Review

CS168, Fall 2014

Sylvia Ratnasamy

http://inst.eecs.berkeley.edu/~cs168/

Logistics

- Test is closed book, closed notes
- Will start on time + “settling in” delay
- Two two-sided “cheat sheet”, hand written
- No calculators, electronic devices, etc.
  - Test does not require any complicated calculation

General Guidelines (1)

- Exam format (tentative!)
  - Similar to the midterm but between 50-75% longer
  - (Q1) A set of multiple choice questions
    - ordered roughly from easiest to hardest
  - (Q2) Design scenarios
    - ordered roughly from easiest to hardest within each scenario
  - (Q3+) more traditional problem solving
    - ordered from easiest to hardest

- Pace yourself accordingly.

General Guidelines (2)

- Test based on material covered in lecture & sections from the entire semester
  - Text: only to clarify details and context for the above
- More emphasis on post-midterm material
  - But you really can't tackle post-midterm material without thoroughly understanding the pre-midterm material!

General Guidelines (3)

- Be prepared to:
  - Weigh design options outside of the context we studied them in

    e.g., run BGP between datacenter sites?

  - Contemplate new designs we haven't talked about
    - *e.g., HTTP-like protocol built on top of UDP*
  - Consider the `complete picture' (esp. Q1 and Q2)
    - *e.g., persistent TCP connection when my DNS server fails…*
- If you're unsure, put down your assumptions

From here on…

- Walk through what I expect you to know: key topics, important aspects of each
  - Focus on post-midterm material; see midterm review for pre-midterm material
- Just because I didn't cover it in review doesn't mean you don't need to know it
  - But if I covered it today, you should know it
- My plan: summarize, not explain
  - Stop me when you want to discuss something further!

Expect a question on …

- Congestion control: TCP and advanced CC
- HTTP performance
- E2E operation: “steps when A talks to B” (lec 18)
- Wireless
- Ethernet spanning tree and self learning
- Datacenters

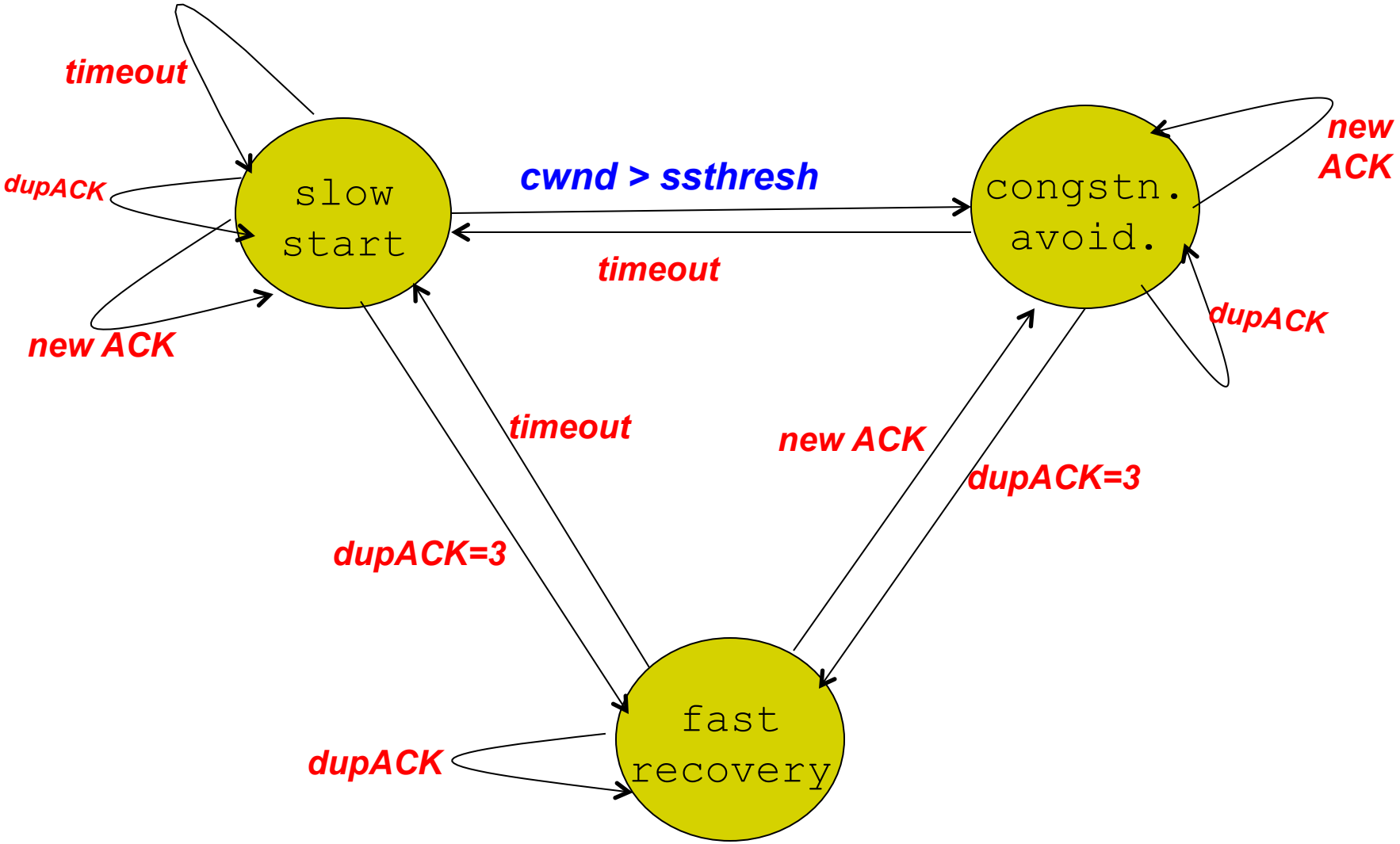
TCP Congestion Control (lecture 12)

