CSC 458/2209 – Computer Networking Systems

Handout # 23: Queueing Mechanisms; Middleboxes

Professor Yashar Ganjali
Department of Computer Science
University of Toronto

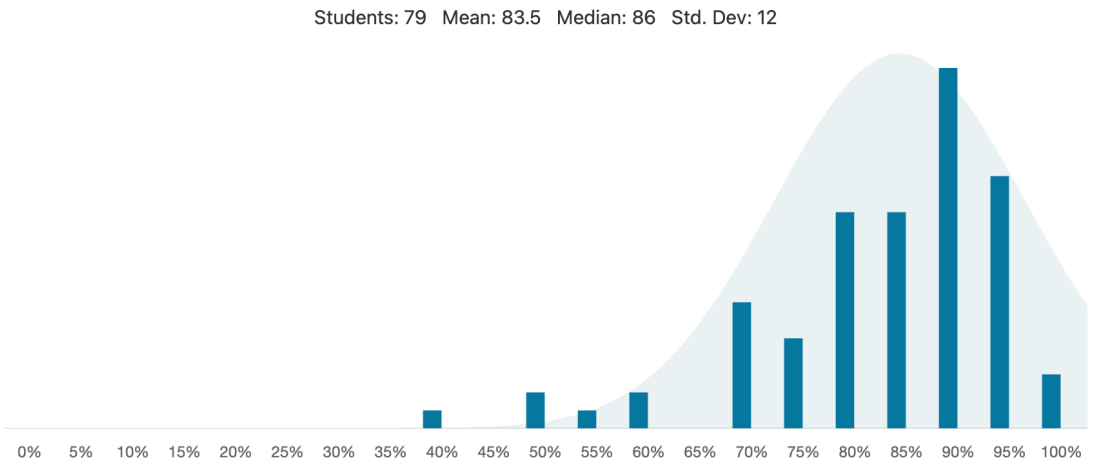
ganjali7@cs.toronto.edu
http://www.cs.toronto.edu/~yganjali

Announcements

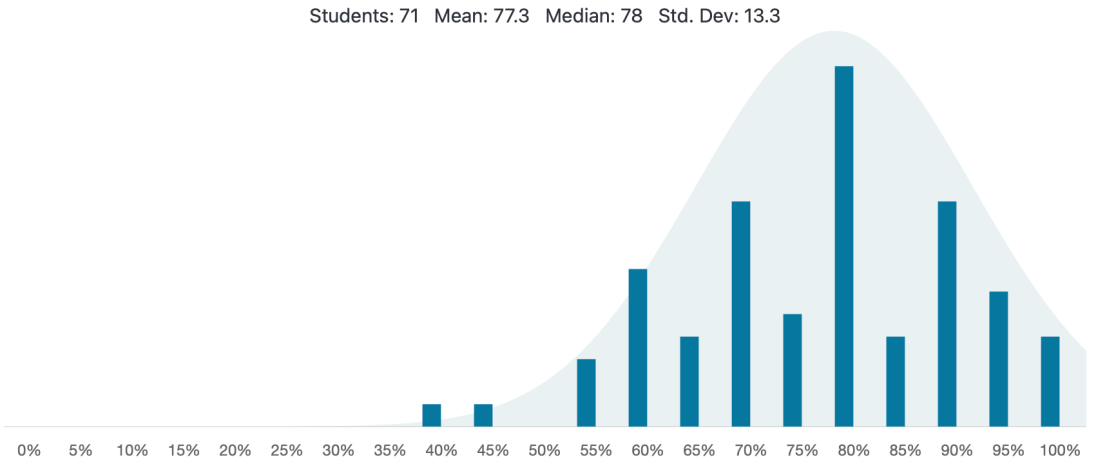
- Problem Set 2
  - Will be posted on class website tonight.
  - Submit electronically as ps2.pdf.
  - Due: Tuesday, November 11, at 5pm.
- Programming Assignment 2: Buffer Sizing
  - Due: Friday, November 28 at 5pm.
- This week's tutorial:
  - Programming Assignment 2 Overview

Midterm Results

- SEC0101
  - Mean: 83.5
  - Median: 86



- SEC5101
  - Mean: 77.3
  - Median: 78



Very strong results. Great job!

The Story ...

- Network layers
  - Link layer
    - Framing, switches, hubs, bridges, error detection, correction, ...
  - Network layer
    - Addressing, routing, autonomous systems, BGP, ...
  - Transport layer
    - Flow control, congestion control, ...

- Ready or applications
  - ... almost!

Today

- Queuing Mechanisms
  - Random Early Detection (RED)
  - Explicit Congestion Notification (ECN)
- Middleboxes
  - Network Address Translation
  - Firewalls
  - Web Proxies

End-to-end Principle

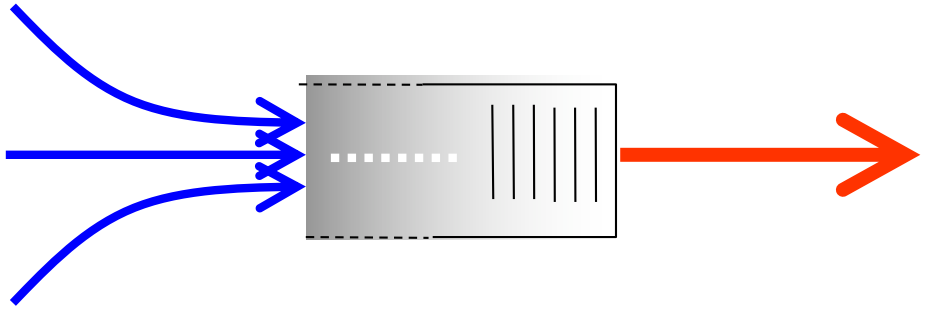
- Design principle for the Internet that says you should keep functionalities at the end-hosts
  - Application specific functions

- **Example**: congestion control in the Internet

- Power at the end-hosts
  - **Pros**: flexible, easy to change and innovate, ...
  - **Cons**: trust at the hands of least trusted component, not necessarily optimal, high overhead, ...

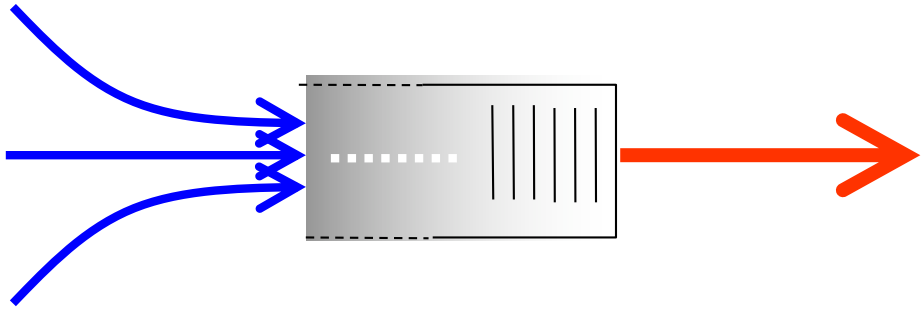
Bursty Loss From Drop-Tail Queuing

- TCP depends on packet loss
  - Packet loss is the indication of congestion
  - In fact, TCP drives the network into packet loss
  - … by continuing to increase the sending rate
- Drop-tail queuing leads to bursty loss
  - When a link becomes congested…
  - … many arriving packets encounter a full queue
  - And, as a result, many flows divide sending rate in half
  - … and, many individual flows lose multiple packets



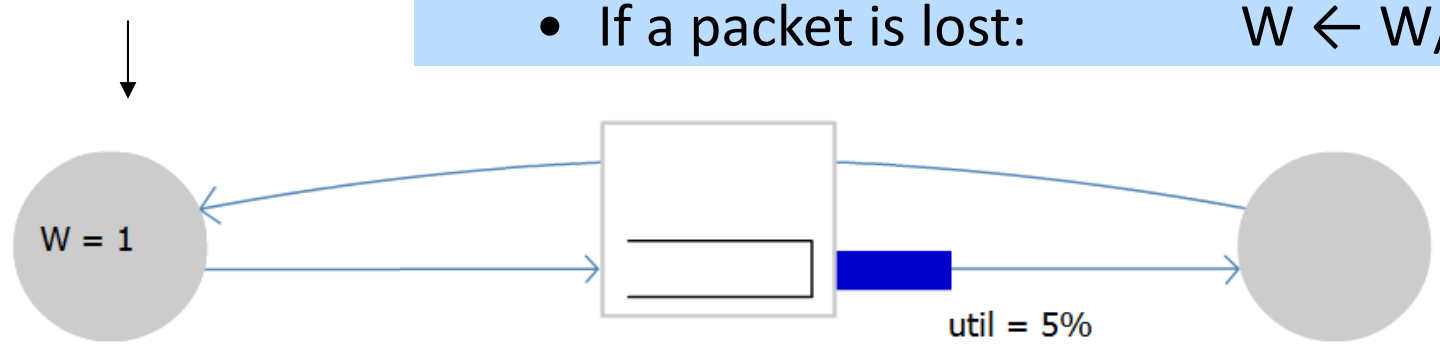
Slow Feedback from Drop Tail

- Feedback comes when buffer is completely full
  - … even though the buffer has been filling for a while
- Plus, the filling buffer is increasing RTT
  - … and the variance in the RTT
- Might be better to give early feedback
  - Get one or two flows to slow down, not all of them
  - Get these flows to slow down before it is too late



