Quality of Service (continued)

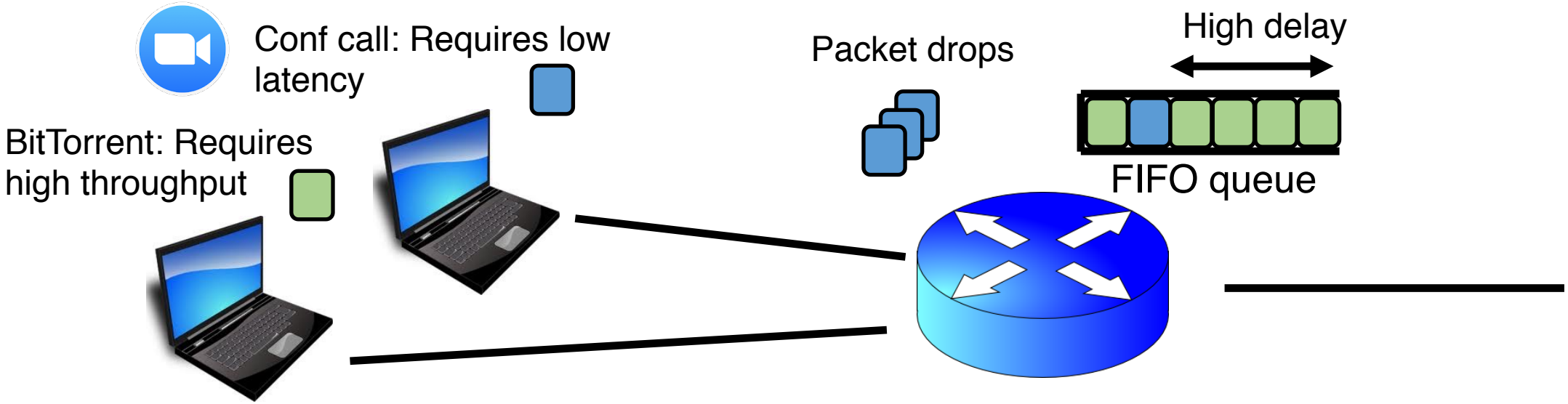
Lecture 26

http://www.cs.rutgers.edu/~sn624/352-S22

Srinivas Narayana

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Best effort networking isn't enough



- Resource contention occurs in the core of the network
- Congestion control will react, but may be too little & too late
- Instead, network should differentiate classes of traffic
- Network should provide service guarantees for each class
  - Prioritization, rate limiting, fair queueing
- Implemented in the router's packet scheduler

Why care about service guarantees?

- Influences how packets are treated at contentious resources in the core of the network
  - Regardless of the endpoint transport

- Implementations of scheduling (QoS) within large networks have implications for debates on network neutrality

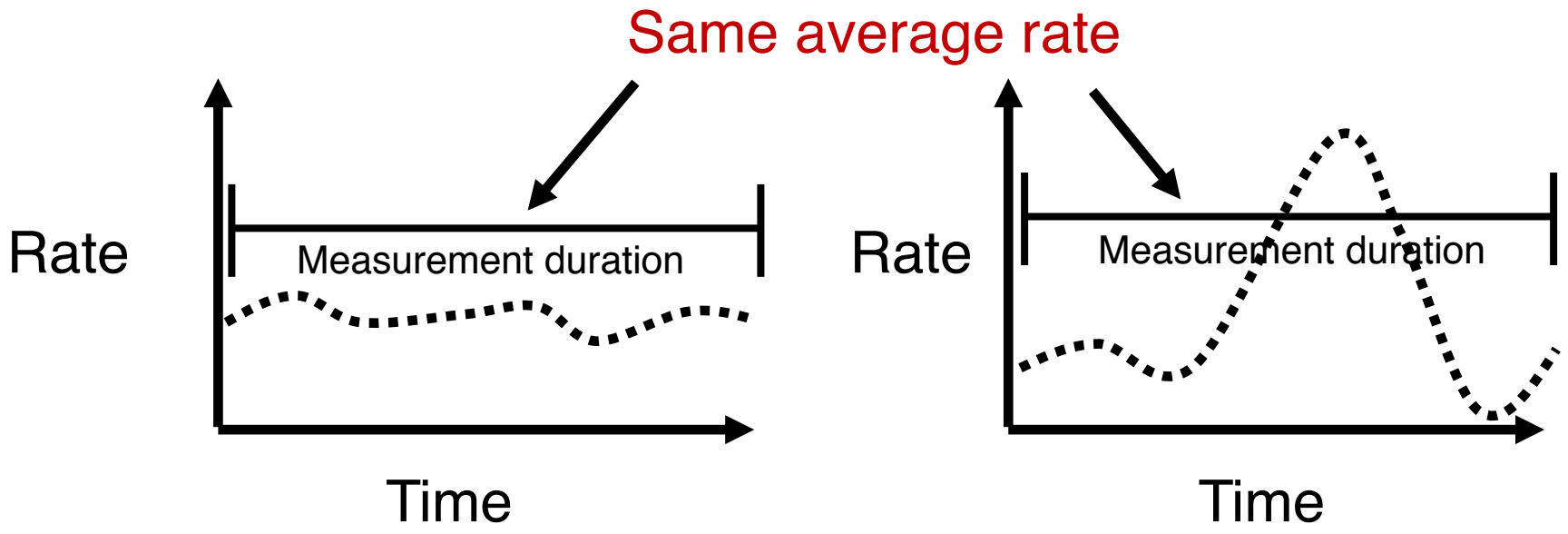
- Scheduling is a fundamental problem in computer networks

