Course and Exam Review

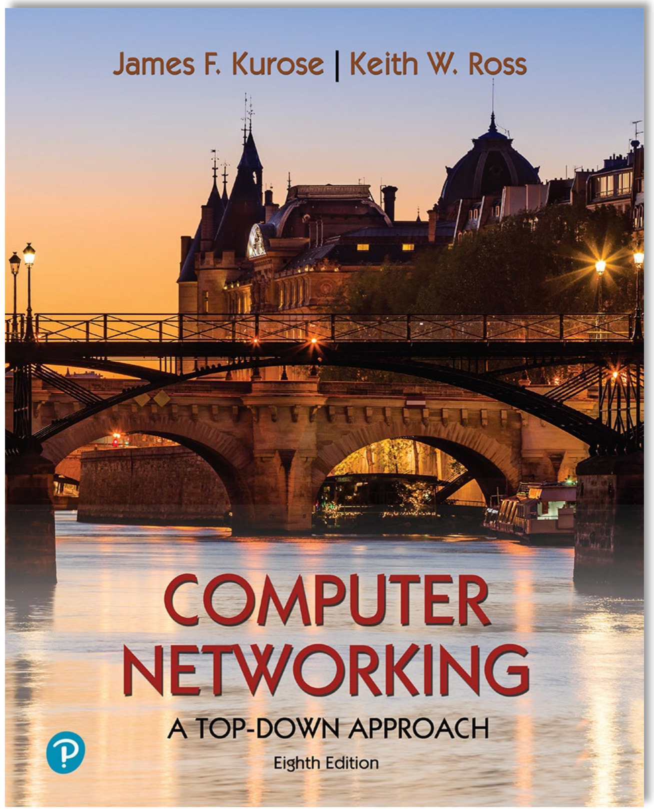
A note on the use of these PowerPoint slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

For a revision history, see the slide note for this page.

Thanks and enjoy! JFK/KWR

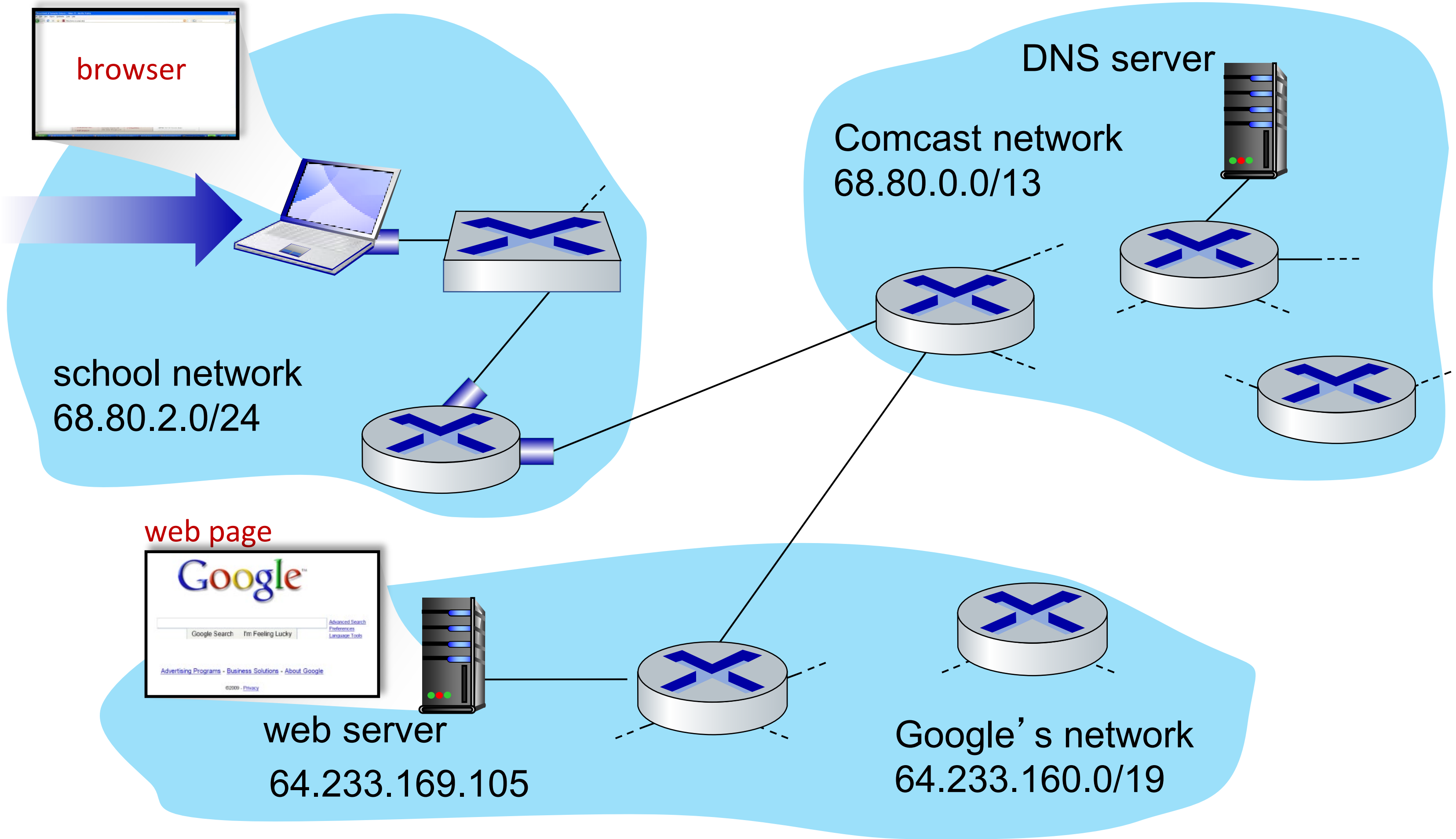


Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Synthesis: a day in the life of a web request

- our journey down the protocol stack is now complete!
  - application, transport, network, link
- putting-it-all-together: synthesis!
  - *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

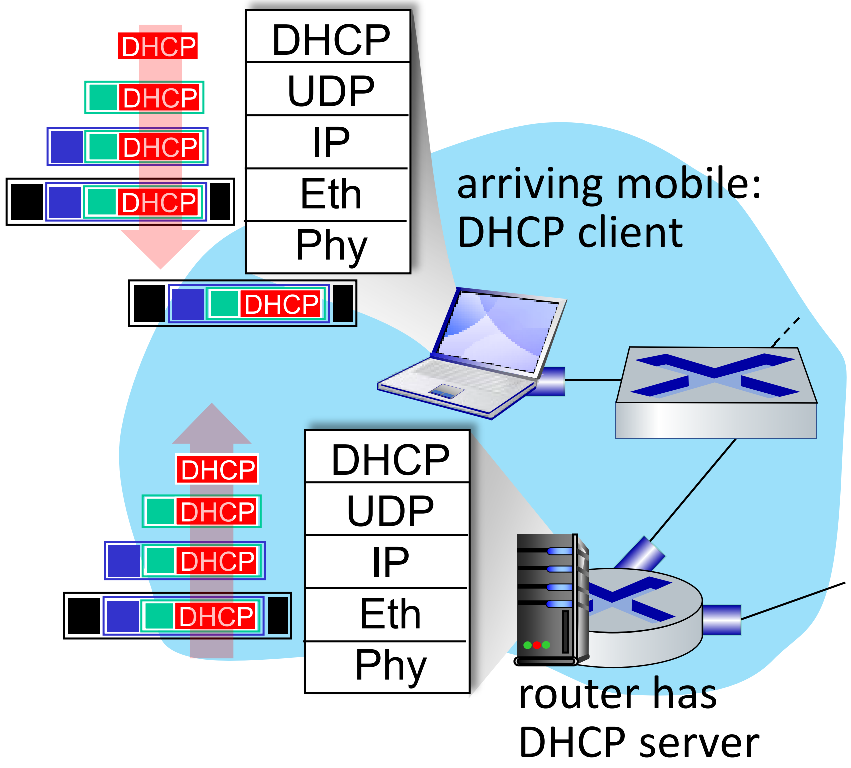


scenario:

- arriving mobile client attaches to network …
- requests web page: www.google.com

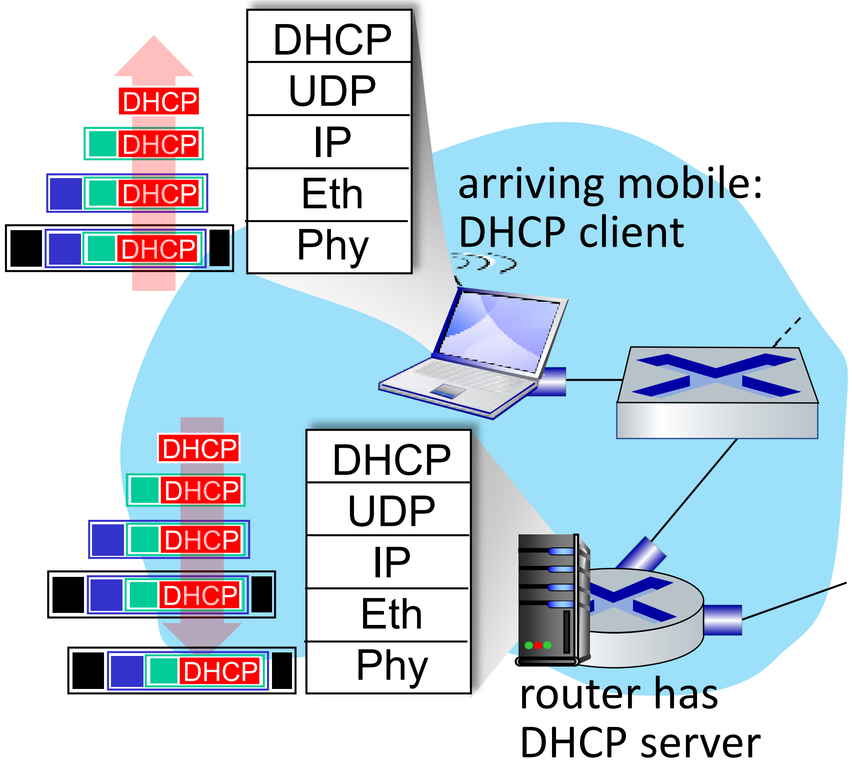
Sounds simple!

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 802.3 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

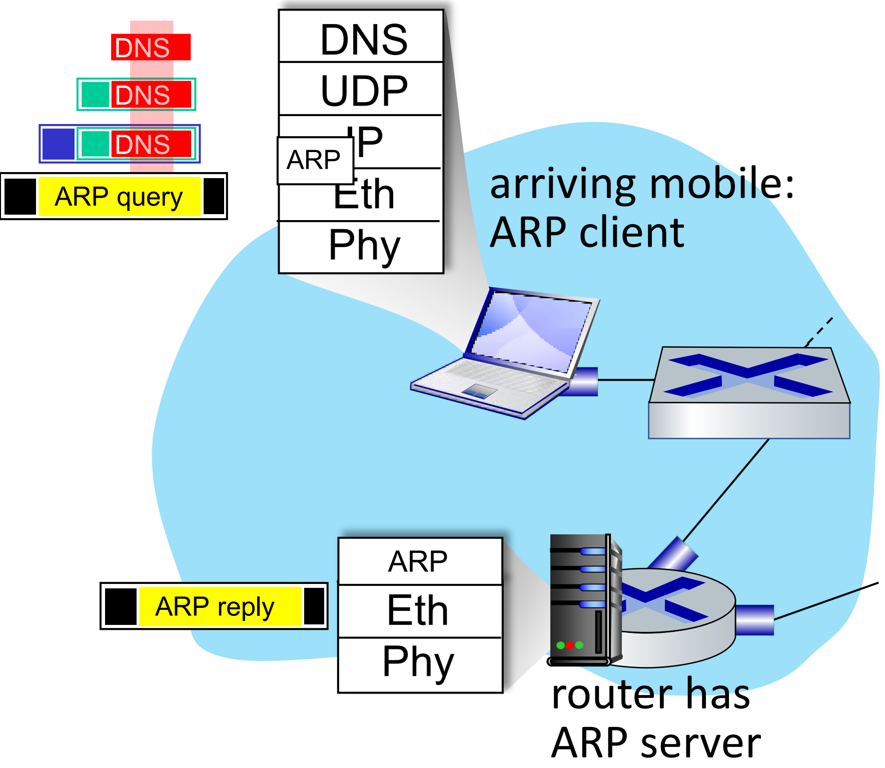
A day in the life: connecting to the Internet



- DHCP server formulates DHCP ACK containing client’s IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