- Why?
  - Because a sender shouldn't overload the network itself
  - But yet, should make efficient use of available network capacity
  - While sharing available capacity fairly with other flows
  - And adapting to changes in available capacity

- How?
  - Quickly find current available capacity (slow start)
  - Then adapt to changes (congestion avoidance, AIMD)
  - With optimizations (fast recovery)

  - *Study the TCP state machine diagram from the text!*

TCP State Machine (partial)



Congestion Control: Details to know

- Slow Start:
  - Why: discover available bandwidth from a cold start
  - How: exponential increase per RTT (every ACK, cwnd = cwnd + 1)

- Congestion Avoidance:
  - Why: adapt to variations in available bandwidth
  - How: AIMD (every ACK, cwnd = cwnd + 1/cwnd)
  - Why AIMD? For fairness

- Fast Recovery:
  - Keeps packets “in flight” after an isolated loss (why?)
  - How: artificially inflate the CWND on every duplicate ACK (for a while)

Advanced CC and Fairness (lecture 14)

- **TCP throughput equation**
  - Know the equation
  - Understand the implications

$$\text{Throughput} = \sqrt{\frac{3}{2}} \frac{1}{RTT\sqrt{p}}$$

- **Max-Min Fairness / Fair queuing**
  - Definition, know how to calculate fair rate (e.g., HW3)
  - Know pros and cons of FQ (isolation, router complexity)

- **Router assisted congestion control**
  - pros (better info.) and cons (complexity in routers)
  - e.g., explicit rate allocation (RCP): basic idea + pros/cons
  - e.g., explicit cong. notification (ECN) : basic idea only

Application Layer (lecture 15, 16)

- Domain Name System (DNS)
  - What's behind (e.g.) xyz.cs.berkeley.edu
- HTTP and the Web
  - What happens when you click on a link?

Internet Names & Addresses

- Machine addresses: *e.g., 169.229.131.109*
  - router-usable labels for machines
  - conforms to network structure (the “where”)
- Machine names: *e.g., inst.eecs.berkeley.edu*
  - human-usable labels for machines
  - conforms to organizational structure (the “who”)
- The Domain Name System (DNS) is how we map from one to the other

Key to DNS design: Hierarchy

Three intertwined hierarchies

- Hierarchical namespace
  - As opposed to original flat namespace
- Hierarchically administered
  - As opposed to centralized
- (Distributed) hierarchy of servers
  - As opposed to centralized storage