- Next: Rate limiting

Rate Limiting

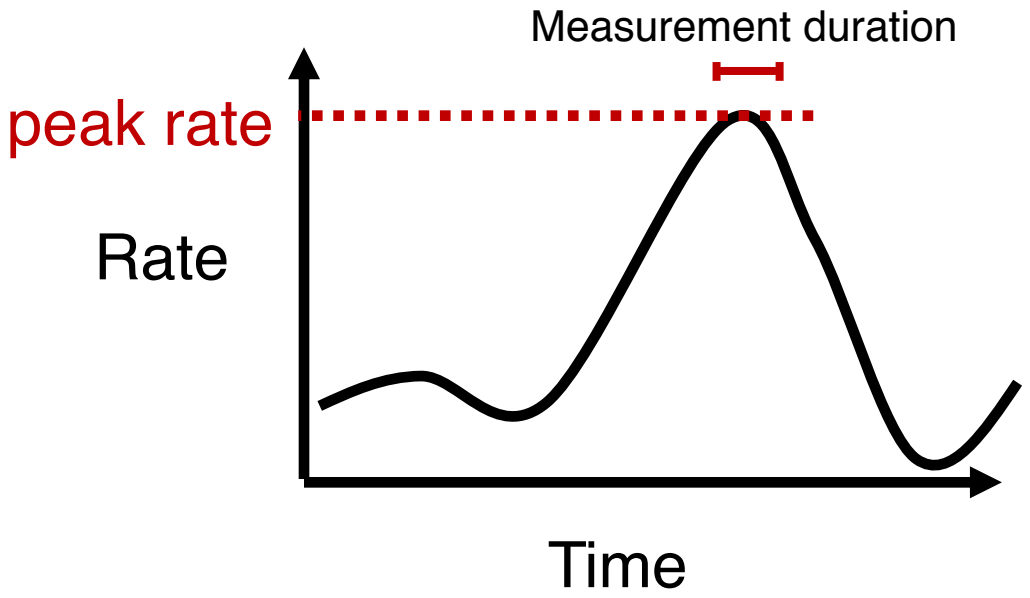
Measures of transmission rate

- Long-term/average rate: data rate transmitted per unit time, over a long period
  - Crucial question: what is the time interval over which rate is measured?
- Average and instantaneous behaviors can be very different



Measures of transmission rate

- Peak rate: largest instantaneous rate that is transmitted
  - Measurement duration is typically very small
- Burst size: maximum amount of data sent consecutively without any intervening idle periods



Rate enforcement

- There are two kinds of rate enforcement policies:
  - shaping and policing
- Two specific mechanisms to implement those:
  - leaky buckets and token buckets
- Basic mechanism: wait for the passage of appropriate time before releasing packets for transmission

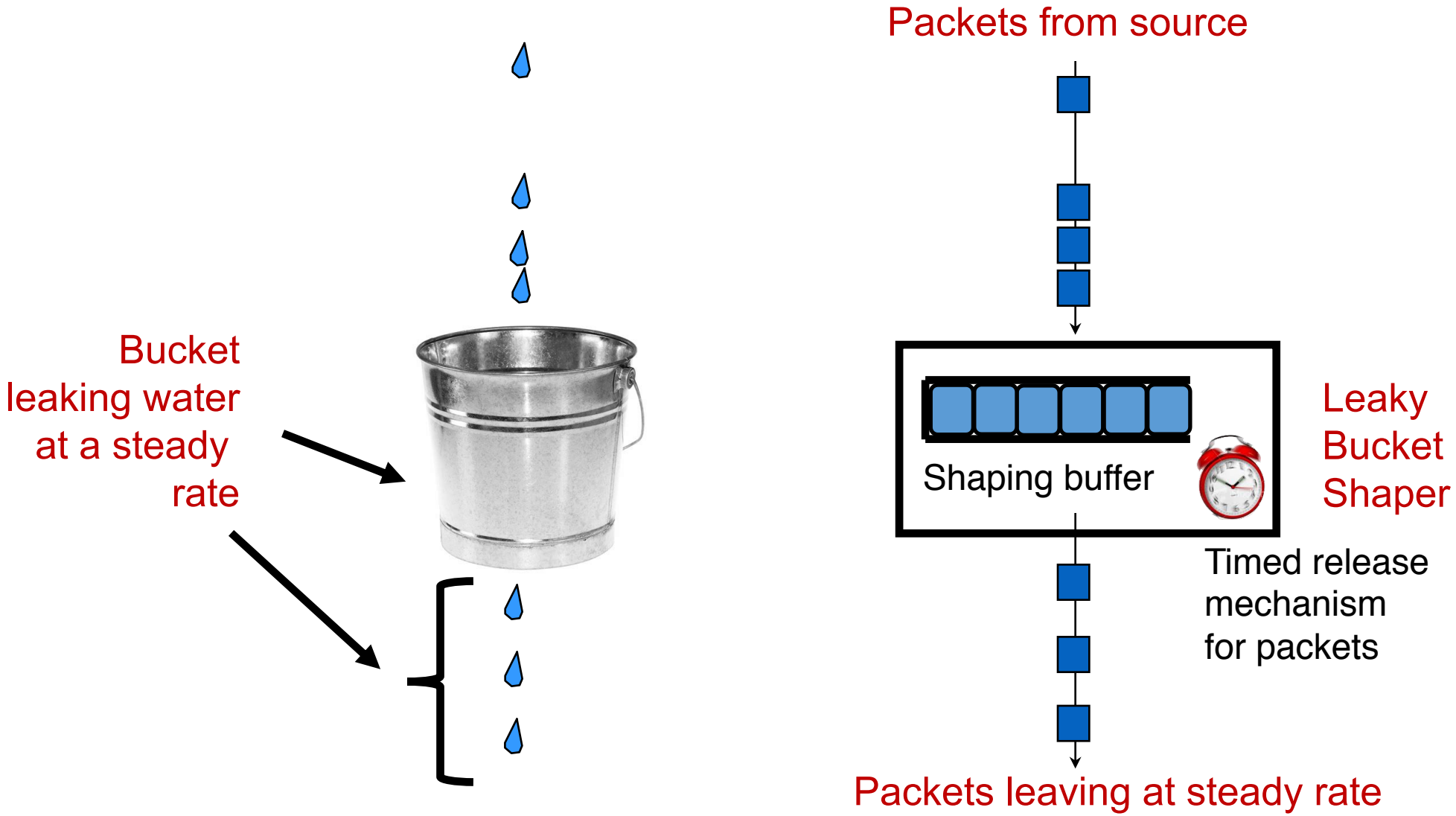
Shaping vs. Policing

- Enforces rate by queueing excess packets in a buffer
  - Drop only if buffer is full
- Requires memory to buffer packets
- Can inflate round-trip time (queueing in shaping buffer)

- Enforces rate by dropping excess packets immediately
  - Can result in high loss rates
- Does not require a memory buffer
- No additional inflation in round-trip times

Leaky bucket shaper

Intuition: release packets at steady rate



Leaky Bucket Shaper

- Packets may enter in a bursty manner
- However, once they pass through the leaky bucket, they are evenly spaced
- The shaping buffer holds packets up to a certain point
  - If the buffer is full, packets are dropped
- Setting the rate is a policy concern
  - Assume an admin provides us the rate
- Shapers may be used in the core of a network to limit bandwidth use, or at the edge to pace packets entering the network in the first place

