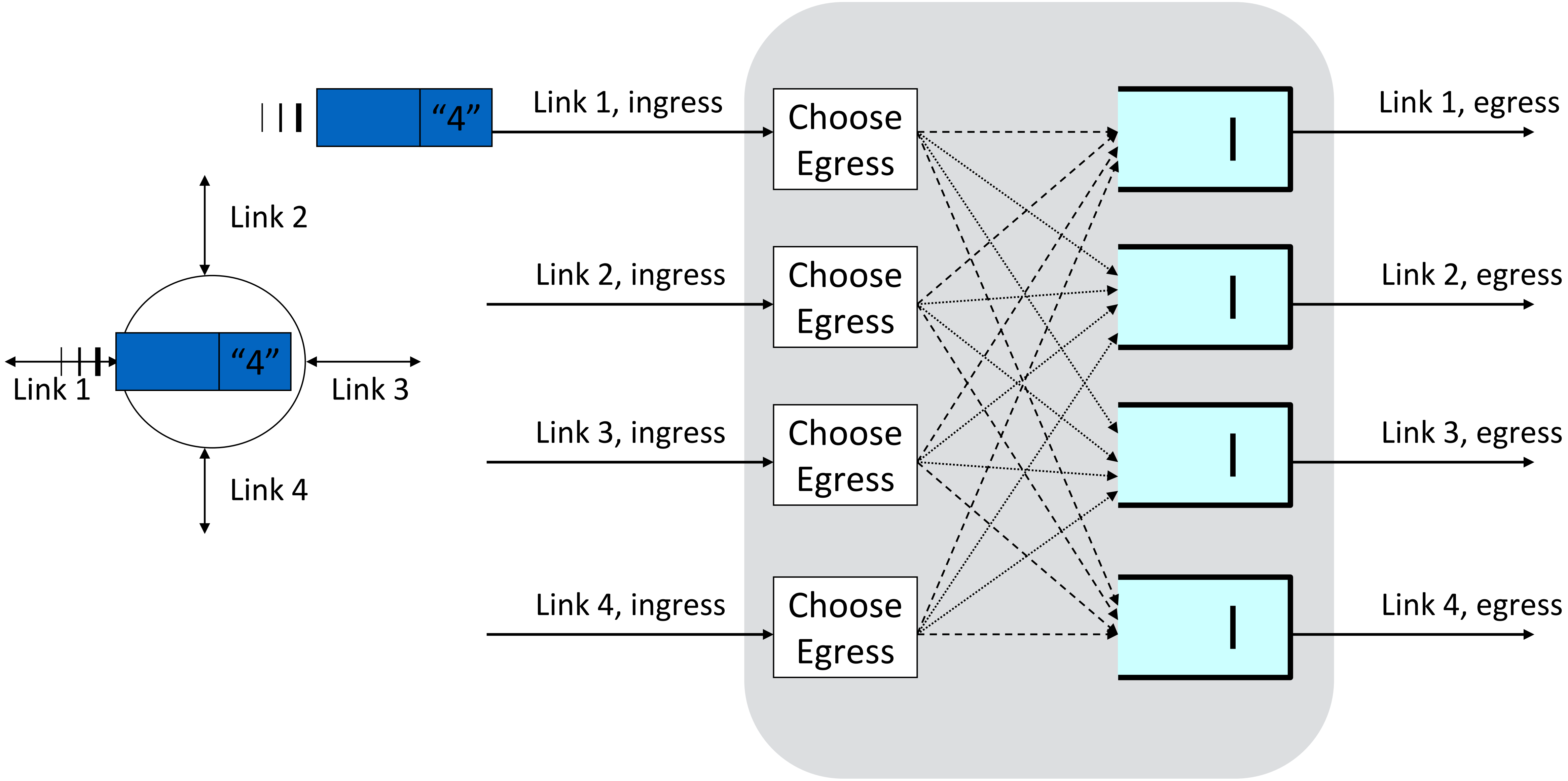
control plane
Processor
data plane
Line card
Line card
Line card
Switching
Fabric
Line card
Line card
Line card

Line Cards (Interface Cards, Adaptors)

- Packet handling
  - Packet forwarding
  - Buffer management
  - Link scheduling
  - Packet filtering
  - Rate limiting
  - Packet marking
  - Measurement



Packet Switching and Forwarding: An **Output Queue** Structure

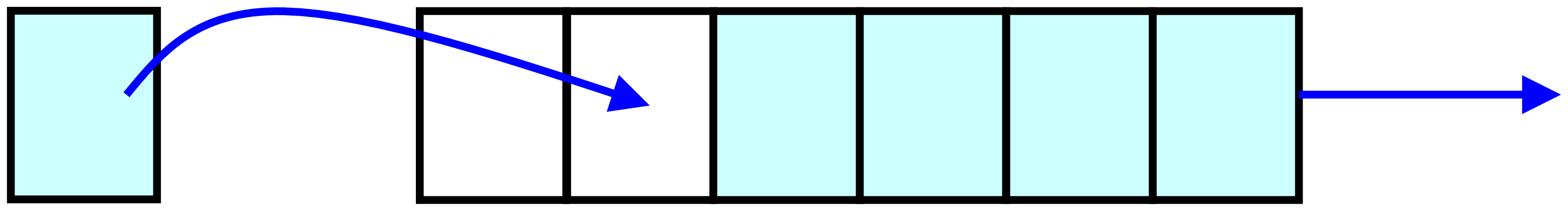


Queue Management Issues

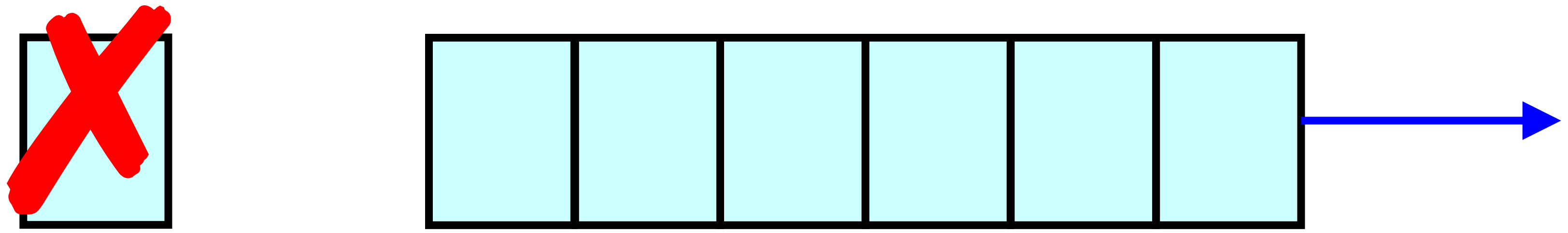
- Scheduling discipline
  - Which packet to send?
  - Some notion of fairness? Priority?
- Drop policy
  - When should you discard a packet?
  - Which packet to discard?
- Goal: balance throughput and delay
  - Huge buffers minimize drops, but add to queuing delay (thus higher RTT, longer slow start, …)

FIFO Scheduling and Drop-Tail

- Access to the bandwidth: first-in first-out queue
  - Packets only differentiated when they arrive

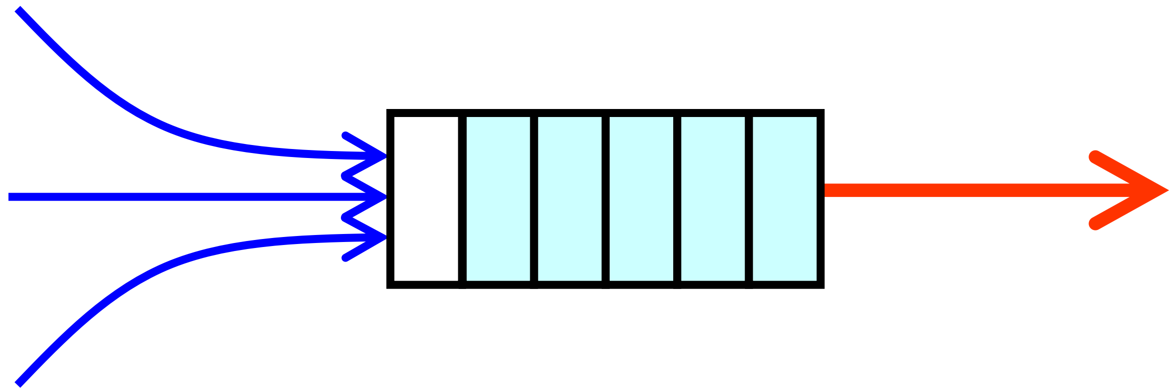


- Access to the buffer space: drop-tail queuing
  - If the queue is full, drop the incoming packet



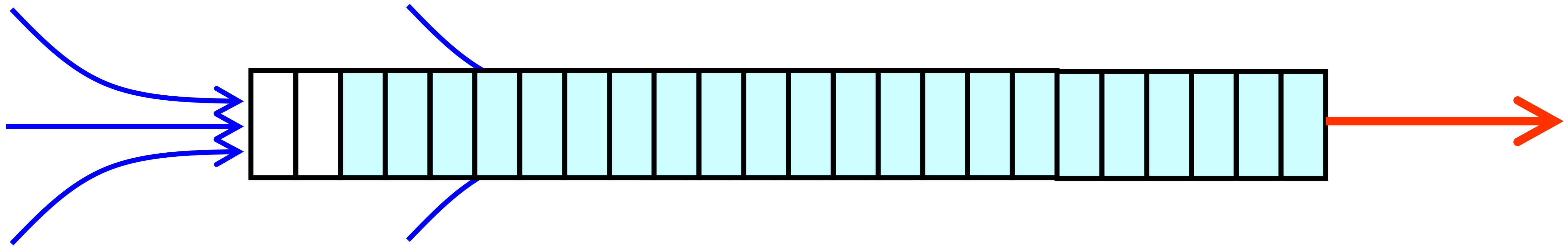
Bursty Loss From Drop-Tail Queuing

- *Most Current* congestion control algorithms depend on packet loss
  - Packet loss is indication of congestion
  - TCP additive increase drives network into loss
- Drop-tail leads to *bursty* loss
  - Congested link: many packets encounter full queue
  - Synchronization: many connections lose packets at once



Slow Feedback from Drop Tail

- Feedback comes when buffer is completely full
  - … even though the buffer has been filling for a while
- Plus, the filling buffer is increasing RTT
  - … making detection even slower

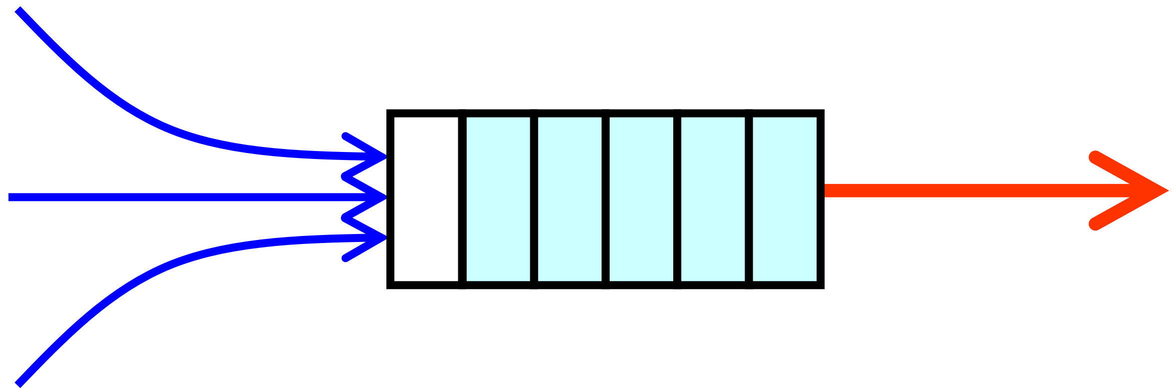


Any suggestions to resolve the Slow Feedback issue of Drop-Tail?