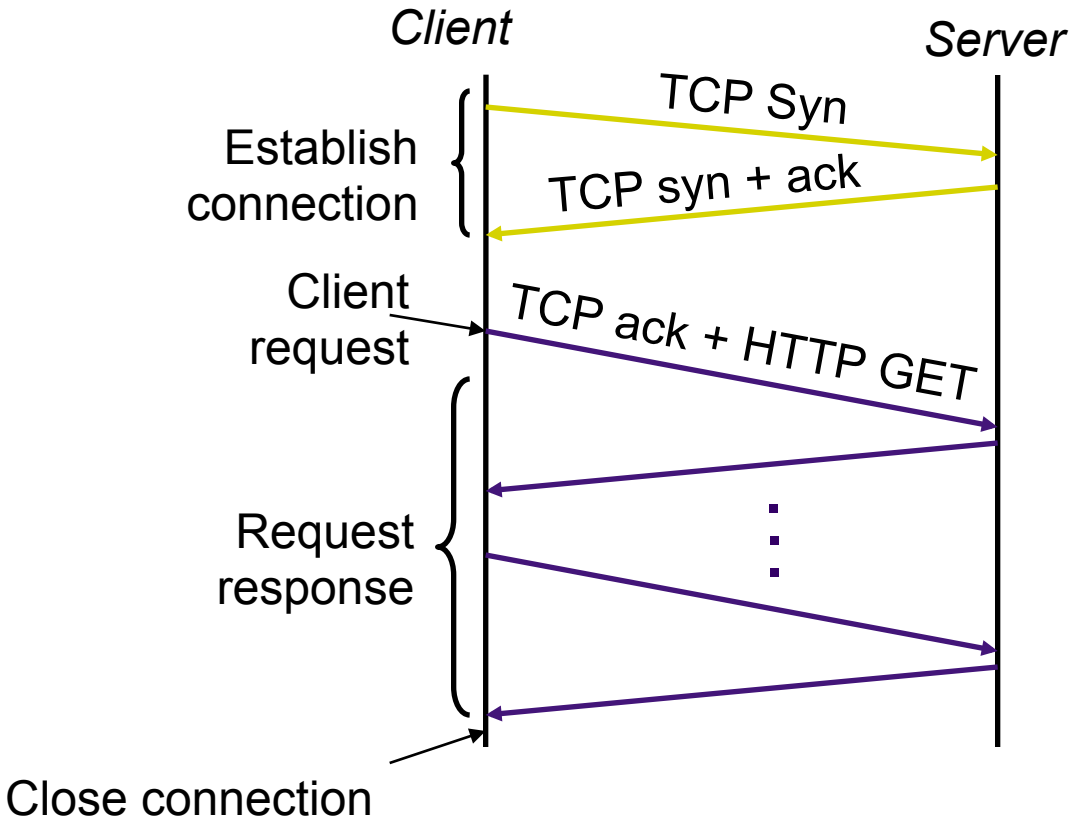
Things to know about DNS

- Steps in resolving a DNS request
  - from the viewpoint of three different hierarchies
  - make sure you can walk through the sequence of messages exchanged between different servers
- Role of caching
  - impact on performance, availability, consistency
  - repeat above walk-through with “cold” vs. “warm” cache
- Pros/cons of the design

Web and HTTP

- Web content is named using URLs
  - URLs use DNS hostnames
  - Thus, content names are tied to specific hosts
- HTTP is the protocol for exchanging information
  - Synchronous request/reply protocol
  - Runs over TCP, Port 80
  - Stateless (modulo cookies)
- Client-server architecture
  - server is “always on” and “well known”
  - clients initiate contact to server

Steps in HTTP Request/Response



Things to know about HTTP

- Steps in HTTP request/response
- Broad form of request/response messages
  - only to the level of detail covered in lecture/section
  - not details of request/response headers
- **Performance**
  - persistent *vs.* concurrent *vs.* pipelined connections
    - why they're needed; what performance benefit they offer
  - when and how caching and replication help performance

Broadcast Ethernet (Lecture 16)

- Traditional Ethernet: broadcast medium
  - dedicated *vs.* shared
- Challenge with broadcast media
  - Must avoid having multiple nodes speaking at once
  - Otherwise, collisions lead to garbled data
  - Need algorithm that determines which node can transmit
- (Old) Ethernet used *random* access protocols
  - Contrast: *a priori* partitioning among nodes