Congestion Window Evolution

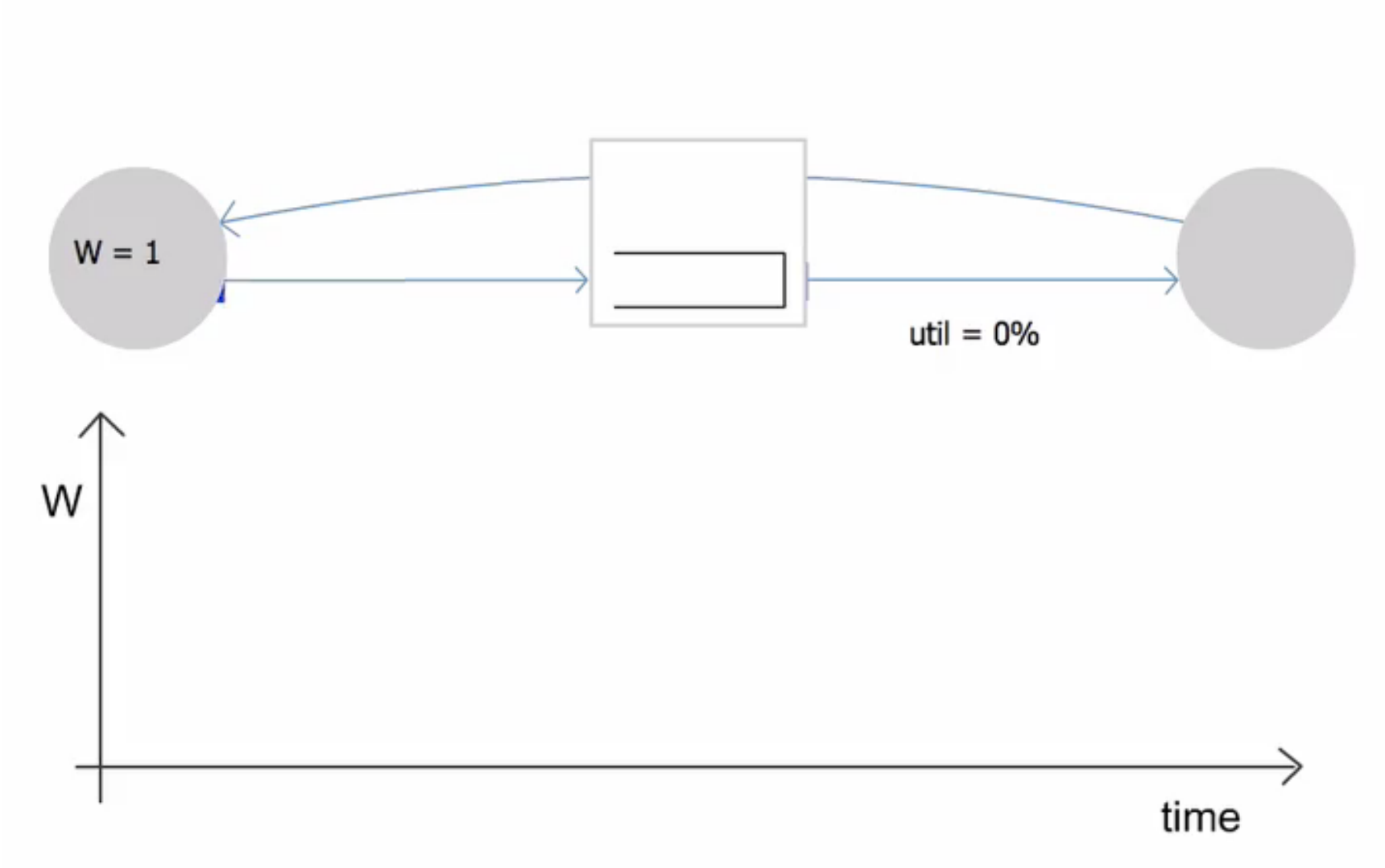
Only W packets
may be outstanding



Rule for adjusting W

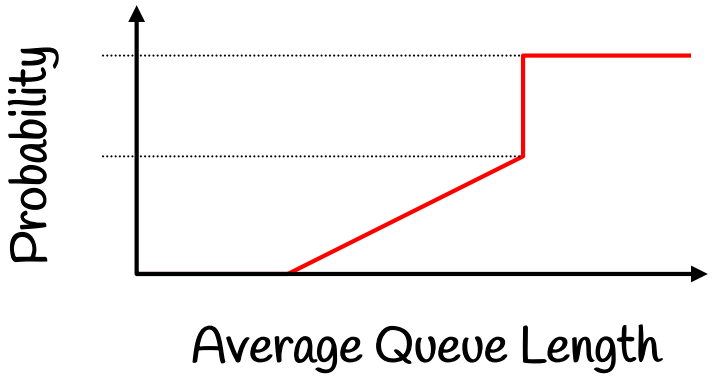
- If an ACK is received: W ← W+1/W
- If a packet is lost: W ← W/2

Congestion Window Evolution

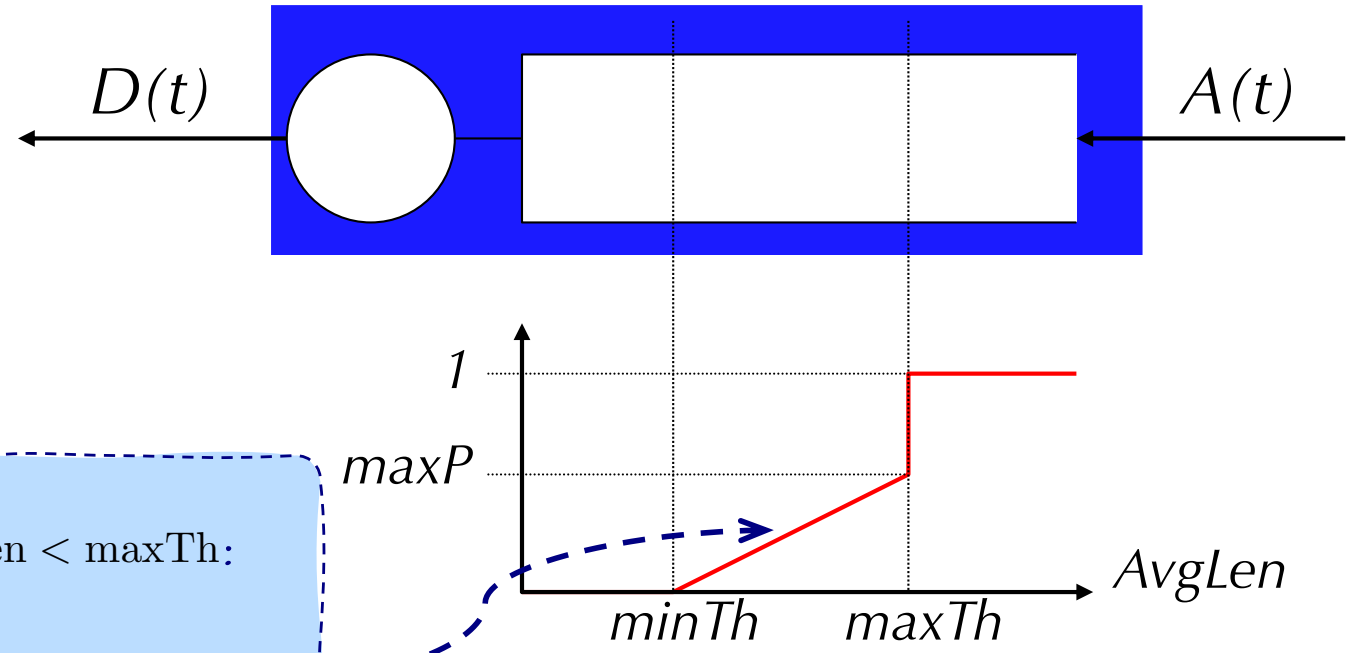


Random Early Detection (RED)

- Basic idea of RED
  - Router notices that the queue is getting backlogged
  - … and randomly drops packets to signal congestion
- Packet drop probability
  - Drop probability increases as queue length increases
  - If buffer is below some level, don't drop anything
  - … otherwise, set drop probability as function of queue



RED Drop Probabilities



If $\text{minTh} < \text{AvgLen} < \text{maxTh}$:

$$\hat{p}_{\text{AvgLen}} = \text{maxP} \times \frac{\text{AvgLen} - \text{minTh}}{\text{maxTh} - \text{minTh}}$$

$$Pr(\text{Packet Drop}) = \frac{\hat{p}_{\text{AvgLen}}}{1 - \text{count} \times \hat{p}_{\text{AvgLen}}}$$

Need to cap by 1

The variable "count" counts how long we've been in

$$\text{minTh} < \text{AvgLen} < \text{maxTh}$$

since we last dropped a packet.
Drops are spaced out in time, reducing likelihood of re-entering slow-start.

RED Average Queue Length

- Drop probability is increased as the average queue length increases.
- (Geometric) moving average of the queue length is used so as to detect long term congestion, yet allow short term bursts to arrive.

