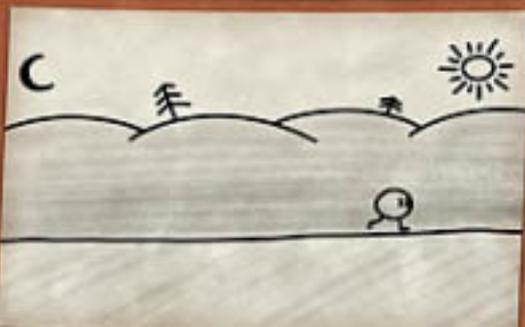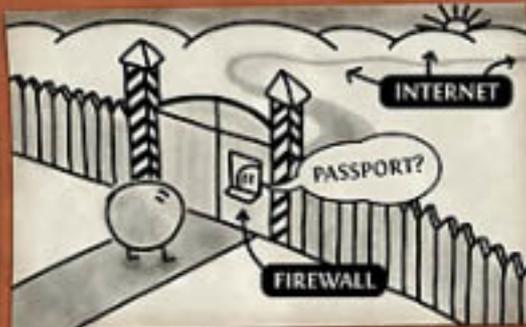# CE 443: Computer Networks

Acknowledgments: Lecture slides are from Computer networks course thought by Jennifer Rexford at Princeton University. When slides are obtained from other sources, a a reference will be noted on the bottom of that slide. A full list of references is provided on the last slide.

Goals for Today's Class

- Overview
  - Goals of the course
  - Structure of the course
  - Course grading

- Key concepts in data networking
  - Protocols
  - Layering
  - Resource allocation
  - Naming

What You Learn in This Course

- Knowledge: how the Internet works
  - IP protocol suite
  - Internet architecture
  - Applications (Web, e-mail, P2P, VoIP, …)

- Insight: key concepts in networking
  - Protocols
  - Layering
  - Resource allocation
  - Naming

- Skill: network programming
  - Socket programming
  - Designing and implementing protocols

Structure of the Course (1st Half)

- Start at the top
  - Sockets: how applications view the Internet
  - Protocols: essential elements of a protocol

- Then study the "narrow waist" of IP
  - IP best-effort packet-delivery service
  - IP addressing and packet forwarding

- And how to build on top of the narrow waist
  - Transport protocols (TCP, UDP)
  - Domain Name System (DNS)
  - Glue (ARP, DHCP, ICMP)
  - End-system security and privacy (NAT, firewalls)

- Looking underneath IP
  - Link technologies (Ethernet, wireless, …)

Structure of the Course (2nd Half)

- And how to get the traffic from here to there
  - Internet routing architecture (the "inter" in Internet)
  - Intradomain and interdomain routing protocols

- Building applications
  - Web and content-distribution networks
  - E-mail
  - Peer-to-peer file sharing
  - Multimedia streaming and voice-over-IP

- Other approaching to building networks
  - Circuit switching (e.g., ATM, MPLS, …)
  - More on wireless networks, multicast, …

Learning the Material: Mailing List

- Mailing list
  - E-mail ce443@lists.ce.sharif.edu
  - Sign up instruction on course webpage

- Read often
  - Good place to ask questions
  - But please do not post your solutions

- Reply, too
  - Good place to *answer* questions

Learning the Material: Books

- Required textbook
  - *Computer Networks: A Systems Approach (4th edition)*, by Peterson and Davie

- Optional textbooks
  - Networking text books
    - *Computer Networking: A Top-Down Approach Featuring the Internet (3rd edition),* by Kurose and Ross
    - *Computer Networks (4th edition)*, by Tanenbaum
  - Network programming references
    - *TCP/IP Illustrated, Volume 1: The Protocols*, by Stevens
    - *Unix Network Programming, Volume 1: The Sockets Networking API (3rd Edition)*, by Stevens, Fenner, & Rudolf

- Online resources
  - E.g. on socket programming

Grading (Tentative)

- Quiz 10%

- Assignments 40%

- Midterm 20%

- Final 30%

Policies

- Late Homework
  - One day late will cost you 25%, two days 50%, and three days 75%.
  - No homework will be accepted after the third day.

- Cellphones
  - Please turn them off before entering class.

- Cheating and Copying
  - First time you are caught you will get a zero for the task at hand.
  - Second time you are caught you will fail the course.
  - Providing your assignment to someone else is considered cheating.

Okay, so let's get started… with a crash course in data networking

Key Concepts in Networking

- Protocols
  - Speaking the same language
  - Syntax and semantics

- Layering
  - Standing on the shoulders of giants
  - A key to managing complexity

- Resource allocation
  - Dividing scare resources among competing parties
  - Memory, link bandwidth, wireless spectrum, paths, …
  - Distributed vs. centralized algorithms

- Naming
  - What to call computers, services, protocols, …

Protocols: Calendar Service

- Making an appointment with your advisor
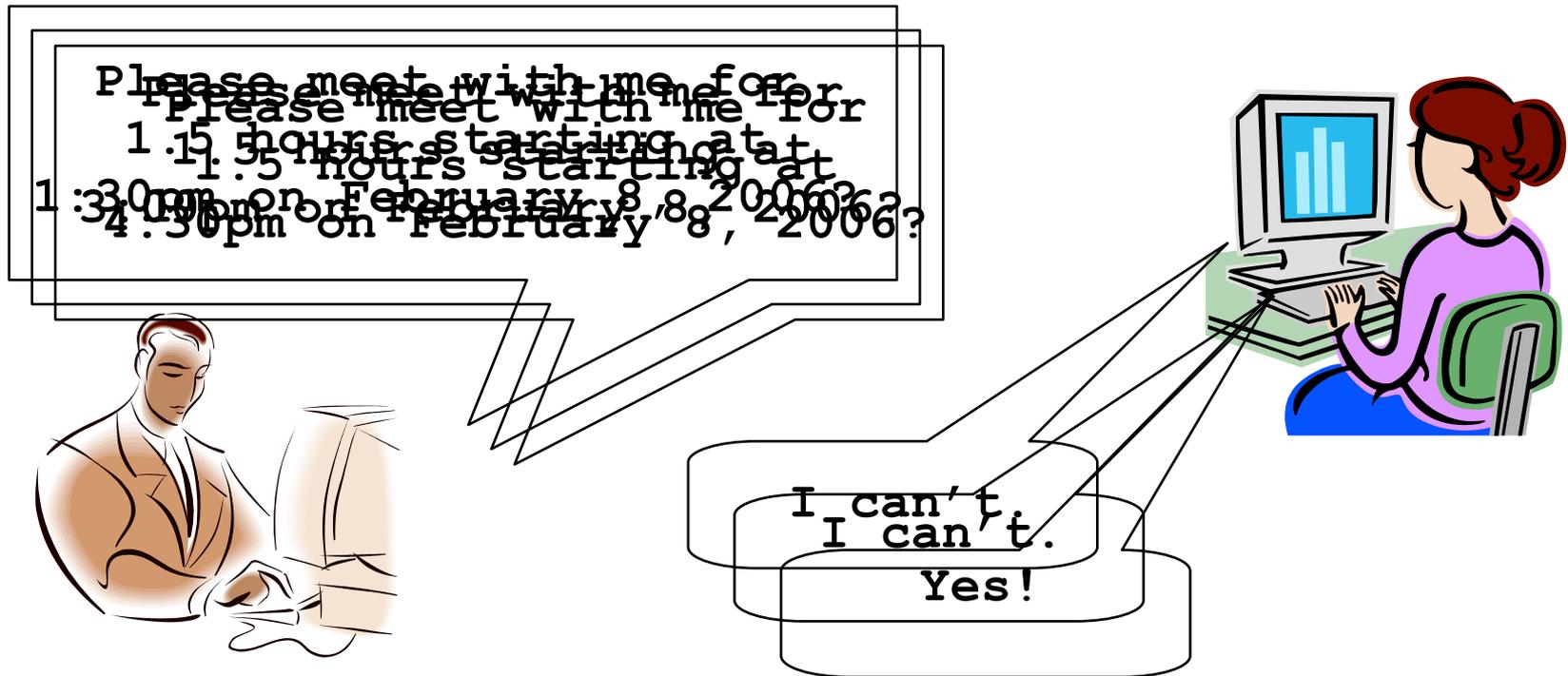
- Specifying the messages that go back and forth
  - And an understanding of what each party is doing