Random Access Protocols

- A node *may* send at any time
  - Contrast: node waits for its turn
- Two or more transmitting nodes ⇒ collision
  - Data lost
- Random access MAC protocol specifies:
  - How to detect or avoid collisions
  - How to recover from collisions

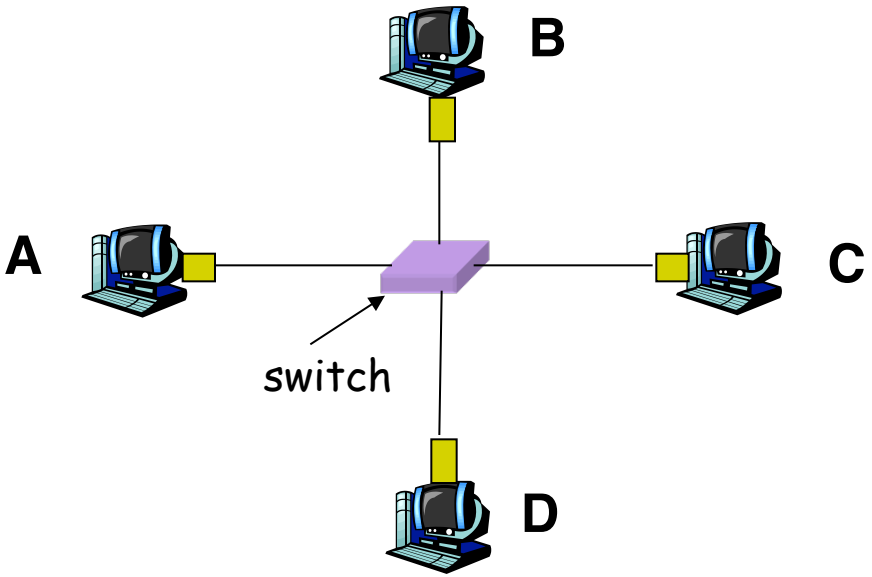
Key Ideas of Random Access

1. Carrier sense
   - Before sending, check if someone else is already sending data
2. Collision detection
   - If someone else starts talking at the same time, stop
     - *But make sure everyone knows there was a collision!*
3. Collision avoidance
   - Explicit ACK from receiver signals (lack of) collision and impending communication
4. Randomness
   - If you can't talk, wait for a random time before trying again

Broadcast Ethernet, CSMA/CD (lecture 16)

- Things to know/understand
  - why CSMA alone does not eliminate all collisions (because of nonzero propagation delay)
  - That collision detection is easy in wired (broadcast) LANs but difficult in wireless LANs (hence CSMA/CA)
  - That collision detection imposes a bound on max length of a wire and minimum length of frame

Switched Ethernet (Lecture 17, 18)



- Why? Concurrent communication
  - Host A can talk to C, while B talks to D
  - No collisions → no need for CSMA, CD
  - No constraints on link lengths, *etc.*

Switched Ethernet (Lecture 17, 18)

- What you should know about switched Ethernet
  - Ethernet frame format
  - Concept of framing and sentinel bits
  - Ethernet addresses
  - Why spanning tree and self-learning are needed
  - How the spanning tree is constructed
    - role of soft state
  - How self-learning works
    - role of caching (see HW3 problem)
  - contrast style of Ethernet vs. IP addressing and routing
    - with implications that follow
      - scalability, plug-n-play, portability, …

Switched Ethernet (Lecture 17, 18)

- How? Two pieces
  1. Build a spanning tree
     - Why? For loop-free flooding
       - Why flooding?: plug-n-play
     - How? Shortest path tree rooted at node with the lowest ID (MAC address)
  2. "Self Learning" switches
     - Why? Optimization; so switches can reach destination without flooding
     - How? If packet from A arrives on port X, switch learns to send packets to A via port X

Naming and Discovery (Lecture 18)

- What you should know
  - Naming schemes at different layers (Ethernet, IP, DNS)
    - format; what they represent; what role they play
  - How we discover and translate between names
    - DNS, ARP, DHCP
    - role of broadcast, soft state and caching

Naming

- Application layer: URLs and domain names
  - names “resources” -- hosts, content, program
- Network layer: IP addresses
  - host’s network location
- Link layer: MAC addresses
  - host identifier
- Use all three for end-to-end communication!