$$AvgLen_{n+1} = (1-\alpha) \times AvgLen_n + \alpha \times Length_n$$

i.e. $$AvgLen_{n+1} = \sum_{i=1}^{n} Length_i (\alpha)(1-\alpha)^{n-i}$$

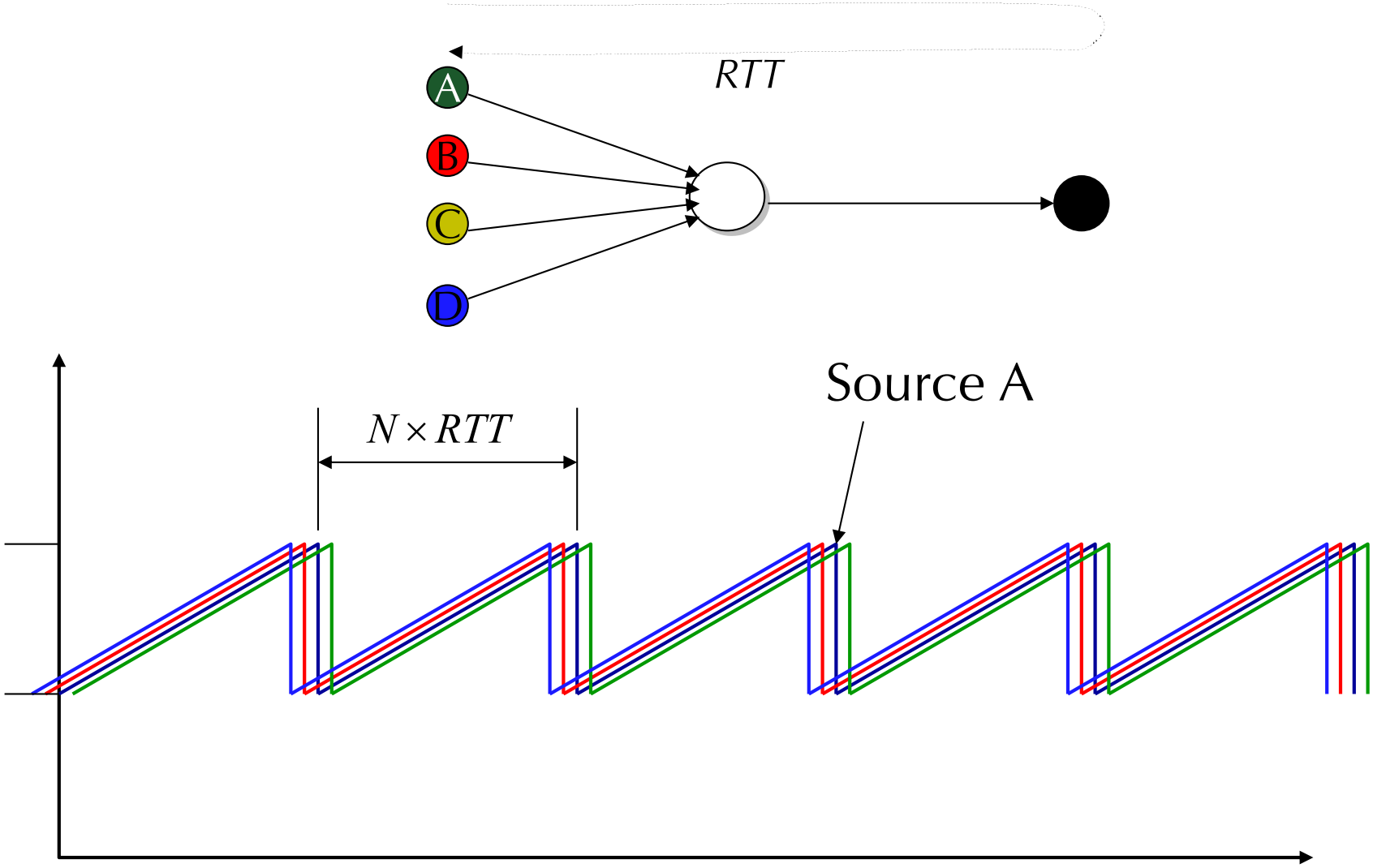
Properties of RED

- 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 have more packets
  - … and, hence, a higher chance of being selected
- Drops are spaced out in time
  - Which should help desynchronize the TCP senders
- Tolerant of burstiness in the traffic
  - By basing the decisions on average queue length

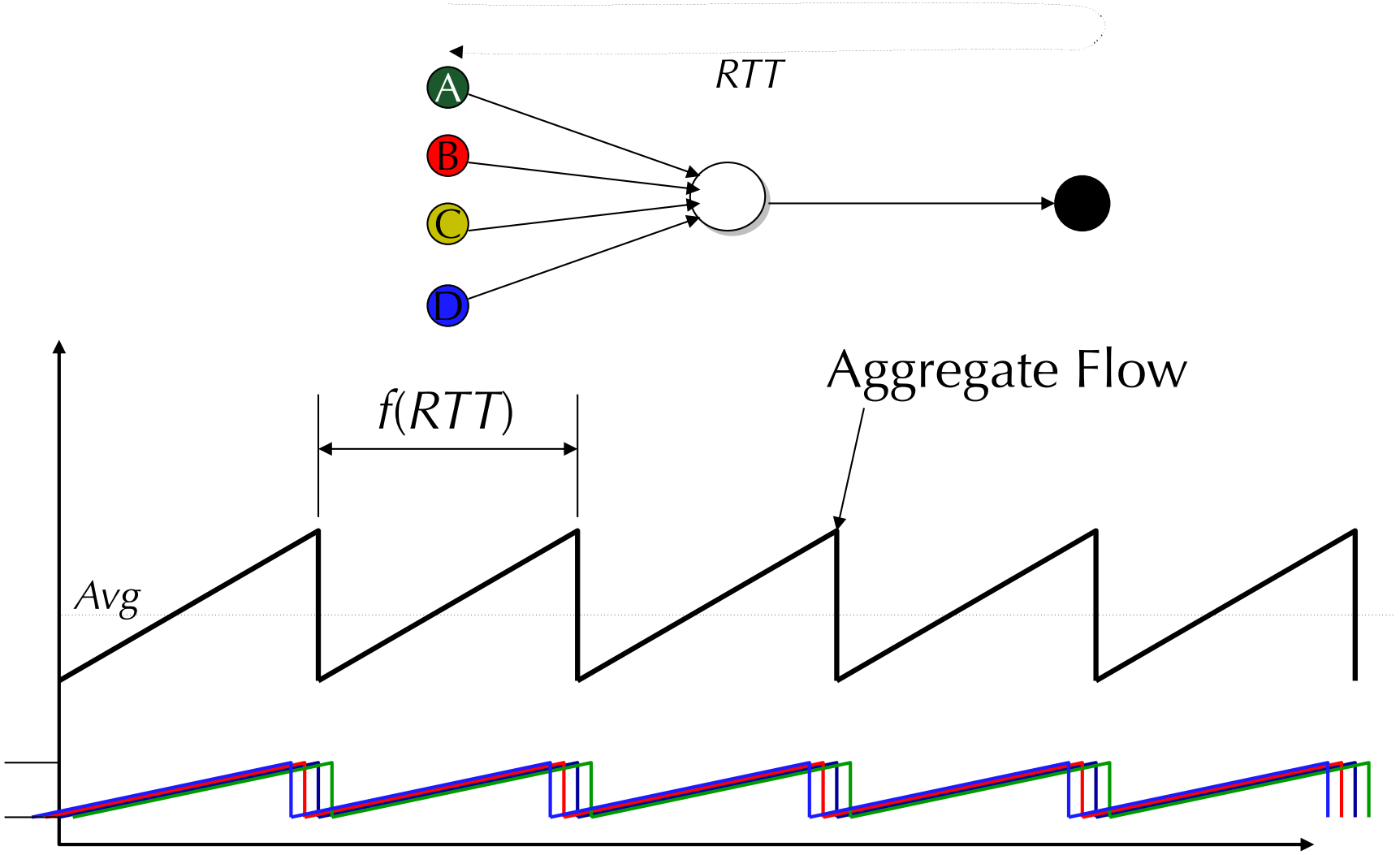
Problems With RED

- Hard to get the tunable parameters just right
  - How early to start dropping packets?
  - What slope for the increase in drop probability?
  - What time scale for averaging the queue length?
- Sometimes RED helps but sometimes not
  - If the parameters aren't set right, RED doesn't help
  - And it is hard to know how to set the parameters
- RED is implemented in practice
  - But, often not used due to the challenges of tuning right
- Many variations
  - With cute names like "Blue" and "FRED"… ☺