Okay, So This is Getting Tedious

- You: When are you free to meet for 1.5 hours during the next two weeks?

- Advisor: 10:30am on Feb 8 and 1:15pm on Feb 9.

- You: Book me for 1.5 hours at 10:30am on Feb 8.

- Advisor: Yes.

Well, Not Quite Enough

- Student #1: When can you meet for 1.5 hours during the next two weeks?

- Advisor: 10:30am on Feb 8 and 1:15pm on Feb 9.

- Student #2: When can you meet for 1.5 hours during the next two weeks?

- Advisor: 10:30am on Feb 8 and 1:15pm on Feb 9.

- Student #1: Book me for 1.5 hours at 10:30am on Feb 8.

- Advisor: Yes.

- Student #2: Book me for 1.5 hours at 10:30am on Feb 8.

- Advisor: Uh… well… I can no longer can meet then. I'm free at 1:15pm on Feb 9.

- Student #2: Book me for 1.5 hours at 1:15pm on Feb 9.

15

Specifying the Details

- How to identify yourself?
  - Name? Student ID?

- How to represent dates and time?
  - Time, day, month, year? In what time zone?
  - Number of seconds since Jan 1, 1970?

- What granularities of times to use?
  - Any possible start time and meeting duration?
  - Multiples of five minutes?

- How to represent the messages?
  - Strings? Record with name, start time, and duration?

- What do you do if you don't get a response?
  - Ask again? Reply again?

Example: HyperText Transfer Protocol

GET /courses/archive/ce443/ HTTP/1.1
Host: www.cs.sharif.edu
User-Agent: Mozilla/4.03
CRLF

Request

Response

HTTP/1.1 200 OK
Date: Mon, 4 Feb 2010 13:09:03 GMT
Server: Netscape-Enterprise/3.5.1
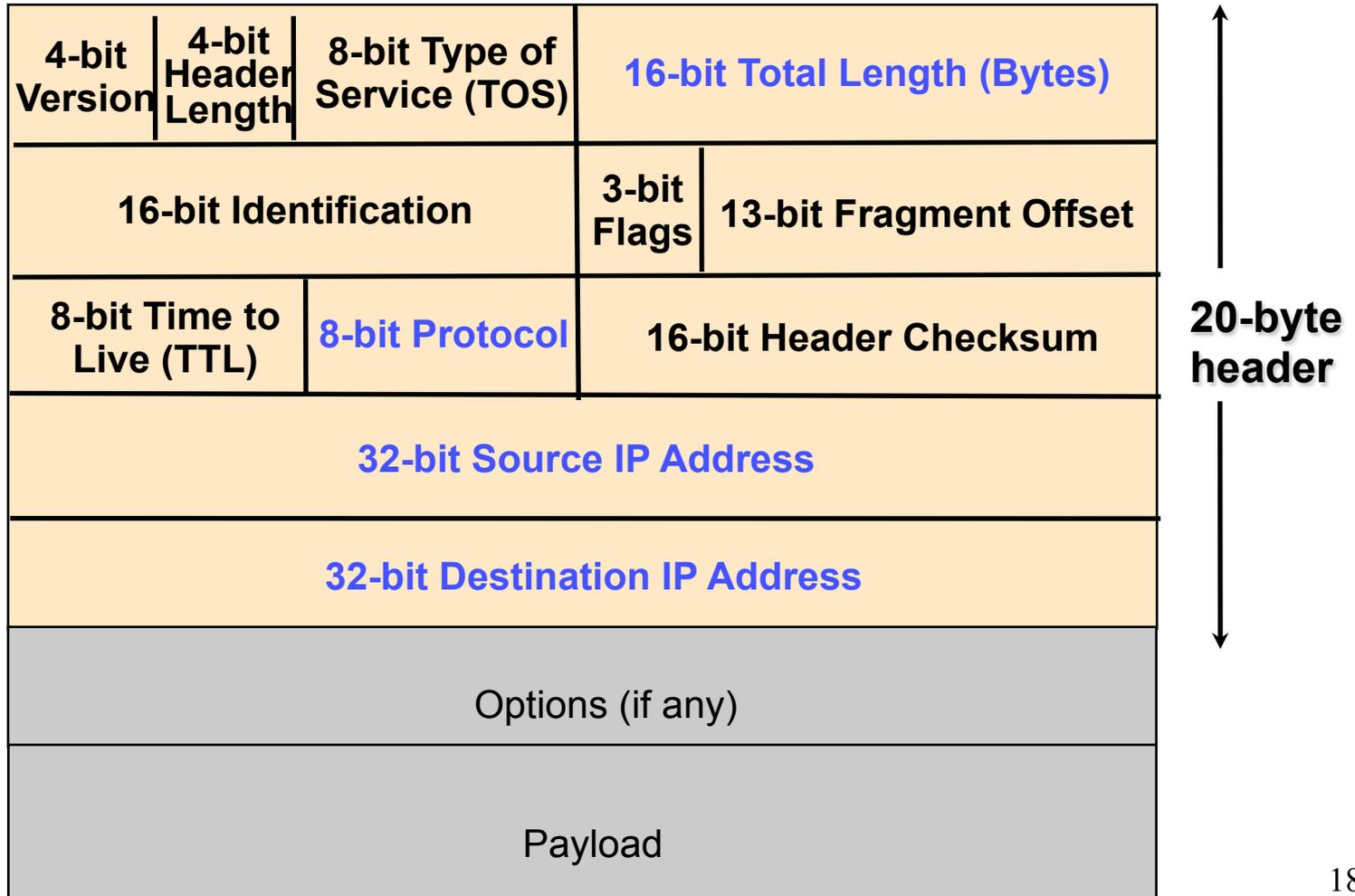Last-Modified: Mon, 4 Feb 2010 11:12:23 GMT
Content-Length: 21
CRLF
Site under construction

Example: IP Packet

4-bit Version	4-bit Header Length	8-bit Type of Service (TOS)	16-bit Total Length (Bytes)	
16-bit Identification			3-bit Flags	13-bit Fragment Offset
8-bit Time to Live (TTL)		8-bit Protocol	16-bit Header Checksum	
32-bit Source IP Address				
32-bit Destination IP Address				
Options (if any)				
Payload				

