# The Network Layer: Addressing and Router Design

CS 352, Lecture 12, Spring 2020

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

Srinivas Narayana

RUTGERS
UNIVERSITY | NEW BRUNSWICK

# Course announcements

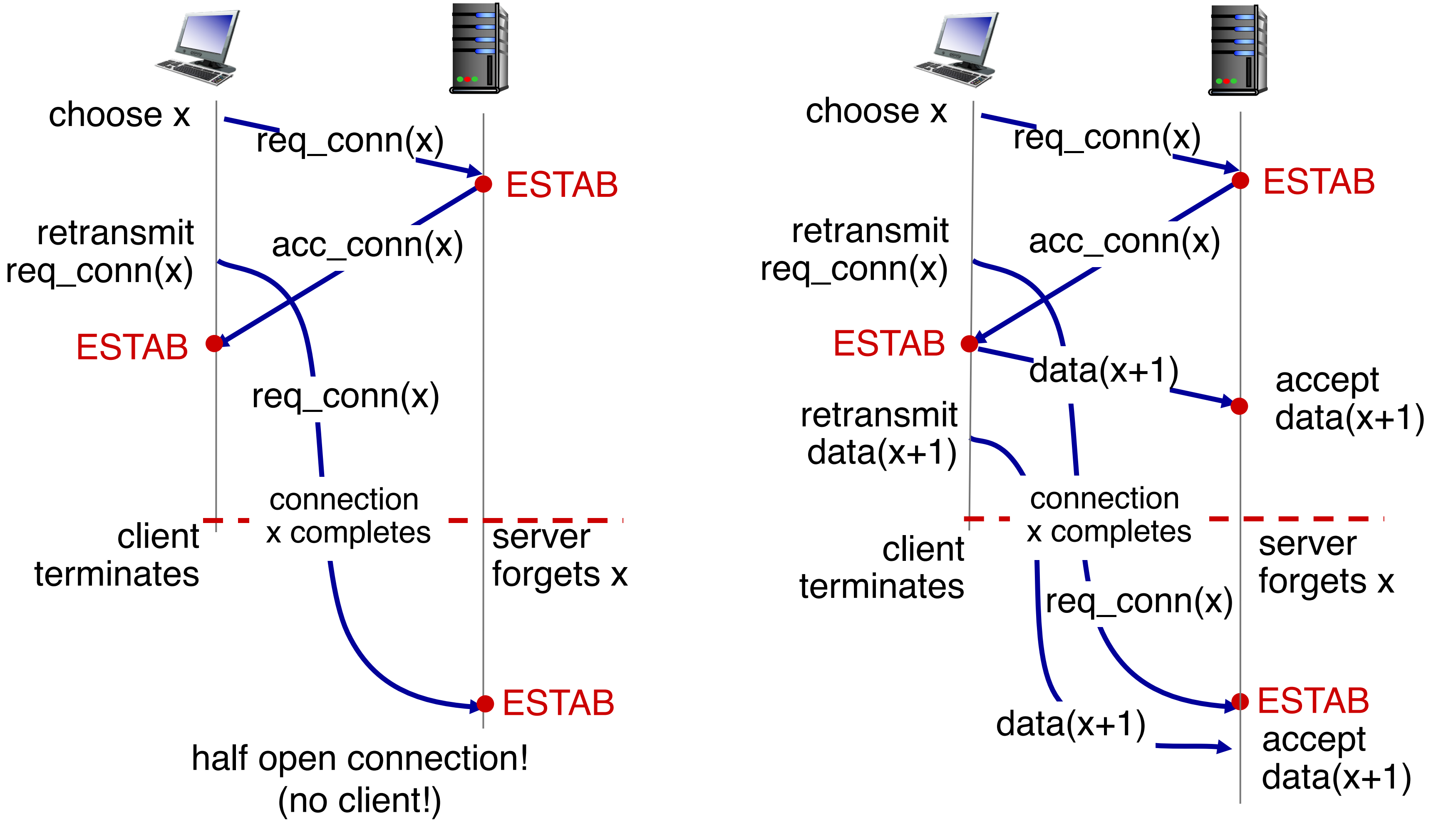
- Mid-term grades available
  - 24/7 grading policy: re-grading requests considered in the school-day period between Thu 03/12 at noon until 03/25 noon
  - Average: 74%
  - Note that final mid-term 1 points will be out of 15

- Online instruction: Likely over WebEx
  - One of the TAs to facilitate
  - Call in using a wired connection or phone if possible
  - Some etiquette and aids to make it as effective as possible