Early Detection of Congestion

Slow Feedback from Drop Tail

- Feedback comes when buffer is completely full
  - … even though the buffer has been filling for a while
- Plus, the filling buffer is increasing RTT
  - … making detection even slower
- Better to give early feedback
  - Get 1-2 connections to slow down before it's too late!



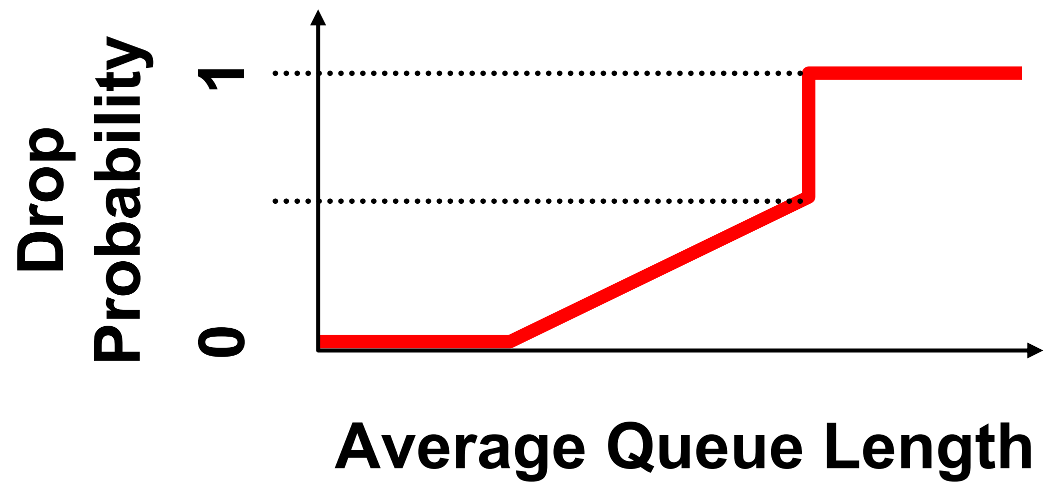
Sally Floyd and Van Jacobson's

Random Early Detection (RED)

1993

- An example algorithm for how we can *better manage packets drops*

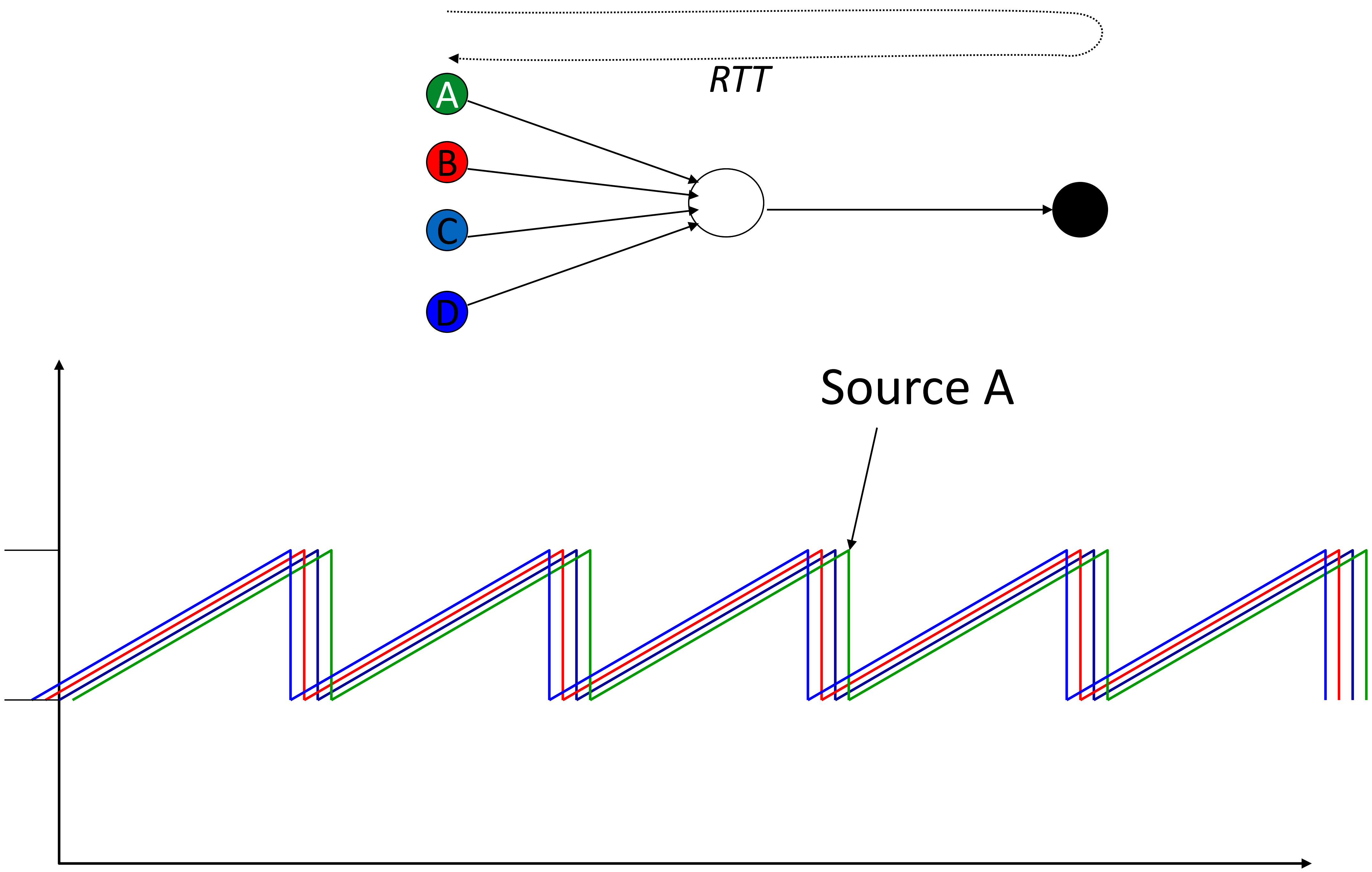
- Router notices that queue is getting full
  - … and **randomly** drops packets to signal congestion
- Packet drop probability
  - Drop probability increases as queue length increases
  - Set drop probability *f*(avg queue length)



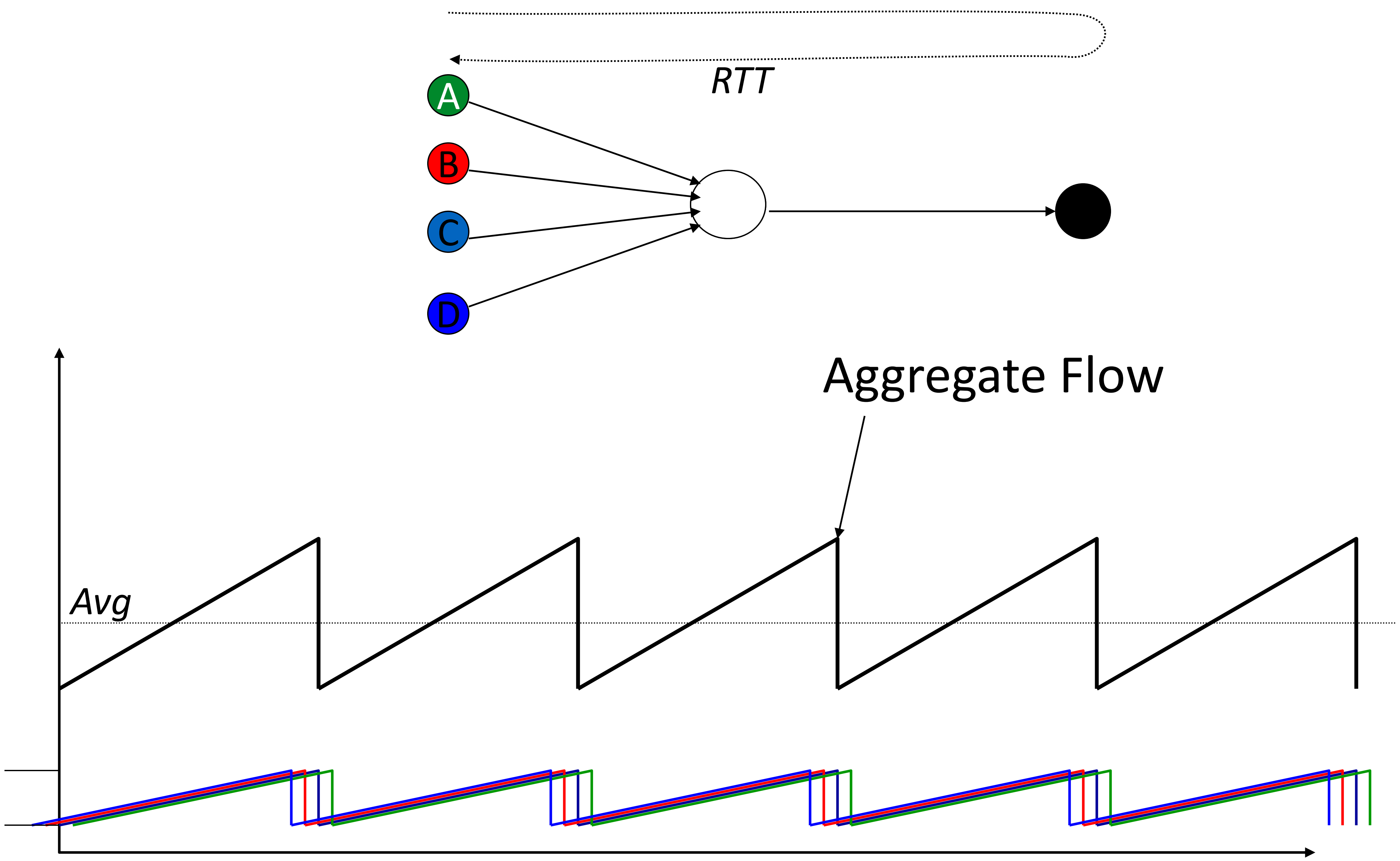
Properties of RED

- Drops packets before queue is full
  - In the hope of reducing the rates of some flows
- Drops packet in proportion to each flow's rate
  - High-rate flows selected more often
- Drops are spaced out in time
  - Helps desynchronize the TCP senders
- Tolerant of burstiness in the traffic
  - By basing the decisions on average queue length