20-byte header

IP: Best-Effort Packet Delivery

- Packet switching
  - Send data in packets
  - Header with source & destination address

- Best-effort delivery
  - Packets may be lost
  - Packets may be corrupted
  - Packets may be delivered out of order

source

destination

IP network
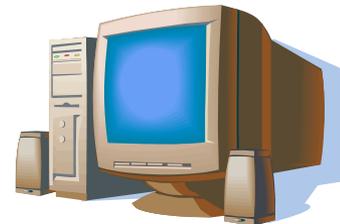
Example: Transmission Control Protocol

- Communication service (socket)
  - Ordered, reliable byte stream
  - Simultaneous transmission in both directions

- Key mechanisms at end hosts
  - Retransmit lost and corrupted packets
  - Discard duplicate packets and put packets in order
  - Flow control to avoid overloading the receiver buffer
  - Congestion control to adapt sending rate to network load

TCP connection

source   network   destination
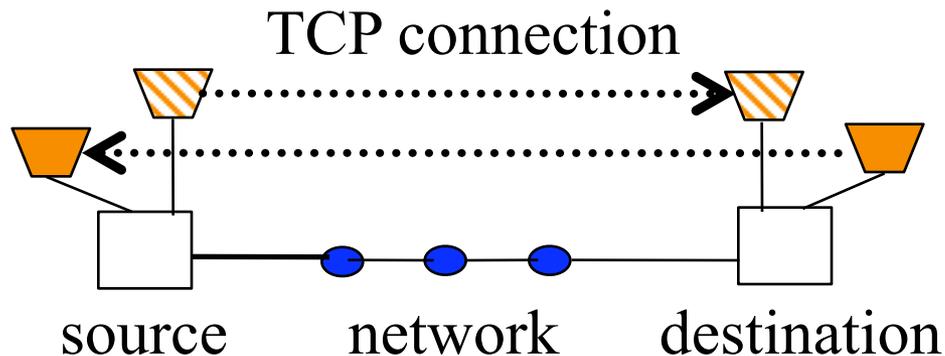
Protocol Standardization

- Communicating hosts speaking the same protocol
  - Standardization to enable multiple implementations
  - Or, the same folks have to write all the software

- Standardization: Internet Engineering Task Force
  - Based on working groups that focus on specific issues
  - Produces "Request For Comments" (RFCs)
    - Promoted to standards via rough consensus and running code
    - E.g., RFC 1945 on "HyperText Transfer Protocol – HTTP/1.0"
  - IETF Web site is http://www.ietf.org

- De facto standards: same folks writing the code
  - P2P file sharing, Skype, <your protocol here>…

Layering: A Modular Approach

- Sub-divide the problem
  - Each layer relies on services from layer below
  - Each layer exports services to layer above

- Interface between layers defines interaction
  - Hides implementation details
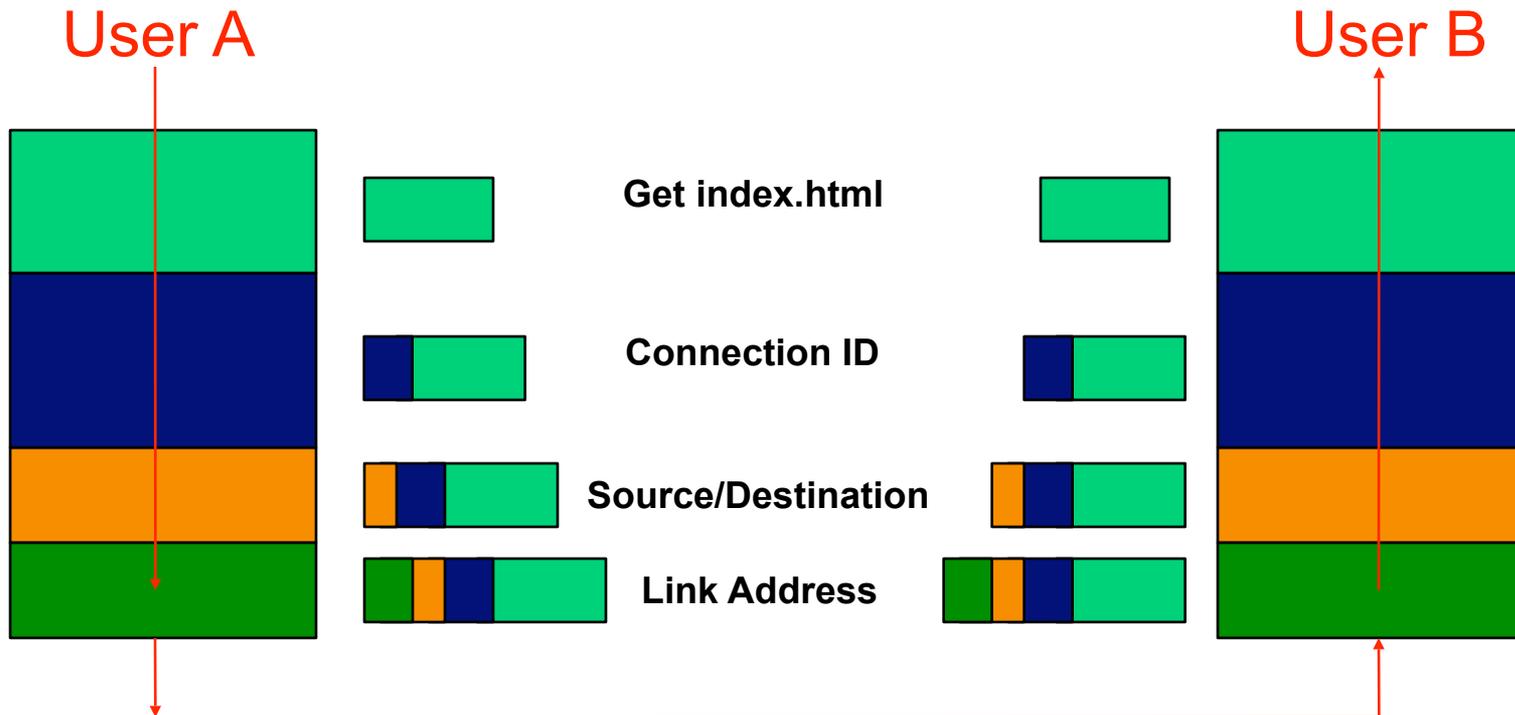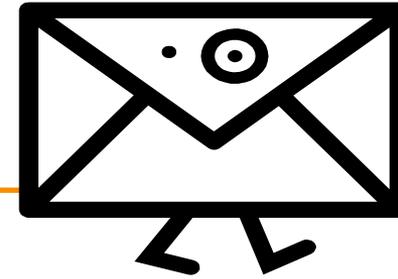 - Layers can change without disturbing other layers