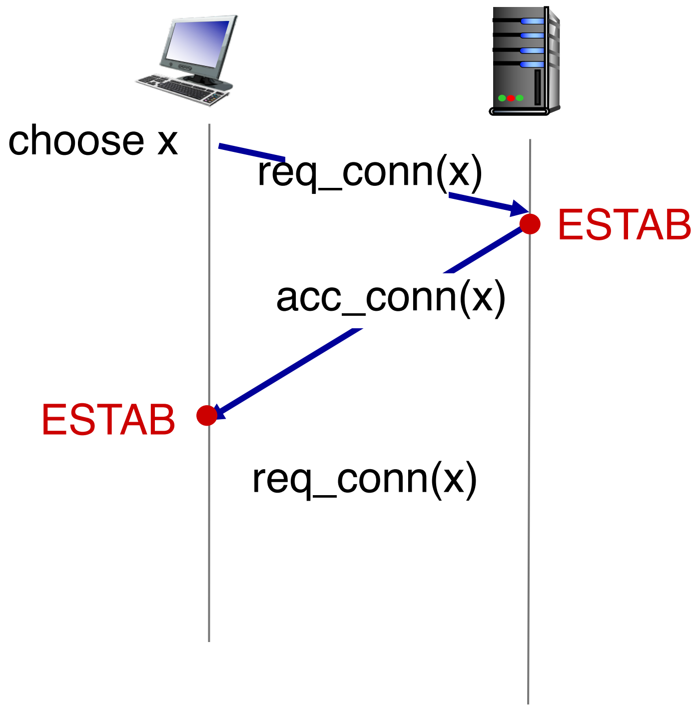
# Review of concepts

- TCP congestion control: need distributed, efficient, fair
    - TCP AIMD helps achieve fairness and efficiency. How?
- TCP timeout: how to set it?
    - Basic intuition: use measured RTTs to find a tight upper bound
    - Estimate RTT using a exponentially weighted moving average
    - Use variance w.r.t. average to create a safety margin
- Retransmission ambiguity: Karn's algorithm
- TCP connection management:
    - TCP is full–duplex: independent seq#, windows, data on each side
    - Three-way handshake: SYN, SYN+ACK, ACK
    - TCP state machine: CLOSED → LISTEN → SYN sent/rcvd → EST → …

# Review: 2-way handshake failure scenarios



# 2-way handshake denial of service




- When server moves into ESTAB state, it:
  - Entry in TCP demultiplexing table
  - Buffer memory for send and recv
  - Code paths to determine sequence numbers, parameters for connection
- Asymmetric work:
  - Client just needs to send a SYN
- Possibility: denial-of-service attack
- TCP standard: client can't send data in SYN
  - Implication: Server won't send data in SYN/ACK either

