Course Overview

Jennifer Rexford
COS 461: Computer Networks
Lectures: MW 10-10:50am in Architecture N101

http://www.cs.princeton.edu/courses/archive/spr12/cos461/

Course Logistics

- Last assignment
  - Posted on the class Web page
  - Due on Dean's Date (5pm Tuesday May 15)
- Final exam
  - Cumulative, emphasis on second half of the class
  - Tuesday May 22th at 1:30-3:30pm in Friend 101
- Office hours
  - Posted soon
- Also, ask questions on the mailing list

3

Key Concepts in Networking

(Exam preparation idea: look for other examples)

Some Key Concepts

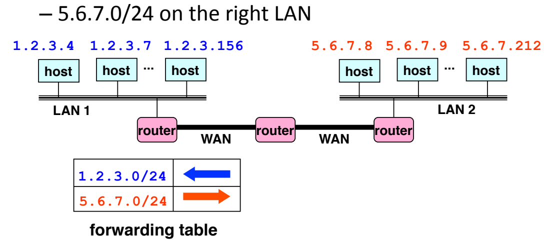
- Course was organized around protocols
  - But a small set of concepts recur in many protocols
- General CS concepts
  - Hierarchy, indirection, caching, randomization
- Networking-specific concepts
  - Soft state, layering, (de)multiplexing
  - End-to-end argument

Hierarchy

- Scalability of large systems
  - Cannot store all information everywhere
  - Cannot centrally coordinate everything
- Hierarchy to manage scale
  - Divide system into smaller pieces
- Hierarchy to divide control
  - Decentralized management
- Examples in the Internet
  - IP addresses, routing protocols, DNS, peer-to-peer

Hierarchy: IP Address Blocks

- Number related hosts from a common subnet
  - 1.2.3.0/24 on the left LAN
  - 5.6.7.0/24 on the right LAN

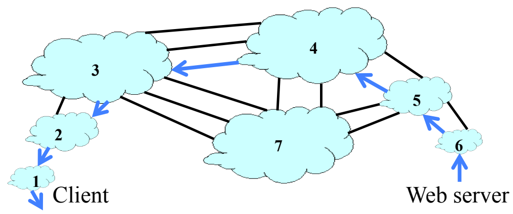


Hierarchy: IP Address Blocks

- Separation of control
  - Prefix: assigned to an institution
  - Addresses: assigned by institution to its nodes
- Who assigns prefixes?
  - Internet Corporation for Assigned Names & Numbers
  - Regional Internet Registries (RIRs)
  - Internet Service Providers (ISPs)
  - Stub networks
  - Regions within an enterprise

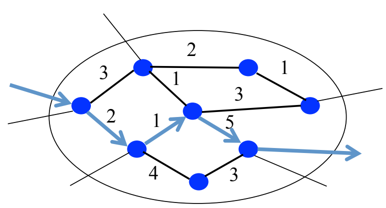
Hierarchy: Routing Protocols

- AS-level topology
  - Nodes are Autonomous Systems (ASes)
  - Edges are links and business relationships
  - Hides the detail within each AS's network



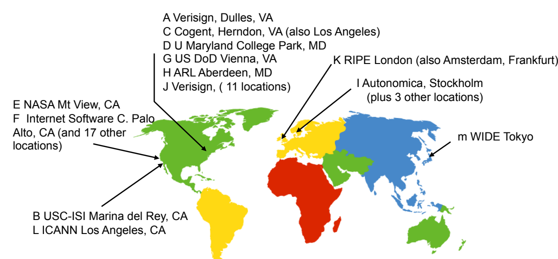
Hierarchy: Routing Protocols

- Interdomain routing ignores details in an AS
  - Routers flood information to learn the topology
  - Routers determine "next hop" to other routers…
  - By computing shortest paths based on link weights