Application
Application-to-application channels
Host-to-host connectivity
Link hardware
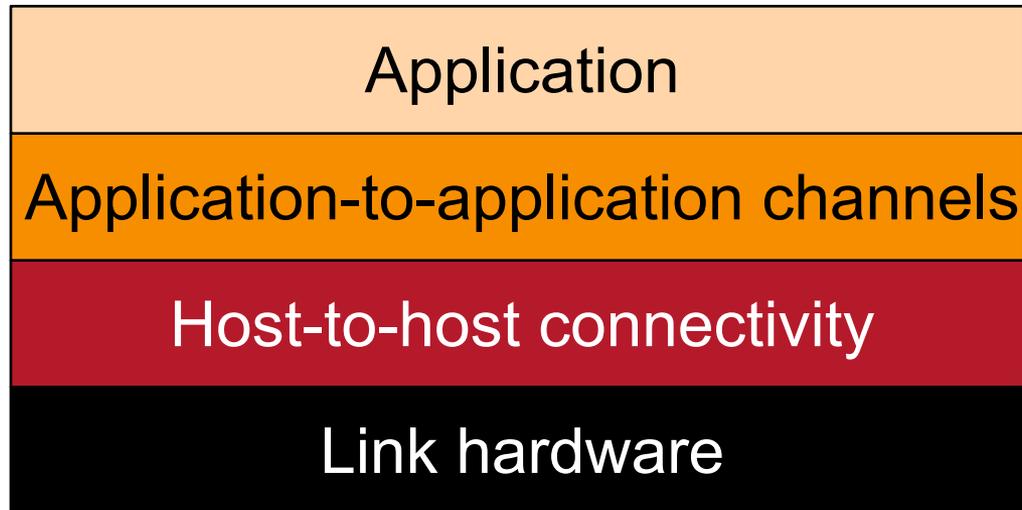
IP Suite: End Hosts vs. Routers
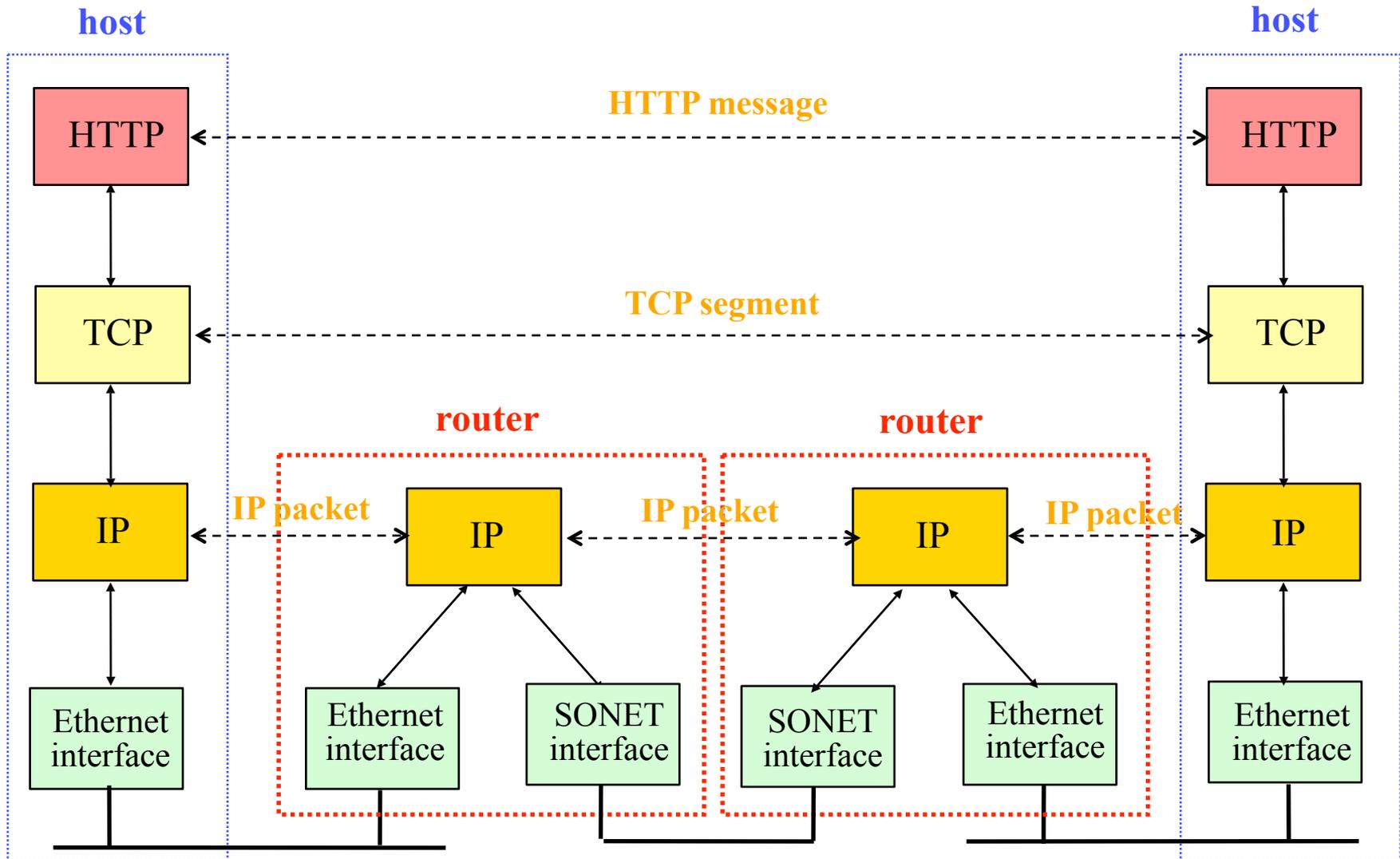
The Internet Protocol Suite
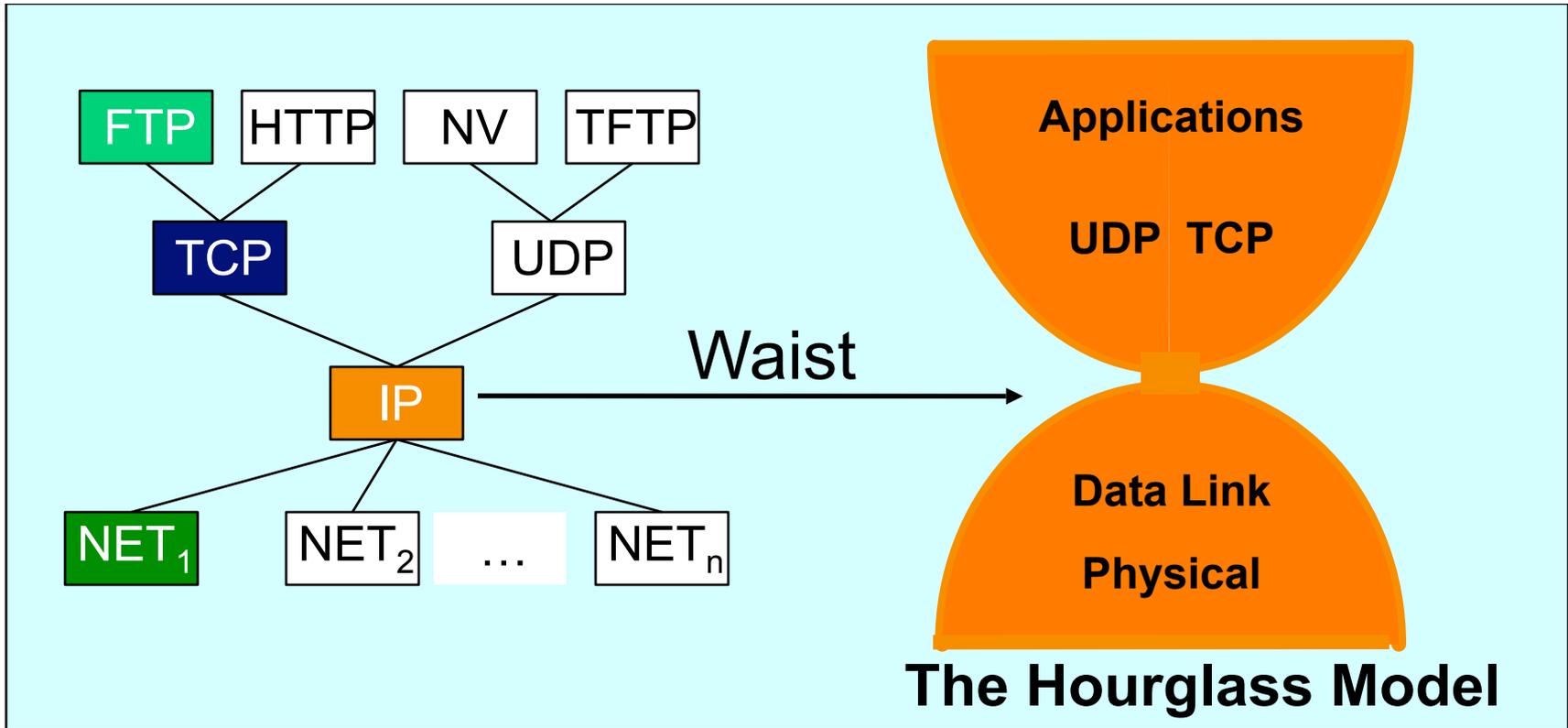
The waist facilitates interoperability