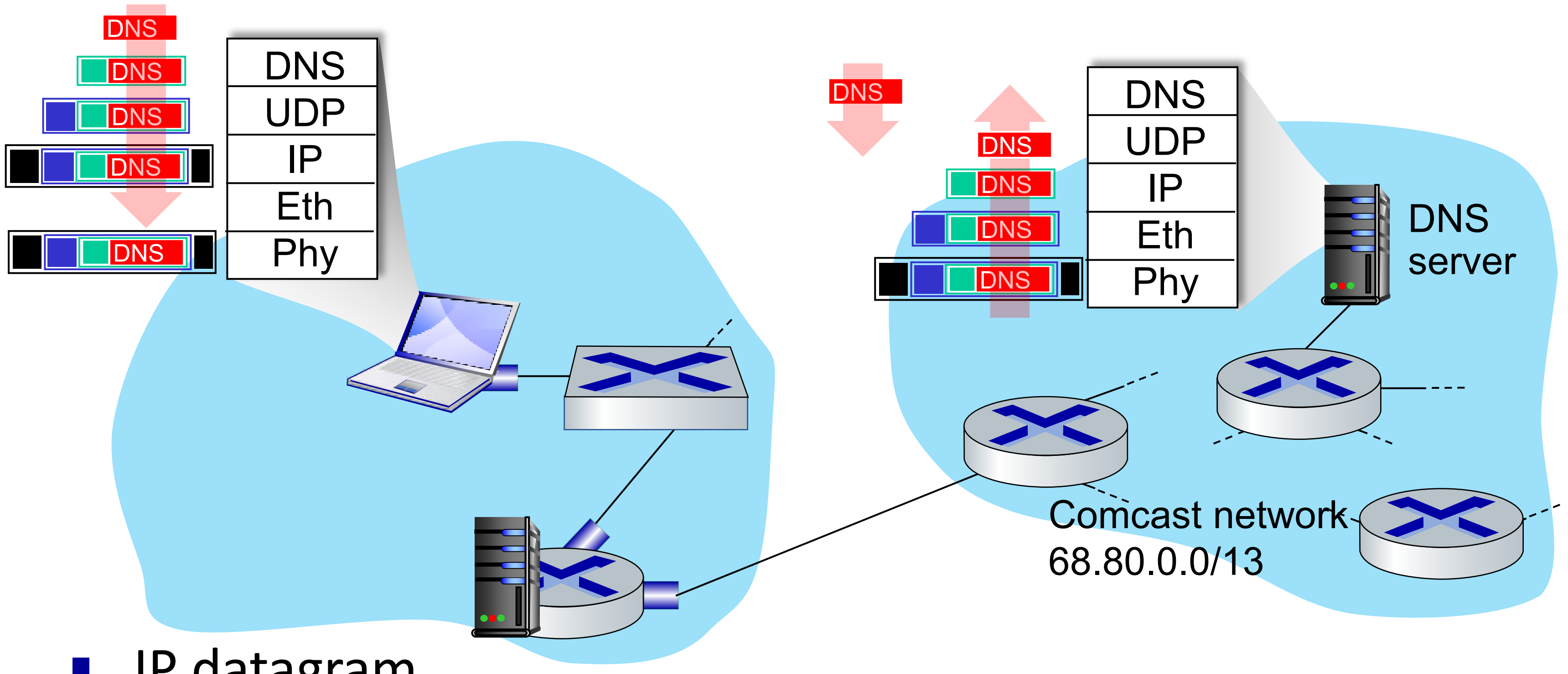
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

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 first hop router, so can now send frame containing DNS query

A day in the life… using DNS



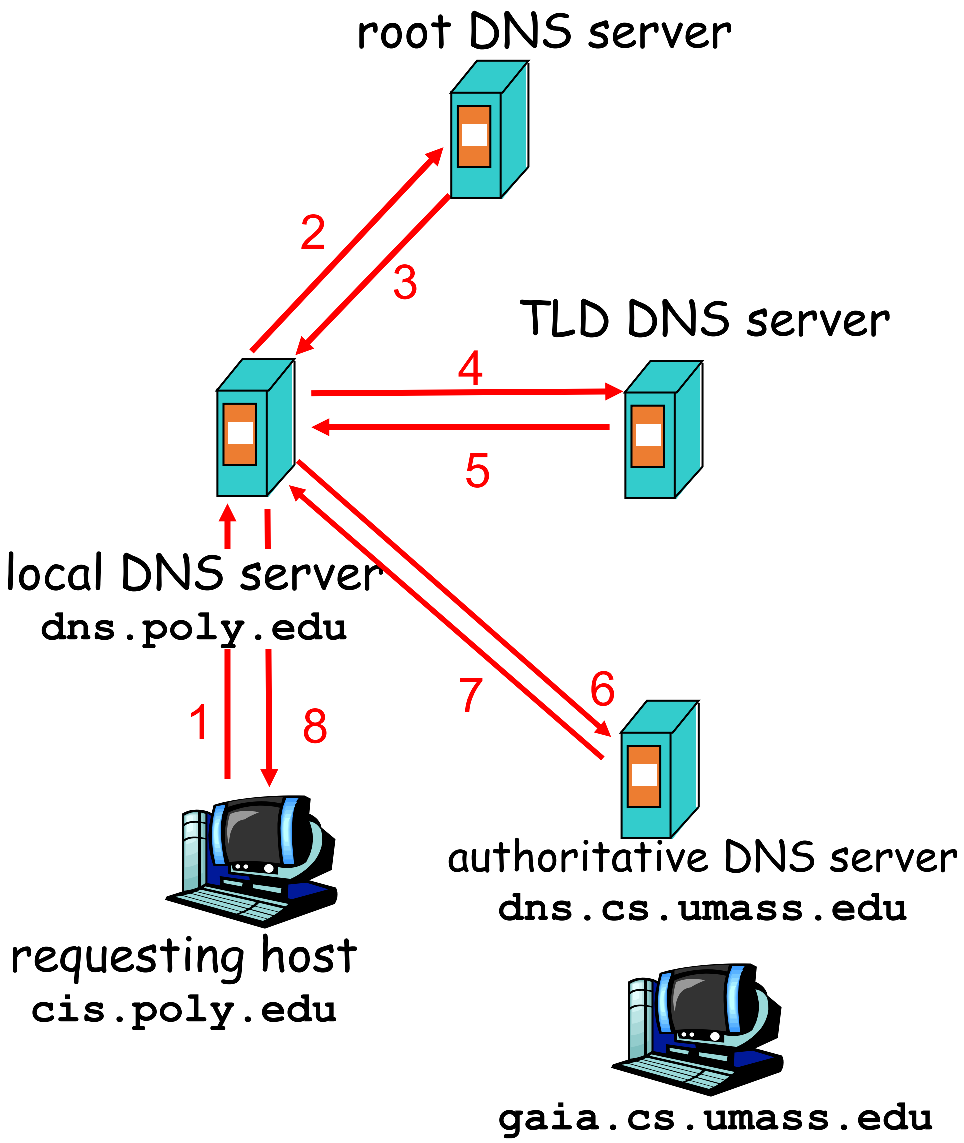
- demuxed to DNS
- DNS replies to client with IP address of www.google.com

- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router

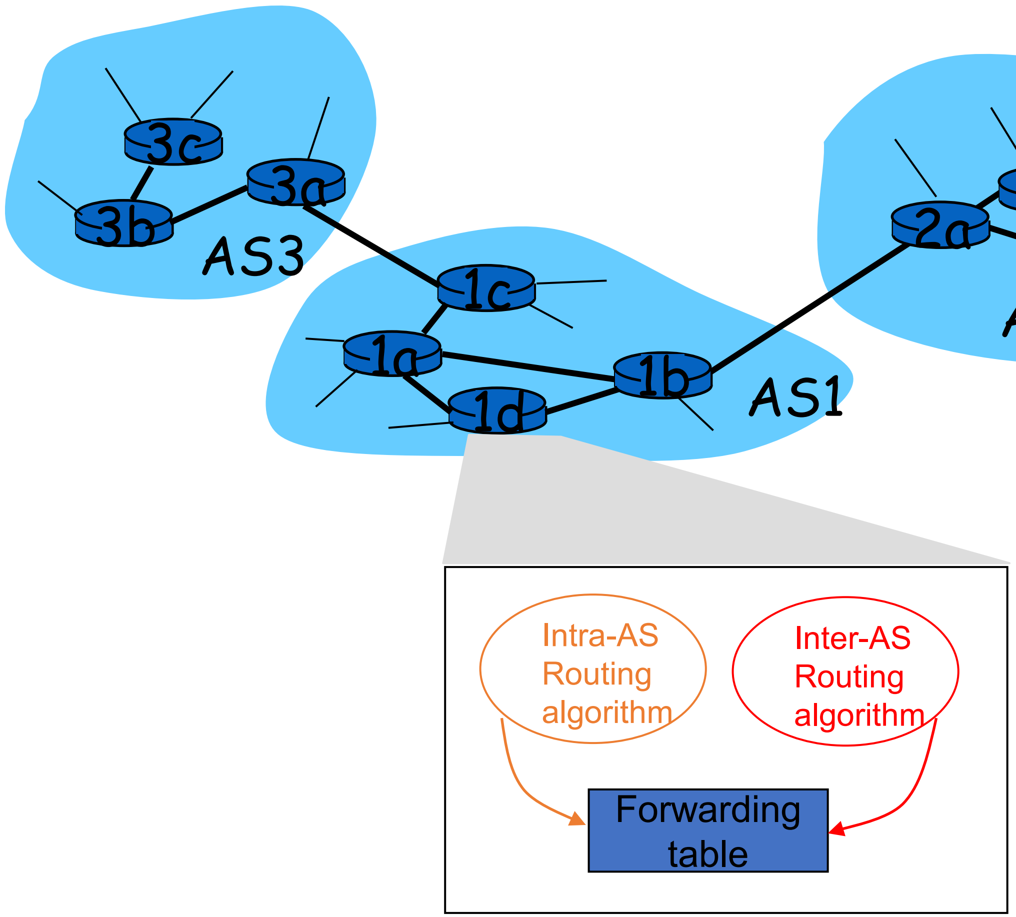
- IP datagram forwarded from campus network into Comcast network, routed (tables created by RIP, OSPF, IS-IS and/or BGP routing protocols) to DNS server

DNS Infrastructure

- Host at cis.poly.edu wants IP address for gaia.cs.umass.edu
- Infrastructure:
  - Client resolver
  - Local DNS server
  - Authoritative DNS Server
  - Root DNS Server
  - Top-Level Domain DNS Server