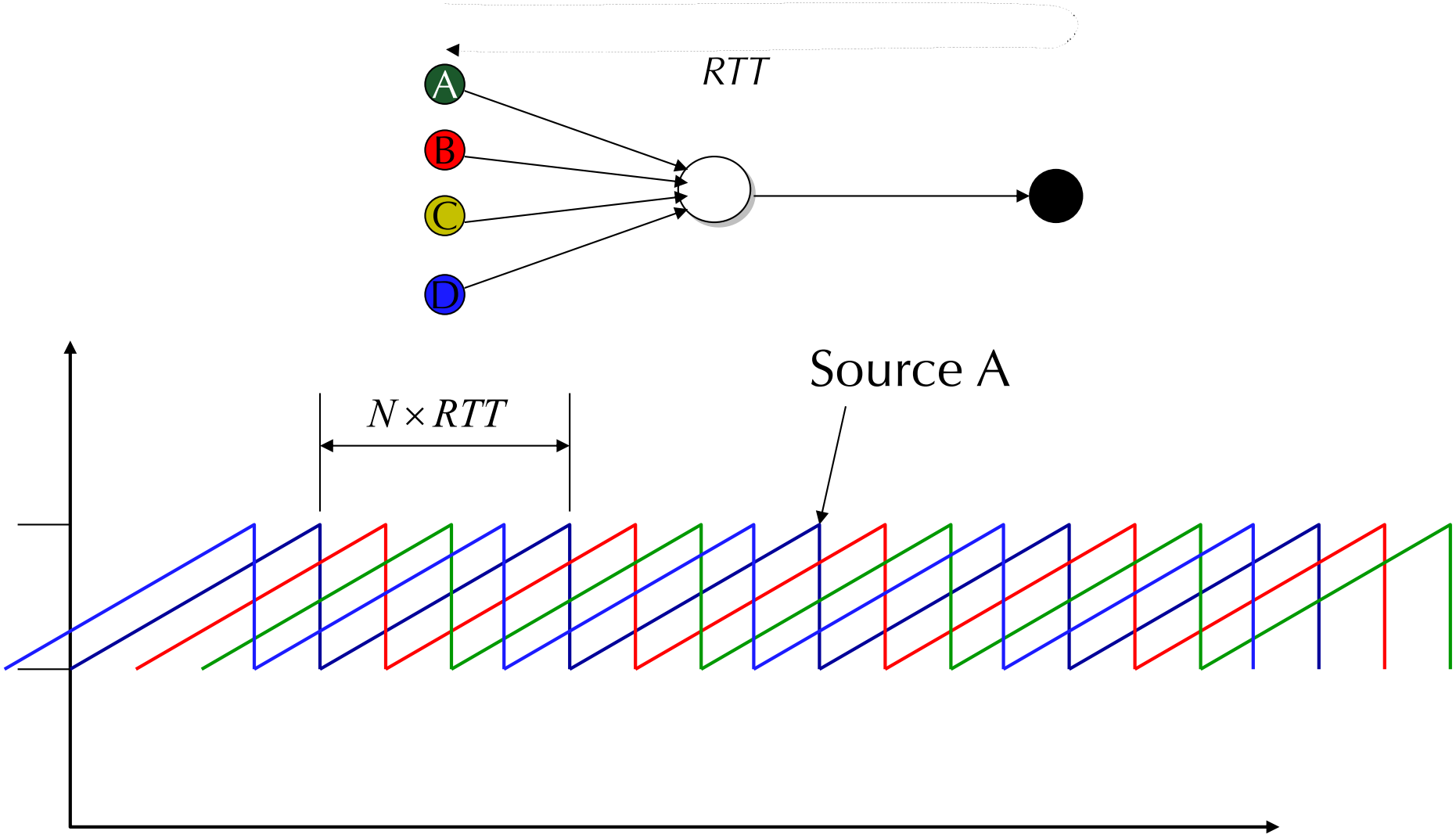
Synchronization of Sources



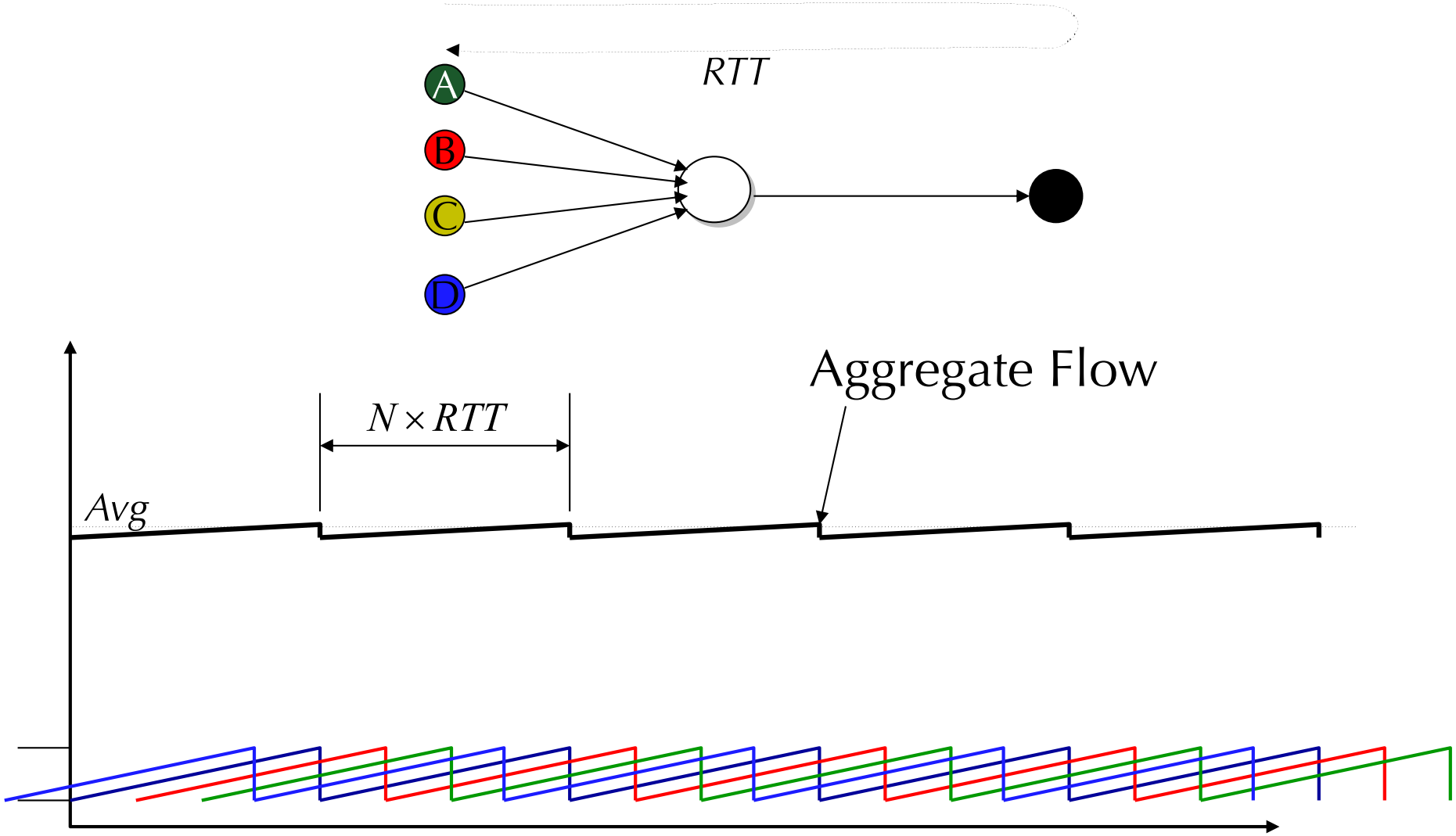
Synchronization of Sources



Desynchronized Sources



Desynchronized Sources



Explicit Congestion Notification

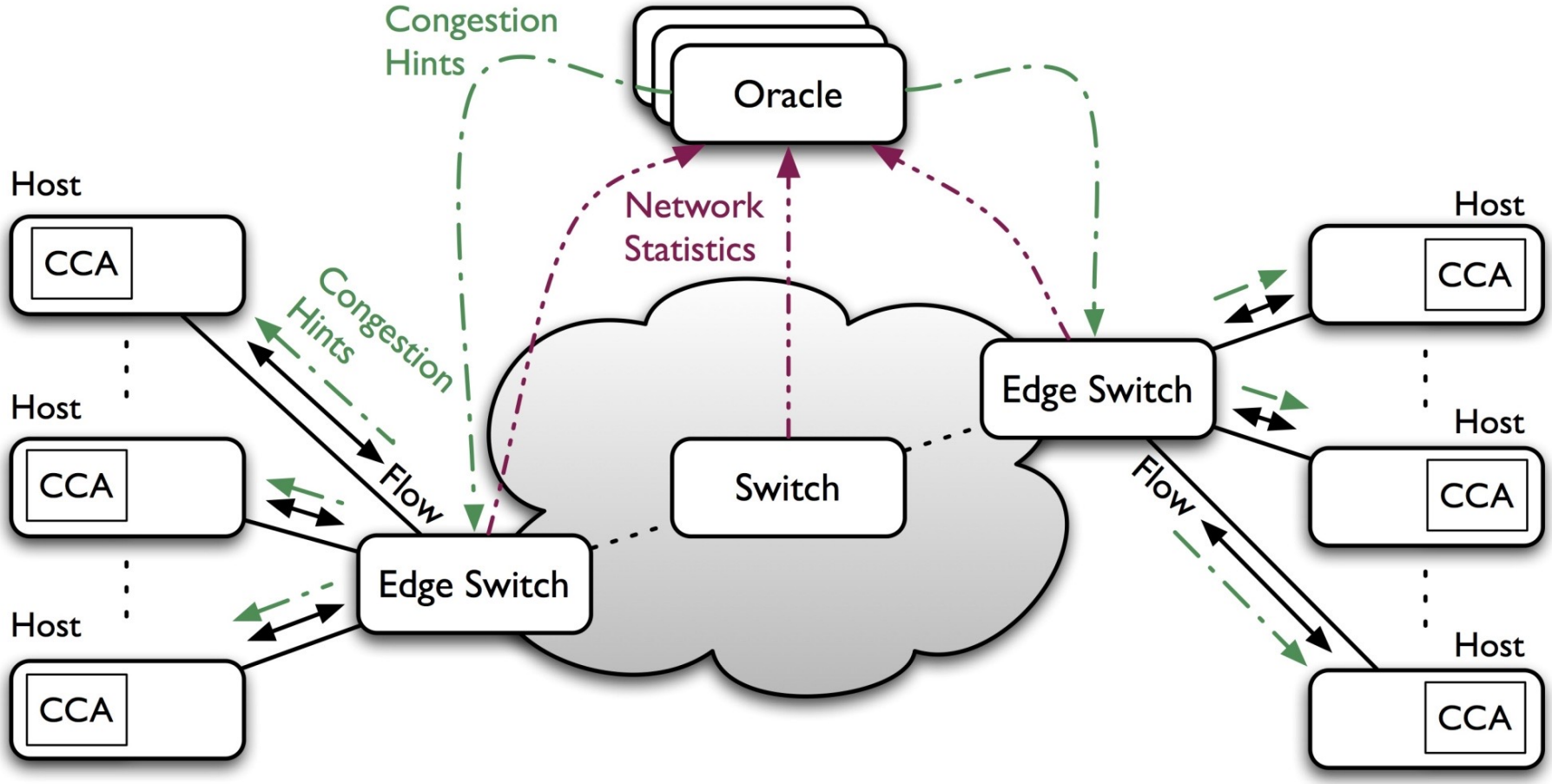
- Early dropping of packets
  - Good: gives early feedback
  - Bad: must drop the packet to give the feedback
- Explicit Congestion Notification
  - Router marks the packet with an ECN bit
  - … and sending host interprets as a sign of congestion
- Surmounting the challenges
  - Must be supported by the end hosts and the routers
  - Requires two bits in the IP header (one for the ECN mark, and one to indicate the ECN capability)
  - Solution: borrow two of the Type-Of-Service bits in the IPv4 packet header

Detour: OpenTCP

- Network can impact congestion control by using AQM schemes.
- Finding the optimal value by probing
  - Costly, and not very efficient
- What if the network could help?
  - Two extremes: end-to-end vs. centralized
- How about a solution in the middle?
  - Network guides the flows without creating dependency.