Layer Encapsulation

What if the Data Doesn't Fit?

Problem: Packet size

- On Ethernet, max IP packet is 1500 bytes

- Typical Web page is 10 kbytes

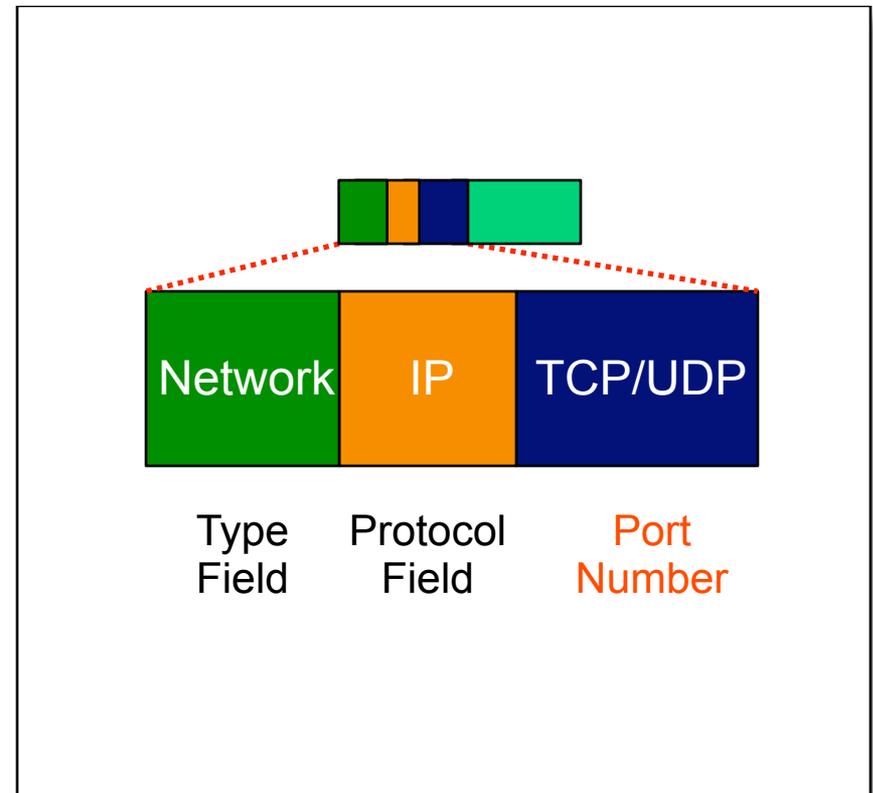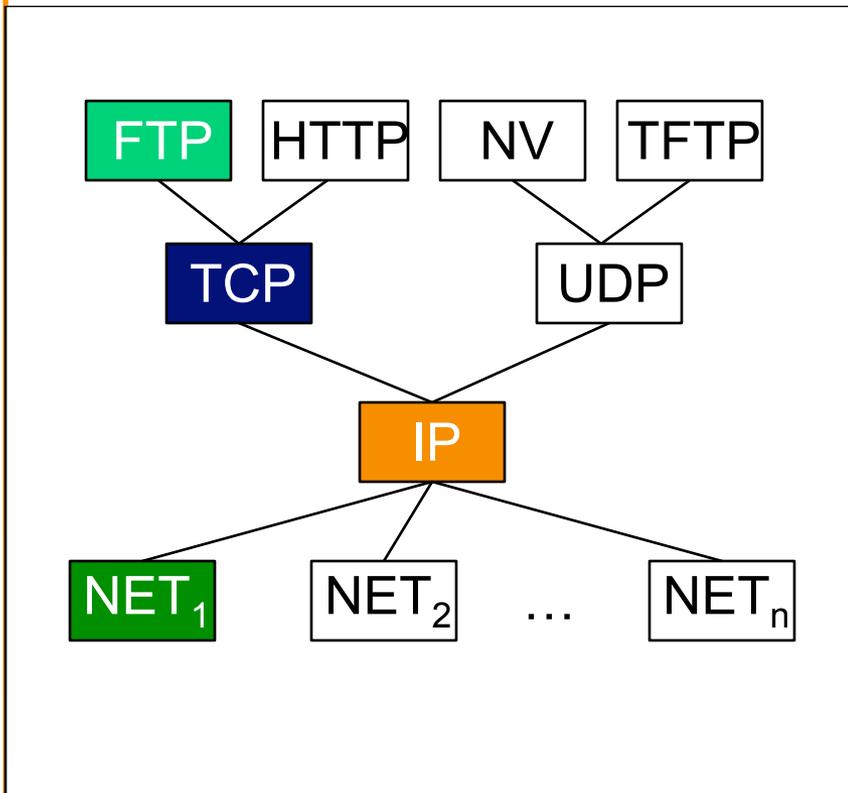Solution: Split the data across multiple packets

ml → x.ht → inde → GET →

GET index.html

Protocol Demultiplexing

- Multiple choices at each layer

Demultiplexing: Port Numbers

- Differentiate between multiple transfers
  - Knowing source and destination host is not enough
  - Need an id for *each transfer* between the hosts

- Specify a particular service running on a host
  - E.g., HTTP server running on port 80
  - E.g., FTP server running on port 21