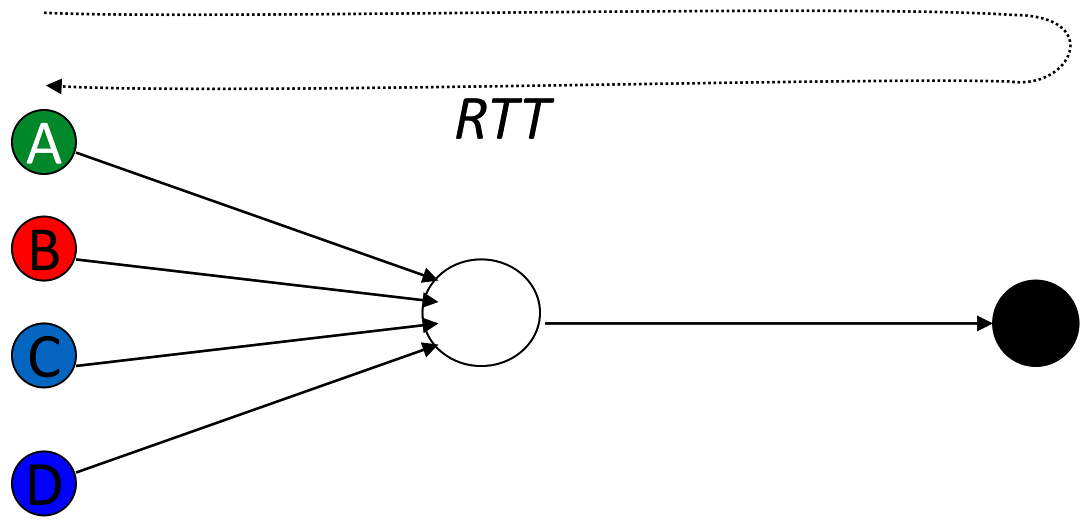
Synchronization of Sources

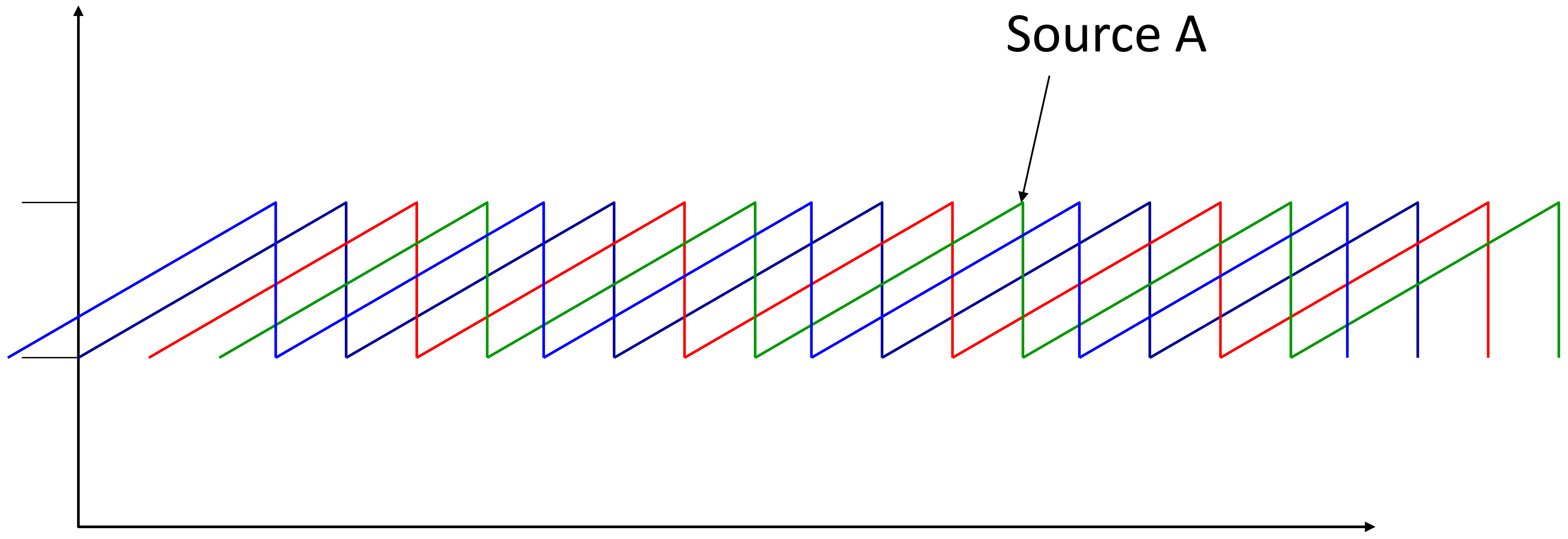


Synchronization of Sources

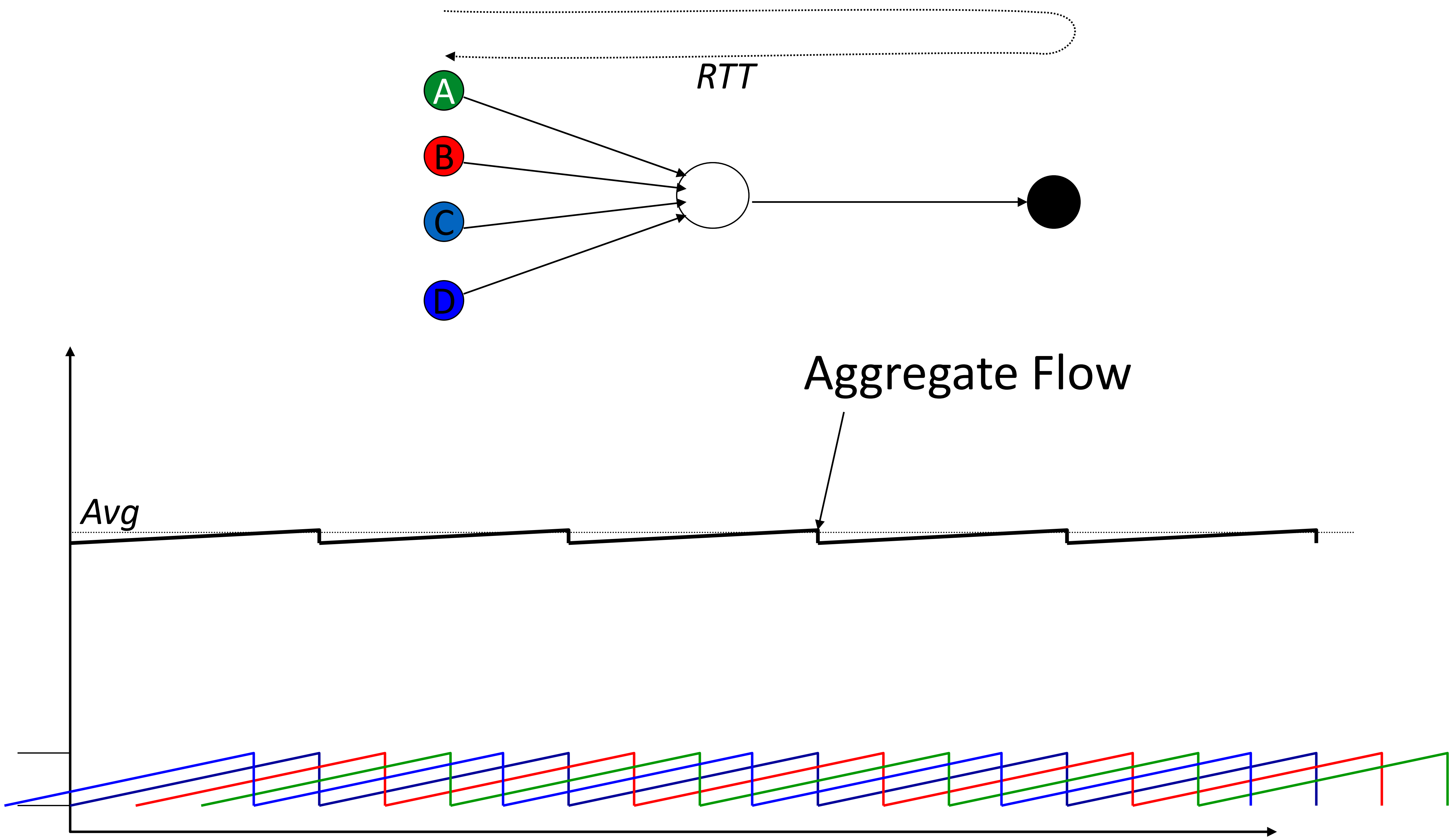


Desynchronized Sources





Desynchronized Sources



Problems With RED

- Hard to get tunable parameters just right
  - How early to start dropping packets?
  - What slope for increase in drop probability?
  - What time scale for averaging queue length?
- This issue was big enough for most people to go and use other solutions!
  - If parameters aren't set right, RED doesn't help
- Many other variations in research community
  - Names like "Blue", "FRED", …

Feedback: From Loss to Notification

- Early dropping of packets
  - Good: gives early feedback
  - Bad: has to drop the packet to give the feedback

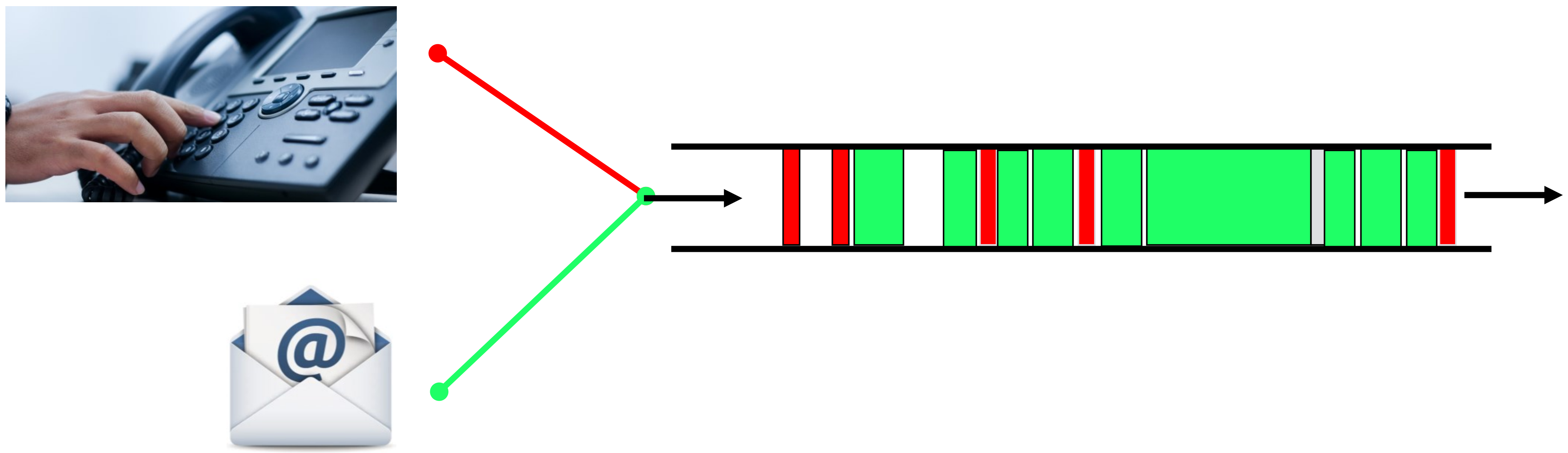
- Explicit Congestion Notification (ECN) (2001)
  - Router marks the packet with an ECN bit
  - Sending host interprets as a sign of congestion
  - Requires participation of hosts and the routers

- Is it a good idea to use ECN on the Internet?
- How about a private network?