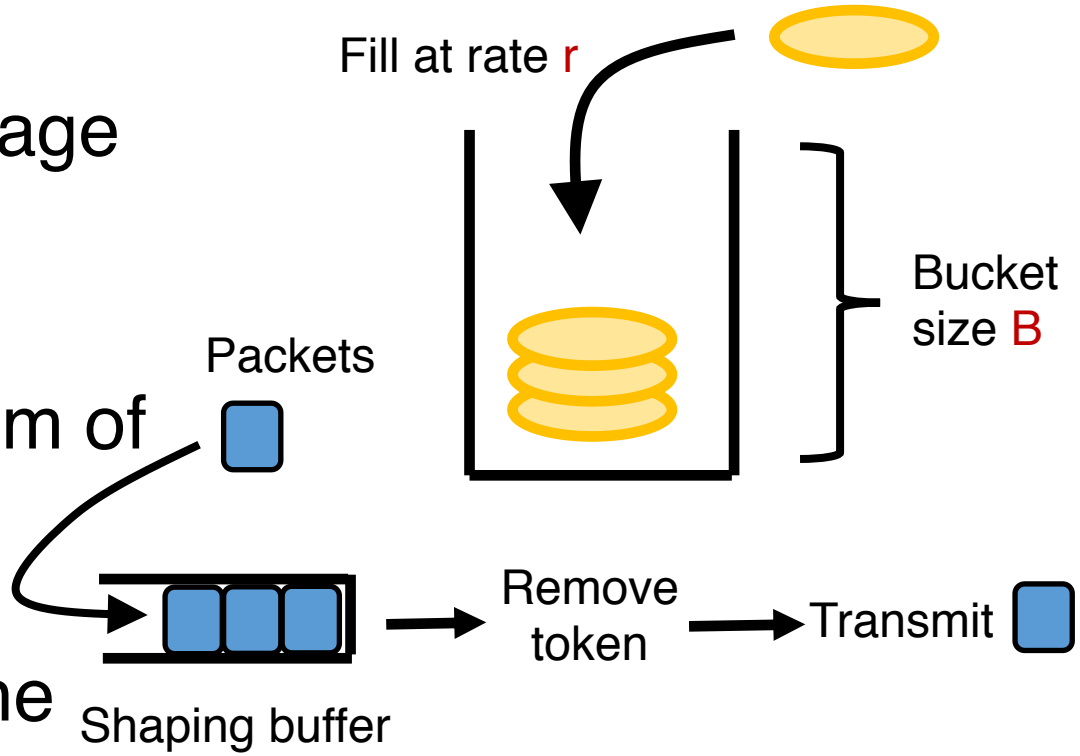
Leaky Bucket Shaper

- For a leaky bucket shaper, assume average rate == peak rate
- However, many Internet transfers just have a few packets
  - For example, web requests and responses
  - Enforcing rate limit for those can significantly delay completion
- We often wish to have peak rate higher than avg rate
  - Especially at the beginning of a connection
  - If so, use a token bucket: burst-tolerant version of a leaky bucket

Token bucket shaper

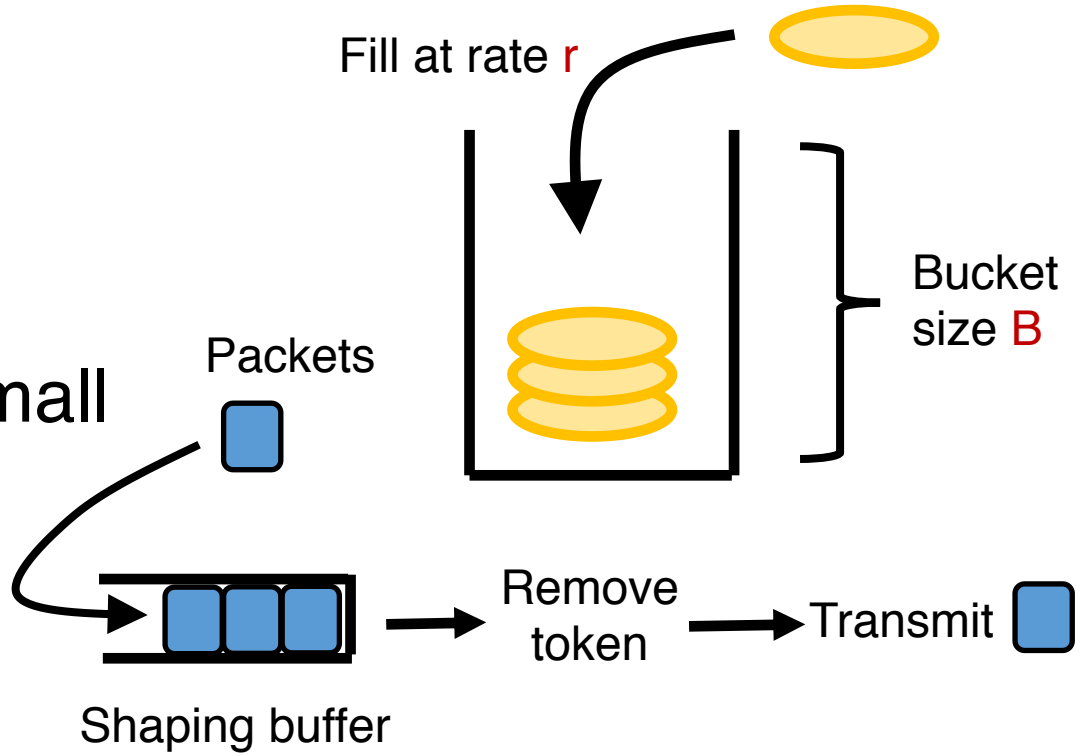
Token bucket shaper

- Limits traffic class to a specified average rate r and burst size B
- Tokens are filled in at rate r
- The token bucket can hold a maximum of B tokens. Further tokens dropped
  - Note: distinct from shaping buffer size
- Suppose a packet is at the head of the shaping buffer
- If a token exists in the bucket, remove token, and transmit the packet
  - If not, wait.