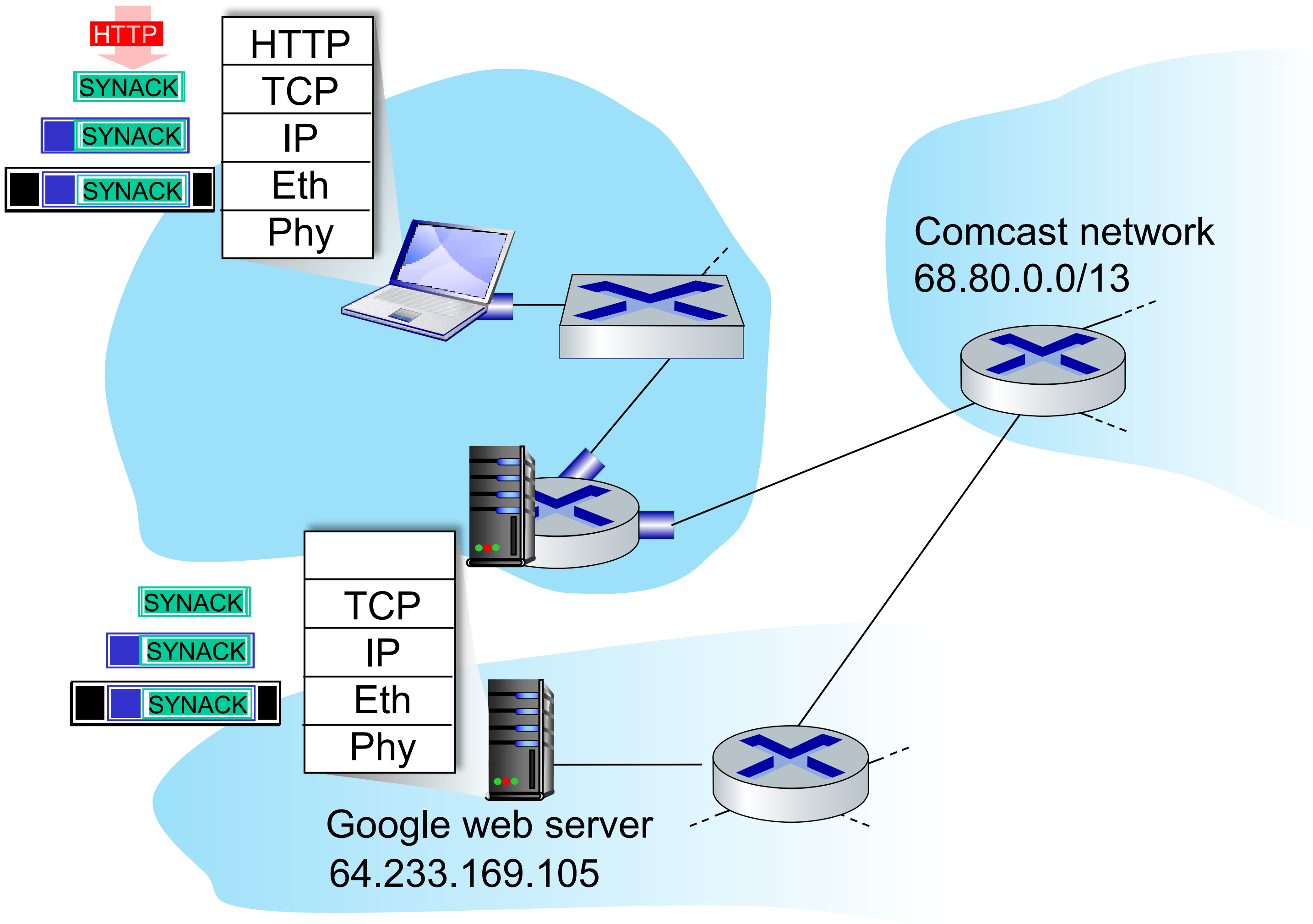


Routing and Forwarding



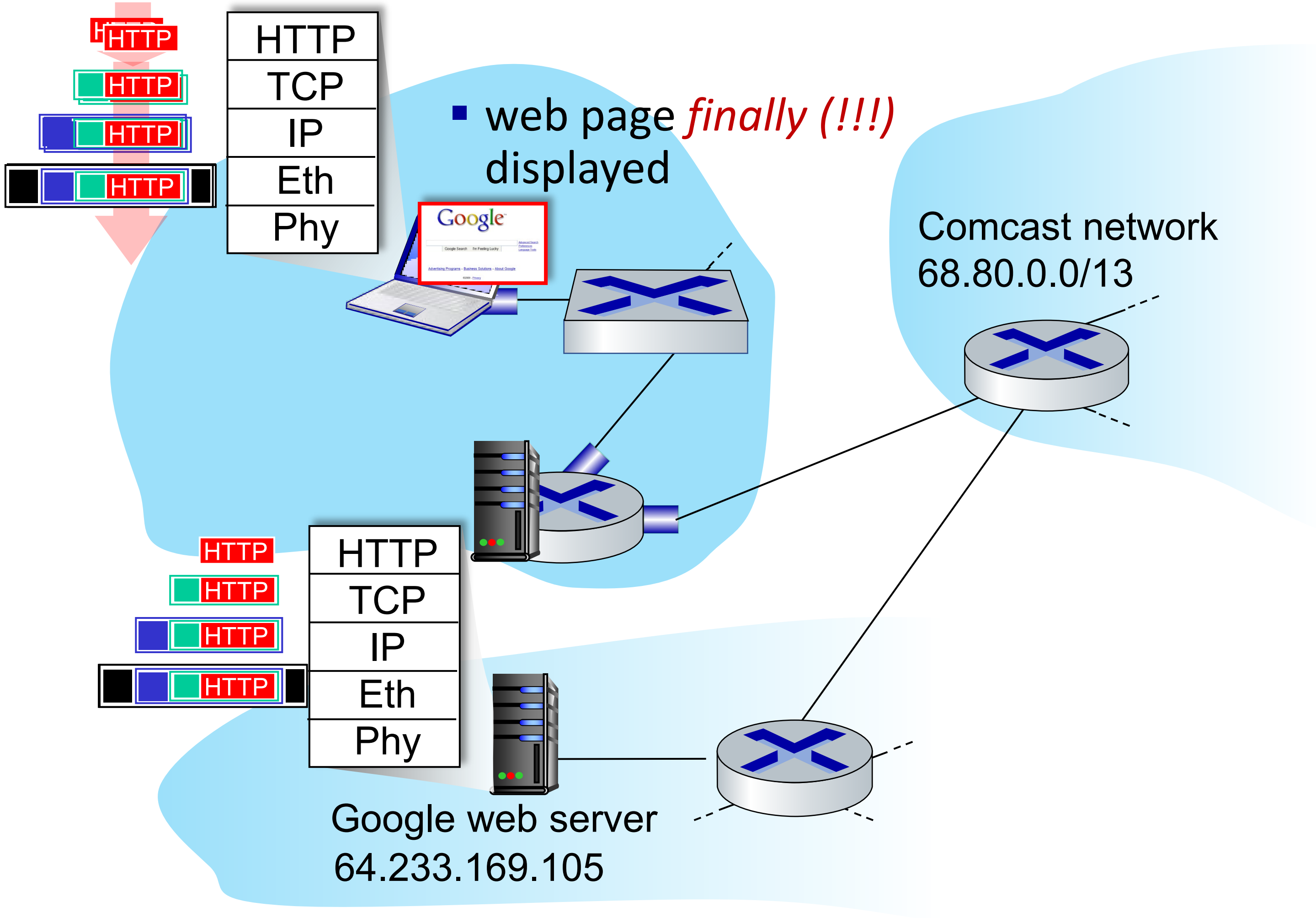
- Forwarding table is configured by both intra- and inter-AS routing algorithm
  - Intra-AS sets entries for internal dests
  - Inter-AS & Intra-As sets entries for external dests

A day in the life…TCP connection carrying HTTP



- to send HTTP request, client first opens TCP socket to web server
- TCP SYN segment (step 1 in TCP 3-way handshake) inter-domain routed to web server
- web server responds with TCP SYNACK (step 2 in TCP 3-way handshake)
- TCP connection established!

A day in the life… HTTP request/reply

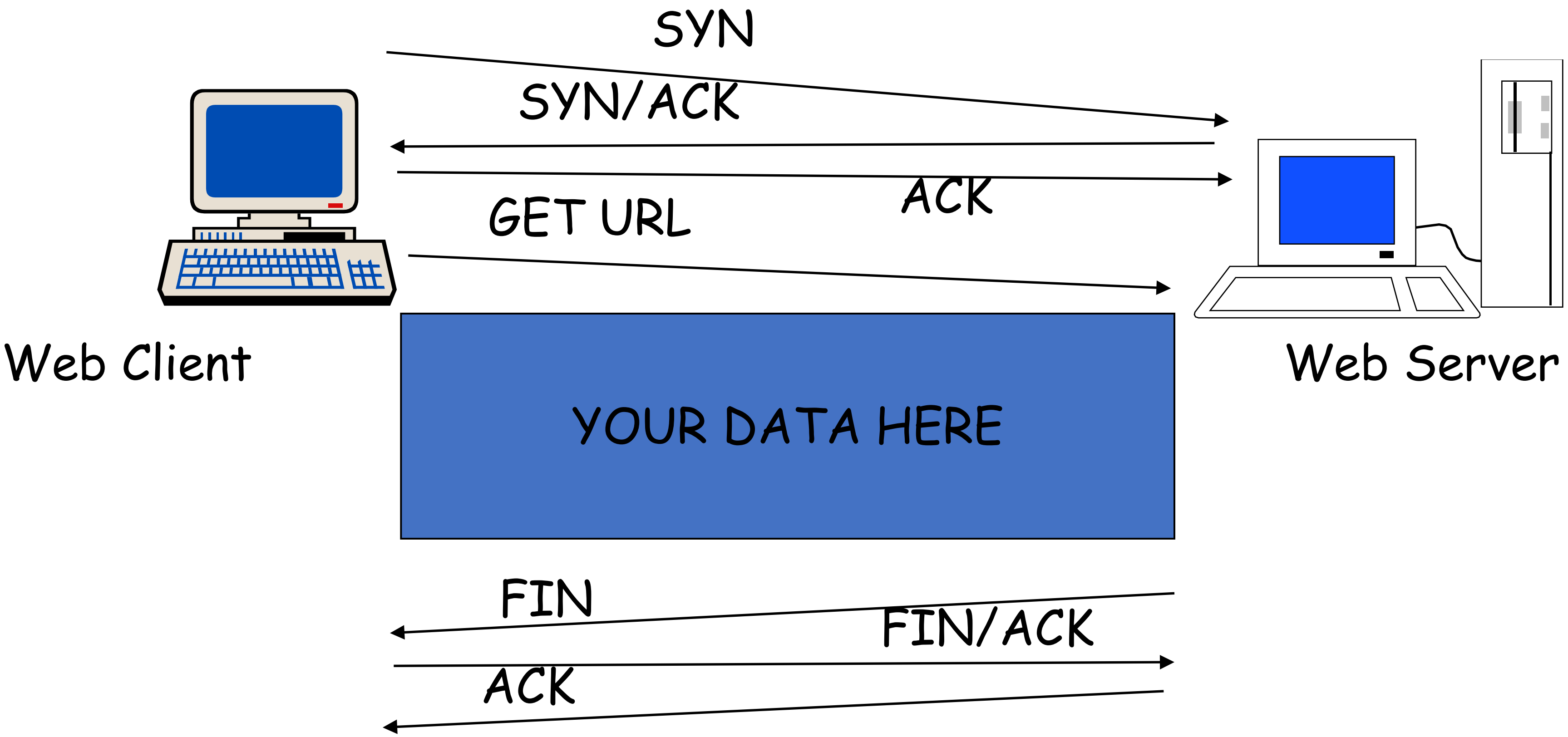


- web page *finally (!!!)* displayed

- 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

Network View: HTTP and TCP

- TCP is a connection-oriented protocol



How do we avoid sending too much for the receiver and network to handle?

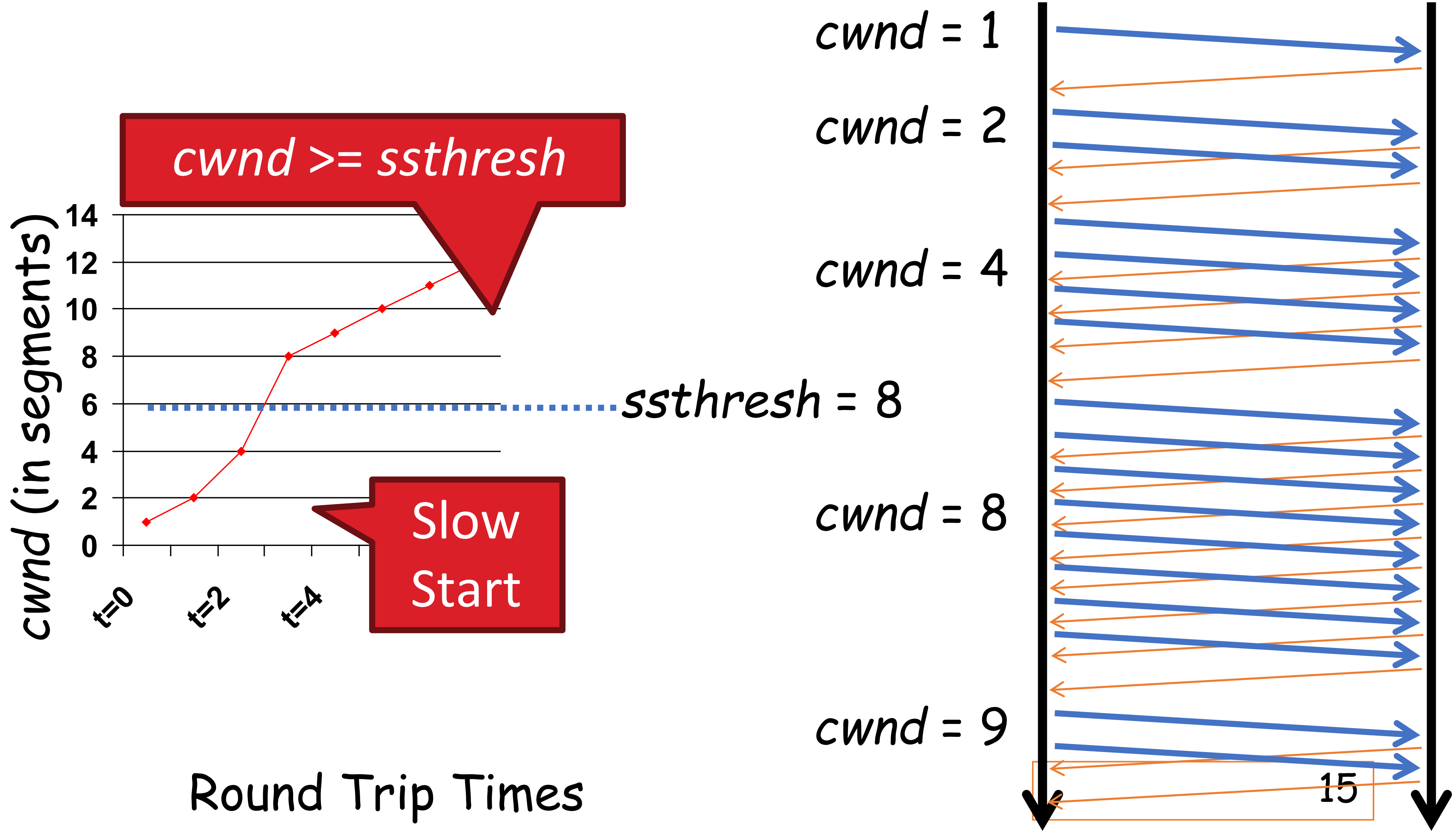
TCP Tahoe: Summary

- Basic ideas
  - Gently probe network for spare capacity
  - Drastically reduce rate on congestion
  - Windowing: self-clocking
  - Other functions: round trip time estimation, error recovery

```
for every ACK {
        if (W < ssthresh) then W++        (SS)
        else          W += 1/W            (CA)

    }
    for every loss {
            ssthresh = W/2
            W  = 1
    }
```