Discovery

- A host is “born” knowing only its MAC address

- Must discover lots of information before it can communicate with a remote host B
  - what is my IP address?
  - what is B’s IP address? (remote)
  - what is B’s MAC address? (if B is local)
  - what is my first-hop router’s address? (if B is not local)
  - …

ARP and DHCP

- Link layer discovery protocols
- Serve two functions
  - Discovery of local end-hosts
    - for communication between hosts on the same LAN
  - Bootstrap communication with remote hosts
    - what's my IP address?
    - who/where is my local DNS server?
    - who/where is my first hop router?

DHCP

- “Dynamic Host Configuration Protocol”

- A host uses DHCP to discover
  - its own IP address
  - its netmask
  - IP address(es) for its DNS name server(s)
  - IP address(es) for its first-hop “default” router(s)

DHCP: operation

1. One or more local DHCP servers maintain required information
2. Client broadcasts a DHCP discovery message
3. One or more DHCP servers responds with a DHCP "offer" message
4. Client broadcasts a DHCP request message
5. Selected DHCP server responds with an ACK

ARP: Address Resolution Protocol

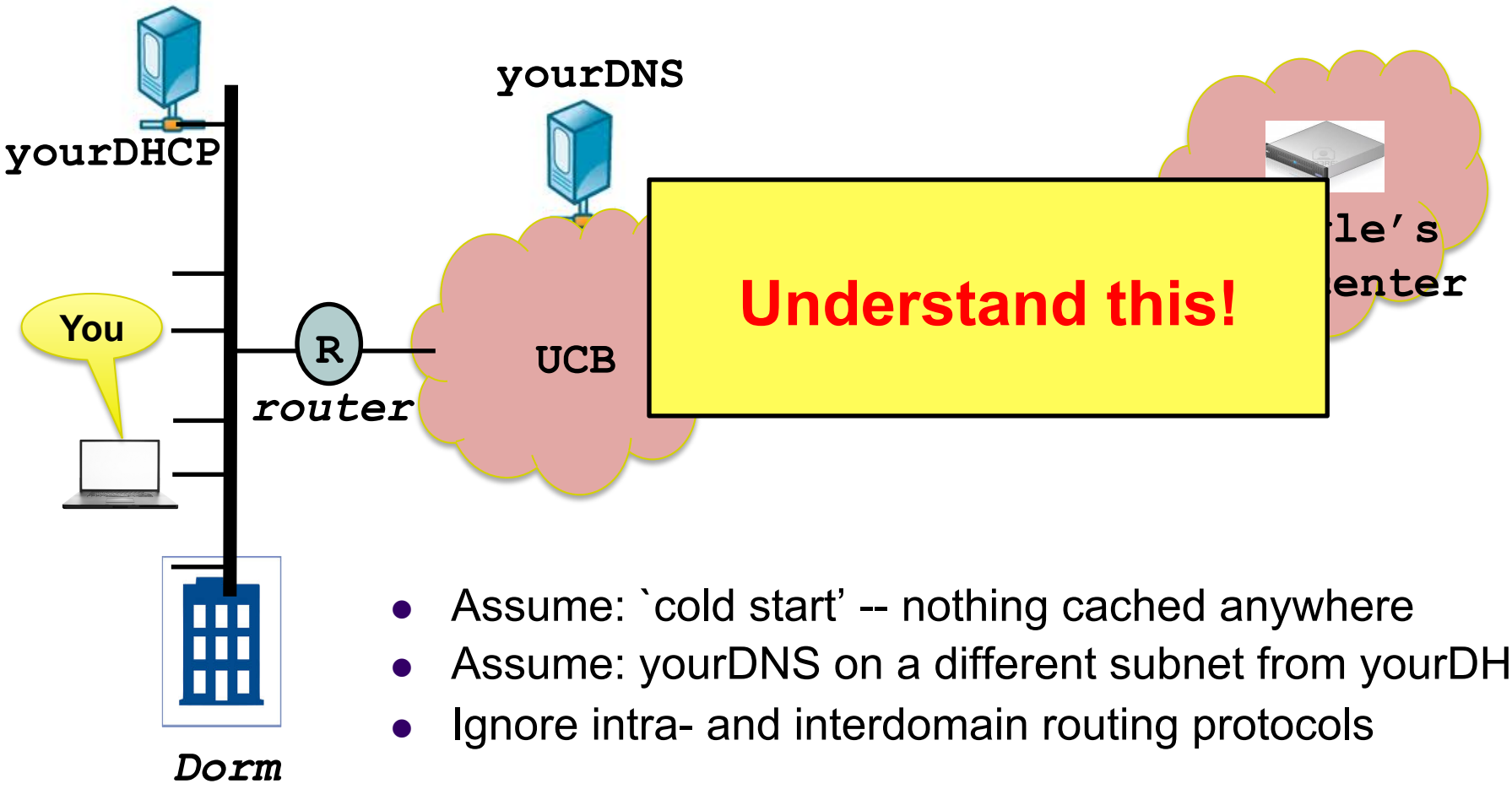
- ARP provides IP-to-MAC translation
- Every host maintains an ARP table
  - list of (IP address → MAC address) pairs
- But: what if IP address not in the table?
  - Sender broadcasts: "**Who has IP address 1.2.3.156**?"
  - Receiver responds: "**MAC address 58-23-D7-FA-20-B0**"
  - Sender caches result in its ARP table