Token bucket shaper

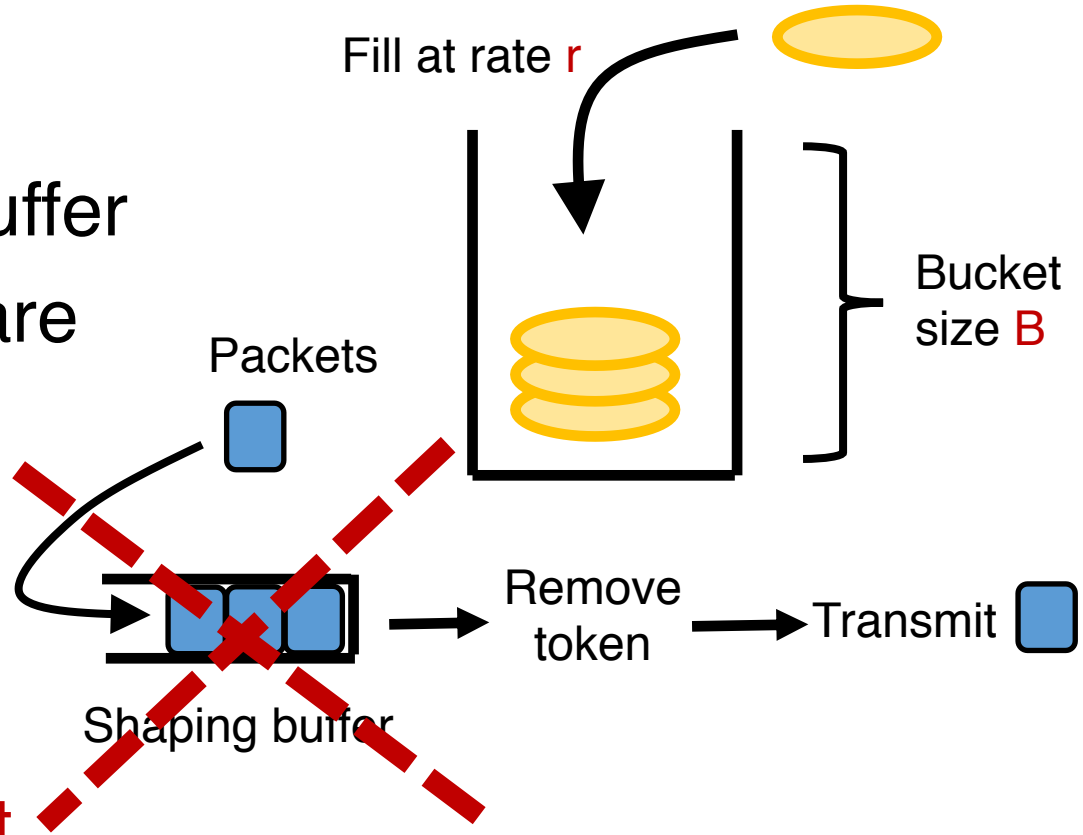
- In time t, the maximum number of packets that depart the shaper is (r * t) + B
- A full bucket of tokens would allow small flows to go through unaffected
  - A maximum burst of B packets
- Longer flows have average rate r
  - Bucket emptied initially, the rest of the flow must respect the token fill rate
  - As t → ∞, the average rate approaches r
  - That is, (1/t) * (r*t + B) → r



Token bucket policers

Token bucket policer

- A token bucket policer is just a token bucket shaper without the shaping buffer
- No place for packets to wait if there are no tokens
- If token exists, packet transmitted.
- If not, packet dropped
- Simple and efficient to implement.
- The internet has tons of token bucket policers



Google study from 2016

Region	Policed segments (among lossy)	Policed segments (overall)	Loss rate (policed)	Loss rate (non-pol.)
India	6.8%	1.4%	28.2%	3.9%
Africa	6.2%	1.3%	27.5%	4.1%
Asia (w/o India)	6.5%	1.2%	22.8%	2.3%
South America	4.1%	0.7%	22.8%	2.3%
Europe	5.0%	0.7%	20.4%	1.3%
Australia	2.0%	0.4%	21.0%	1.8%
North America	2.6%	0.2%	22.5%	1.0%

Table 2: % segments policed among lossy segments ($\geq$ 15 losses, the threshold to trigger the policing detector), and overall. Avg. loss rates for policed and unpoliced segments.

Small but non-trivial fraction of policed links

Significant impact on packet loss rate

Flach et al., An Internet-Wide Analysis of Traffic Policing, SIGCOMM 2016

Impact on TCP

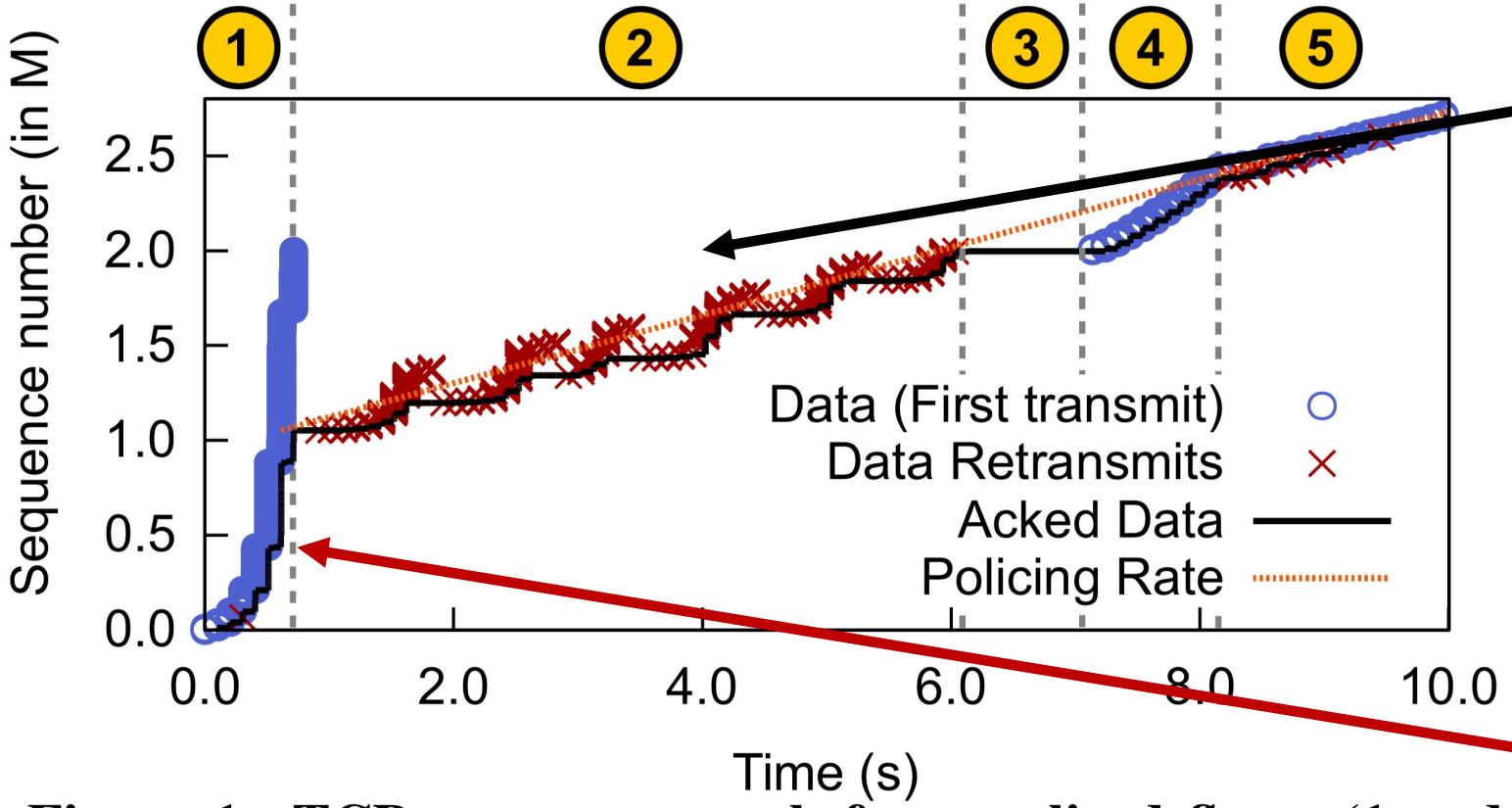


Figure 1: TCP sequence graph for a policed flow: (1 and 4) high throughput until token bucket empties, (2 and 5) multiple rounds of retransmissions to adjust to the policing rate, (3) idle period between chunks pushed by the application.