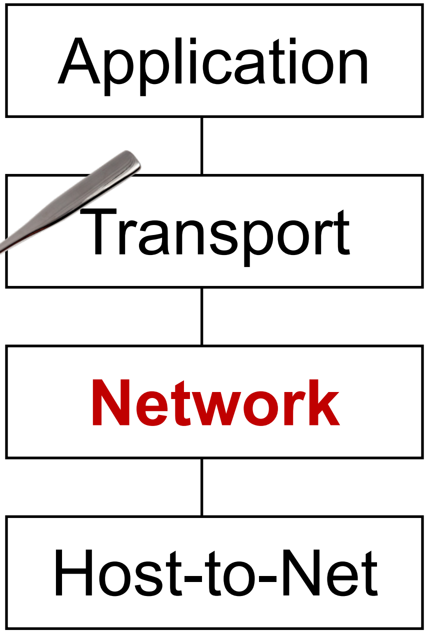
# TCP summary

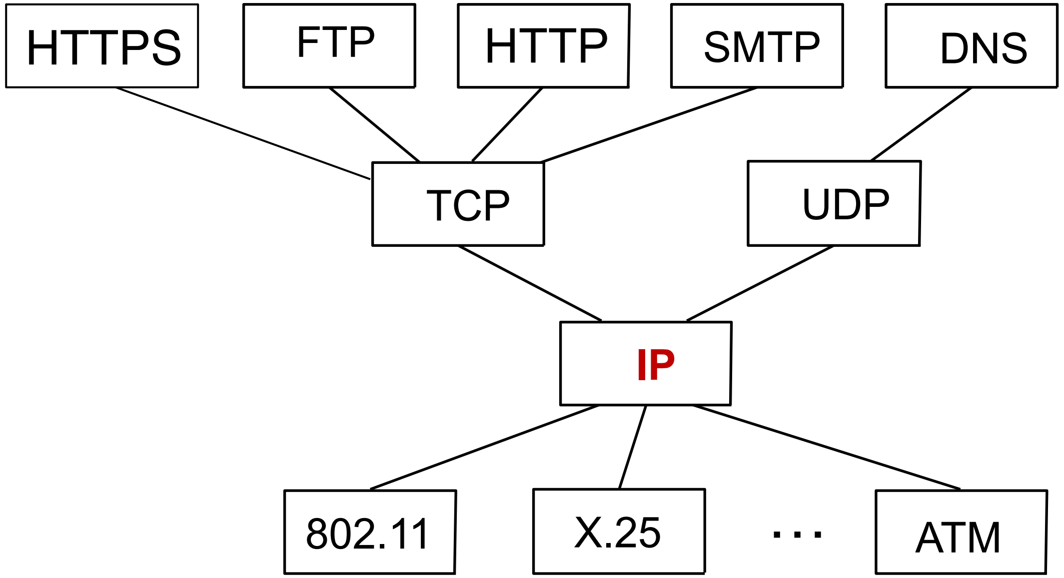
- Reliability
- Ordering
- Flow control
- Congestion control
- Timeout computation
- Connection management
- When you tune in on WebEx, reflect on that…

# The Network Layer

Introduction

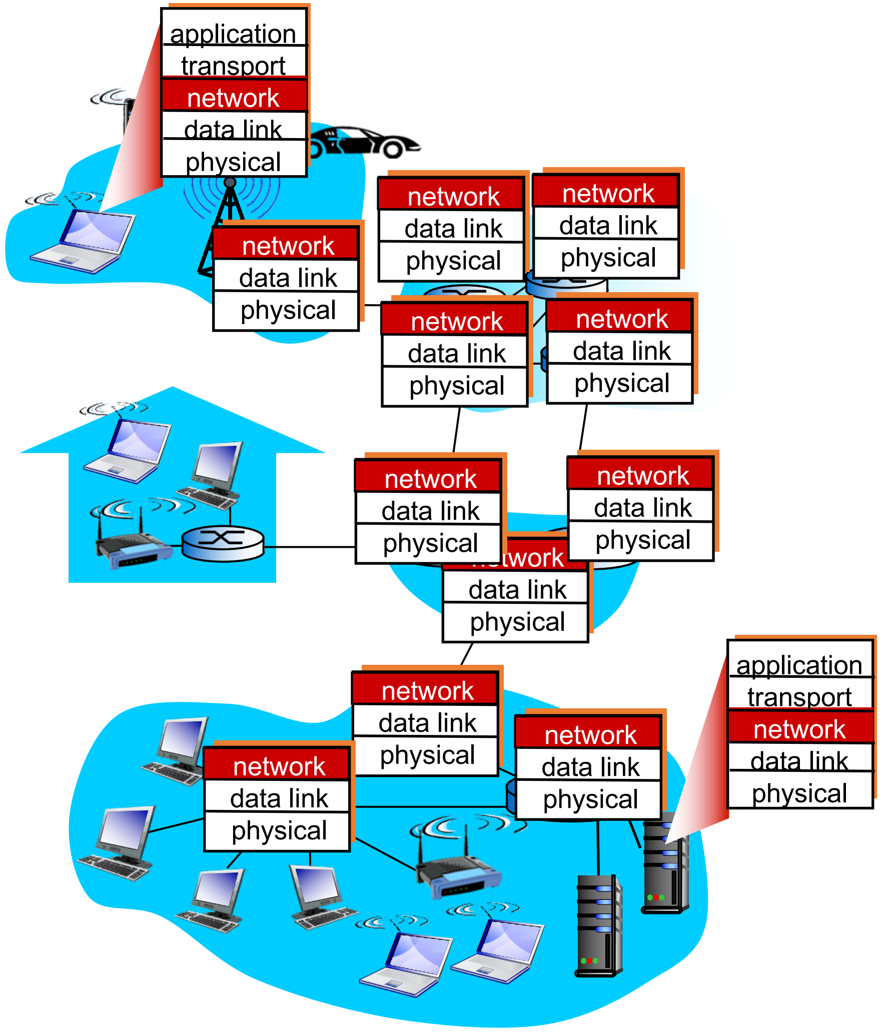
# Where we are: The network layer



# Network layer functions

- Move data from sending to receiving endpoint
- on sending endpoint, encapsulates transport segments into datagrams
- on receiving endpoint, deliver datagrams to transport layer
- The network layer also runs in every router!
- The router examines header fields in all network-layer (IP) datagrams passing through it



# Two key network-layer functions

- Forwarding: move packets from router's input to appropriate router output

- Routing: determine route taken by packets from source to destination
  - routing algorithms

- The network layer solves the routing problem.