Congestion Avoidance Example



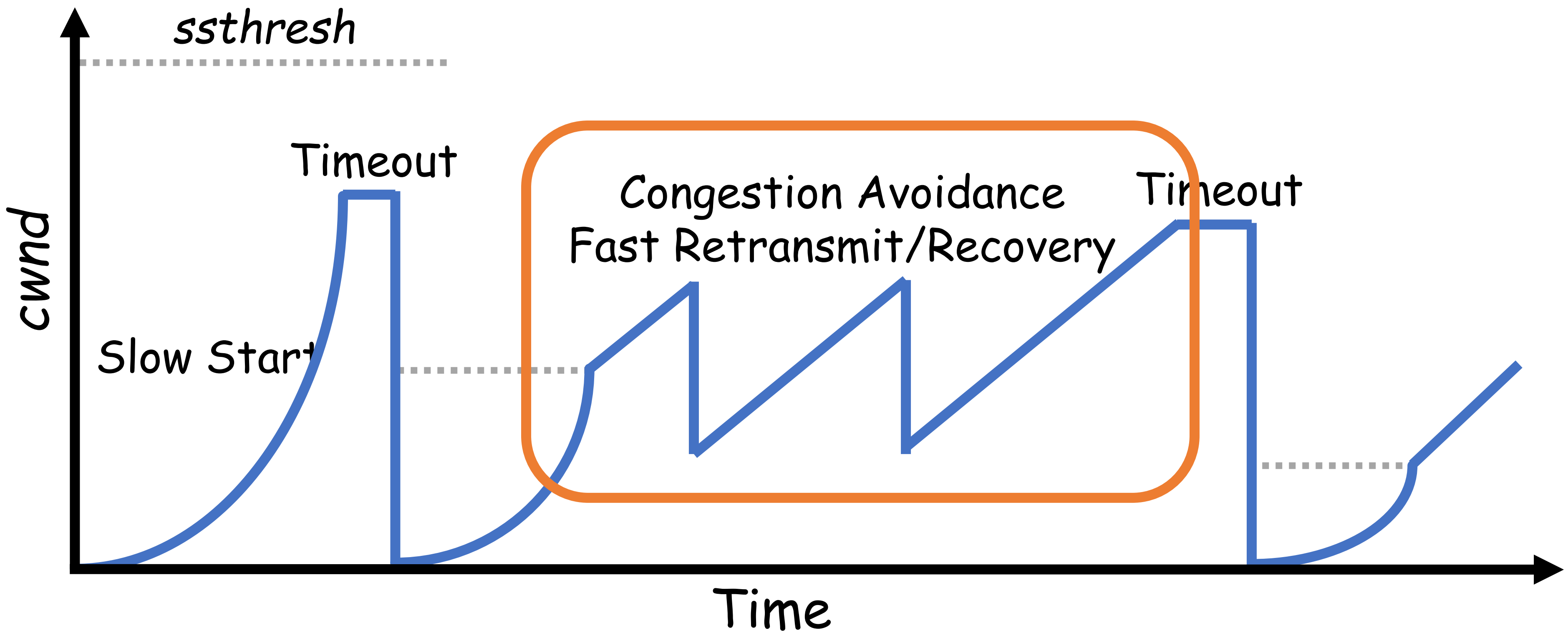
cwnd >= ssthresh

Slow Start

TCP Reno: Summary

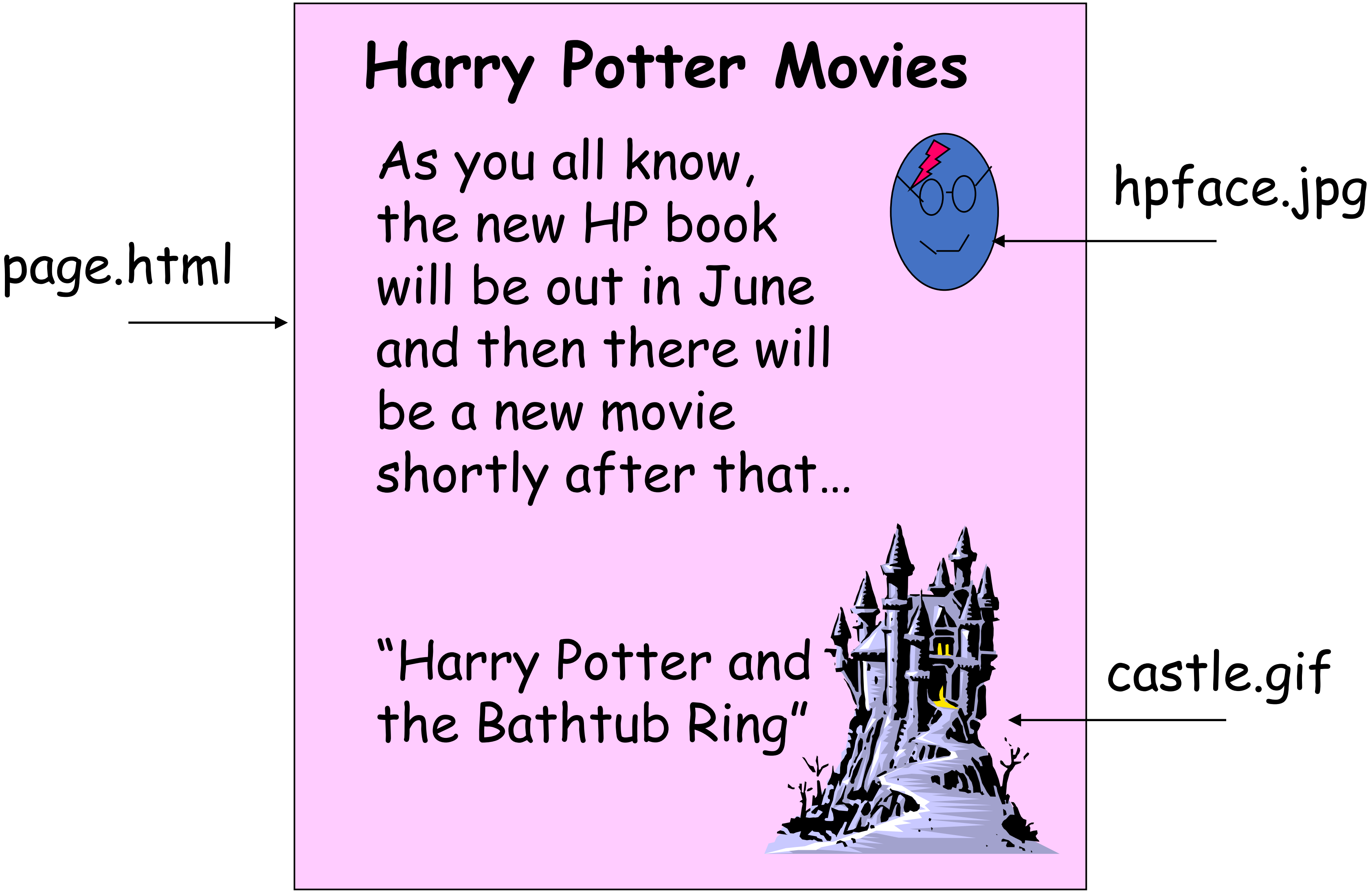
- Fast Recovery along with Fast Retransmit used to avoid slow start
- On 3 duplicate ACKs
  - Fast retransmit and fast recovery
- On timeout
  - Retransmit last segment and slow start

Fast Retransmit and Fast Recovery



- At steady state, *cwnd* oscillates around the optimal window size
- TCP always forces packet drops

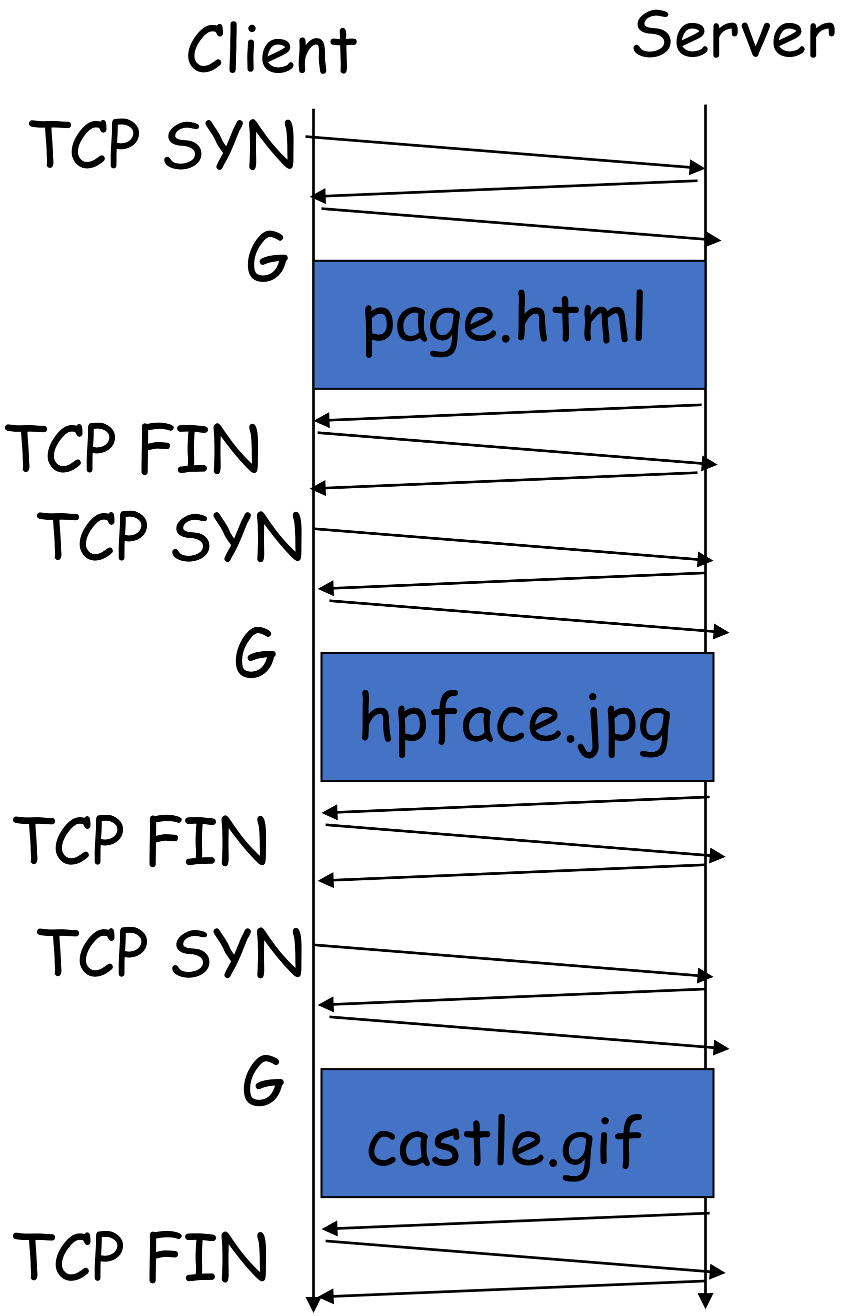
Example Web Page



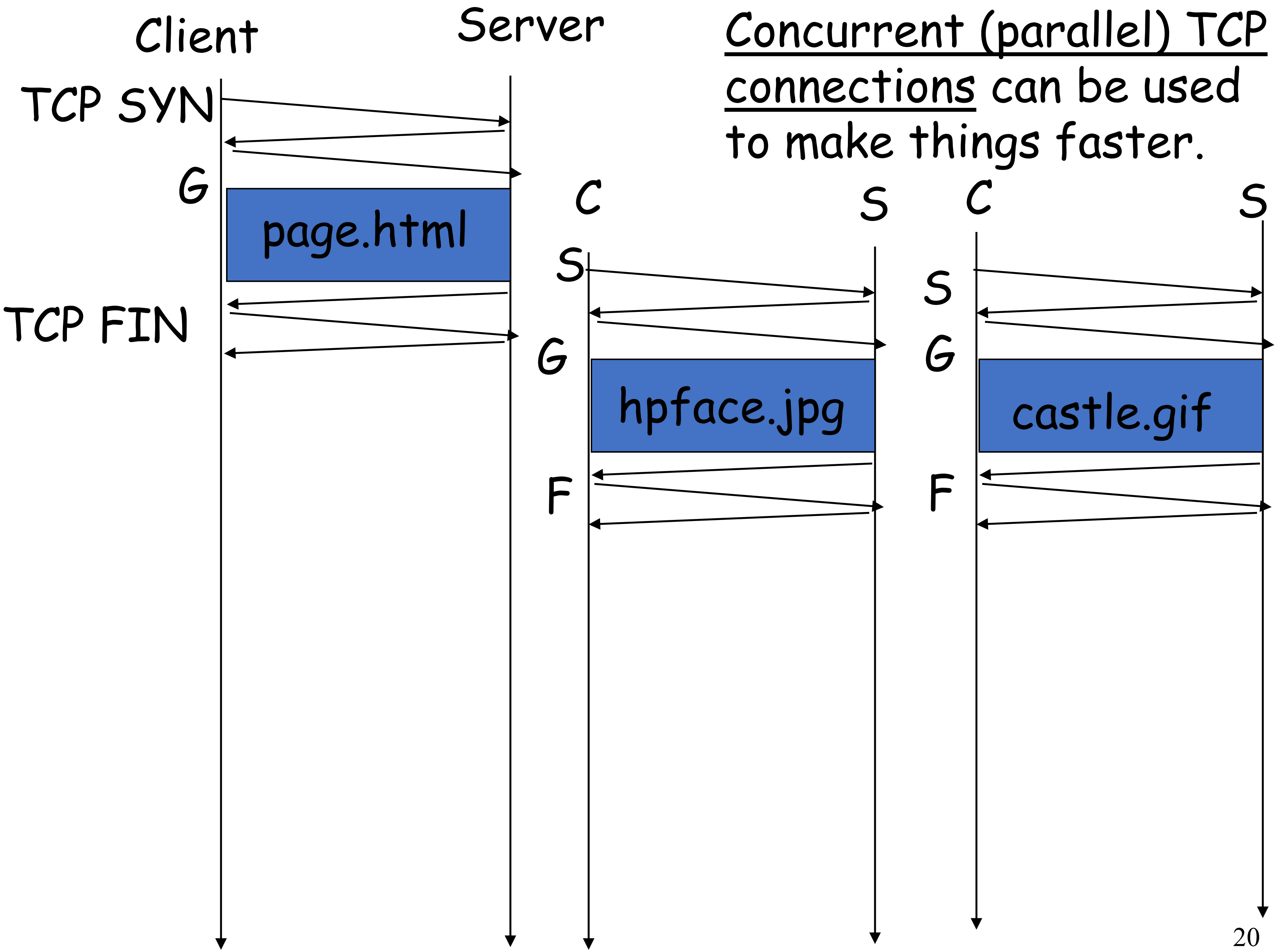
Harry Potter Movies

As you all know,
the new HP book
will be out in June
and then there will
be a new movie
shortly after that…