Link Scheduling

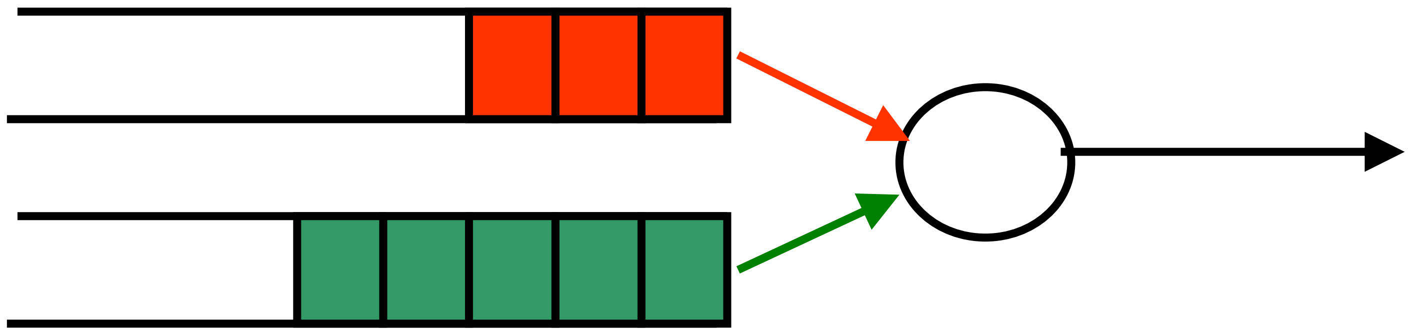
First-In First-Out Scheduling

- First-in first-out scheduling
  - Simple, but restrictive
- Example: two kinds of traffic
  - Voice over IP needs low delay
  - E-mail is not that sensitive about delay
- Voice traffic waits behind e-mail



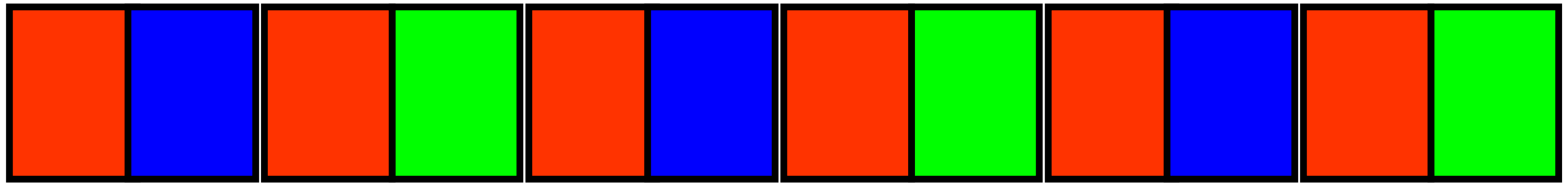
Strict Priority

- Multiple levels of priority
  - Always transmit high-priority traffic, when present
- Isolation for the high-priority traffic
  - Almost like it has a dedicated link
  - Except for (small) delay for packet transmission
- What is the problem with this?
  - Lower priority traffic may starve



Weighted Fair Scheduling

- Weighted fair scheduling
  - Assign each queue a fraction of the link bandwidth
  - Rotate across queues on a small time scale



50% red, 25% blue, 25% green

- Work-conserving
  - Send extra traffic from one queue if others are idle

Implementation Trade-Offs

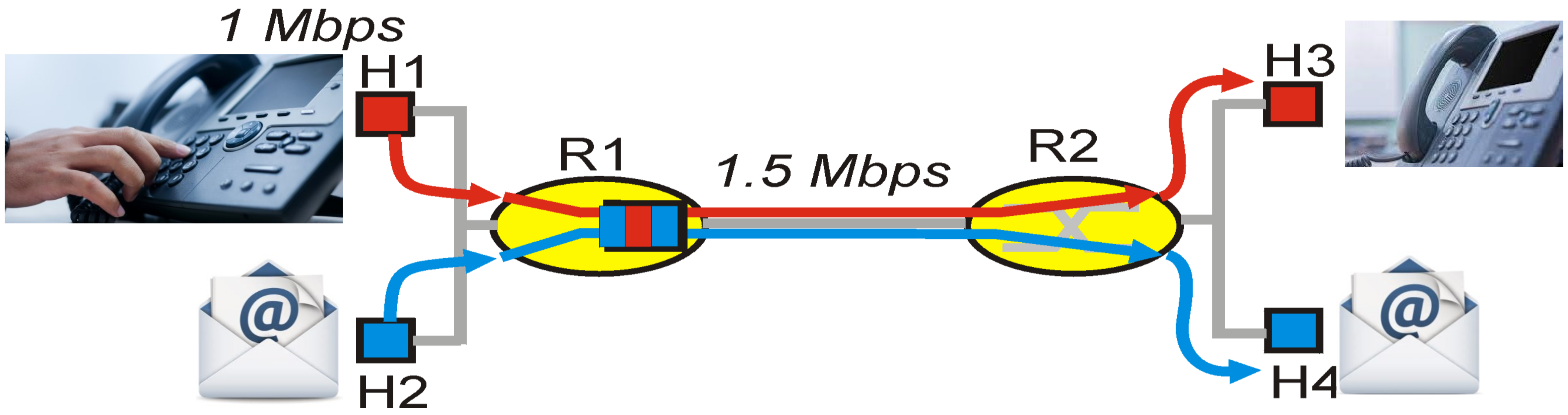
- FIFO
  - One queue, trivial scheduler
- Strict priority
  - One queue per priority level, simple scheduler
- Weighted fair scheduling
  - One queue per class, and more complex scheduler

Quality of Service Guarantees

Distinguishing Traffic

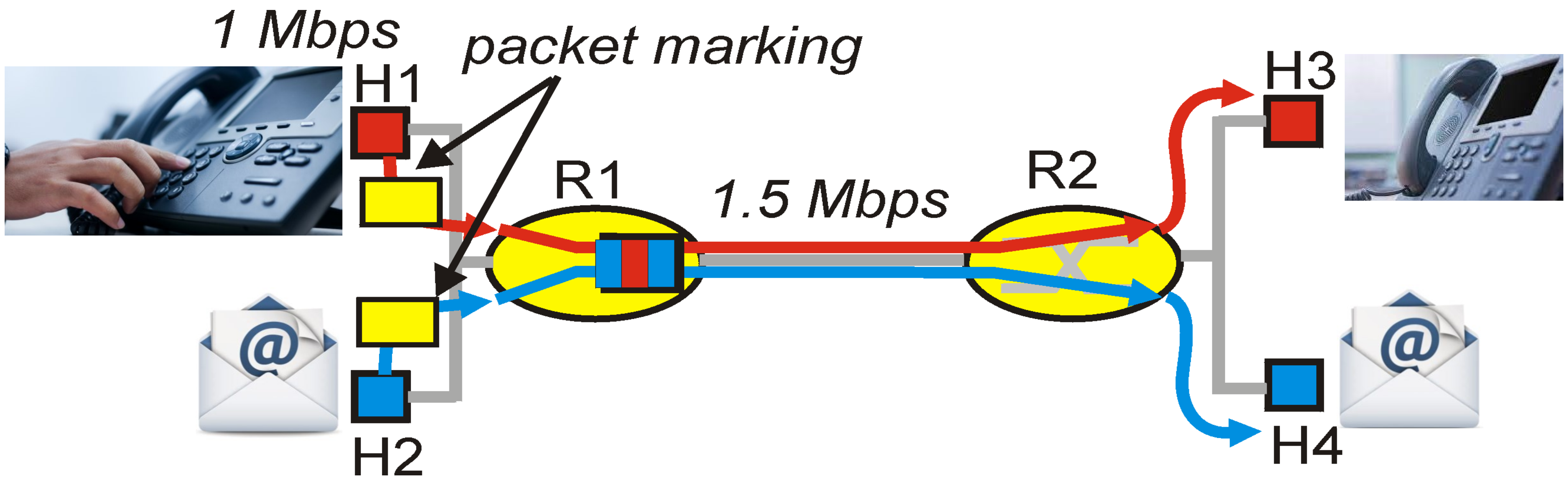
- Applications compete for bandwidth
  - VoIP and email sharing a link
  - E-mail traffic can cause congestion and losses
- Principle 1: **Packet marking**
  - So router can distinguish between classes
  - E.g., Type of Service (ToS) bits in IP header

What if someone marks
her email packets with ToS of VoIP?!



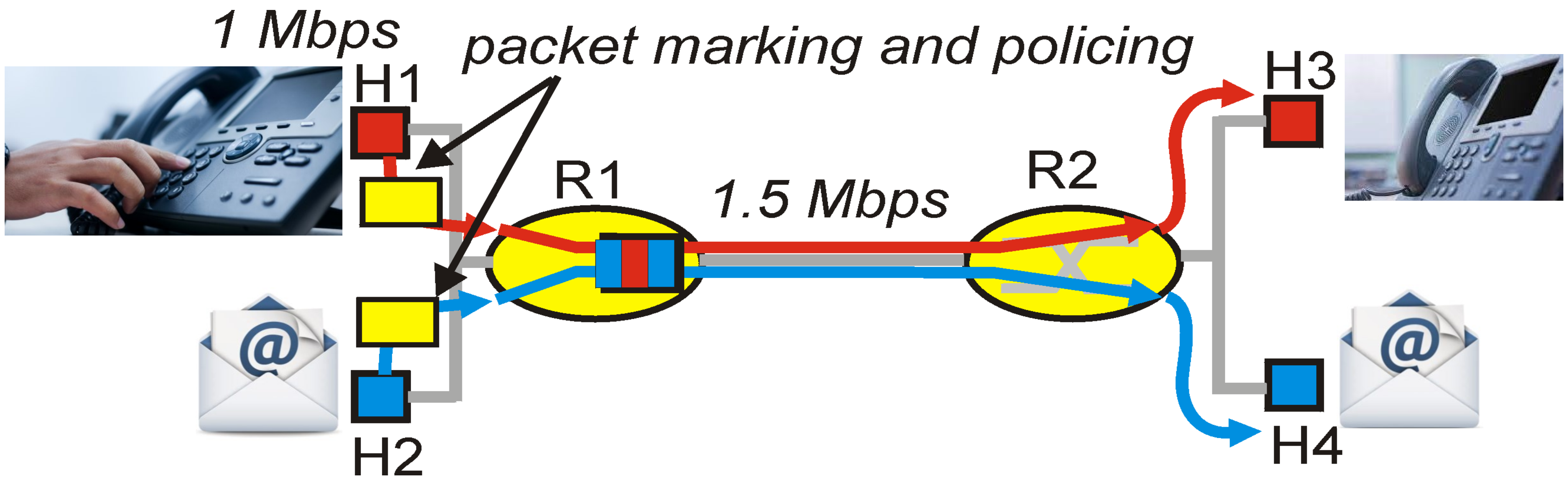
Preventing Misbehavior

- Applications misbehave
  - VoIP sends packets faster than 1 Mbps



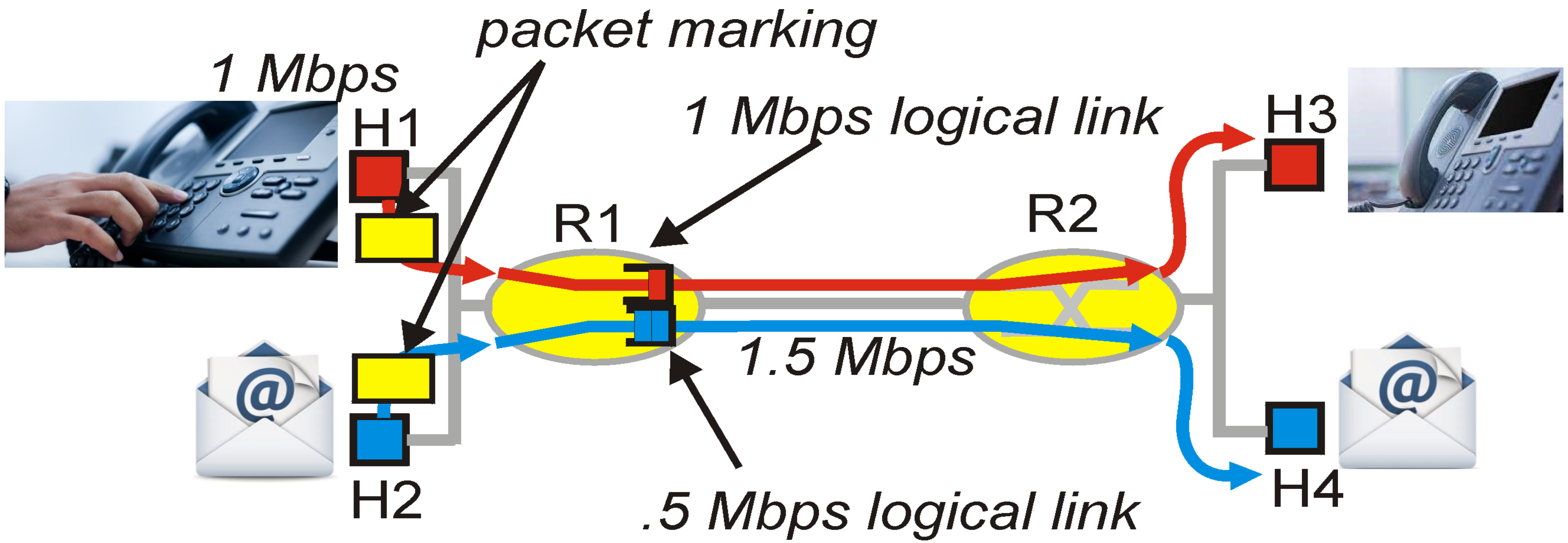
Preventing Misbehavior

- Applications misbehave
  - VoIP sends packets faster than 1 Mbps
- Principle 2: **Policing**
  - Protect one traffic class from another
  - By enforcing a rate limit on the traffic