Detour:

OpenTCP in Software-Defined Networks



Conclusions

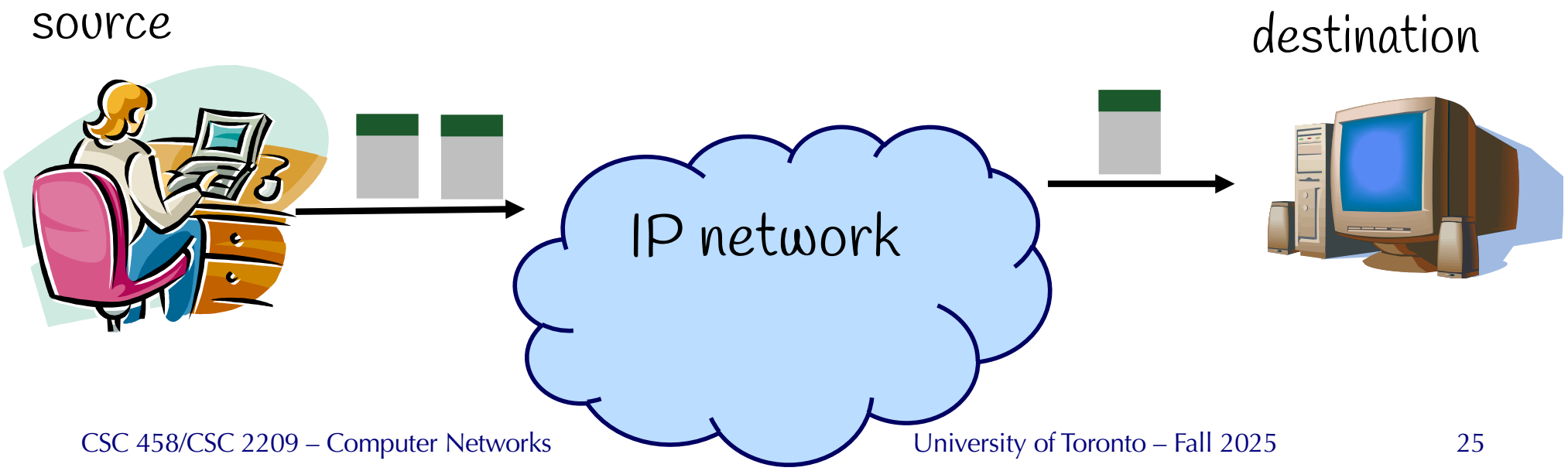
- Congestion is inevitable
  - Internet does not reserve resources in advance
  - TCP actively tries to push the envelope
- Congestion can be handled
  - Additive increase, multiplicative decrease
  - Slow start, and slow-start restart
- Active Queue Management can help
  - Random Early Detection (RED)
  - Explicit Congestion Notification (ECN)

Today

- Queuing Mechanisms
  - Random Early Detection (RED)
  - Explicit Congestion Notification (ECN)
- Middleboxes
  - Network Address Translation
  - Firewalls
  - Web Proxies

Network-Layer Principles

- 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
  - Changes in IP addresses as hosts move
- IP address depletion
  - Dynamic assignment of IP addresses
  - Use of private addresses
- Security concerns
  - Discarding suspicious or unwanted packets
  - Detecting suspicious traffic
- Performance concerns
  - Controlling how link bandwidth is allocated
  - Storing popular Web content near the clients

Middleboxes

- Middleboxes are intermediaries
  - Interposed in-between the communicating hosts
  - Often without knowledge of one or both parties
- Examples
  - Network address translators
  - Firewalls
  - Traffic shapers
  - Intrusion detection systems
  - Transparent Web proxy caches

Two Views of Middleboxes

- An abomination
  - Violation of layering
  - Cause confusion in reasoning about the network
  - Responsible for many subtle bugs
- A necessity
  - Solving real and pressing problems
  - Needs that are not likely to go away

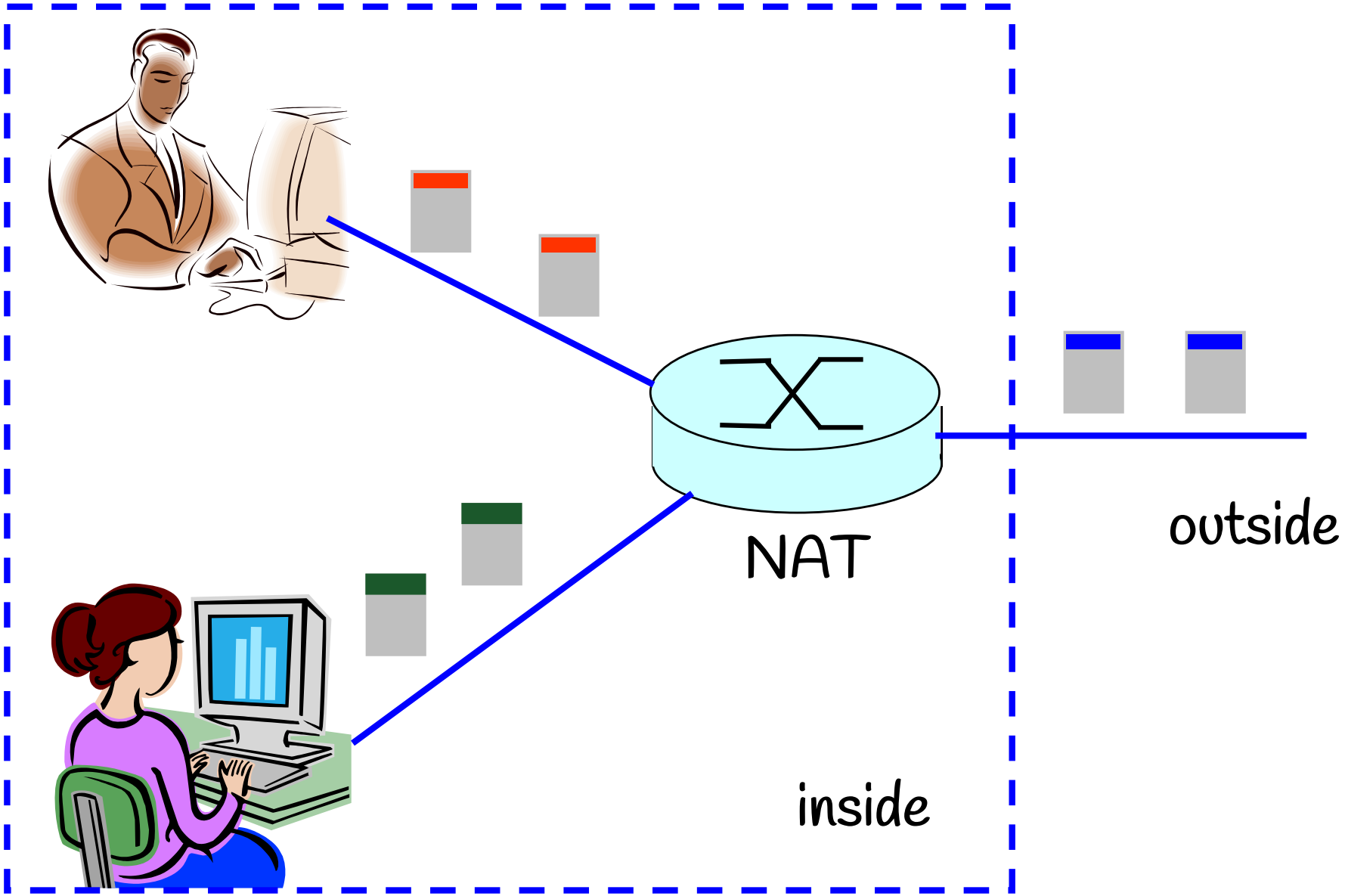
Today

- Queuing Mechanisms
  - Random Early Detection (RED)
  - Explicit Congestion Notification (ECN)
- Middleboxes
  - Network Address Translation
  - Firewalls
  - Web Proxies

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 to provide temporary relief
  - Intended as a short-term remedy
  - Now, NAT are very widely deployed
  - … much more so than IPv6

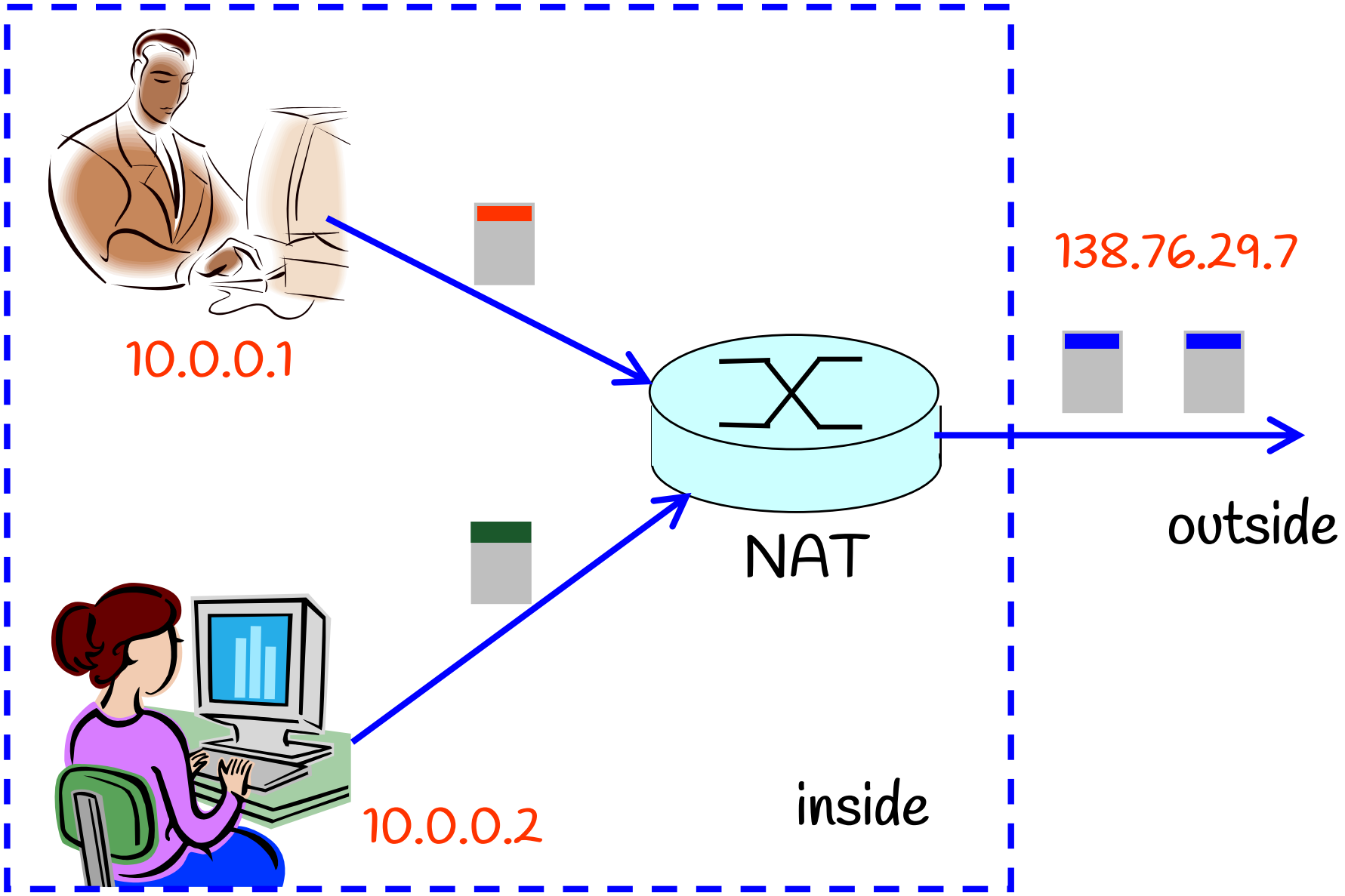
Active Component in the Data Path



IP Header Translators

- Local network addresses not globally unique
  - E.g., private IP addresses (in 10.0.0.0/8)
- NAT box rewrites the IP addresses
  - Make the "inside" look like a single IP address
  - … and change header checksums accordingly
- Outbound traffic: from inside to outside
  - Rewrite the source IP address
- Inbound traffic: from outside to inside
  - Rewrite the destination IP address