**HTTP transfers**

FTP transfer
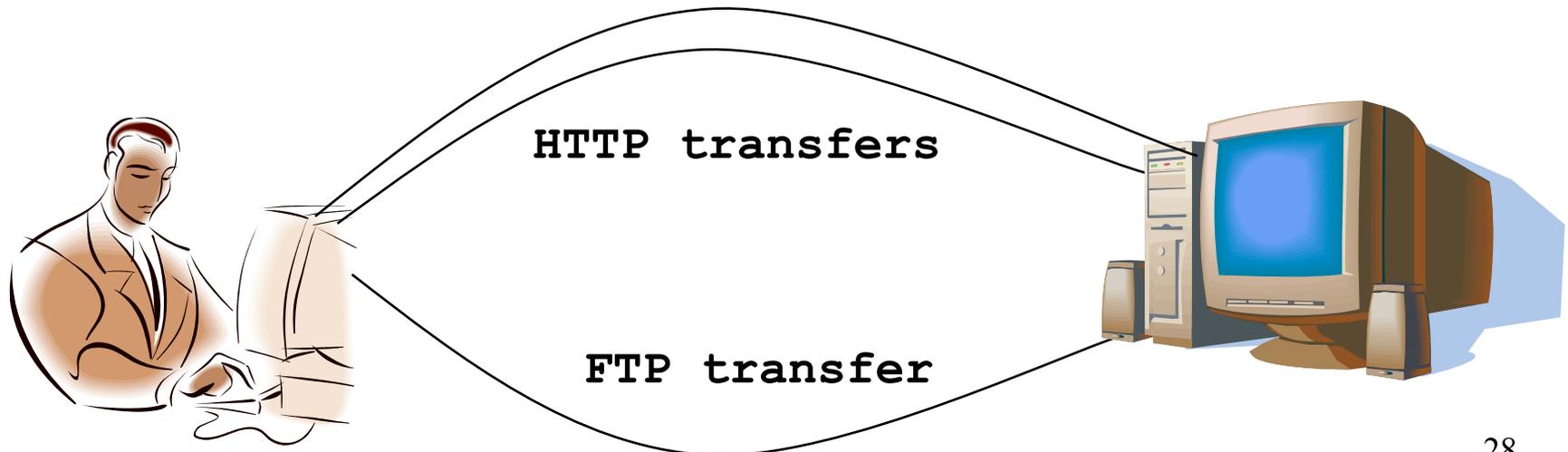
Is Layering Harmful?

- Layer N may duplicate lower level functionality
  - E.g., error recovery to retransmit lost data

- Layers may need same information
  - E.g., timestamps, maximum transmission unit size

- Strict adherence to layering may hurt performance
  - E.g., hiding details about what is really going on

- Some layers are not always cleanly separated
  - Inter-layer dependencies for performance reasons

- Headers start to get really big
  - Sometimes more header bytes than actual content

Resource Allocation: Queues
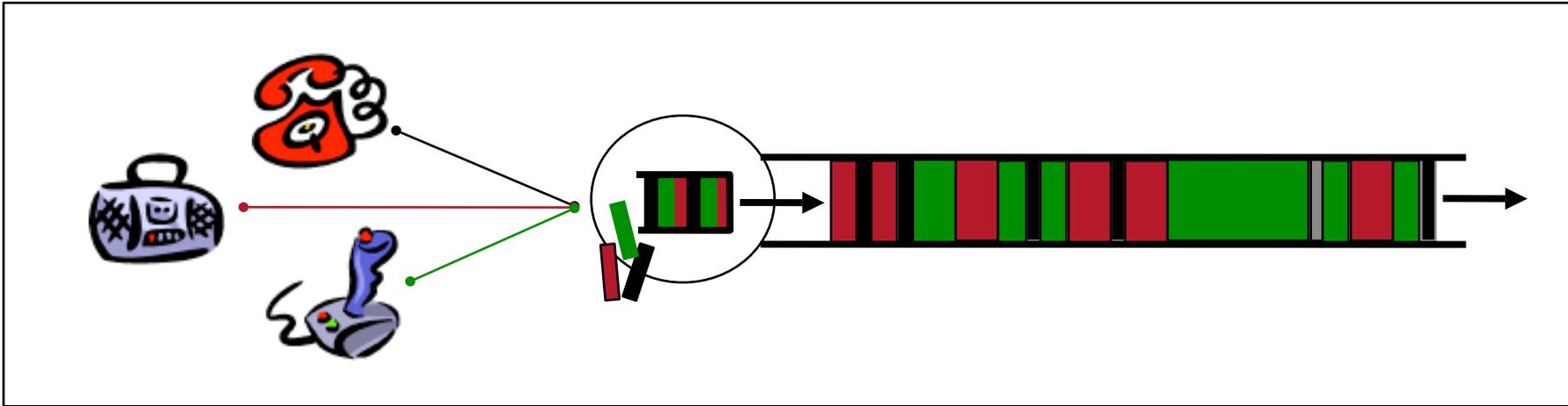
- Sharing access to limited resources
  - E.g., a link with fixed service rate

- Simplest case: first-in-first out queue
  - Serve packets in the order they arrive
  - When busy, store arriving packets in a buffer
  - Drop packets when the queue is full

What if the Data gets Dropped?