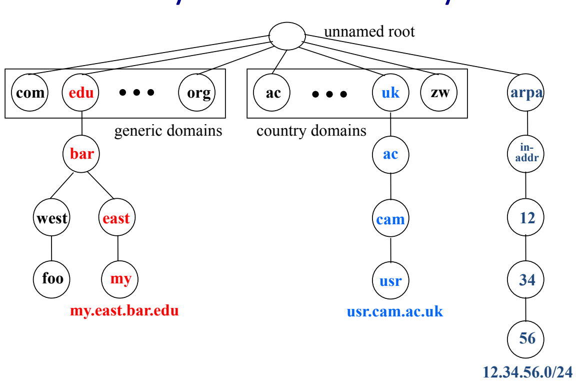


Hierarchy: Domain Name System

- 13 root servers (see http://www.root-servers.org/)
- Labeled A through M

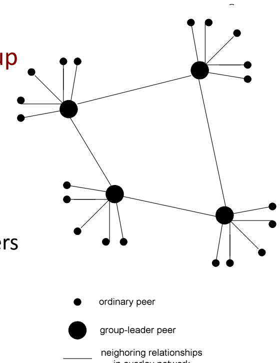


Hierarchy: Domain Name System



Hierarchy: Super Peers in KaZaA

- Each peer is either a group leader or assigned to a group leader
  - TCP connection between peer and its group leader
  - TCP connections between some pairs of group leaders
- Group leader tracks the content in all its children



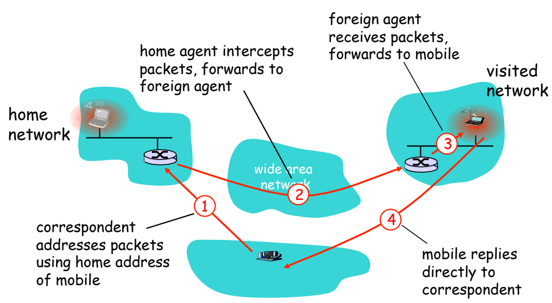
Indirection

- Referencing by name
  - Rather than the value itself
  - E.g., manipulating a variable through a pointer
- Benefits of indirection
  - Human convenience
  - Reducing overhead when things change
- Examples of indirection in the Internet
  - Names vs. addresses
  - Mobile IP

Indirection: Names vs. Addresses

- Host name to IP address
  - Mnemonic names to location-dependent addresses
  - E.g., from www.cnn.com to 64.236.16.20
  - Using the Domain Name System (DNS)
- From IP address to MAC address
  - From hierarchical global address to interface card
  - E.g., from 64.236.16.20 to 00-15-C5-49-04-A9
  - Using the Address Resolution Protocol (ARP)

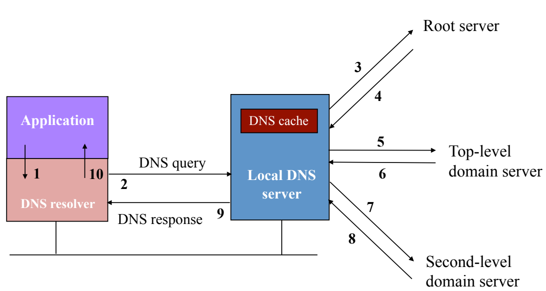
Indirection: Mobile IP



Caching

- Duplicating data stored elsewhere
  - To reduce latency for accessing the data
  - To reduce resources consumed
- Caching is often quite effective
  - Speed difference between cache and primary copy
  - Locality of reference, and small set of popular data
- Examples from the Internet
  - DNS caching, Web caching

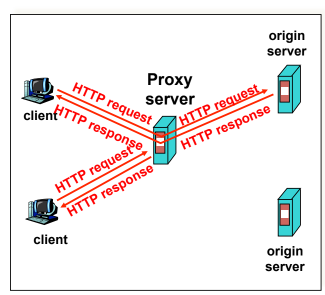
Caching: DNS Caching



18

Caching: Web Caching

- Caching location
  - Proxy cache
  - Browser cache
- Better performance
  - Lower RTT
  - Existing connection
  - Less network load