Analogy: taking a trip

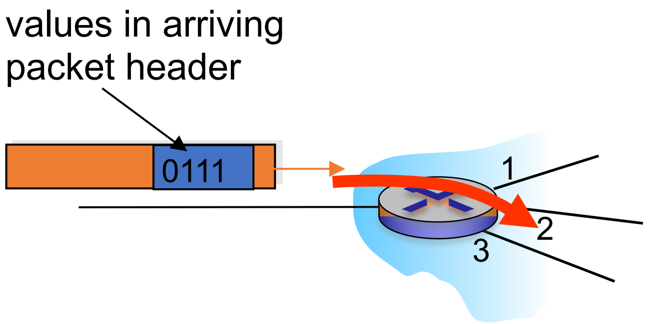
- Forwarding: process of getting through single interchange

- Routing: process of planning trip from source to destination

# Data plane and Control Plane

## Data plane

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port
- forwarding function




## Control plane

- network-wide logic
- determines how datagram is routed along end-to-end path from source to destination endpoint
- two control-plane approaches:
  - Distributed routing algorithms: implemented in routers
  - Centralized routing: software-defined networking (SDN) implemented on (remote) servers

# Internet Protocol: Addressing

# We need addresses

- Allow endpoints to talk to each other
- Allow routers to determine how to move a packet
- The primary function of IP addresses is to help implement efficient routing and forwarding

# IPv4 Addresses

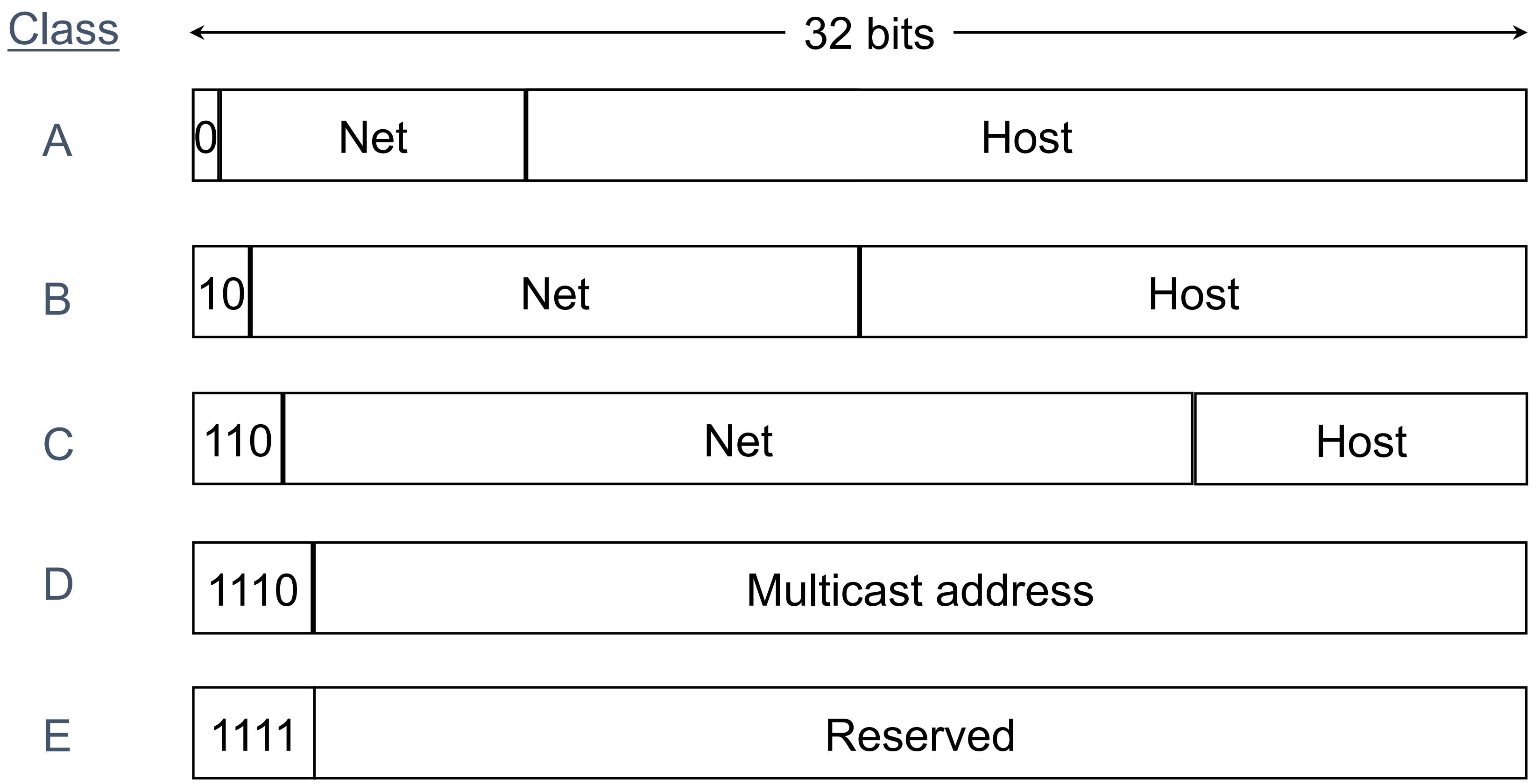
- 32 bits long
- Identifier for host, router interface
  - Corresponds to the point of attachment
  - Not an identifier for the endpoint
- Notation:
  - Each byte is written in decimal in MSB order, separated by dots
  - Example: 128.195.1.80 stands for the 32-bit IP address