Policers drop multiple packets in a burst: causing RTOs and retransmissions after emptying of token bucket

Slow start period: burst allowed with a full bucket of tokens

Effect on actual apps: YouTube

- Video rebuffer rate: rebuffer time / overall watch time

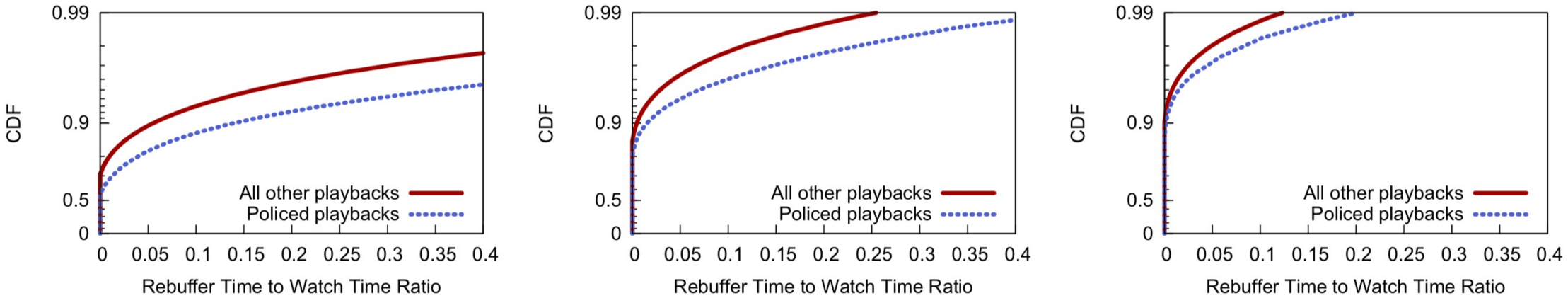


Figure 9: Rebuffer to watch time ratios for video playbacks. Each had at least one chunk with a goodput of 300 kbps, 1.5 Mbps, or 3 Mbps (±15%).

Summary of rate limiting

- Rate limiting is a useful mechanism to isolate traffic classes from each other
- Two strategies: policing and shaping
  - Leaky bucket and token bucket
- The Internet has a lot of token bucket policers, causing real impact on TCP connections and app performance
- Understand how ISPs treat consumer Internet traffic