Using a Single Source Address



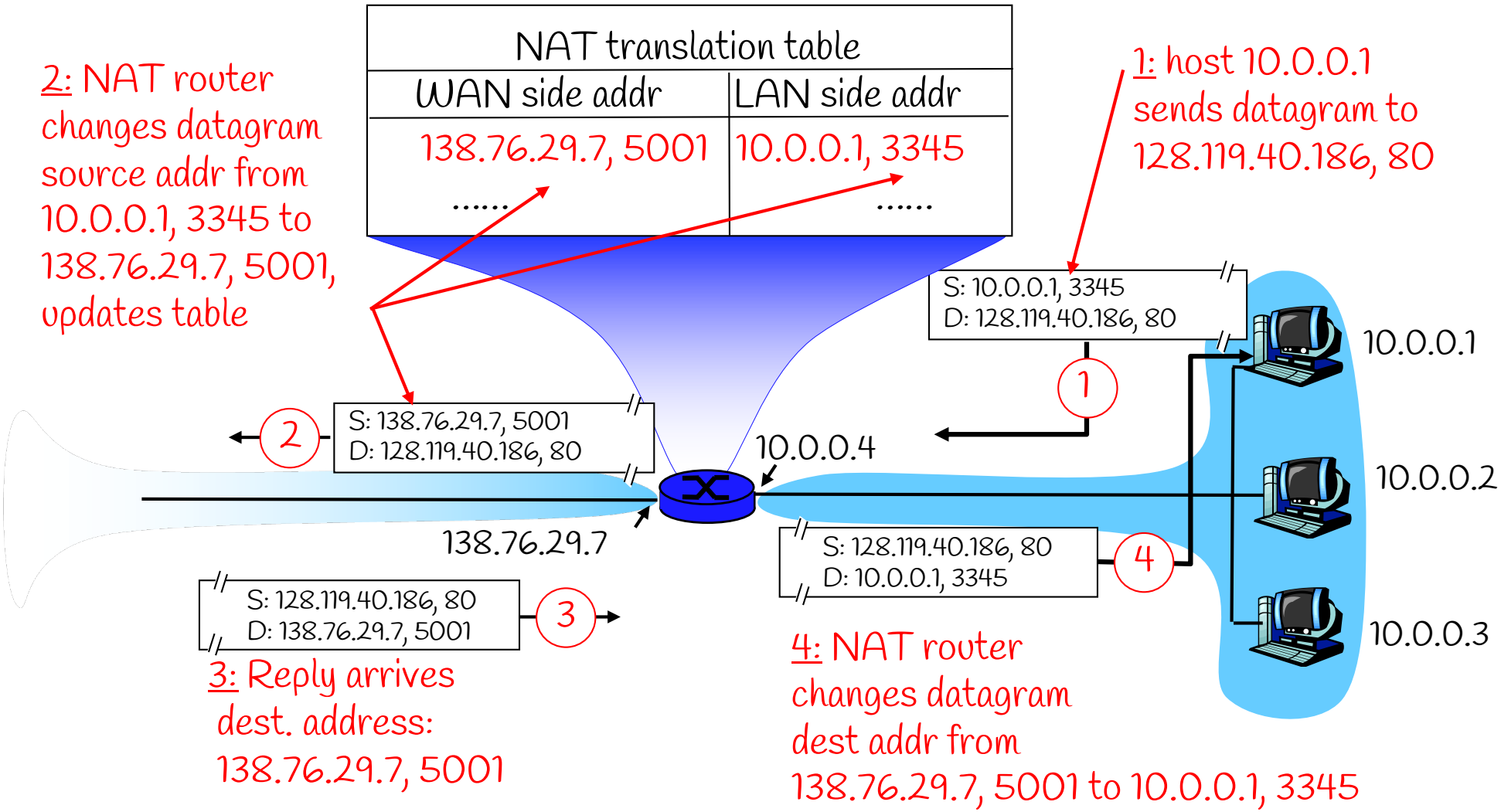
What if Both Hosts Contact Same Site?

- Suppose hosts contact the same destination
  - E.g., both hosts open a socket with local port 3345 to destination 128.119.40.186 on port 80
- NAT gives packets same source address
  - All packets have source address 138.76.29.7
- Problems
  - Can destination differentiate between senders?
  - Can return traffic get back to the correct hosts?

Port-Translating NAT

- Map outgoing packets
  - Replace source address with NAT address
  - Replace source port number with a new port number
  - Remote hosts respond using (NAT address, new port #)
- Maintain a translation table
  - Store map of (source address, port #) to (NAT address, new port #)
- Map incoming packets
  - Consult the translation table
  - Map the destination address and port number
  - Local host receives the incoming packet

Network Address Translation Example

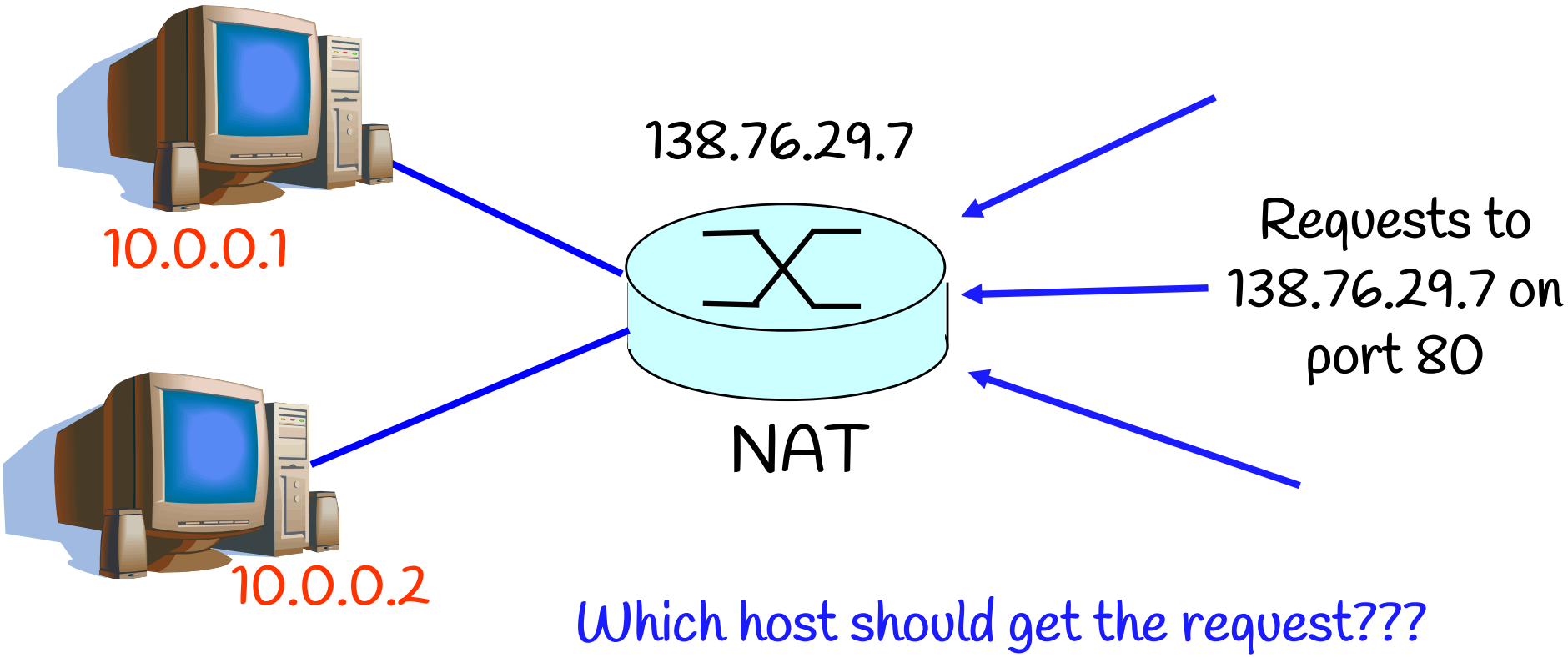


Maintaining the Mapping Table

- Create an entry upon seeing a packet
  - Packet with new (source addr, source port) pair
- Eventually, need to delete the map entry
  - But when to remove the binding?
- If no packets arrive within a time window
  - … then delete the mapping to free up the port #s
- Yet another example of "soft state"
  - I.e., removing state if not refreshed for a while

Objections Against NAT

- Port #s are meant for addressing processes
  - Yet, NAT uses them to identify end hosts
  - Makes it hard to run a server behind a NAT



Objections Against NAT

- Difficult to support peer-to-peer applications
  - P2P needs a host to act as a server
  - … difficult if both hosts are behind NATs
- 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

Where is NAT Implemented?