Problem: Lost Data

GET index.html

Internet

Solution: Timeout and Retransmit

GET index.html
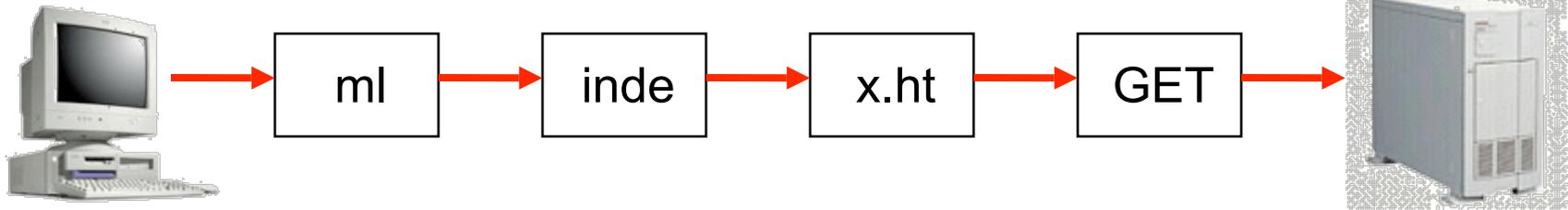
GET index.html

Internet

GET index.html
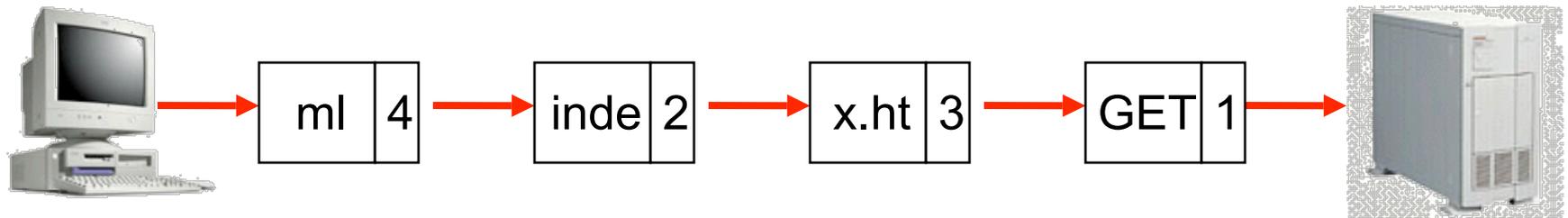
What if the Data is Out of Order?
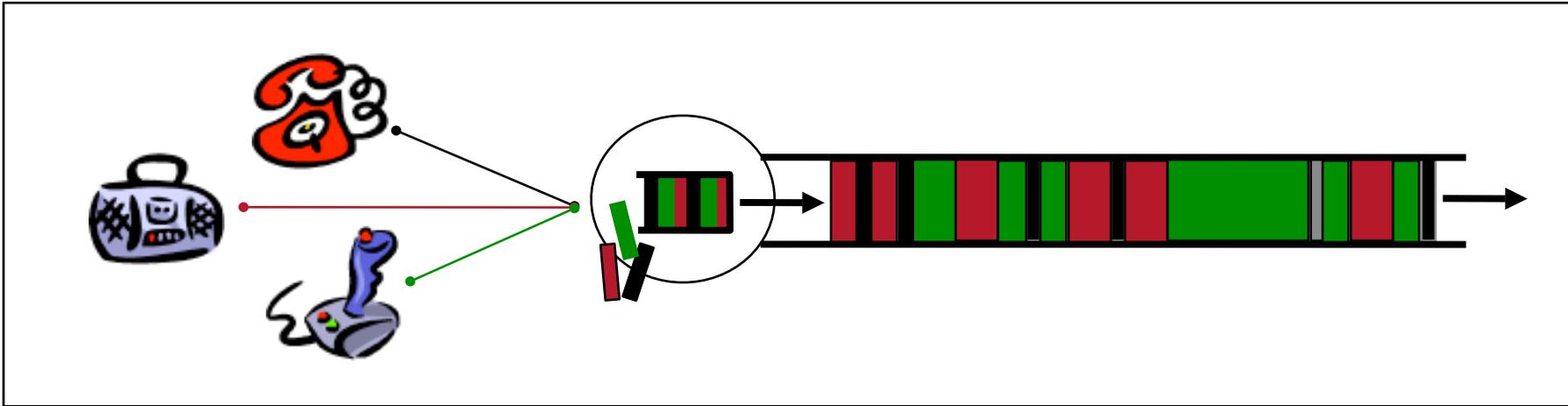
Problem: Out of Order

ml	→	inde	→	x.ht	→	GET

GET x.htindeml

Solution: Add Sequence Numbers

ml	4	→	inde	2	→	x.ht	3	→	GET	1

GET index.html

Resource Allocation: Congestion Control

- What if too many folks are sending data?
  - Senders agree to slow down their sending rates
  - … in response to their packets getting dropped

- The essence of TCP congestion control
  - Key to preventing congestion collapse of the Internet

Transmission Control Protocol

- Flow control: window-based
  - Sender limits number of outstanding bytes (window size)
  - *Receiver window* ensures data does not overflow receiver

- Congestion control: adapting to packet losses
  - *Congestion window* tries to avoid overloading the network (increase with successful delivery, decrease with loss)
  - TCP connection starts with small initial congestion window

congestion avoidance

slow start

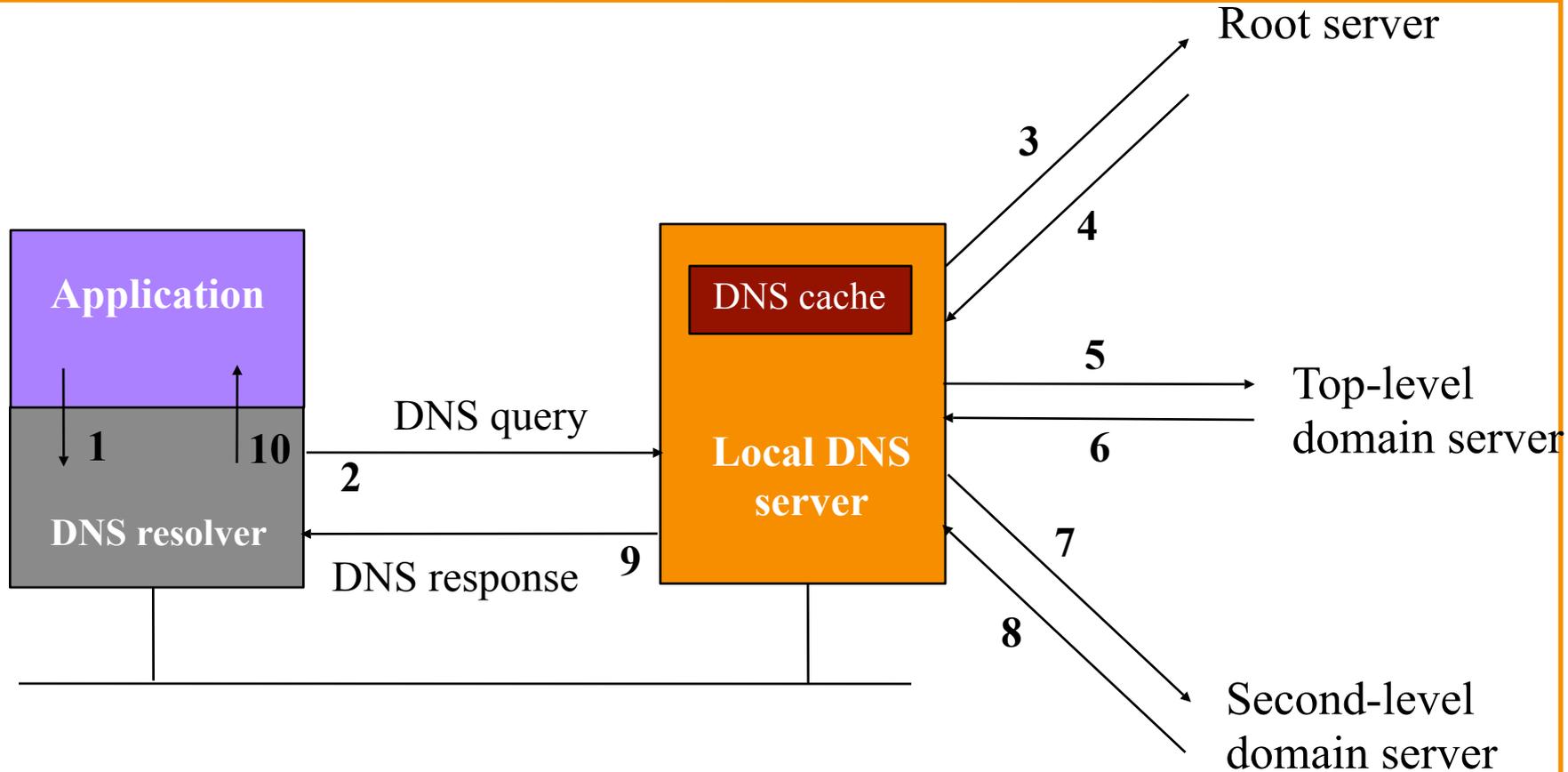**time**
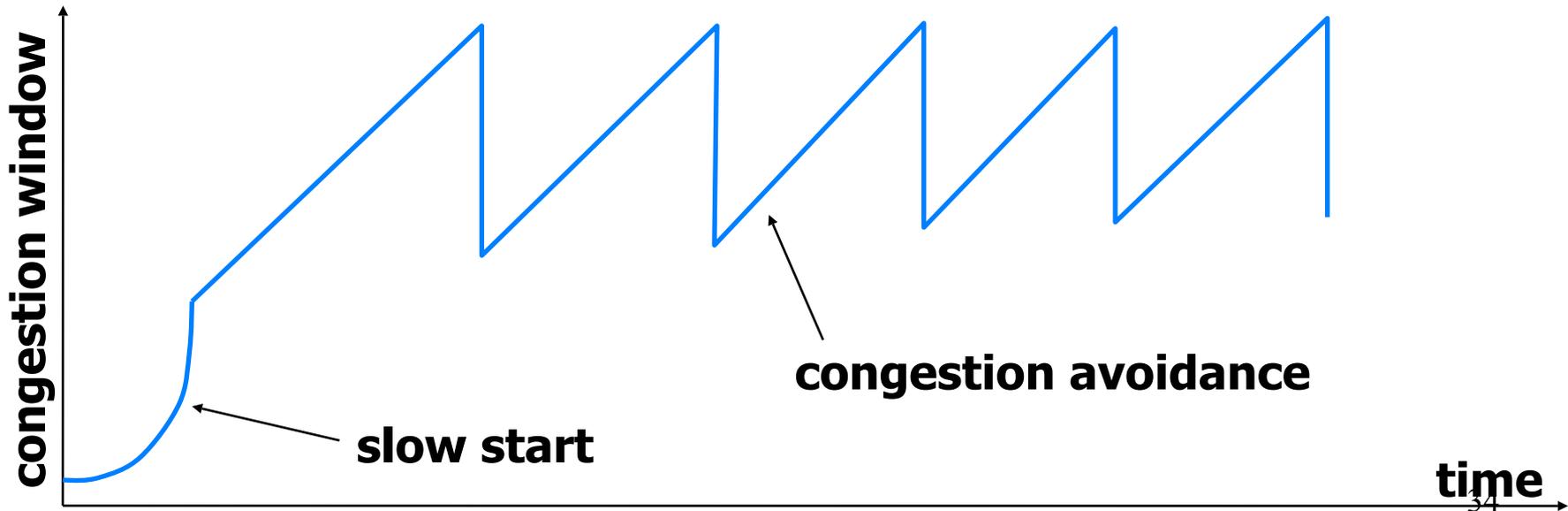
Naming: Domain Name System (DNS)