Key Ideas in Both ARP and DHCP

- Broadcasting: used for initial bootstrap
- Caching: remember the past for a while
  - Store the information you learn to reduce overhead
  - *Key optimization for performance*
- Soft state: eventually forget the past
  - Associate a time-to-live field with the information
  - … and either refresh or discard the information
  - *Key for robustness*

Putting the pieces together (Lec 18)

Walk through the steps required to download www.google.com/index.html from your laptop



- Assume: `cold start' -- nothing cached anywhere
- Assume: yourDNS on a different subnet from yourDHCP
- Ignore intra- and interdomain routing protocols

Wireless (lecture 19)

Things you need to know:

- Properties of the medium
  - broadcast
    → collisions happen
  - but broadcast has limited range
    →no concept of a “global” collision
    → simultaneous transmissions are possible
  - can’t receive while transmitting
    → can’t detect collisions

Wireless (lecture 19)

Things you need to know:

- Properties of the medium
- Canonical scenarios
  - hidden terminal (carrier sense fails to prevent collisions)
  - exposed terminal (carrier sense needlessly limits commn.)

Wireless (lecture 19)

Things you need to know:

- Properties of the medium
- Canonical scenarios
- Techniques for collision avoidance
  - carrier sense
  - explicit request/response (RTS/CTS)
  - backoff

Wireless (lecture 19)

Things you need to know:

- Properties of the medium
- Canonical scenarios
- Techniques for collision avoidance
- How to analyze a given media access protocol that uses the above techniques
  - We'll give you the protocol rules; you analyze how (and how well) data exchange proceeds
  - You don't need to memorize protocol rules
  - e.g., Q7 on HW3

Wireless (lecture 19)

Things you need to know:

- Properties of the medium
- Canonical scenarios
- Techniques for collision avoidance
- How to analyze a given media access protocol that uses the above techniques

Things we don't expect you to know

- mathematical understanding of wireless signals (free space loss, interference, attenuation, *etc.*)
- details of specific wireless protocols (e.g., 802.11)

Datacenters (lectures 20, 21)

What you should know

- How and why DC networks are different (*vs.* WAN)
  - in terms of workload, goals, characteristics
- How traditional solutions fare in this environment
  - e.g., IP, Ethernet, TCP
- Understand Fat-Tree DC networks
  - Why “fat”; how we do addressing and routing on a fat-tree
- High level understanding of DC-TCP and pFabric
  - mostly the “why” for the approach taken

Typical datacenter architecture

- Servers organized in racks
- Each rack has a `Top of Rack' (ToR) switch
- An `aggregation fabric' interconnects ToR switches
- Connected to the outside via `core' switches

Typical datacenter traffic workload

- "North-South traffic"
  - Traffic between outside world and the datacenter
  - Primarily latency sensitive
- "East-West traffic"
  - Traffic between machines in the datacenter
  - Commn. *within* "big data" computations (e.g. Map Reduce)
  - Throughput intensive
- "Elephants and Mice"