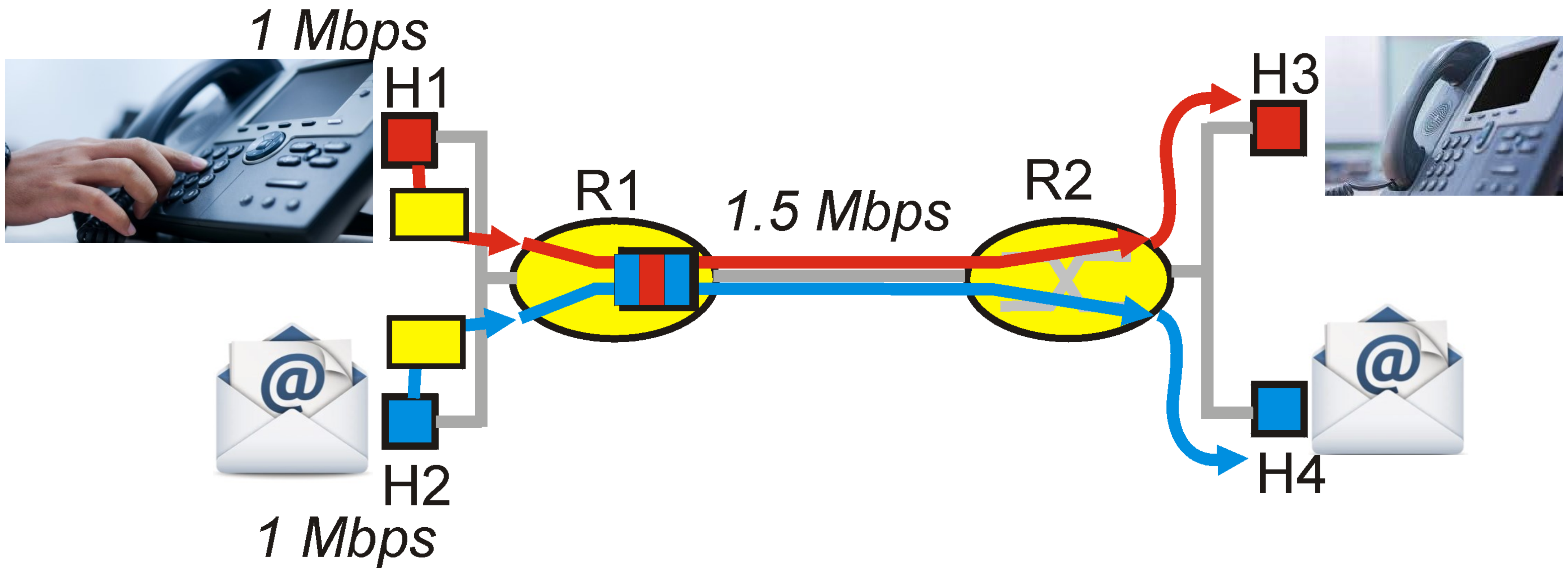
Subdividing Link Resources

- Principle 3: **Link scheduling**
  - Ensure each application gets its share
  - … while (optionally) using any extra bandwidth
  - E.g., weighted fair scheduling



Reserving Resources, and Saying No

- Traffic cannot exceed link capacity
  - Deny access, rather than degrade performance
- Principle 4: **Admission control**
  - Application declares its needs in advance
  - Application denied if insufficient resources available



Quality of Service (QoS)

- Guaranteed performance
  - Alternative to best-effort delivery model
- QoS protocols and mechanisms
  - Packet classification and marking
  - Traffic shaping
  - Link scheduling
  - Resource reservation and admission control
  - Identifying paths with sufficient resources

5-min Break!

Internet Ideal: Simple Network Model

- Globally unique identifiers
  - Each node has a unique, fixed IP address
  - … reachable from everyone and everywhere
- Simple packet forwarding
  - Network nodes simply forward packets
  - … rather than modifying or filtering them

Internet Reality

- Host mobility
  - Host changing address as it moves
- IP address depletion
  - Multiple hosts using the same address
- Security concerns
  - Detecting and blocking unwanted traffic
- Replicated services
  - Load balancing over server replicas
- Performance concerns
  - Allocating bandwidth, caching content, …
- Incremental deployment
  - New technology deployed in stages

Middleboxes

- Middleboxes are intermediaries
  - Interposed between communicating hosts
  - Often without knowledge of one or both parties
- Myriad uses
  - Address translators
  - Firewalls
  - Traffic shapers
  - Intrusion detection
  - Transparent proxies
  - Application accelerators

“An abomination!”

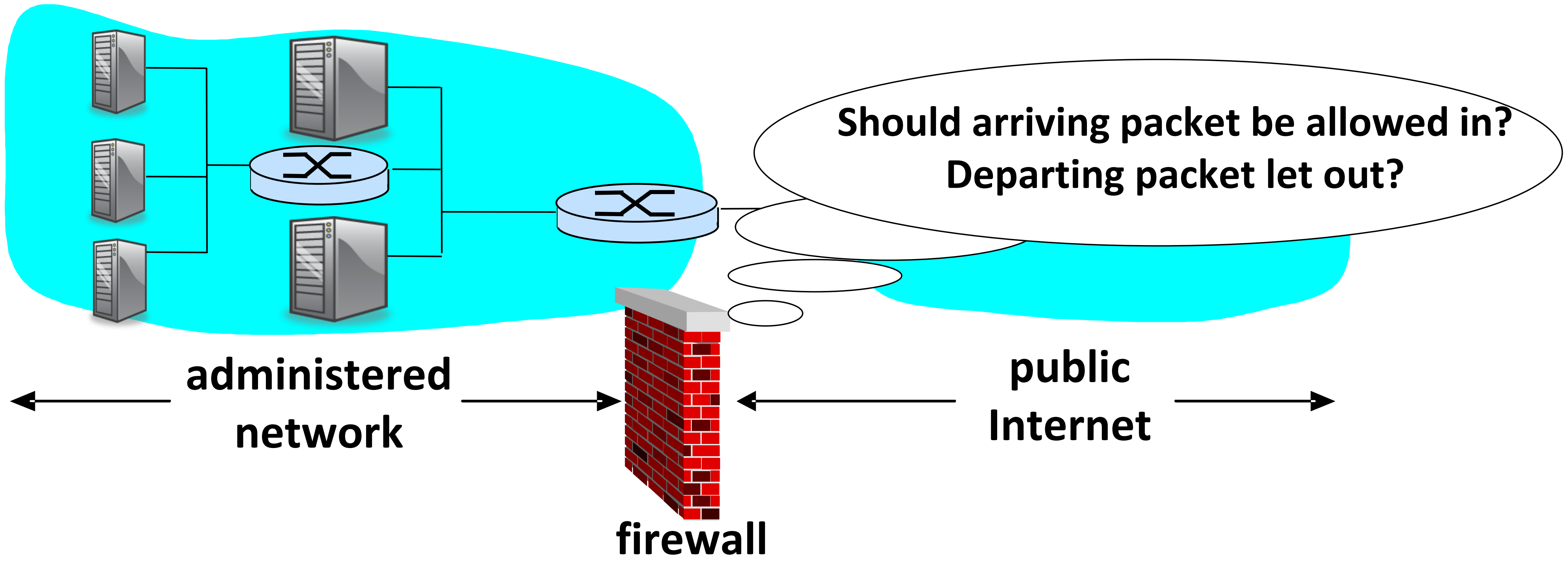
- **Violation of layering**
- **Hard to reason about**
- **Responsible for subtle bugs**

“A practical necessity!”

- **Solve real/pressing problems**
- **Needs not likely to go away**

Firewalls

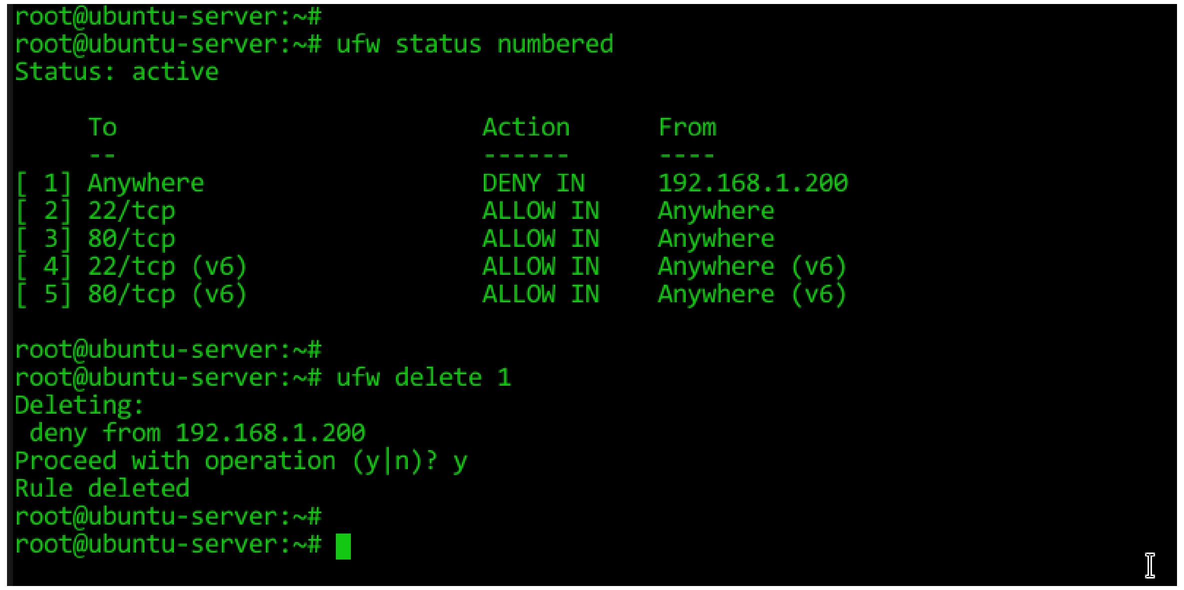
Firewalls



- Firewall filters packet-by-packet, based on:
  - Source and destination IP addresses and port numbers
  - TCP SYN and ACK bits; ICMP message type
  - Deep packet inspection on packet contents (DPI)