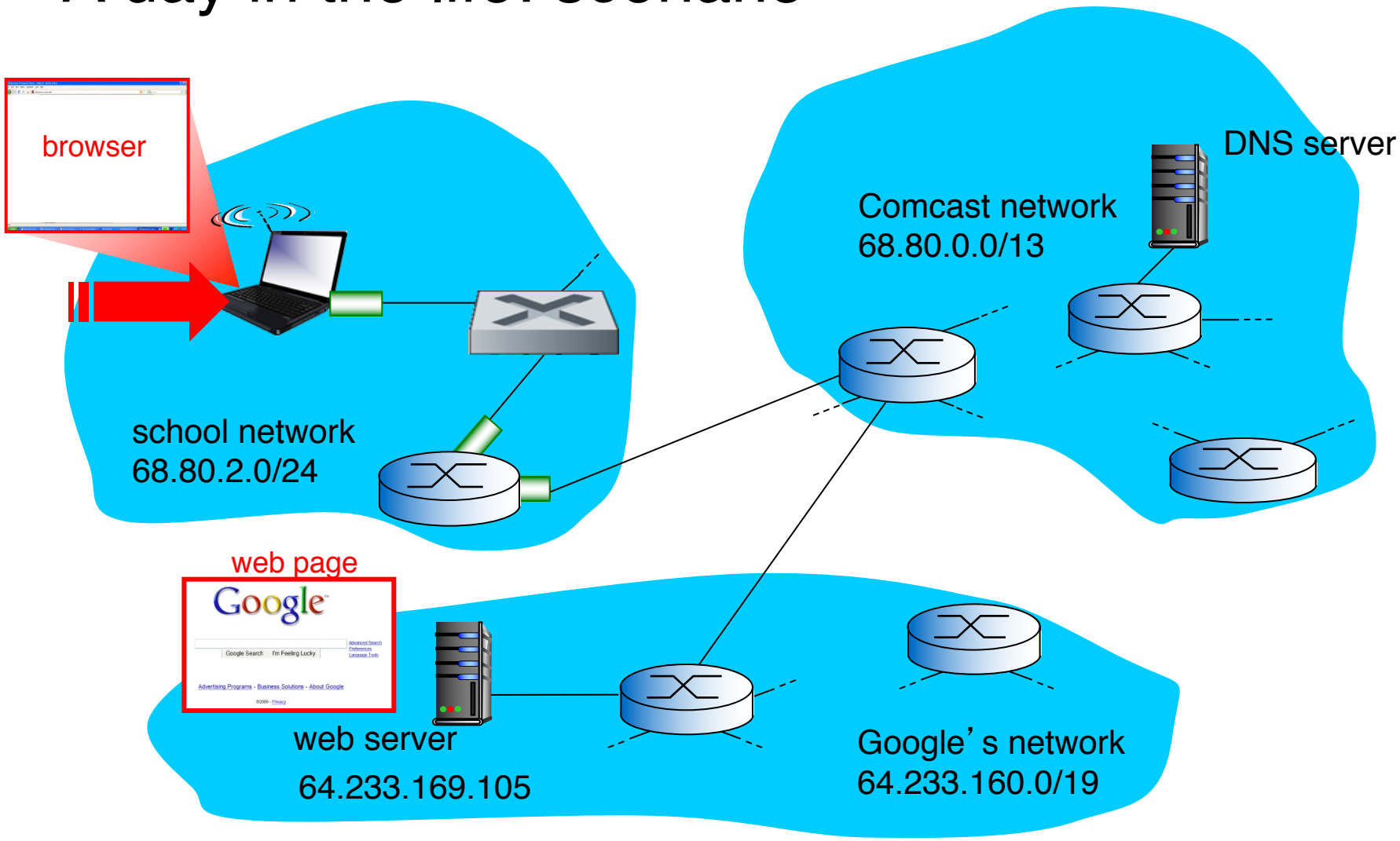
Synthesis of protocols

Synthesis: a day in the life of a web request

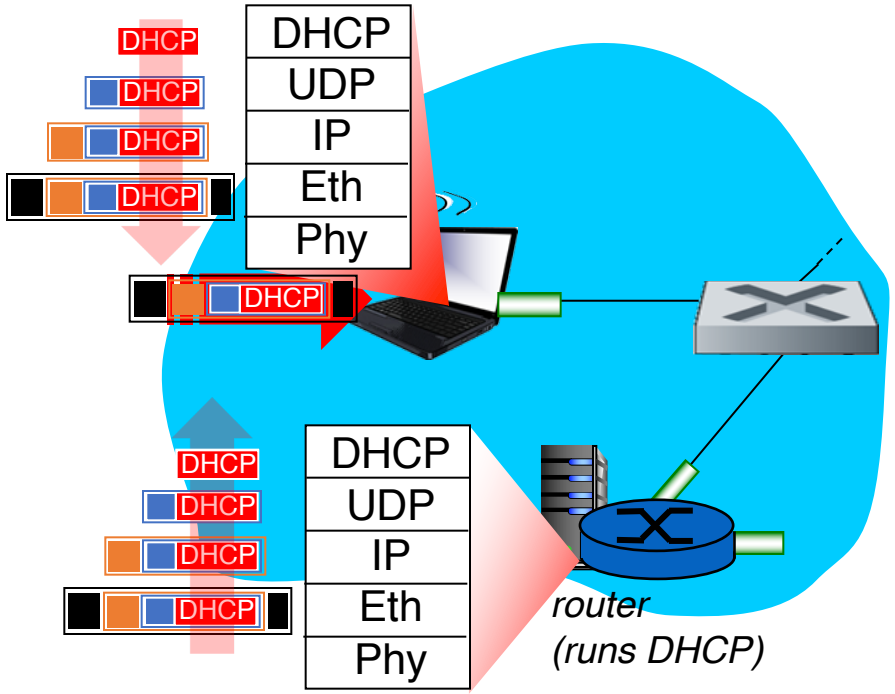
- Goal: identify, review, understand protocols (at all layers) involved in seemingly simple scenario: requesting www page

- Scenario: student attaches laptop to campus network, requests/receives www.google.com

A day in the life: scenario



A day in the life… connecting to the Internet



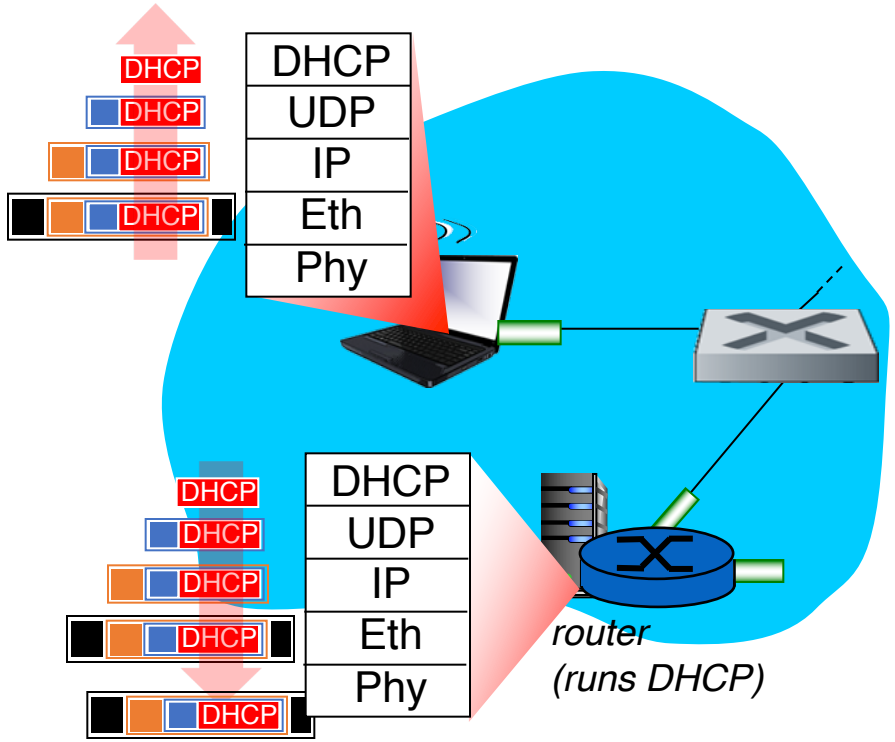
- connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use *DHCP*

- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in link layer Ethernet

- Packet broadcast (dest: FFFFFFFFFFFF) on the local network, received at a router running DHCP server

- Ethernet decapsulated to IP decapsulated to UDP decapsulated to DHCP

A day in the life… connecting to the Internet

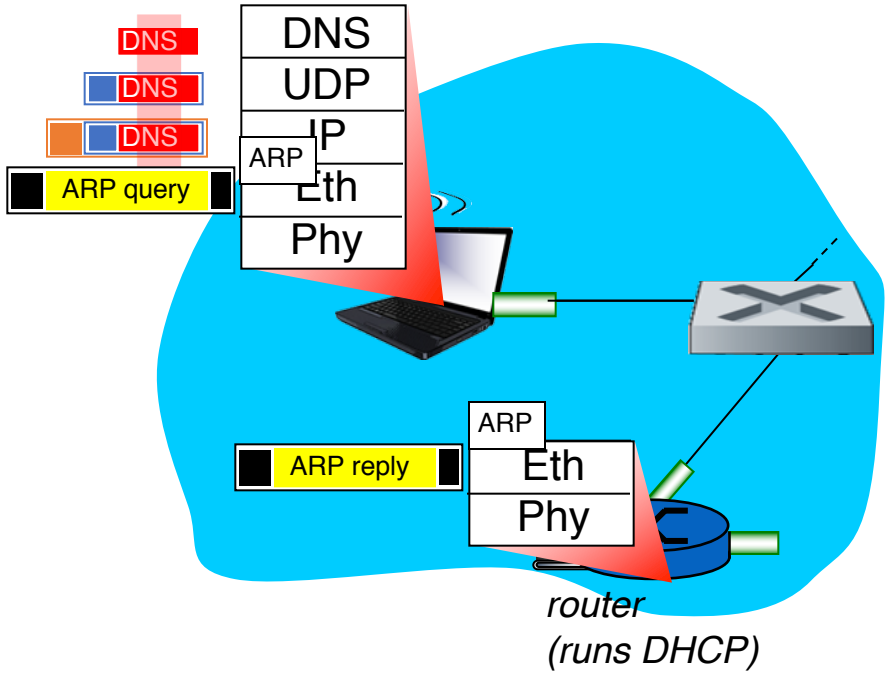


- DHCP server formulates *DHCP ACK* containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server