Characteristics of a datacenter network

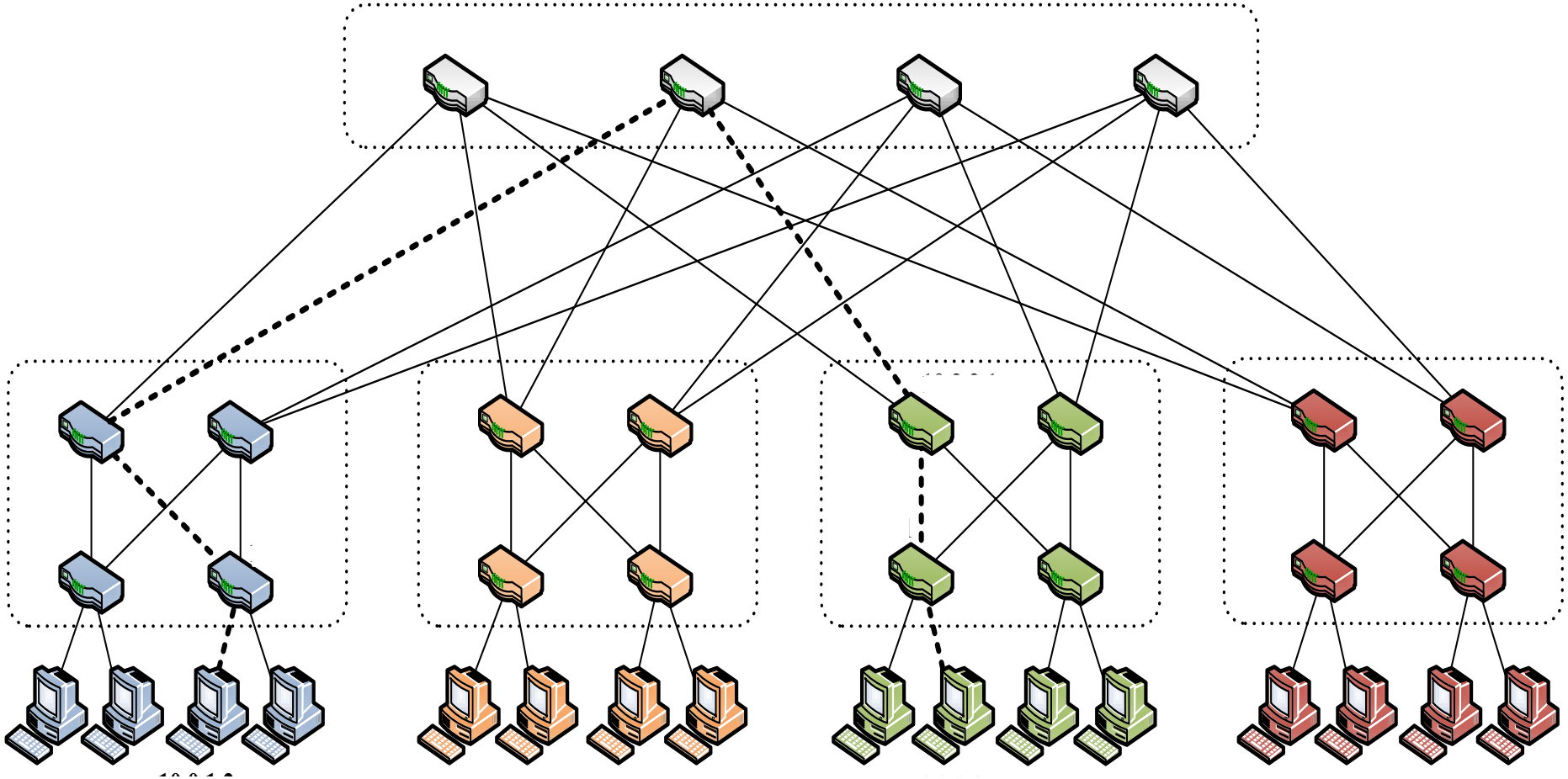
- Huge scale
- Limited geographic scope
- Limited heterogeneity
  - regular topologies, link speeds, technologies, latencies, …
- Single administrative domain
- Control over network *and* endpoints
- Control over the *placement* of traffic sources/sinks
- Control over topology (e.g., trees/fat-trees)
- Lower concern: interoperability, backward compatibility

New degrees of design freedom

Fat-Tree Topologies

- What: “scale out” approach to building a high bisection bandwidth network
  - “high bisection BW” → high BW between servers
- Why?
  - Lots of east-west traffic
    - (think shuffle in map-reduce)
  - But traditional tree topologies expensive to scale
    - (“scale up” → links near root exorbitantly expensive)
- How? Topology in which each switch has many links going “up”