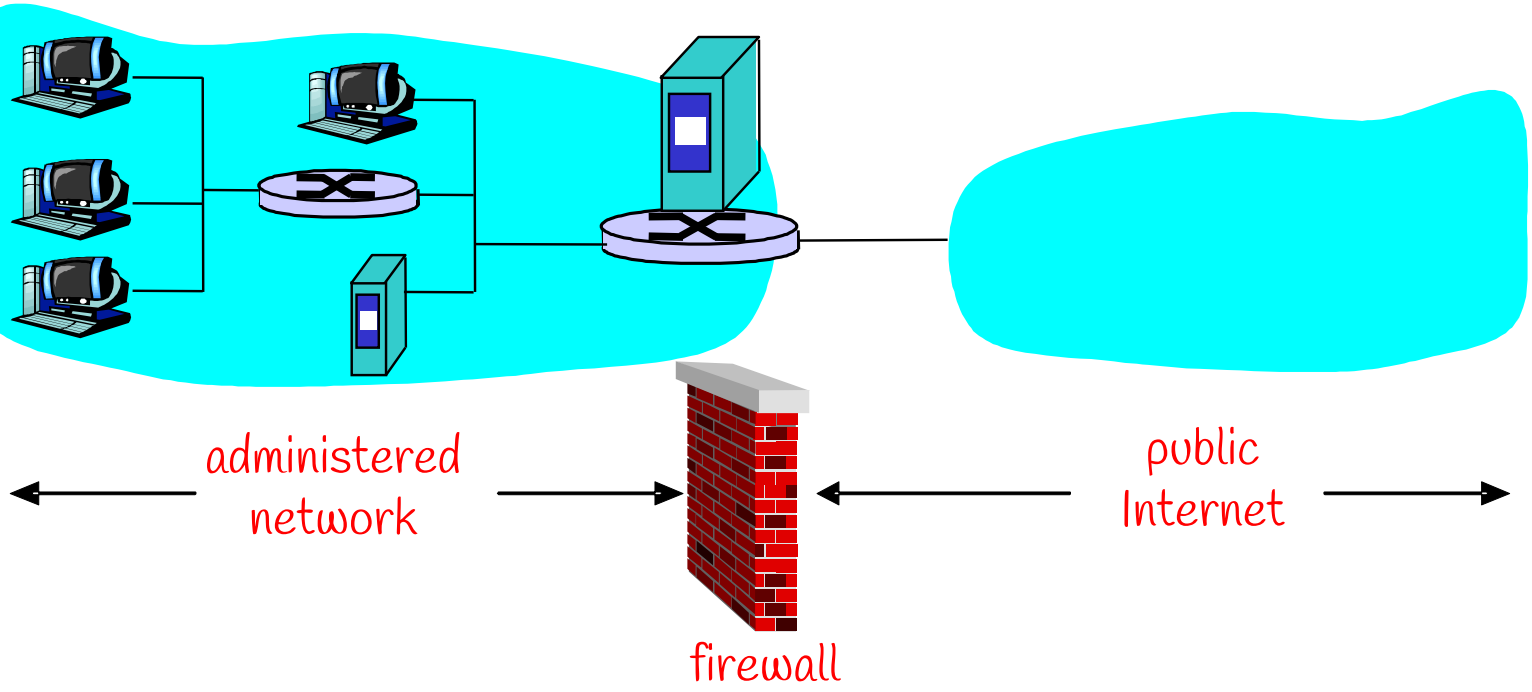
- Home router (e.g., Linksys box)
  - Integrates router, DHCP server, NAT, etc.
  - Use single IP address from the service provider
  - … and have a bunch of hosts hiding behind it
- Campus or corporate network
  - NAT at the connection to the Internet
  - Share a collection of public IP addresses
  - Avoid complexity of renumbering end hosts and local routers when changing service providers

Today

- Queuing Mechanisms
  - Random Early Detection (RED)
  - Explicit Congestion Notification (ECN)
- Middleboxes
  - Network Address Translation
  - Firewalls
  - Web Proxies

Firewalls

- Isolates organization's internal net from larger Internet, allowing some packets to pass, blocking others.



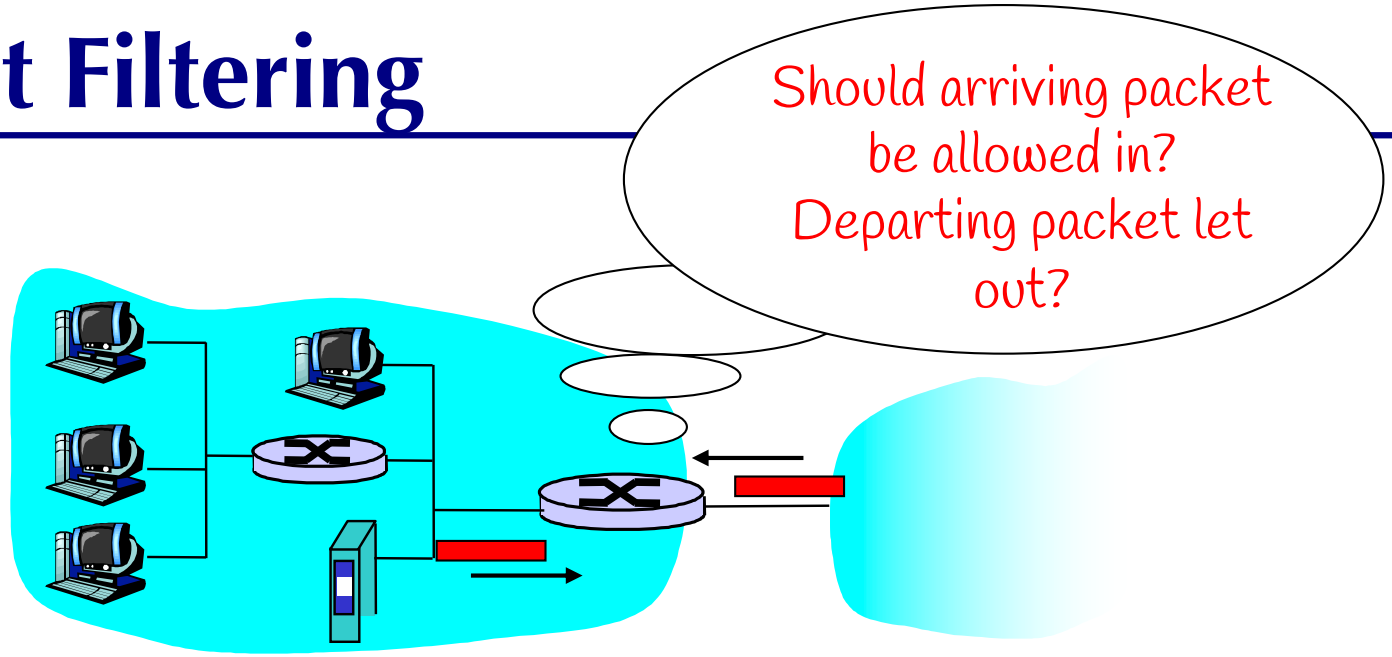
Internet Attacks: Denial of Service

- Denial-of-service attacks
  - Outsider overwhelms the host with unsolicited traffic
  - … with the goal of preventing any useful work
- Example
  - Bad guys take over a large collection of hosts
  - … and program these hosts to send traffic to your host
  - Leading to excessive traffic
- Motivations for denial-of-service attacks
  - Malice (e.g., just to be mean)
  - Revenge (e.g. for some past perceived injustice)
  - Greed (e.g., blackmailing)

Internet Attacks: Break-Ins

- Breaking in to a host
  - Outsider exploits a vulnerability in the end host
  - … with the goal of changing the behavior of the host
- Example
  - Bad guys know a Web server has a buffer-overflow vulnerability
  - … and, say, send an HTTP request with a long URL
  - Allowing them to break in
- Motivations for break-ins
  - Take over the machine to launch other attacks
  - Steal information stored on the machine
  - Modify/replace the content the site normally returns

Packet Filtering



- Internal network connected to Internet via firewall
- Firewall filters packet-by-packet, based on:
  - Source IP address, destination IP address
  - TCP/UDP source and destination port numbers
  - ICMP message type
  - TCP SYN and ACK bits

Packet Filtering Examples

- Block all packets with IP protocol field = 17 and with either source or dest port = 23.
  - All incoming and outgoing UDP flows blocked
  - All ssh connections are blocked
- Block inbound TCP packets with SYN but no ACK
  - Prevents external clients from making TCP connections with internal clients
  - But allows internal clients to connect to outside
- Block all packets with TCP port of Fortnight

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 the packet matches a rule, the decision is done

Firewall Configuration Example

- Alice runs a network in 222.22.0.0/16
  - Wants to let Bob's school access certain hosts
    - Bob is on 111.11.0.0/16
    - Alice's special hosts on 222.22.22.0/24
  - Alice doesn't trust Trudy, inside Bob's network
    - Trudy is on 111.11.11.0/24
  - Alice doesn't want any other traffic from Internet
- Rules
  - #1: Don't let Trudy machines in
    - Deny (src = 111.11.11.0/24, dst = 222.22.0.0/16)
  - #2: Let rest of Bob's network in to special dests
    - Permit (src=111.11.0.0/16, dst = 222.22.22.0/24)
  - #3: Block the rest of the world
    - Deny (src = 0.0.0.0/0, dst = 0.0.0.0/0)

A Variation: Traffic Management

- Permit vs. deny is too binary a decision
  - Maybe better to classify the traffic based on rules
  - … and then handle the classes of traffic differently
- Traffic shaping (rate limiting)
  - Limit the amount of bandwidth for certain traffic
  - E.g., rate limit on Web or P2P traffic
- Separate queues
  - Use rules to group related packets
  - And then do round-robin scheduling across the groups
  - E.g., separate queue for each internal IP address

Firewall Implementation Challenges

- Per-packet handling
  - Must inspect every packet
  - Challenging on very high-speed links
- Complex filtering rules
  - May have large # of rules
  - May have very complicated rules
- Location of firewalls
  - Complex firewalls near the edge, at low speed
  - Simpler firewalls in the core, at higher speed