- Properties of DNS
  - Hierarchical name space divided into zones
  - Translation of names to/from IP addresses
  - Distributed over a collection of DNS servers

- Client application
  - Extract server name (e.g., from the URL)
  - Invoke system call to trigger DNS resolver code
    - E.g., *gethostbyname()* on "www.cs.sharif.edu"

- Server application
  - Extract client IP address from socket
  - Optionally invoke system call to translate into name
    - E.g., *gethostbyaddr()* on "12.34.158.5"

Domain Name System
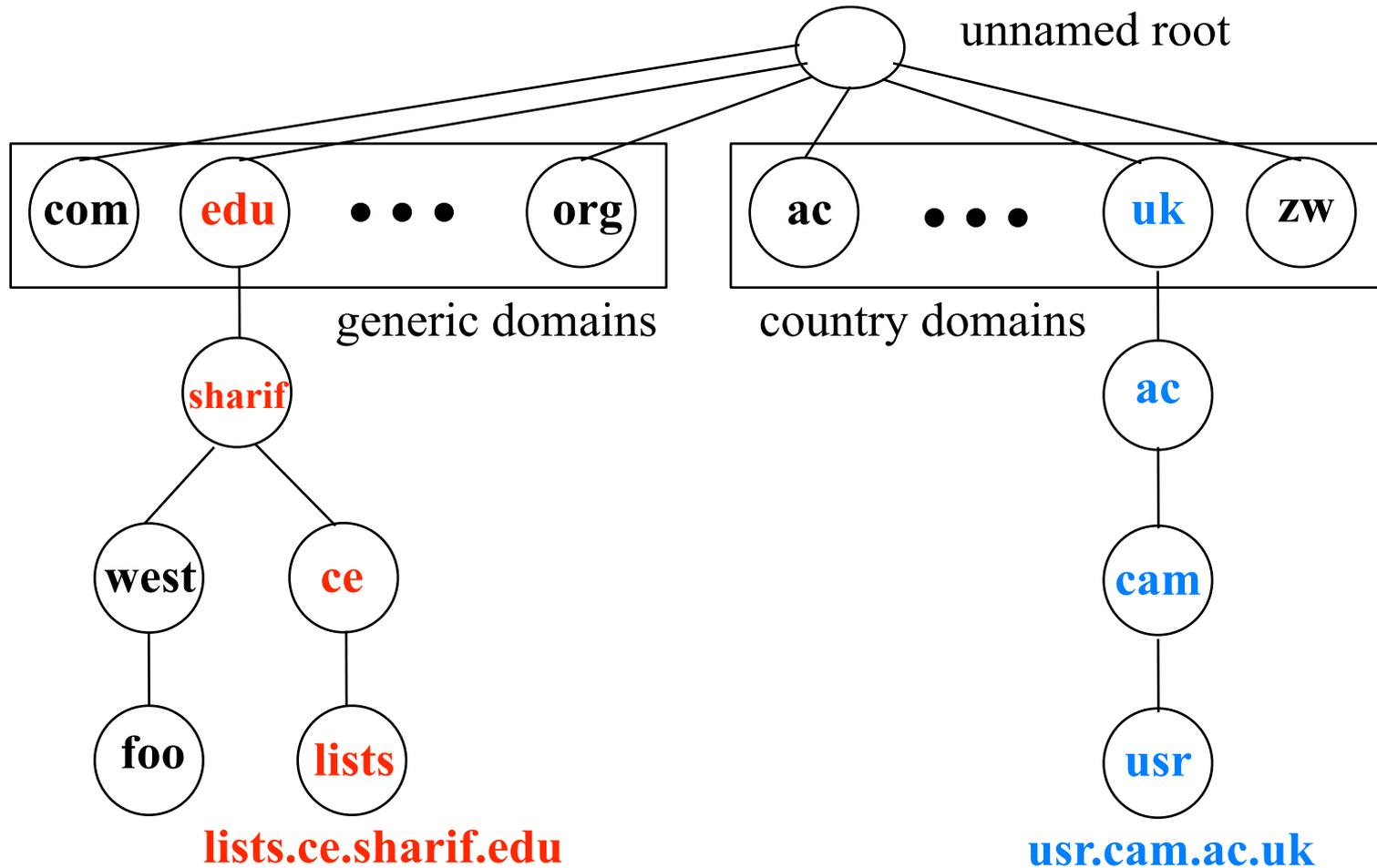
DNS Resolver and Local DNS Server

Caching based on a time-to-live (TTL) assigned by the DNS server responsible for the host name to reduce latency in DNS translation.

Conclusions

- Course objectives
  - How the Internet works, key concepts in networking, and Network programming

- Key concepts in networking
  - Protocols, layers, resource allocation, and naming

- Next lecture:
  - Read Chapter 1 of the Peterson/Davie book
  - Skim the online reference material on sockets
  - (Re)familiarize yourself with C programming