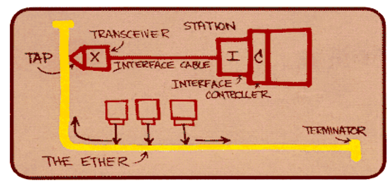


Randomization

- Distributed adaptive algorithms
  - Multiple distributed parties
  - Adapting independently
- Risk of synchronization
  - Many parties reacting at the same time
  - Leading to bad aggregate behavior
- Randomization can desynchronize
  - Ethernet back-off, Random Early Detection
- Rather than imposing centralized control

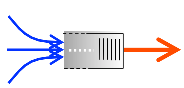
Randomization: Ethernet Back-off

- Random access: exponential back-off
  - After collision, wait random time before retrying
  - After m^{th}, choose K randomly from {0, ..., 2^m-1}
  - Wait for K*512 bit times before trying again



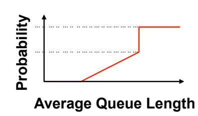
Randomization: Dropping Packets Early

- Congestion on a link
  - Eventually the queue becomes full
  - And new packets must be dropped
- Drop-tail queuing leads to bursty loss
  - Many packets encounter a full queue
  - Many TCP senders reduce their sending rates



Randomization: Dropping Packets Early

- Better to give early feedback
  - Get a few connections to slow down
  - ... before it is too late
- Random Early Detection (RED)
  - Randomly drop packets when queue (near) full
  - Drop rate increases as function of queue length



Soft State

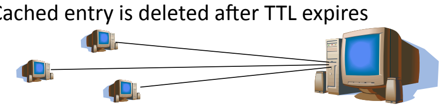
- State: stored in nodes by network protocols
  - Installed by receiver of a set-up message
  - Updated when conditions change

- Hard state: valid unless told otherwise
  - Removed by receiver of tear-down message
  - Requires error handling to deal with sender failure
- Soft state: invalid if not told to refresh
  - Periodically refreshed, removed by timeout

- Soft state reduces complexity
  - DNS caching, DHCP leases

Soft State: DNS Caching

- Cache consistency is a hard problem
  - Ensuring the cached copy is not out of date
- Strawman: explicit revocation or updates
  - Keep track of everyone who has cached information
  - If name-to-host mapping changes, update caches
- Soft state solution
  - DNS responses include a “time to live” (TTL) field
  - Cached entry is deleted after TTL expires

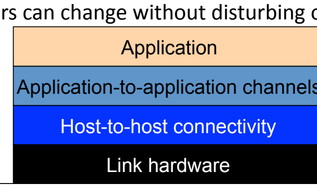


Soft State: DHCP Leases

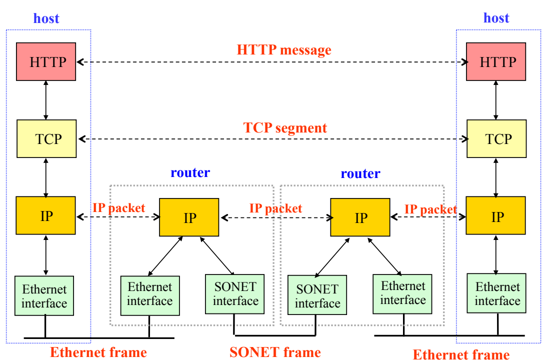
- DHCP “offer message” from the server
  - Configuration parameters (proposed IP address, mask, gateway router, DNS server, ...)
  - Lease time (the time information remains valid)
- Why is a lease time necessary?
  - Client can release address (DHCP RELEASE)
    - E.g., “ipconfig /release” or clean shutdown of computer
  - But, the host might not release the address
    - E.g., the host crashes or buggy client software
  - You don’t want address to be allocated forever

Layering: A Modular Approach

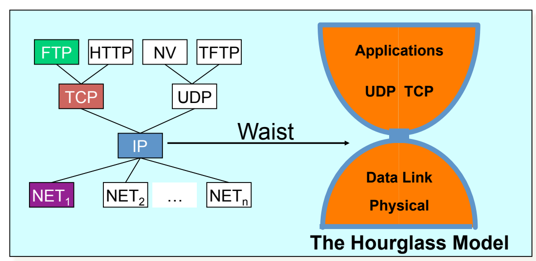
- Sub-divide the problem
  - Each layer relies on services from layer below
  - Each layer exports services to layer above
- Interface between layers defines interaction
  - Hides implementation details
  - Layers can change without disturbing other layers