- DHCP client receives DHCP ACK reply

Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

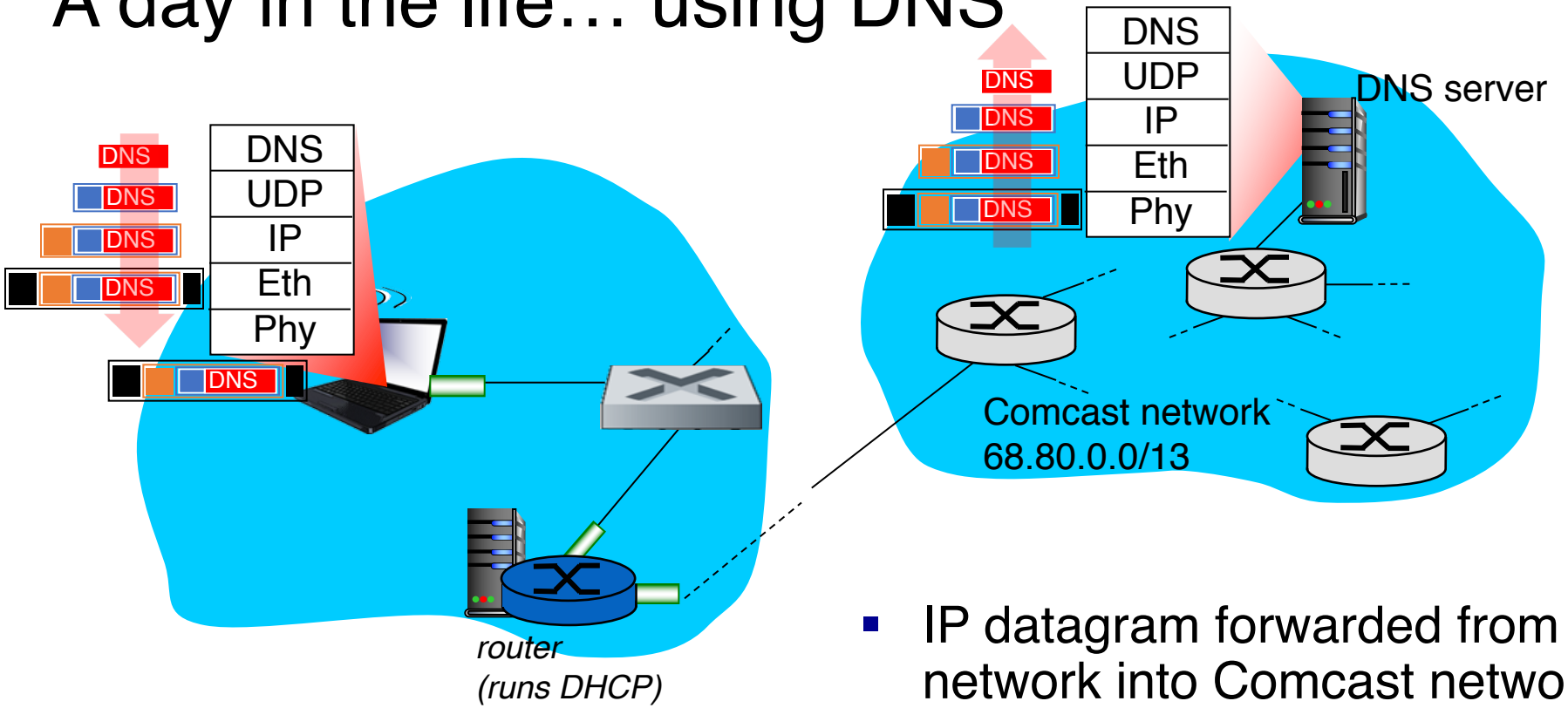
A day in the life… ARP (before DNS, before HTTP)



- before sending *HTTP* request, need IP address of www.google.com: *DNS*

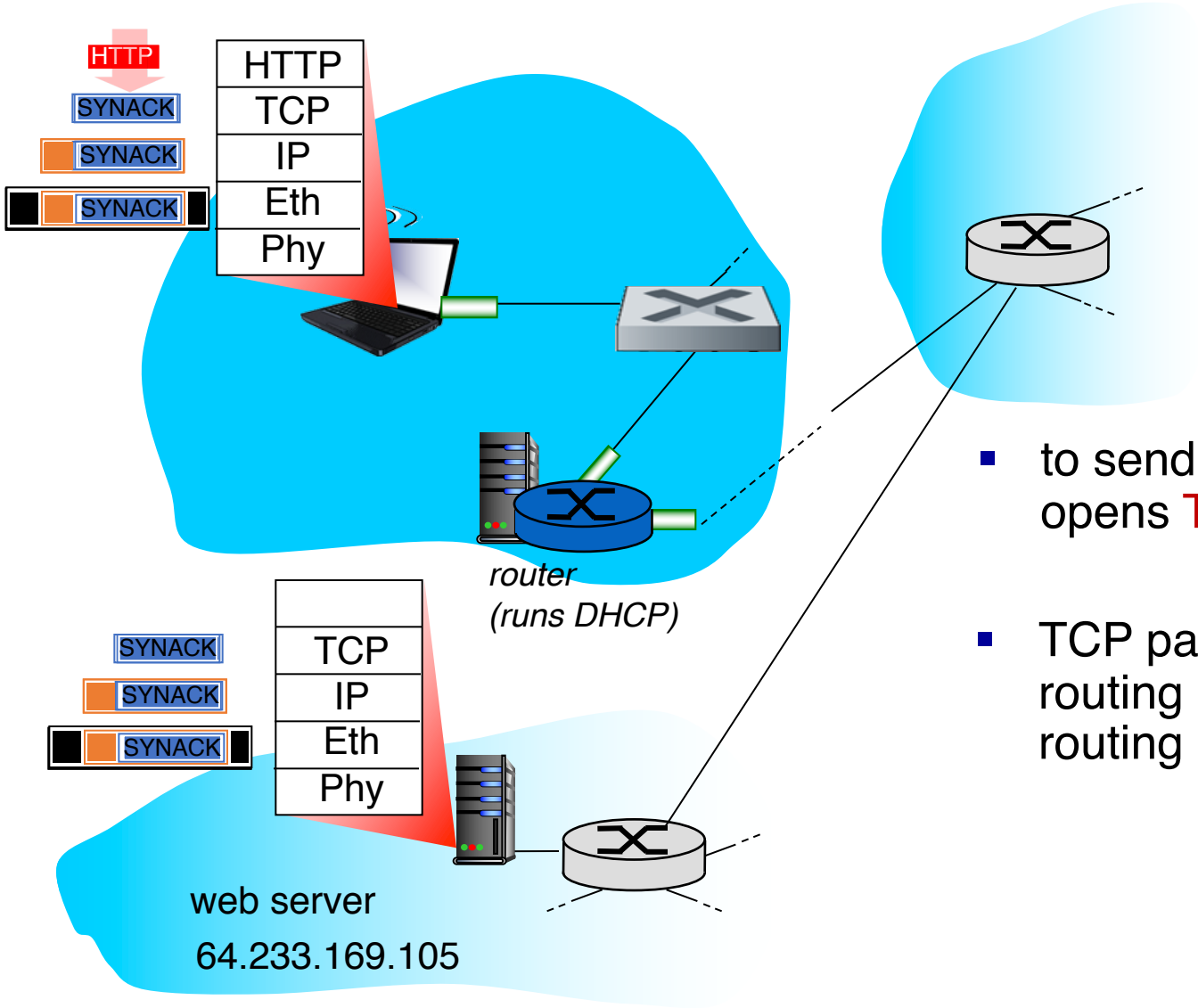
  - DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: ARP
  - ARP query broadcast, received by router, which replies with ARP reply giving MAC address of router interface
  - client now knows MAC address of gateway router, so can now send packet containing DNS query