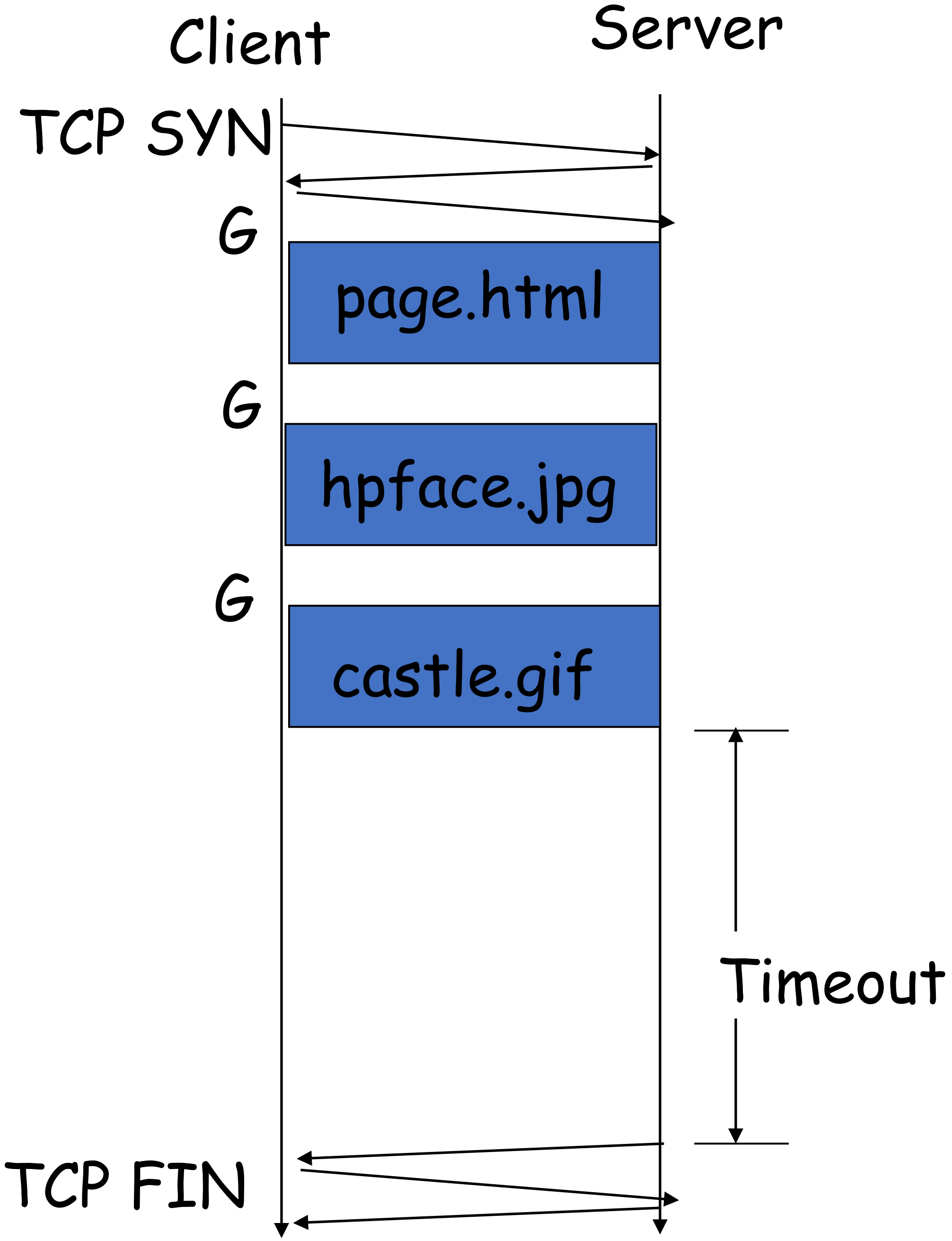
"Harry Potter and
the Bathtub Ring"



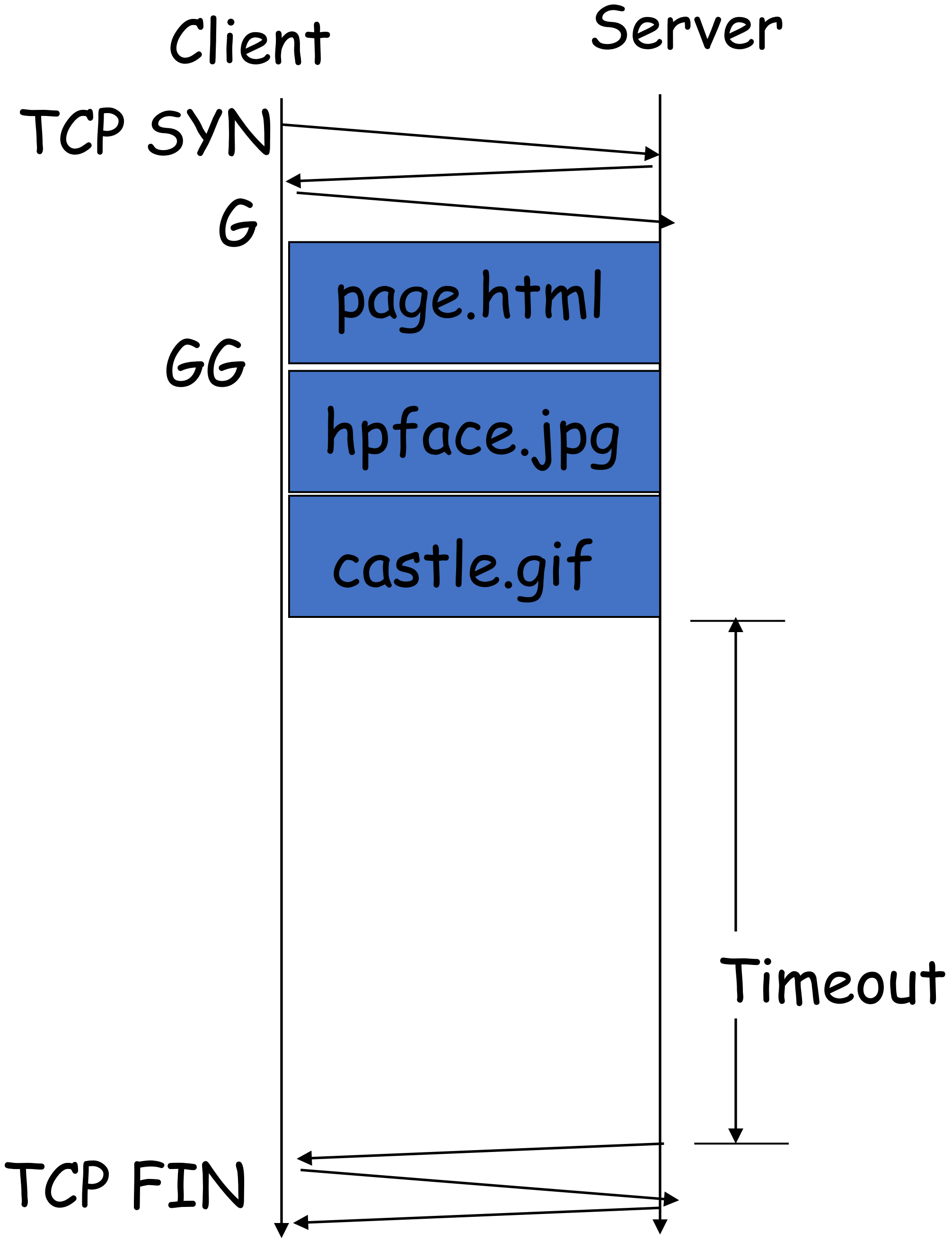
The “classic” approach in HTTP/1.0 is to use one HTTP request per TCP connection, serially.



<u>Concurrent (parallel) TCP connections</u> can be used to make things faster.

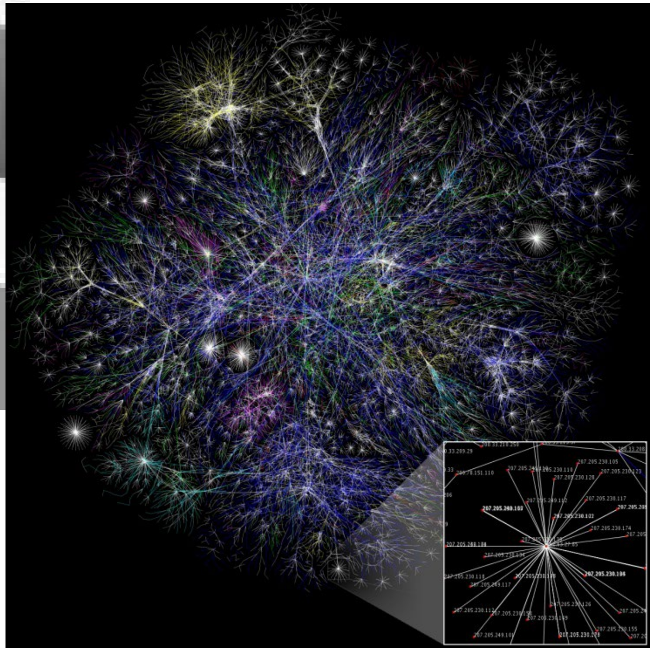


The "persistent HTTP" approach can re-use the same TCP connection for Multiple HTTP transfers, one after another, serially. Amortizes TCP overhead, but maintains TCP state longer at server.



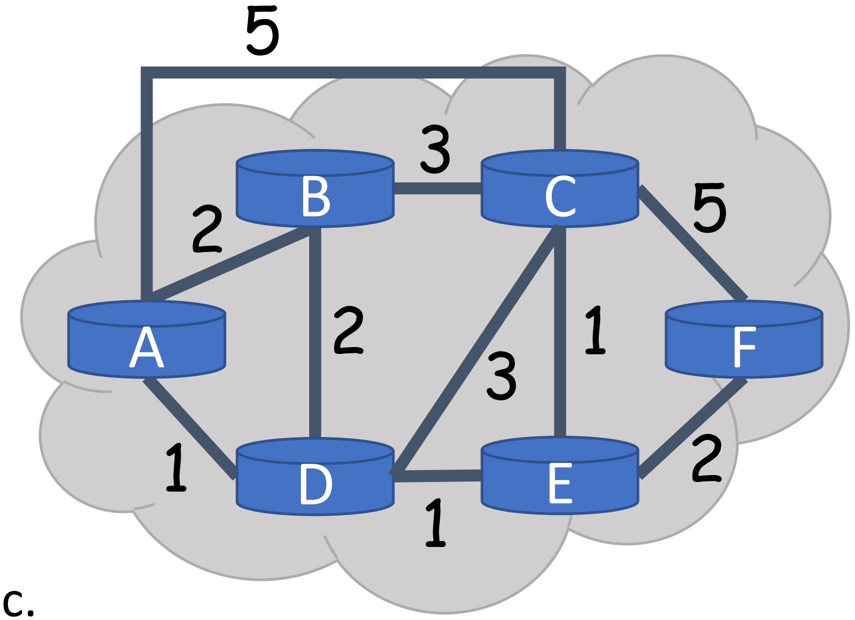
The "pipelining" feature in HTTP/1.1 allows requests to be issued asynchronously on a persistent connection. Requests must be processed in proper order. Can do clever packaging.

How do we find a path?



Routing on a Graph

- Goal: determine a "good" path through the network from source to destination
- What is a good path?
  - Usually means the shortest path
  - Load balanced
  - Lowest $$$ cost
- Network modeled as a graph
  - Routers → nodes
  - Link → edges
    - Edge cost: delay, congestion level, etc.

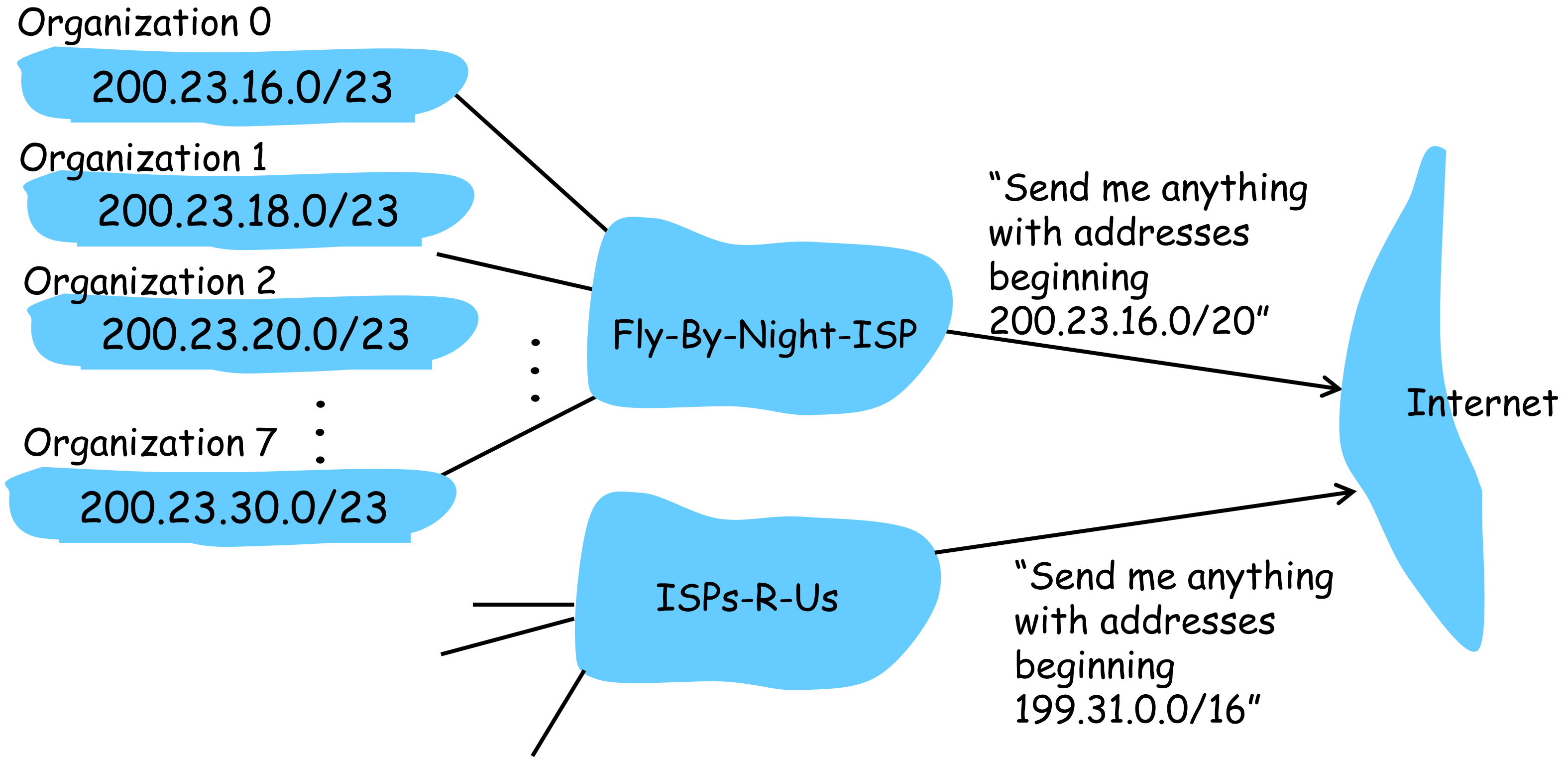


Intra-domain Routing Protocols

- Distance vector
  - Routing Information Protocol (RIP), based on Bellman-Ford
  - Routers periodically exchange reachability info with neighbors
- Link state
  - Open Shortest Path First (OSPF), based on Dijkstra
  - Each network periodically floods neighbor information to all routers
  - Routers locally compute routes

Hierarchical addressing: route aggregation

ISP has an address block; it can further divide this block into sub blocks and assign them to subscriber organizations.