Firewalls

Software



Hardware

A simple Linux-based firewall

- UFW: Uncomplicated Firewall!
- For some details check this: https://ubuntu.com/server/docs/security-firewall

Packet Filtering Examples

- Block all packets with IP protocol field = 17 and with either source or dst port = 23
  - All incoming and outgoing UDP flows blocked
  - All Telnet connections are blocked
- Block all packets with TCP/UDP ports used for *Call of Duty*

- Question:
  - Prevent external clients from making TCP connections with internal clients
  - **But** allow internal clients to connect to outside
  - **How?**

Firewall Configuration

- Firewall applies a set of rules to each packet
  - To decide whether to permit or deny the packet
- Each rule is a test on the packet
  - Comparing IP and TCP/UDP header fields
  - … and deciding whether to permit or deny
- Order matters
  - Once packet matches a rule, the decision is done

Firewall Configuration Example

- Ali runs a network in 222.22.0.0/16

- Wants to let Bao’s school access certain hosts
  - Boa is on 111.11.0.0/16
  - Ali’s special hosts on 222.22.22.0/24

- Ali doesn’t trust Donald, inside Bao’s network
  - Donald is on 111.11.11.0/24

- Ali doesn’t want any other Internet traffic

Firewall Configuration Rules

#1: Allow Bao's network in to special dsts

- **ALLOW** (src=111.11.0.0/16, dst = 222.22.22.0/24)

#2: Don't let Donald's machines in

- **DENY** (src = 111.11.11.0/24, dst = 222.22.0.0/16)

#3: Block the rest of the world

- **DENY** (src = 0.0.0.0/0, dst = 0.0.0.0/0)

- **Order?**
  - **#2, #1, #3**

Stateful Firewall

- Stateless firewall:
  - Treats each packet independently
- Stateful firewall
  - Remembers connection-level information
  - E.g., client initiating connection with a server
  - … allows the server to send return traffic

A Variation: Traffic Management

- Permit vs. deny is too binary a decision
  - Classify the traffic based on rules
  - … and handle each class differently

- Traffic shaping (rate limiting)
  - Limit the amount of bandwidth for certain traffic

- Separate queues
  - Use rules to group related packets
  - And then do weighted fair scheduling across groups

Clever Users Subvert Firewalls

- Example: filtering dorm access to a server
  - Firewall rule based on IP addresses of dorms
  - … and the server IP address and port number
  - Problem: users may log in to another machine

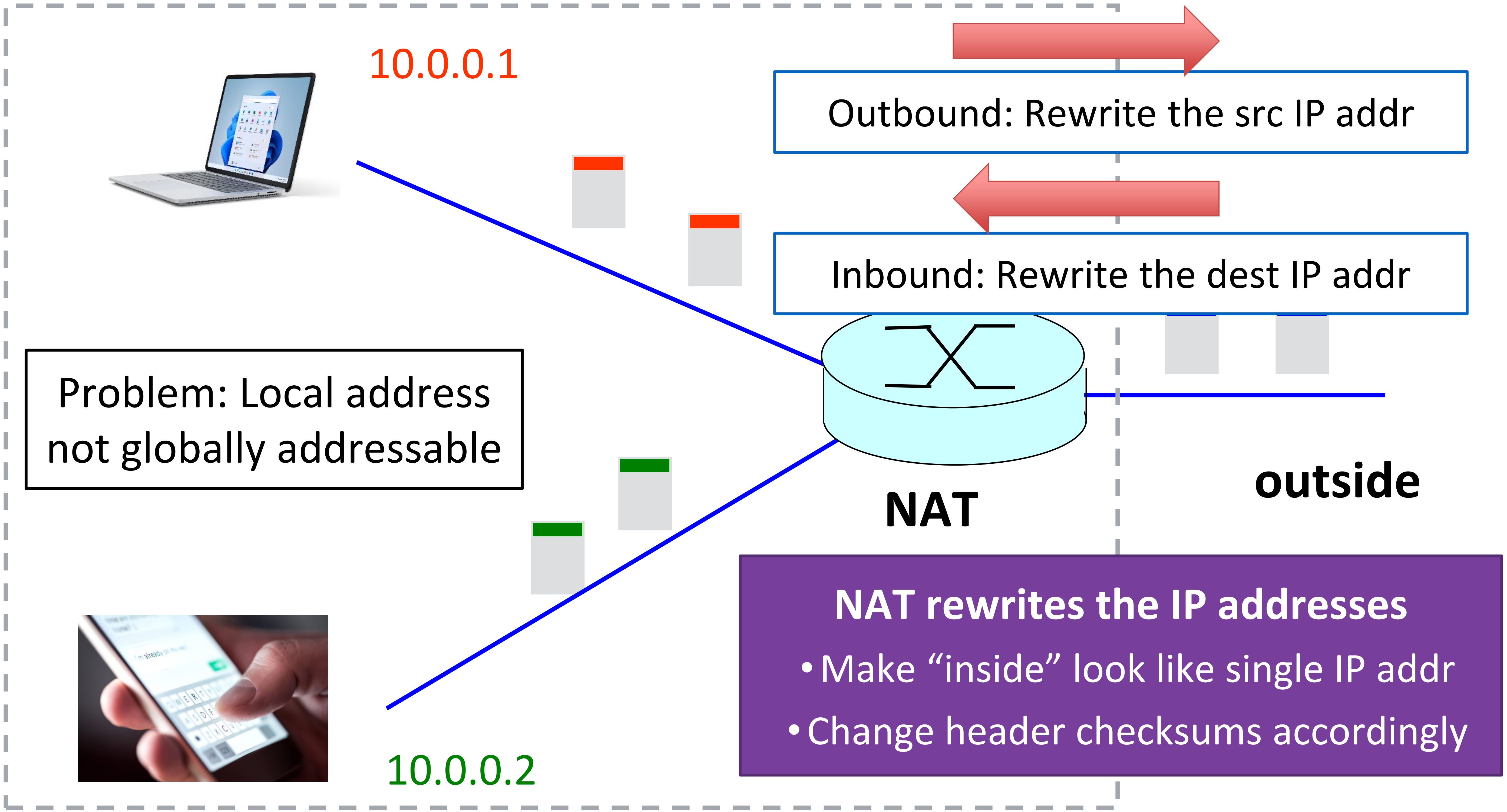
- Example: filtering P2P based on port #s
  - Firewall rule based on TCP/UDP port numbers
    - E.g., allow only port 80 (e.g., Web) traffic
  - Problem: software using non-traditional ports
    - E.g., write P2P client to use port 80 instead

Network Address Translation

History of NATs

- IP address space depletion
  - Clear in early 90s that 2^{32} addresses not enough
  - Work began on a successor to IPv4
- In the meantime…
  - Share addresses among numerous devices
  - … without requiring changes to existing hosts
- Meant as a short-term remedy
  - Now: NAT is widely deployed, much more than IPv6

Network Address Translation

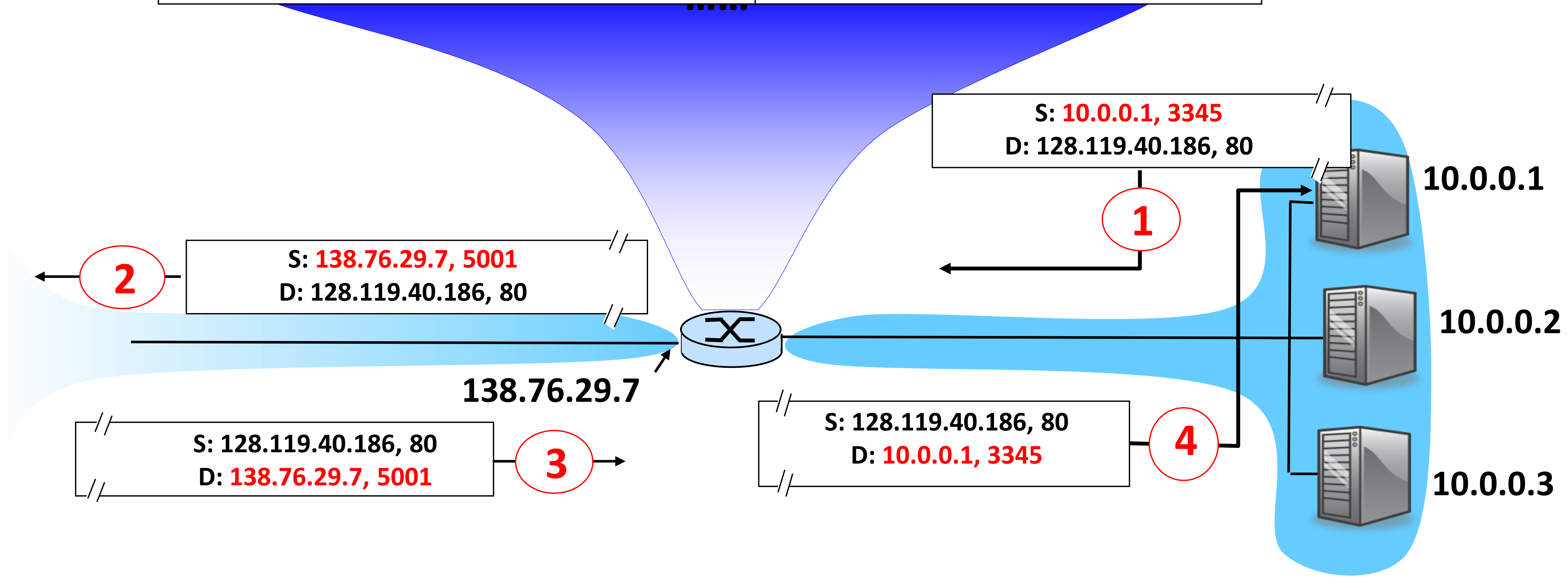


Port-Translating NAT

- Two hosts communicate with same destination
  - Destination needs to differentiate the two
- Map outgoing packets
  - Change source address and source port
- Maintain a translation table
  - Map of (src addr, port #) to (NAT addr, new port #)
- Map incoming packets
  - Map the destination address/port to the local host

Network Address Translation Example

NAT translation table	
WAN side addr	LAN side addr
138.76.29.7, 5001	10.0.0.1, 3345
……	



Maintaining the Mapping Table

- Create an entry upon seeing an outgoing packet
  - Packet with new (source addr, source port) pair
- Eventually, need to delete entries to free up #'s
  - When? If no packets arrive before a timeout
  - (At risk of disrupting a temporarily idle connection)
- Yet another example of "soft state"
  - I.e., removing state if not refreshed for a while