E.g., “Fat Tree” Topology [Sigcomm’08]



All links have same (reasonable) speed and #ports

Routing / addressing on a Fat Tree

- Option 1: Extend DV/LS
  - With per packet/flow load balancing
  - Simple, incremental, but scales poorly
- Option 2: Topology-aware addressing
  - IP address reflects position in the topology
  - Simple, scalable, not migrateable
- Option 3: SDN w/ source routing
  - Centralized route calculation.; path vector as address
  - Simple, scalable (as SDN), not backward compatible

Datacenter Transport Designs

- Know: rationale and approach at a high level only

- E.g., DC-TCP
  - Why? Datacenters need both high throughput *and* low latency but TCP isn't good at this
  - What? Incremental change to TCP to achieve both
  - How?: two ideas
    - react early (using ECN)
    - react in proportion to the *extent* of congestion (by adapting CWND based on fraction of packets with ECN set)

Datacenter Transport Designs

- Know: rationale and approach at a high level only

- E.g., pFabric
  - Why? Minimizing flow completion time (FCT) is the ultimate goal but TCP isn't good at this
    - Why? Mice get stuck behind elephants → both finish late
  - What? Clean-slate design for low FCT
  - How?:
    - Each packet has priority value set to #remaining bytes in flow
    - Switches send highest priority / drop lowest priority / packet
    - Loose intuition: mice finish first (approx. shortest job first sched.)

IPv6 (lecture 22)

- Know: see detailed post on piazza

- Motivation: IPv4 address exhaustion
- What IPv6 does/doesn't change in overall architecture
- Details: IPv6 address and header formats
  - with rationale for changes
  - and understanding of what gets simpler/harder
- Basics of IP address assignment
  - EUI, network-interface split, *etc.*

- Note: unless stated otherwise, assume "IP" means "IPv4"

SDN (lecture 23)

- Why?: modern networks face a multitude of control requirements (access control, isolation, etc.) but we have no coherent management architecture
- How?
  - 10,000 ft view: modularity with abstractions
  - 1,000 ft view: three new abstractions
    - Forwarding: <match, action> rules
    - Control: global view of physical network + abstract network view

SDN (lecture 23)

- Why?: modern networks face a multitude of control requirements (access control, isolation, etc.) but we have no coherent management architecture
- How?
  - 10,000 ft view: modularity with abstractions
  - 1,000 ft view: three new abstractions
  - 100 ft view: decouple data and control plane
    - Data plane: switches (implement/expose forwarding abstraction)
    - Control plane: servers (implement physical/abstract network views)
      - Runs “Network OS” that queries switches to construct global physical view
      - Virtualization layer runs over NOS; translates physical←→virtual network
      - Management apps highest layer; express policy on virtual view

SDN (lecture 23)

- Why?: modern networks face a multitude of control requirements (access control, isolation, etc.) but we have no coherent management architecture
- How?
  - 10,000 ft view: modularity with abstractions
  - 1,000 ft view: three new abstractions
  - 100 ft view: decouple data and control plane
- What you should know
  - Overall idea to 100 ft. view; no details beyond Scott's slides
  - E.g., do not need to know VLANs, Traffic engg., ACLs, MPLS
  - E.g., do not need to know details of NOS, OpenFlow, Virt. layer, etc

Reminder: Attend the EECS Colloquium!

Software-Defined Networks and the Maturing of the Internet

Wednesday, December 3, 2014
306 Soda Hall (HP Auditorium)
4:00 - 5:00 pm

Nick McKeown
Professor of Computer Science and Electrical Engineering, Stanford

Final points

- On advanced topics:
  - E.g., router-assisted congestion control (RCP, RC3, ECN), datacenter designs such as DC-TCP, pFabric, SDN, IPv6
  - Figure primarily in multiple choice and design questions (Q1,Q2)
  - For most such questions: we'll provide the design goals and options; your job only to *analyze* the design we describe

- If you look over exams from previous years, note changes
  - We did not cover security
  - We covered datacenters, IPv6 and SDN in more detail

- We will hold additional OHs and post review material
  - Will post details on piazza

Thanks
&
Good luck!