10000000 11000011 00000001 01010000

# Types of IPv4 Addresses

- Unicast Address
  - Destination is a single host
- Multicast address
  - Destination is a group of hosts
- Broadcast address
  - 255.255.255.255
  - Destination is all hosts

# IPv4 Address Classes (old, “classful”)



# IP Address Classes

- Class A:
  - For very large organizations
  - 16 million hosts allowed
- Class B:
  - For large organizations
  - 65 thousand hosts allowed
- Class C
  - For small organizations
  - 255 hosts allowed
- Class D
  - Multicast addresses
  - No network/host hierarchy

# Key Principle: Use hierarchy to scale

- IP addresses fall into a class, corresponding to a prefix length

- All those IP addresses with the same prefix can take identical paths from a far-away remote endpoint

- This principle reduces the amount of information needed to route packets in the Internet

- We will also see how it enables delegating prefixes from ISPs to their customers

# Problems with Class-based Routing

- Too many small networks requiring multiple class C addresses

- Running out of class B addresses, not enough nets in class A

- Addressing strategy must allow for greater diversity of network sizes

# Classless IP addressing (CIDR)

# IP addressing: CIDR

CIDR: Classless InterDomain Routing

- subnet portion of address of arbitrary length
- address format: a.b.c.d/x, where x is # bits in subnet portion of address

subnet part ← → host part

11001000 00010111 00010000 00000000

200.23.16.0/23

# CIDR

- An ISP can obtain a block of addresses and partition this further to its customers
  - Say an ISP has 200.8.0.0/16 address (65K addresses).
  - It has another customer who needs only 64 addresses starting from 200.8.4.128
  - Then that block can be specified as 200.8.4.128/26

- 200.8.4.128/26 is “inside” 200.8.0.0/16

# Subnetting

Example: Class B address with 8-bit subnetting

| | 16 bits | 8 bits | 8 bits |
|---|---|---|---|
| | Network id | Subnet id | Host id |
| Example Address: | 165.230 | .24 | .8 |

# Subnet Masks

Subnet masks allow hosts to determine if another IP address is on the same subnet or the same network

| | 16 bits | 8 bits | 8 bits |
|---|---|---|---|
| | Network id | Subnet id | Host id |
| Mask: | 1111111111111111 | 11111111 | 00000000 |
| | 255.255 | .255 | .0 |

# Subnet Masks *(cont'd)*

Assume IP addresses A and B share subnet mask M.

Are IP addresses A and B on the same subnet?

1. Compute logical AND (A & M).
2. Compute logical AND (B & M).
3. If (A & M) == (B & M) then A and B are on the same subnet.

Example: A and B are class B addresses

A = 165.230.82.52
B = 165.230.24.93
M = 255.255.255.0

Same (classful) network?
Same subnet?

# Example of IP Addressing in a network