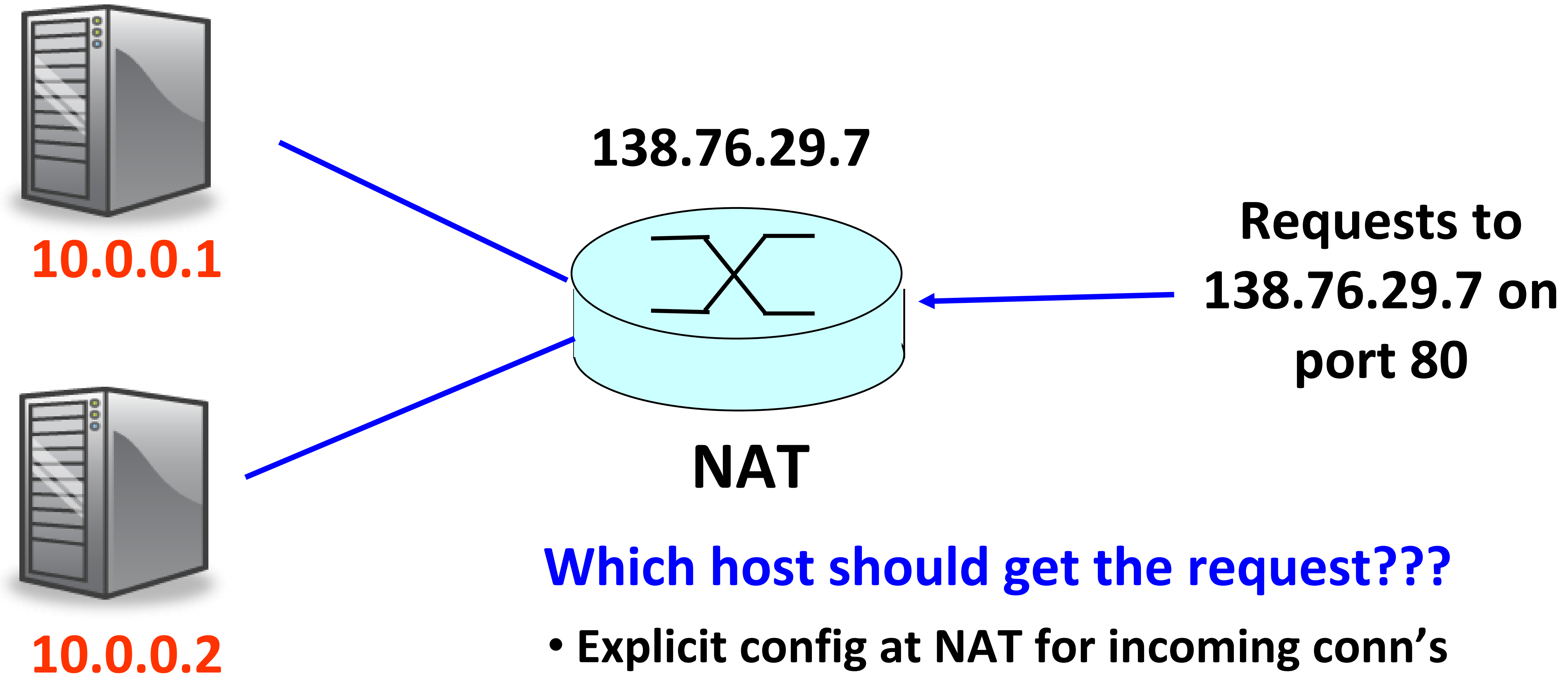
Where is NAT Implemented?

- Home router (e.g., Linksys box)
  - Integrates router, DHCP server, NAT, etc.
  - Use single IP address from the service provider
- Campus or corporate network
  - NAT at the connection to the Internet
  - Share a collection of public IP addresses
  - Avoid complexity of renumbering hosts/routers when changing ISP (w/ provider-allocated IP prefix)

Practical Objections Against NAT

Port numbers are meant to identify sockets

- Yet, NAT uses them to identify end hosts
- Makes it hard to run a server behind a NAT



Which host should get the request???

- **Explicit config at NAT for incoming conn's**

Principled Objections Against NAT

- Routers are not supposed to look at port #s
  - Network layer should care only about *IP* header
  - … and not be looking at the *port numbers* at all

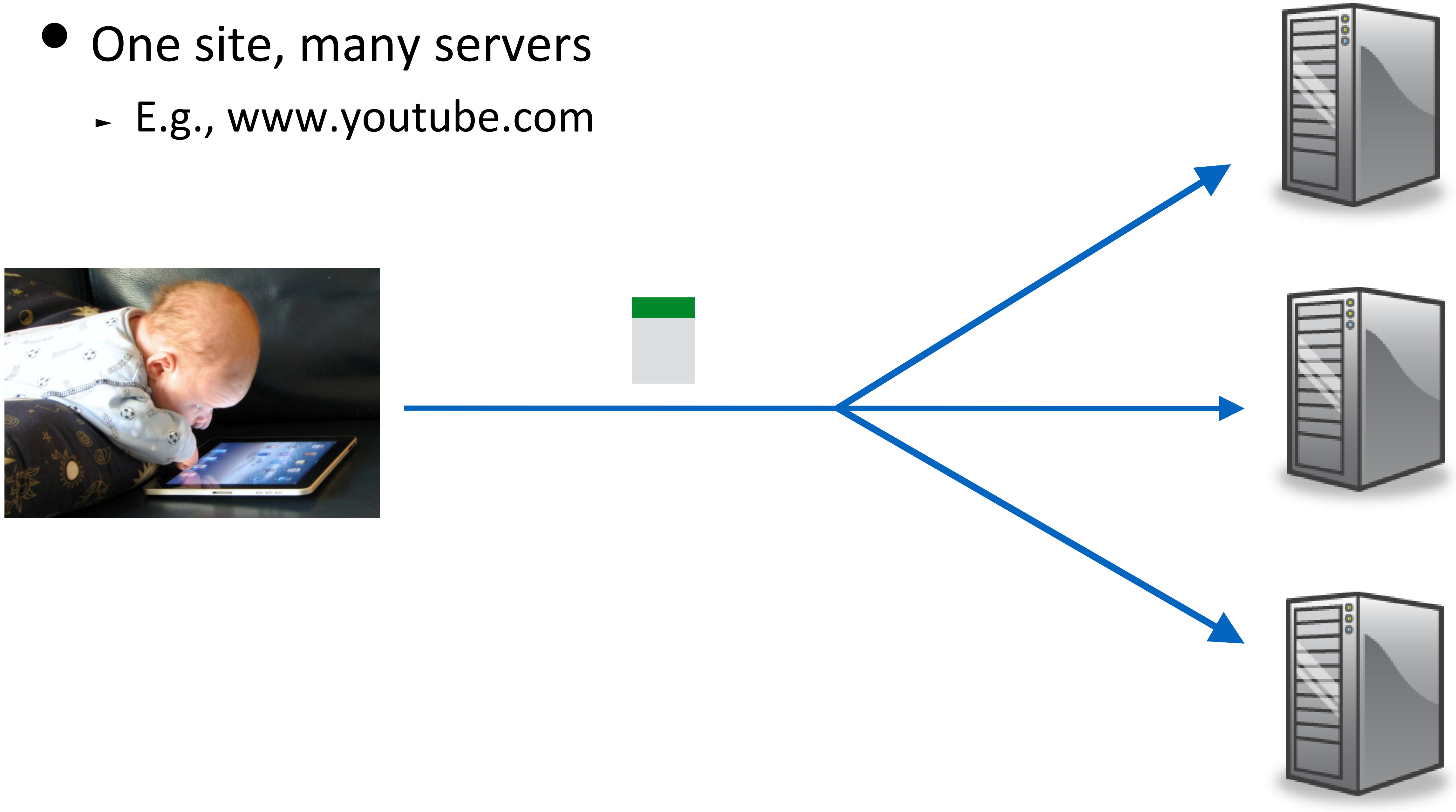
- NAT violates the end-to-end argument
  - Network nodes should not modify the packets

- IPv6 is a cleaner solution
  - Better to migrate than to limp along with a hack