"Send me anything with addresses beginning 200.23.16.0/20"

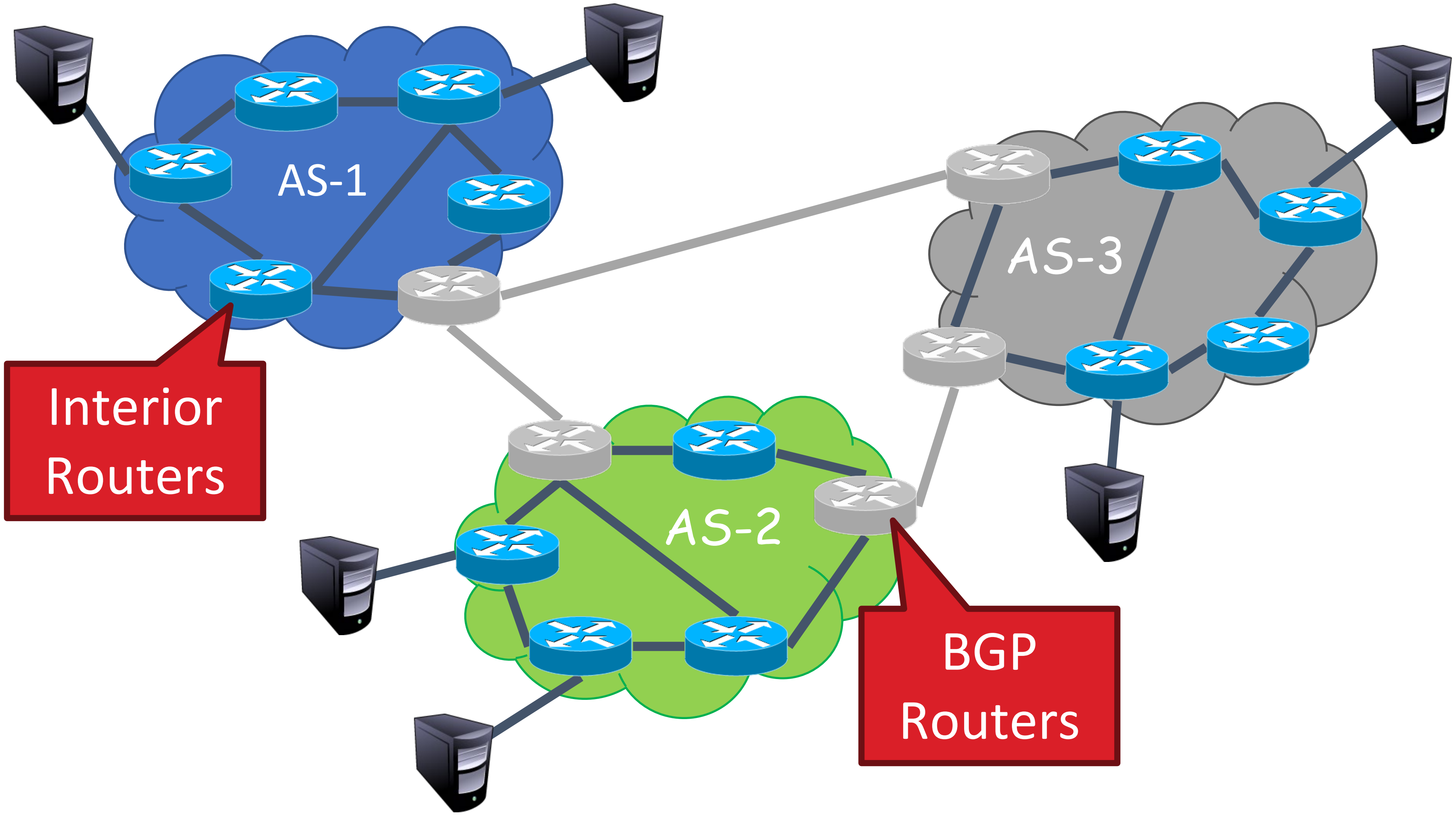
"Send me anything with addresses beginning 199.31.0.0/16"

Example CIDR Routing Table

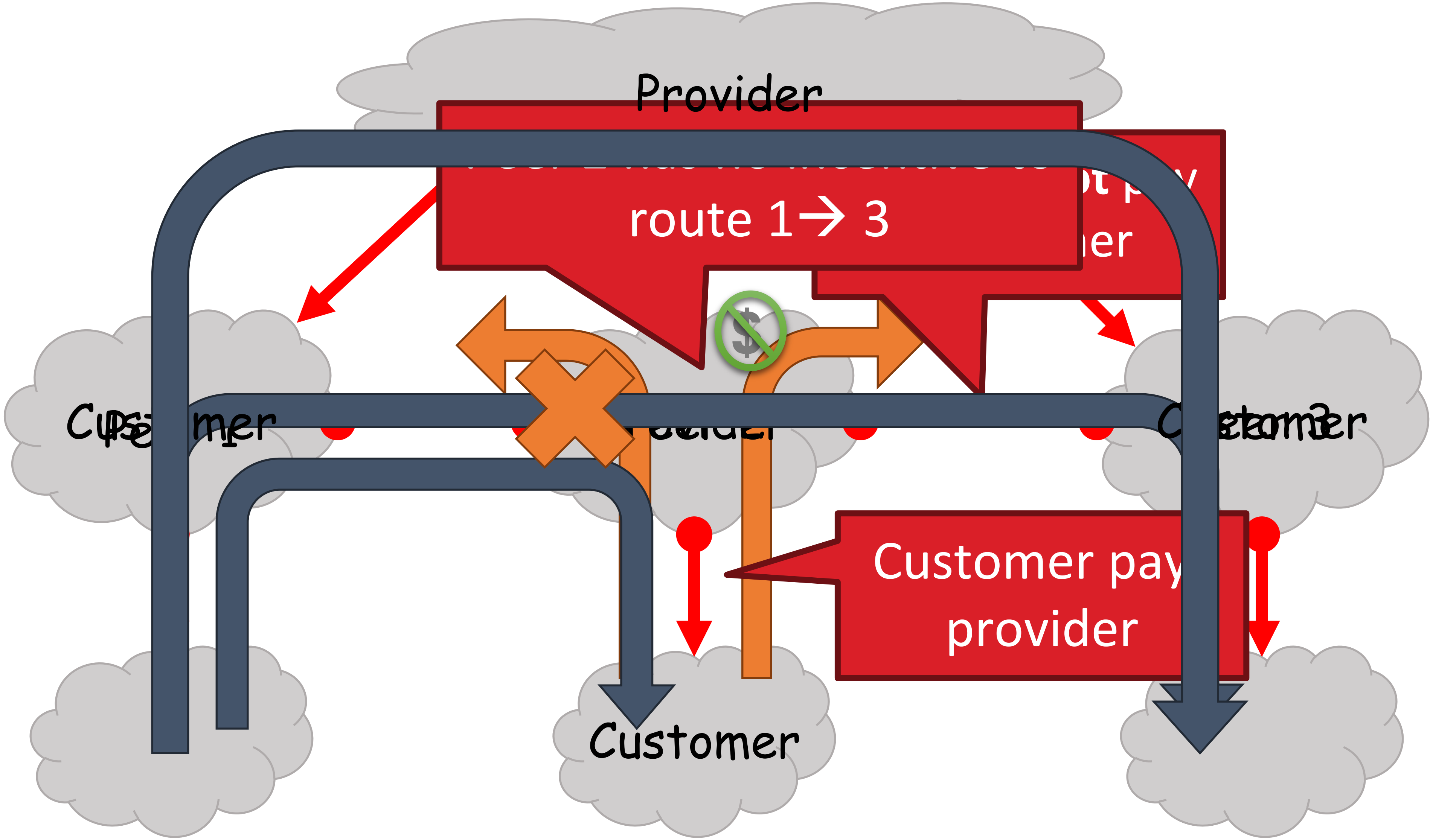
Address	Netmask	Third Byte	Byte Range
207.46.0.0	19	000xxxxx	0 – 31
207.46.32.0	19	001xxxxx	32 – 63
207.46.64.0	19	010xxxxx	64 – 95
207.46.128.0	18	10xxxxxx	128 – 191
207.46.192.0	18	11xxxxxx	192 – 255