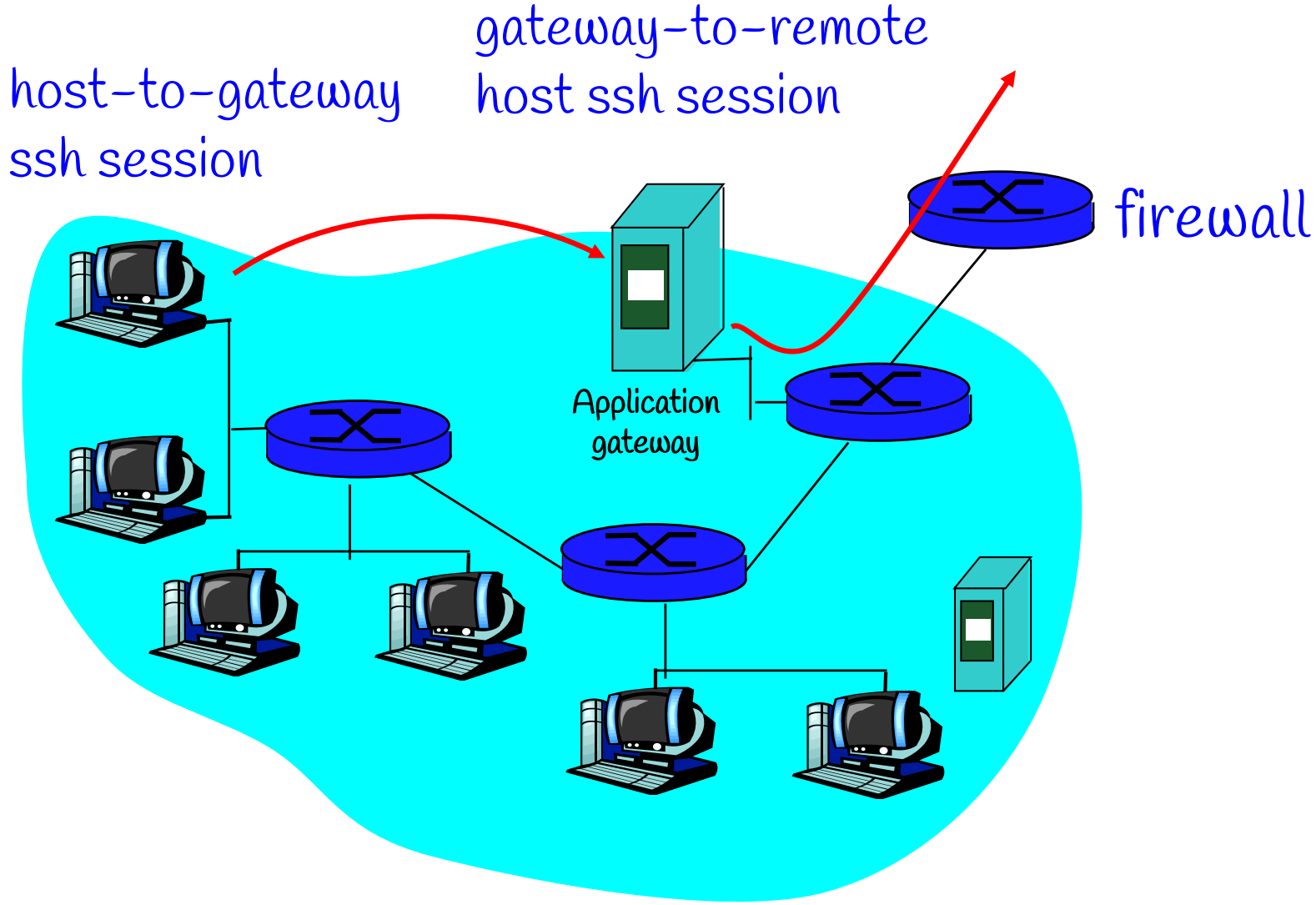
Clever Users Subvert Firewalls

- 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
    - E.g., connect from the dorms to another host
    - … and then onward to the blocked server
- 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

Application Gateways

- Filter packets on application data
  - Not just on IP and TCP/UDP headers
- Example: restricting ssh usage
  - Don't allow any external clients to ssh inside
  - Only allow certain internal users to ssh outside
- Solution: ssh gateway
  - Force all ssh traffic to go through a gateway
  - I.e. filter ssh traffic that doesn't originate from the IP address of the gateway
- At the gateway…
  - Require user to login and provide password
  - Apply policy to decide whether they can proceed

ssh Gateway Example



Motivation for Gateways

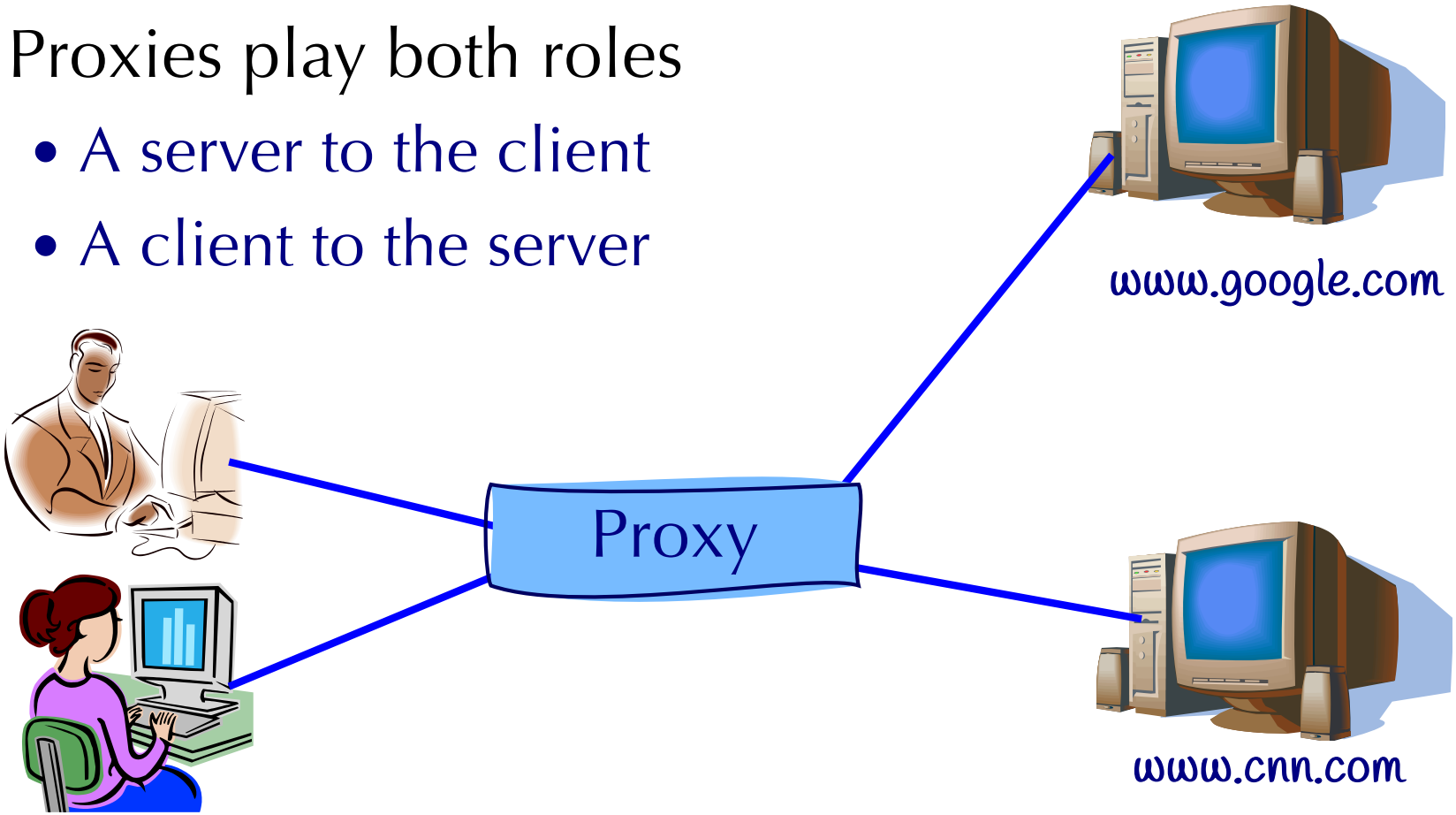
- Enable more detailed policies
  - E.g., login id and password at ssh gateway
- Avoid rogue machines sending traffic
  - E.g., e-mail "server" running on user machines
  - … probably a sign of a spammer
- Enable a central place to perform logging
  - E.g., forcing all Web accesses through a gateway
  - … to log the IP addresses and URLs
- Improve performance through caching
  - E.g., forcing all Web accesses through a gateway
  - … to enable caching of the popular content

Today

- Queuing Mechanisms
  - Random Early Detection (RED)
  - Explicit Congestion Notification (ECN)
- Middleboxes
  - Network Address Translation
  - Firewalls
  - Web Proxies

Web Clients and Servers

- Web is a client-server protocol
  - Client sends a request
  - Server sends a response
- Proxies play both roles
  - A server to the client
  - A client to the server



Proxy Caching

- Client #1 requests http://www.foo.com/fun.jpg
  - Client sends "GET fun.jpg" to the proxy
  - Proxy sends "GET fun.jpg" to the server
  - Server sends response to the proxy
  - Proxy stores the response, and forwards to client
- Client #2 requests http://www.foo.com/fun.jpg
  - Client sends "GET fun.jpg" to the proxy
  - Proxy sends response to the client from the cache
- Benefits
  - Faster response time to the clients
  - Lower load on the Web server
  - Reduced bandwidth consumption inside the network

Getting Requests to the Proxy

- Explicit configuration
  - Browser configured to use a proxy
  - Directs all requests through the proxy
  - Problem: requires user action
- Transparent proxy (or "interception proxy")
  - Proxy lies in path from the client to the servers
  - Proxy intercepts packets en route to the server
  - … and interposes itself in the data transfer
  - Benefit: does not require user action

Challenges of Transparent Proxies

- Must ensure all packets pass by the proxy
  - By placing it at the only access point to the Internet
  - E.g., at the border router of a campus or company
- Overhead of reconstructing the requests
  - Must intercept the packets as they fly by
  - … and reconstruct into the ordered by stream
- May be viewed as a violation of user privacy
  - The user does not know the proxy lies in the path
  - Proxy may be keeping logs of the user's requests

Other Functions of Web Proxies

- Anonymization
  - Server sees requests coming from the proxy address
  - … rather than the individual user IP addresses
- Transcoding
  - Converting data from one form to another
  - E.g., reducing the size of images for cell-phone browsers
- Prefetching
  - Requesting content before the user asks for it
- Filtering
  - Blocking access to sites, based on URL or content

Conclusions

- Middleboxes address important problems
  - Using fewer IP addresses
  - Blocking unwanted traffic
  - Making fair use of network resources
  - Improving Web performance
- Middleboxes cause problems of their own
  - No longer globally unique IP addresses
  - No longer can assume network simply delivers packets