A day in the life… using DNS



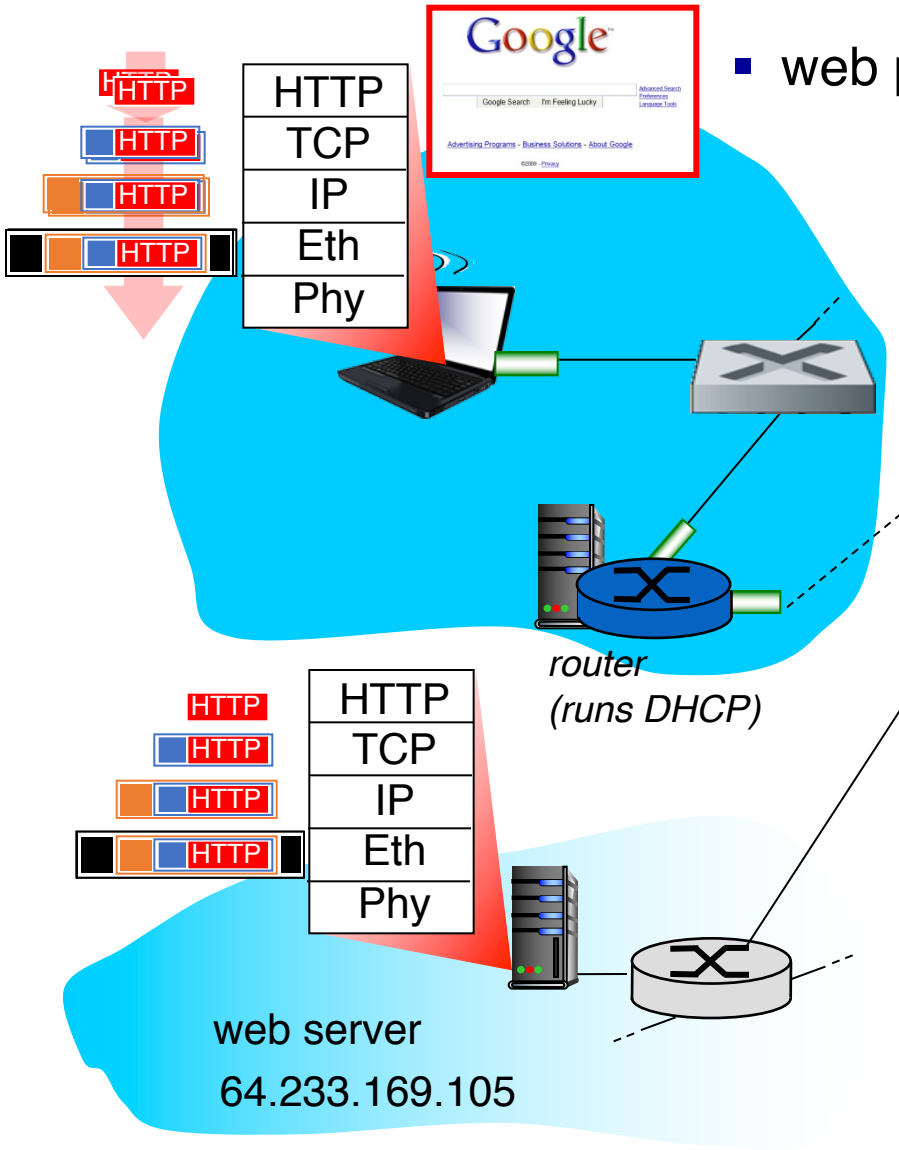
- IP datagram containing DNS query from client to gateway router
- IP datagram forwarded from campus network into Comcast network, routed (tables created by EIGRP, OSPF, and/or BGP routing protocols) to DNS server
- decapsulated to DNS server
- DNS server replies to client with IP address of www.google.com

A day in the life…TCP connection carrying HTTP



- to send HTTP request, client first opens TCP socket to web server
- TCP packet routed using inter-domain routing (BGP) and intra-domain routing (OSPF, EIGRP) to web server

A day in the life… HTTP request/reply



- web page finally (!!!) displayed

- HTTP request sent into TCP socket
- IP datagram containing HTTP request routed to www.google.com
- web server responds with HTTP reply (containing web page)
- IP datagram containing HTTP reply routed back to client

Internet Technology

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Outro

- Computer networks are a stack of layers
  - Built for modularity
  - Each layer does one set of functions very well
  - Each layer depends on the layers beneath it

- Many general and useful principles
  - Applicable to real life (e.g., feedback control)
  - Applicable to computer system design (e.g., indirection & hierarchy)

Outro: Now what?

- Go about life as usual
  - One difference: enhanced abilities to work with Internet-based tech
- Apply your new skills to solve a problem you care about
  - Tons of free and open-source software and platforms. Opportunities
- Deepen your understanding of the Internet and its tech
  - CS 552 Computer Networks (Fall 2022 - Prof Rich Martin)
  - CS 553 Internet services (Spring 2023 - me)
- Push the boundaries of Internet tech
  - Talk to me about research