Load Balancers

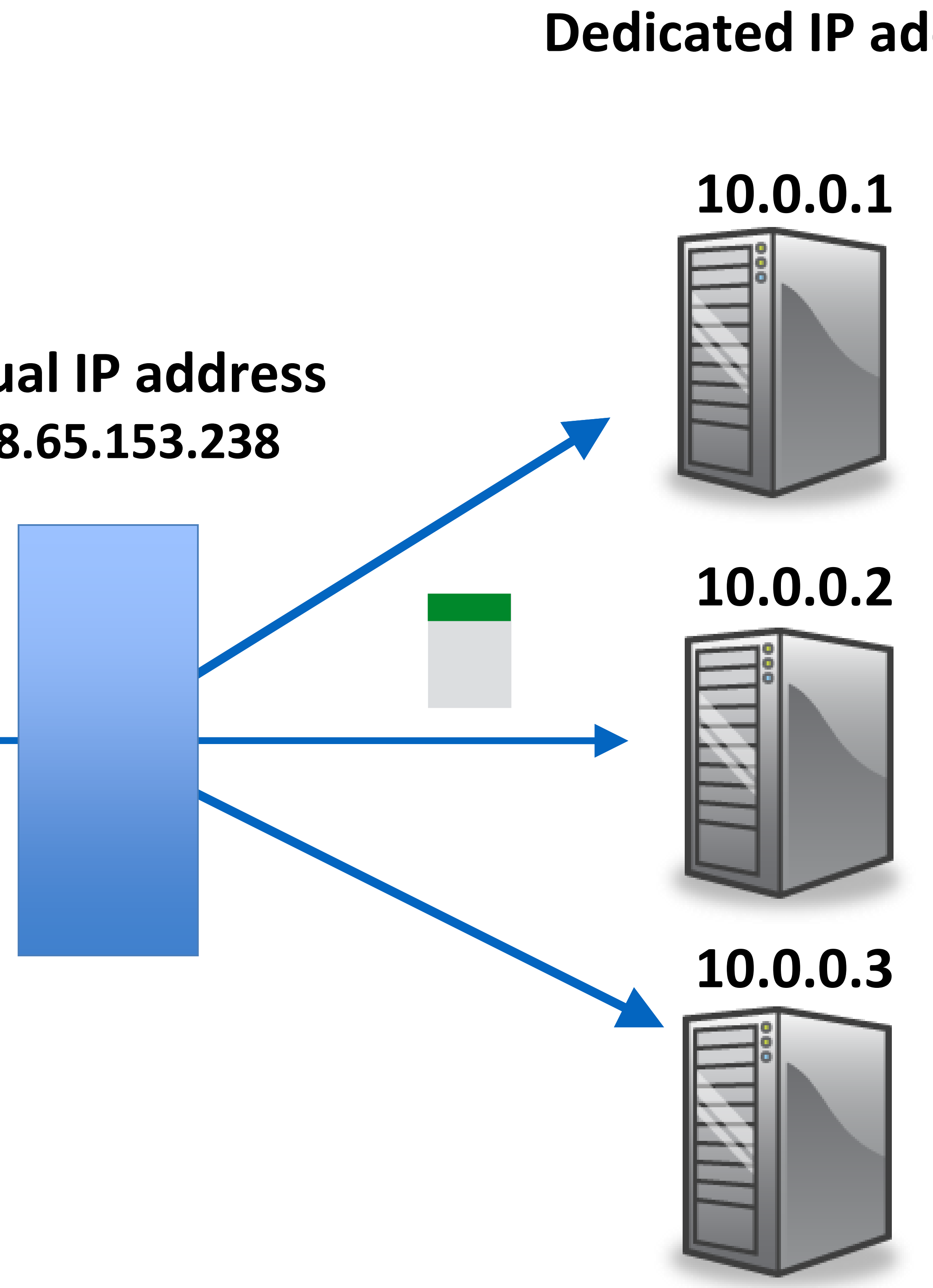
Replicated Servers

- One site, many servers
  - E.g., www.youtube.com



Load Balancer

- Splits load over server replicas
  - At the connection level

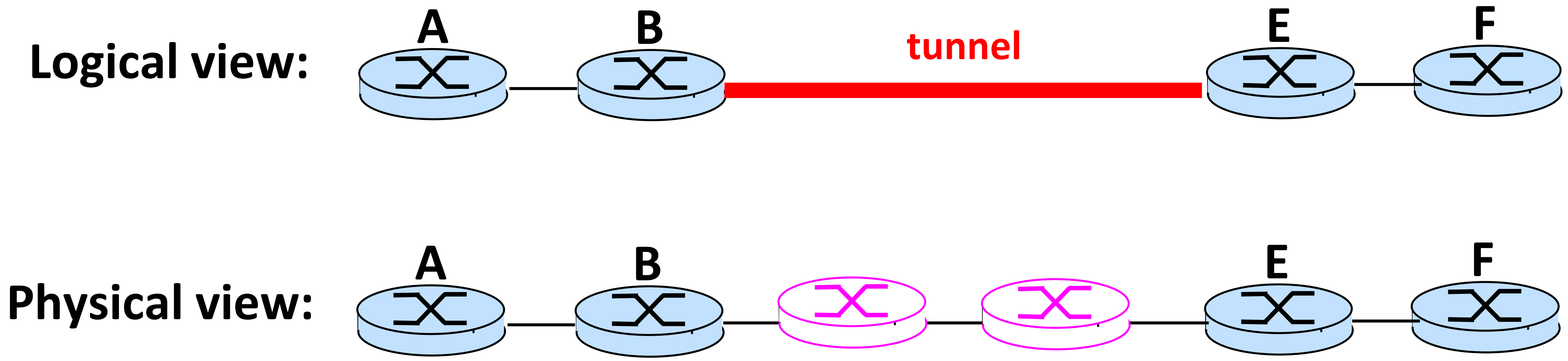


- Apply load balancing policies

Tunneling

IP Tunneling

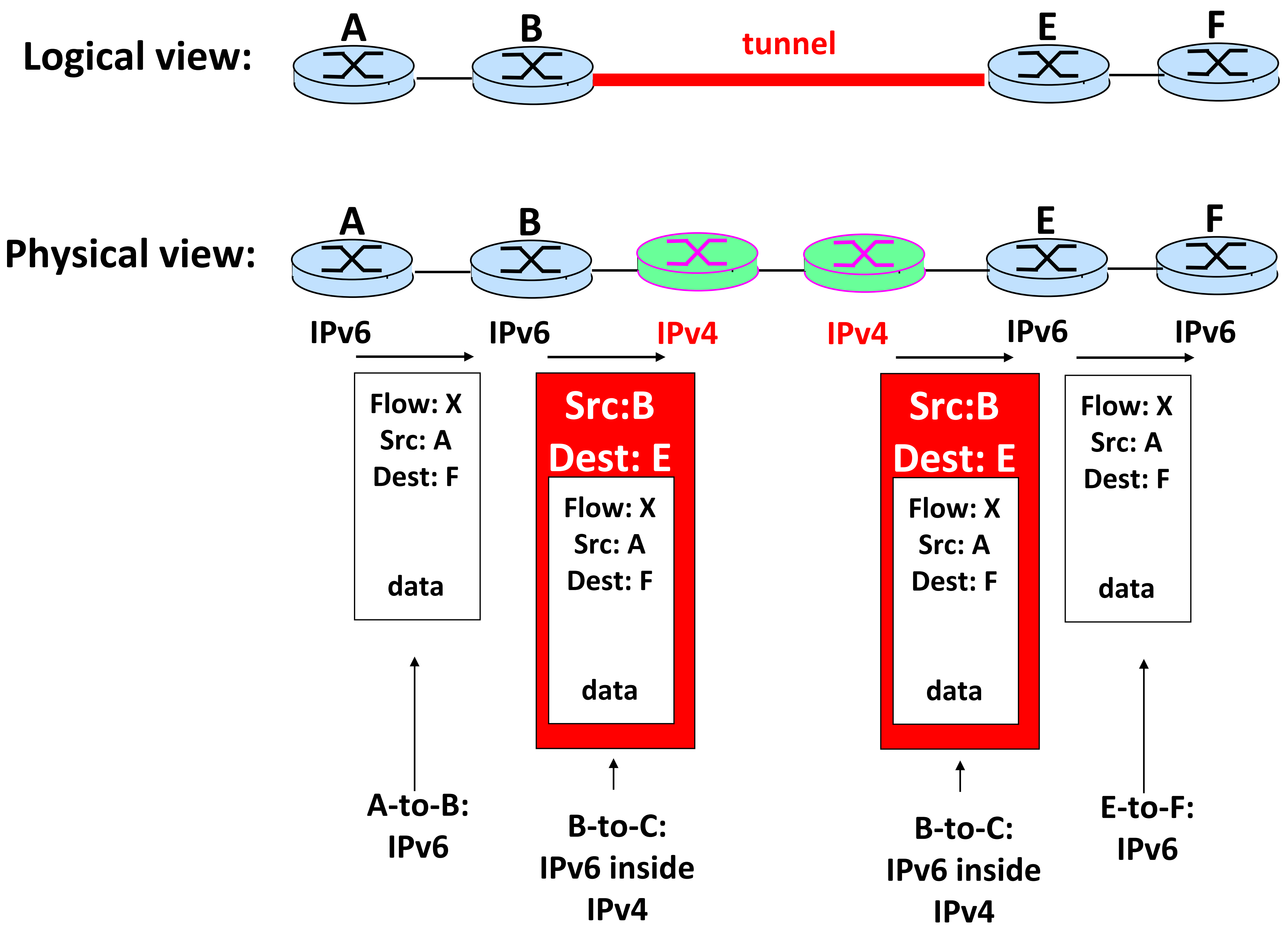
- IP tunnel is a virtual point-to-point link
  - Illusion of a direct link between two nodes



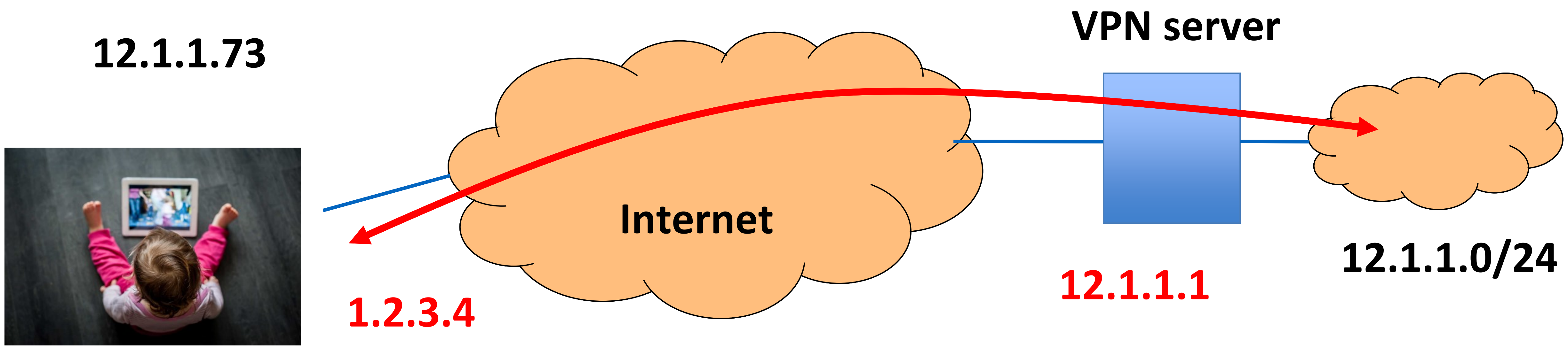
- Encapsulation of the packet inside IP datagram
  - Node B sends a packet to node E
  - … containing another packet as the payload

6Bone: Deploying IPv6 over IP4

A testbed for IPv6 (1996-2006)

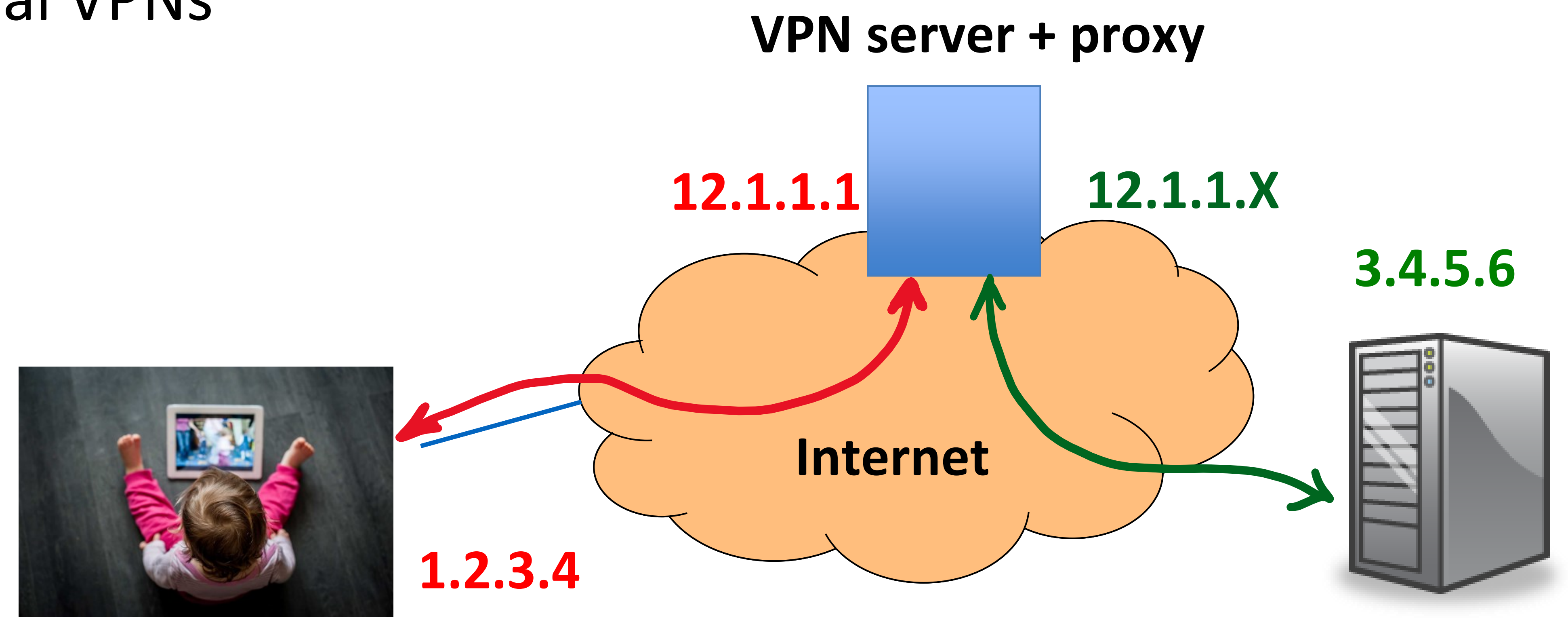


Remote Access Virtual Private Network



- Tunnel from user machine to VPN server
  - A "link" across the Internet to the local network
- Encapsulates packets to/from the user
  - Packet from 12.1.1.73 to 12.1.1.100
  - Inside a packet from 1.2.3.4 to 12.1.1.1

Commercial VPNs



- Tunnel from user machine to VPN server
- VPN server NATs or TCP proxies traffic to origin sites
  - Traffic between client and VPN encrypted
  - VPN “anonymizes” the IP of client to rest of Internet, and can circumvent censorship on client-side
  - Client **must** fully trust VPN provider!
    - Why?!

Wrap up

- Middleboxes address important problems
  - Getting by with fewer IP addresses
  - Blocking unwanted traffic
  - Making fair use of network resources
  - Improving end-to-end performance

- Middleboxes cause problems of their own
  - No longer globally unique IP addresses
  - **Cannot assume network simply delivers packets!**