Layering: Standing on Shoulders



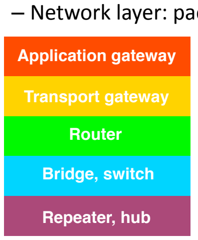
Layering: Internet Protocol Suite

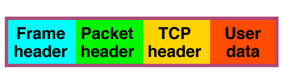


The waist facilitates interoperability

Layering: Encapsulation of Data

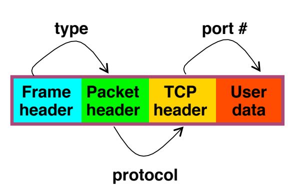
- Different devices switch different things
  - Physical layer: electrical signals (repeaters and hubs)
  - Link layer: frames (bridges and switches)
  - Network layer: packets (routers)



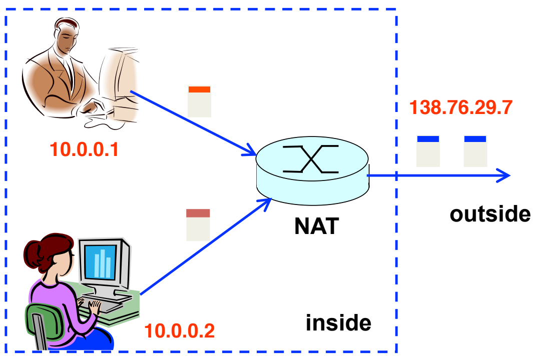


Demultiplexing

- Separating multiple streams out of one
  - Recognizing the separate streams
  - Treating the separate streams accordingly
- Examples in the Internet



(De)multiplexing: With a NAT



31

Power at the End Host

End-to-End Principle
Whenever possible, communications protocol operations should be defined to occur at the end-points of a communications system.

Programmability
With programmable end hosts, new network services can be added at any time, by anyone.

Why No Math in This Course?

- Hypothesis #1: theory not relevant to Internet
  - Body of math created for telephone networks
  - Many of these models don't work in data networks
- Hypothesis #2: too many kinds of theory
  - Queuing: statistical multiplexing works
  - Control: TCP congestion control works
  - Optimization: TCP maximizes aggregate utility
  - Game: reasoning about competing ASes
- Is networking a domain or a discipline?

What Will Happen to the Internet

No Strict Notions of Identity


"On the Internet, nobody knows you're a dog."

- Leads to
  - Spam
  - Spoofing
  - Denial-of-service
  - Route hijacking

Protocols Designed Based on Trust

- That you don't spoof your addresses
  - MAC spoofing, IP address spoofing, spam, …
- That port numbers correspond to applications
  - Rather than being arbitrary, meaningless numbers
- That you adhere to the protocol
  - Ethernet exponential back-off after a collision
  - TCP additive increase, multiplicative decrease
- That protocol specifications are public
  - So others can build interoperable implementations

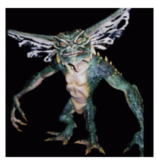
Nobody in Charge

- Traffic traverses many Autonomous Systems
  - Who's fault is it when things go wrong?
  - How do you upgrade functionality?
- Implicit trust in the end host
  - What if some hosts violate congestion control?
- Anyone can add any application
  - Whether or not it is legal, moral, good, etc.
- Spans many countries
  - So no one government can be in charge

Challenging New Requirements

- Disseminating data
- Mobile, multi-homed hosts
- Sometimes-connected hosts
- Large number of hosts
- Real-time applications

The Internet of the Future



- Can we fix what ails the Internet
  - Security, performance, reliability
  - Upgradability, managability
  - <Your favorite gripe here>
- Without throwing out baby with bathwater
  - Ease of adding new hosts
  - Ease of adding new services
  - Ease of adding new link technologies
- An open technical and policy question…

Thank You!