Hole in the Routing Table: No coverage for 96 - 127
207.46.96.0/19

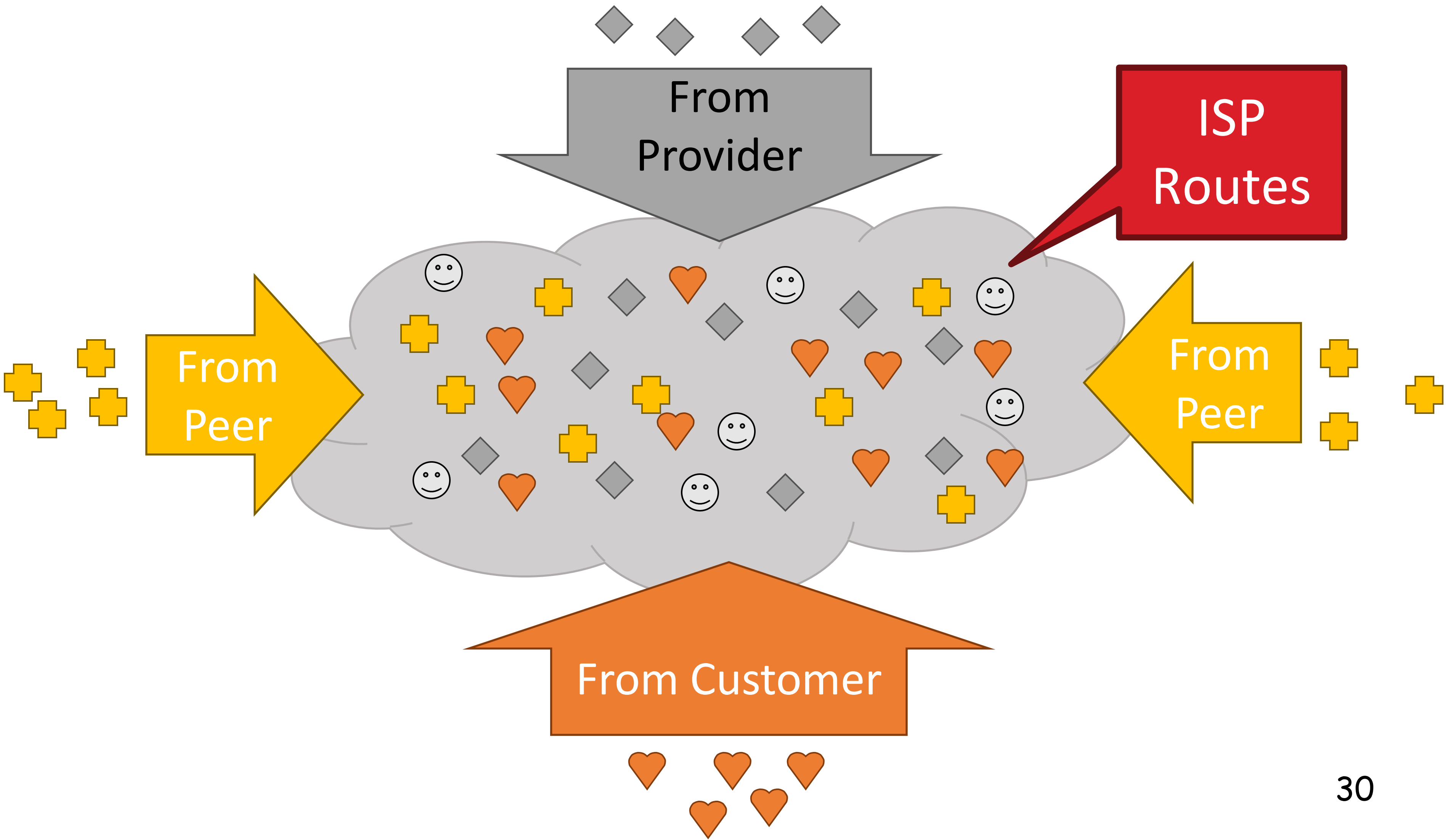
Network of networks: BGP and ASes



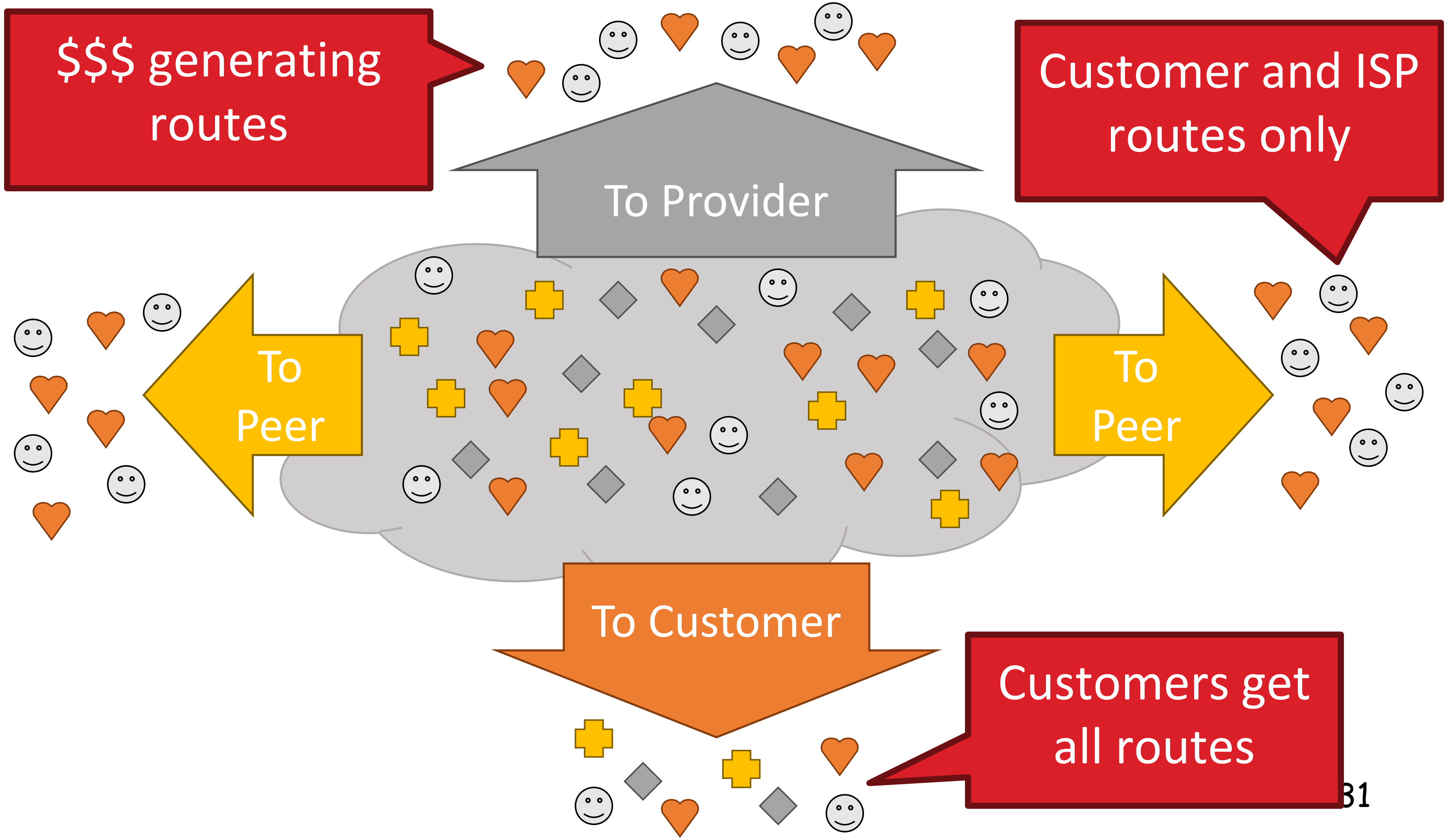
BGP Relationships



Importing Routes



Exporting Routes



$$$ generating routes

Customer and ISP routes only

Customers get all routes

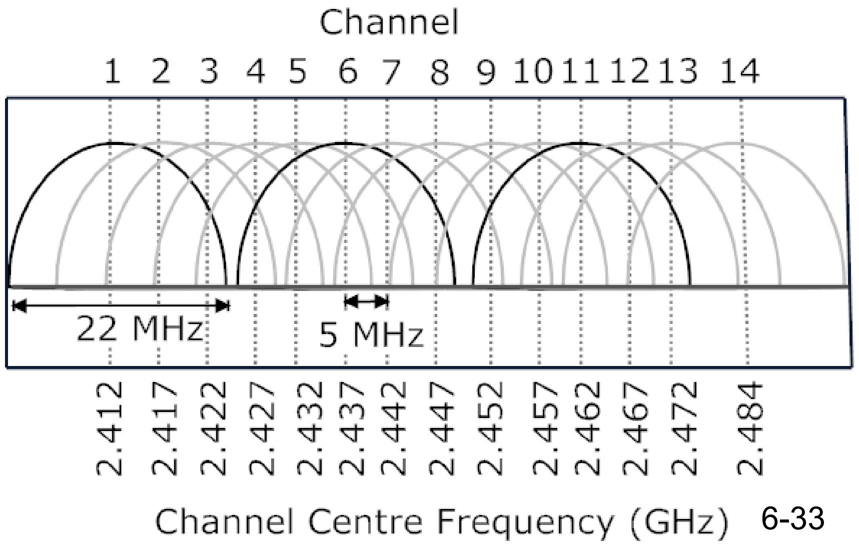
Also cover many other topics …

- For example …
  - Wireless (Ch 7)
  - Multimedia networking (Ch 8)
  - Network security (Ch 9)

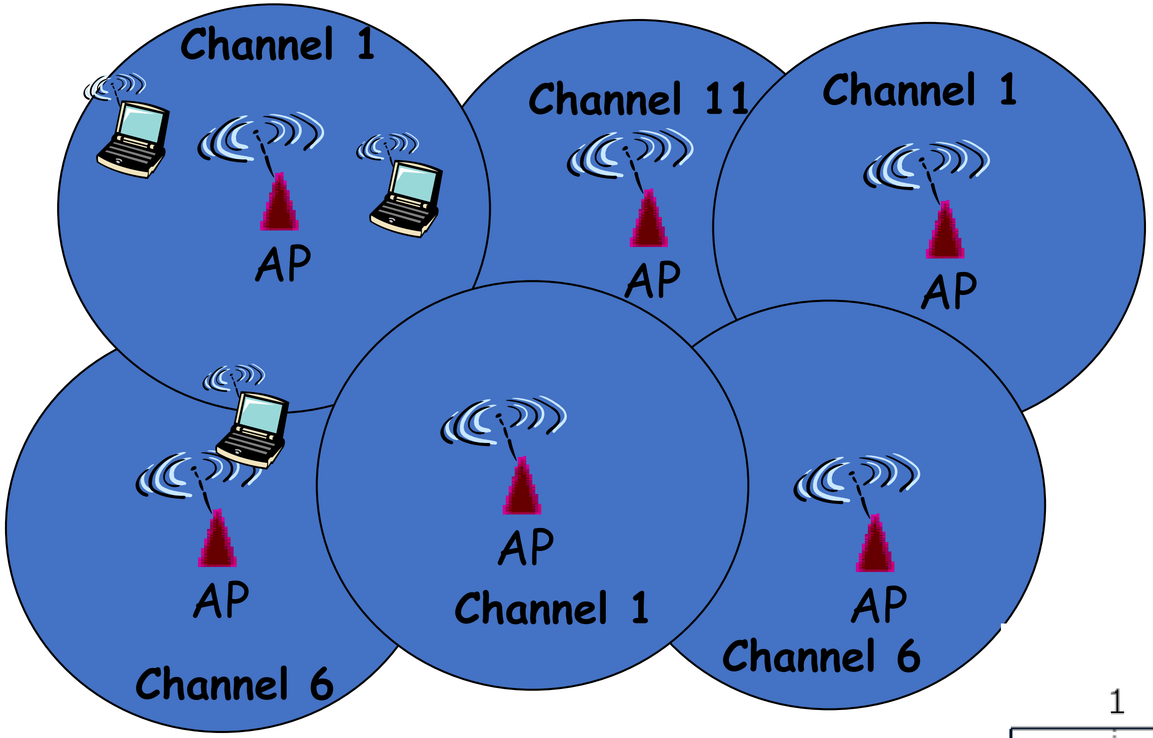
Wireless Cells

- 802.11b has 11 channels
- Channels 1, 6, and 11 are non-overlapping

- Admin chooses frequency for AP

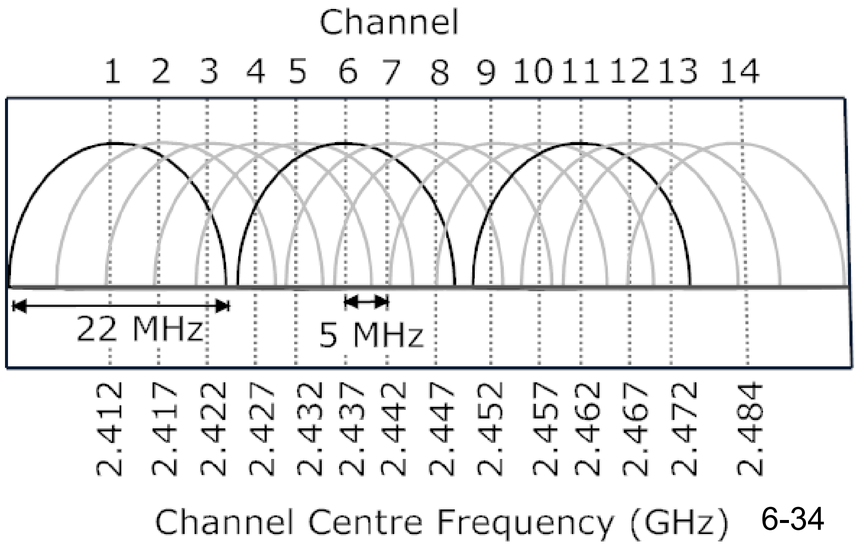


Wireless Cells

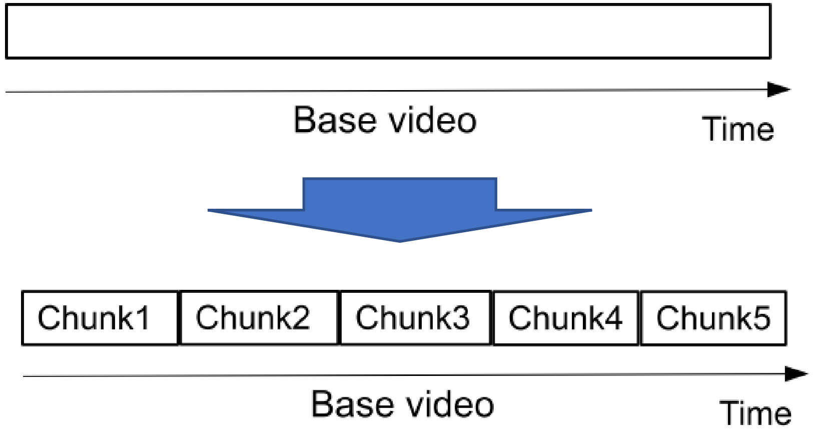


- 802.11b has 11 channels
- Channels 1, 6, and 11 are non-overlapping
- Each AP coverage area is called a “cell”
- Wireless nodes can roam between cells

- Admin chooses frequency for AP
- Interference possible: channel can be same as that chosen by neighboring AP!

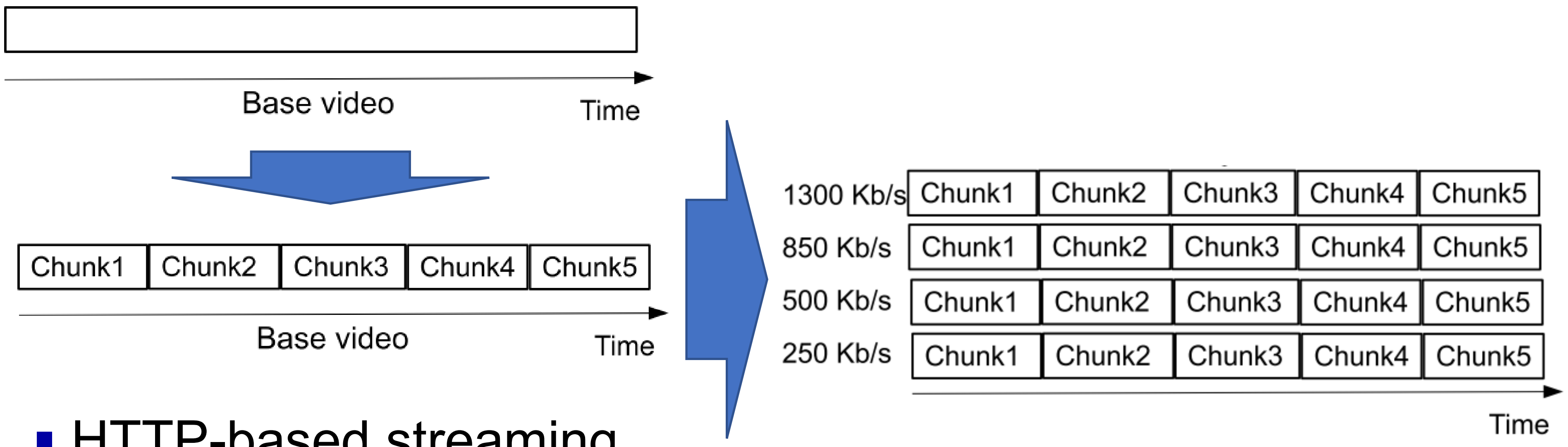


HTTP-based Adaptive Streaming (HAS)



- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD

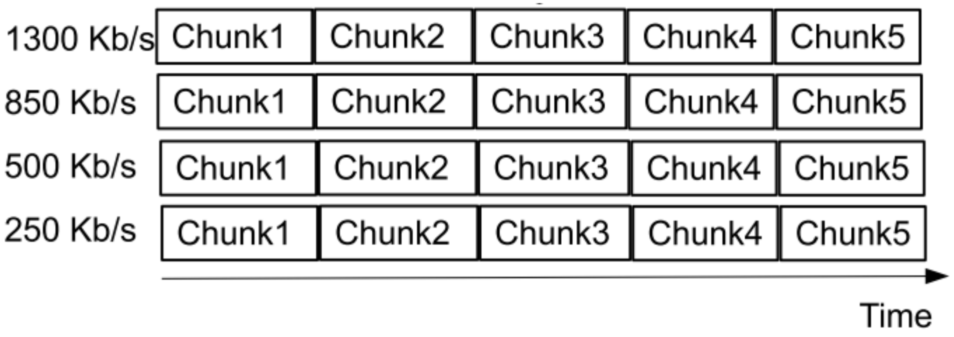
HTTP-based Adaptive Streaming (HAS)



1300 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
850 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
500 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
250 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5

- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each chunk (defined in manifest file)
  -

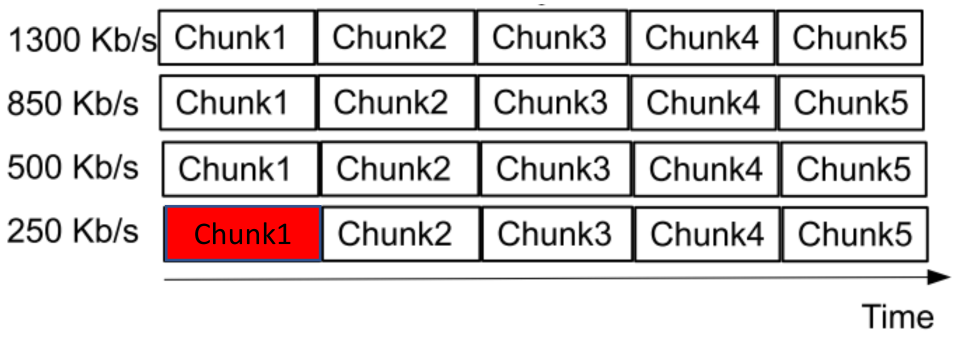
HTTP-based Adaptive Streaming (HAS)



1300 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
850 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
500 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
250 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5

- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each chunk (defined in manifest file)

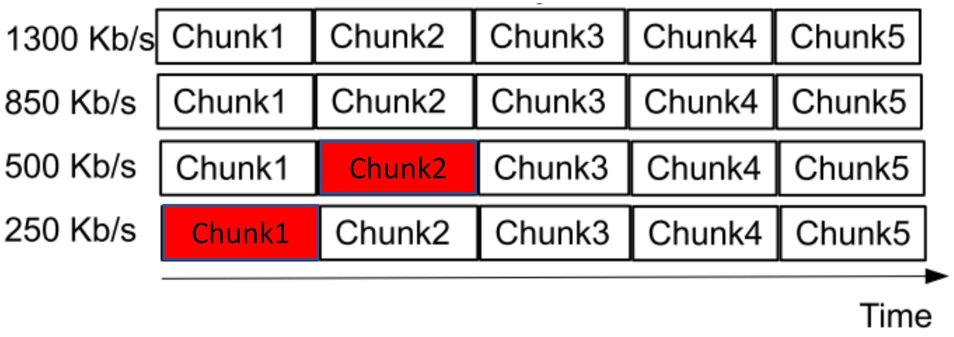
HTTP-based Adaptive Streaming (HAS)



1300 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
850 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
500 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
250 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5

- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each chunk (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

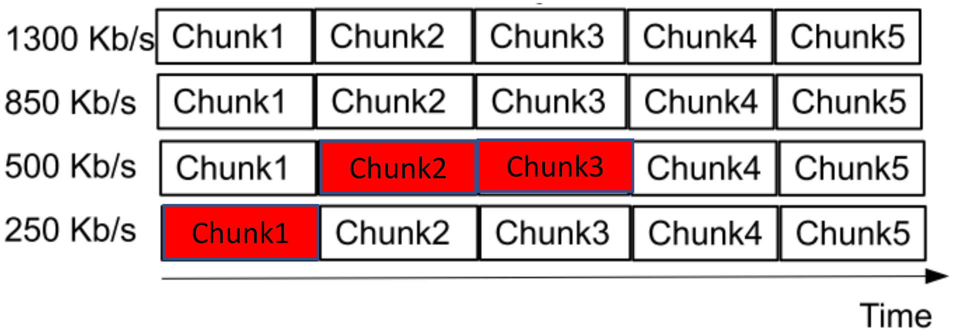
HTTP-based Adaptive Streaming (HAS)



1300 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
850 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
500 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
250 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5

- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each chunk (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

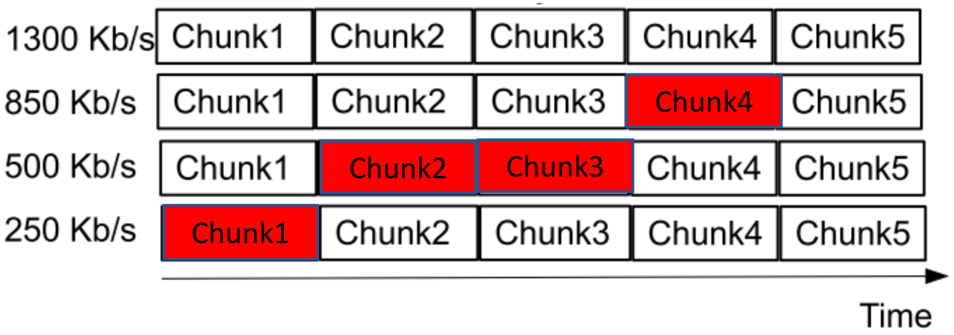
HTTP-based Adaptive Streaming (HAS)



1300 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
850 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
500 Kb/s	Chunk1	**Chunk2**	**Chunk3**	Chunk4	Chunk5
250 Kb/s	**Chunk1**	Chunk2	Chunk3	Chunk4	Chunk5

- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each chunk (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

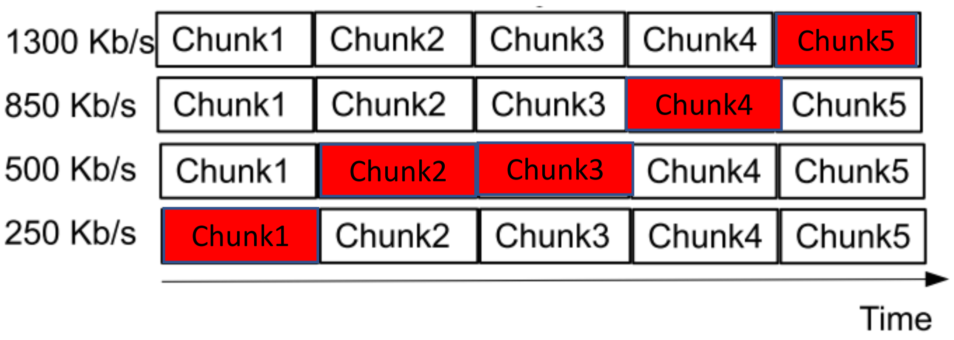
HTTP-based Adaptive Streaming (HAS)



1300 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	Chunk5
850 Kb/s	Chunk1	Chunk2	Chunk3	**Chunk4**	Chunk5
500 Kb/s	Chunk1	**Chunk2**	**Chunk3**	Chunk4	Chunk5
250 Kb/s	**Chunk1**	Chunk2	Chunk3	Chunk4	Chunk5

- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each chunk (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HTTP-based Adaptive Streaming (HAS)

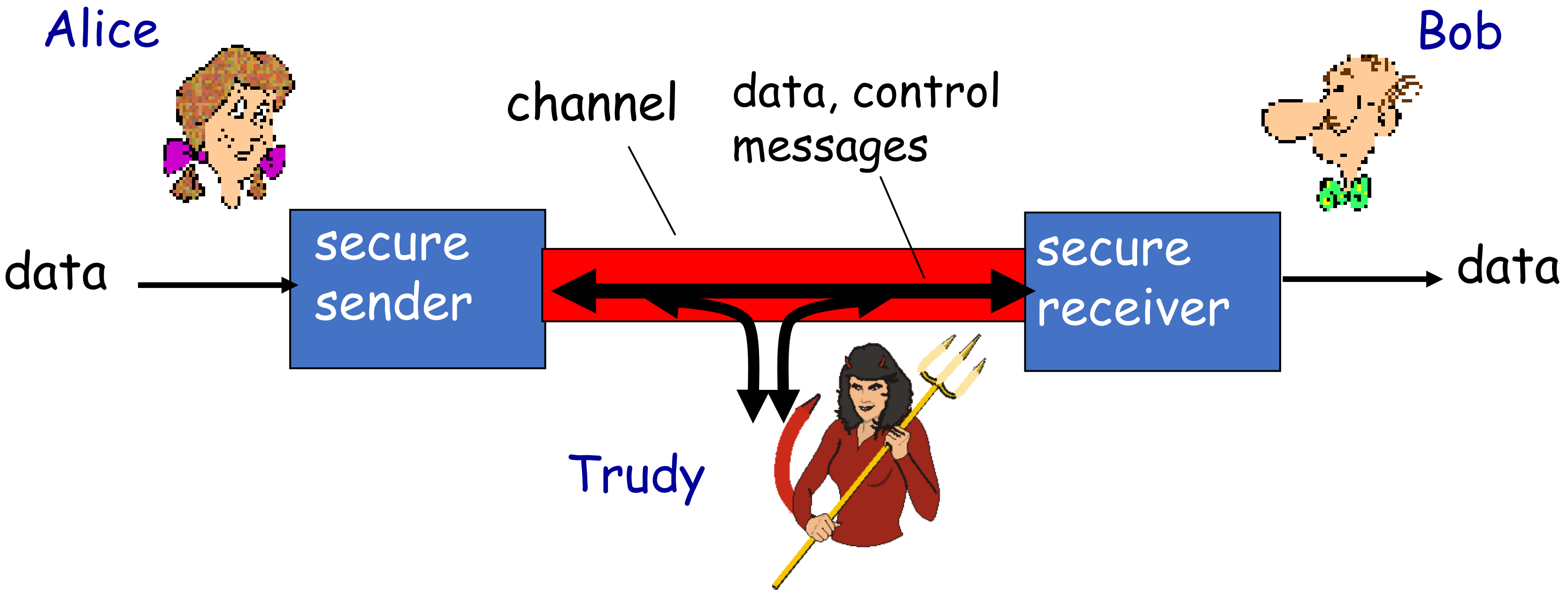


1300 Kb/s	Chunk1	Chunk2	Chunk3	Chunk4	**Chunk5**
850 Kb/s	Chunk1	Chunk2	Chunk3	**Chunk4**	Chunk5
500 Kb/s	Chunk1	**Chunk2**	**Chunk3**	Chunk4	Chunk5
250 Kb/s	**Chunk1**	Chunk2	Chunk3	Chunk4	Chunk5

- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each chunk (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

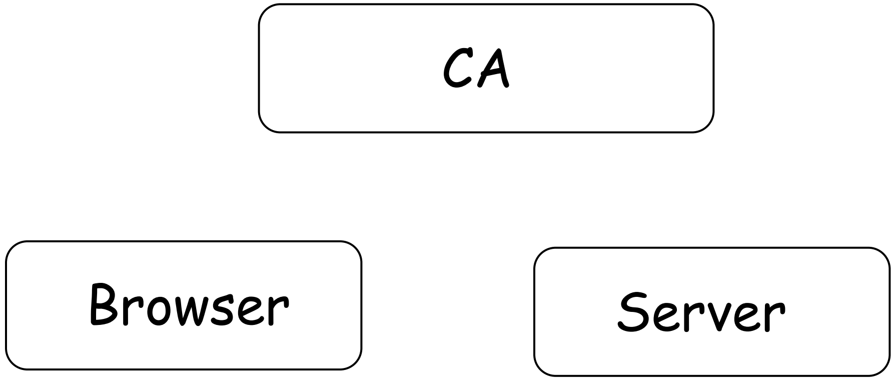
Friends and enemies: Alice, Bob, Trudy

- well-known in network security world
- Bob, Alice (lovers!) want to communicate "securely"
- Trudy (intruder) may intercept, delete, add messages



HTTPS (and TLS/SSL Cipher Suite)

- cipher suite
  - public-key algorithm
  - symmetric encryption algorithm
  - MAC algorithm
- TLS/SSL supports several cipher suites
- negotiation: client, server agree on cipher suite
  - client offers choice
  - server picks one



Exam (pre-COVID)

- Read all instructions carefully
- Please explain how you derived your answers. Your final answers should be clearly stated.
- Write answers legibly; no marks will be given for answers that cannot be read easily.
- Where a discourse or discussion is called for, be concise and precise.
- No assistance: closed book, closed notes, and no electronics ...
- https://www.ida.liu.se/~TDTS06/exam/examples2014.html

Exam (Remote Home)

- https://www.ida.liu.se/~TDTS04/exam/exam_questions_tdts11_04_may2020.pdf
- Edu. code Course name Date Time City Location / Enrolment period
- TDTS04/TEN1 Computer Networks and Distributed Systems 2022-03-22 8 - 12 Linköping 2022-02-20 - 2022-03-12
- Distanstentamen/ Distance exam
- TDTS04/TEN1 Computer Networks and Distributed Systems 2022-06-09 14 - 18 Linköping 2022-05-10 - 2022-05-30
- TDTS04/TEN1 Computer Networks and Distributed Systems 2022-08-26 8 - 12 Linköping 2022-07-27 - 2022-08-16