CS162
Operating Systems and Systems Programming
Lecture 21

Networking

November 12, 2008

Prof. John Kubiatowicz

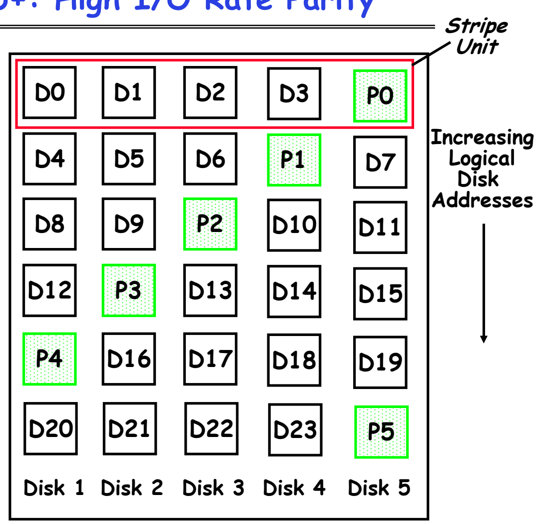
http://inst.eecs.berkeley.edu/~cs162

Review: File System Caching

- Delayed Writes: Writes to files not immediately sent out to disk
  - Instead, `write()` copies data from user space buffer to kernel buffer (in cache)
    - » Enabled by presence of buffer cache: can leave written file blocks in cache for a while
    - » If some other application tries to read data before written to disk, file system will read from cache
  - Flushed to disk periodically (e.g. in UNIX, every 30 sec)
  - Advantages:
    - » Disk scheduler can efficiently order lots of requests
    - » Disk allocation algorithm can be run with correct size value for a file
    - » Some files need never get written to disk! (e..g temporary scratch files written /tmp often don't exist for 30 sec)
  - Disadvantages
    - » What if system crashes before file has been written out?
    - » Worse yet, what if system crashes before a directory file has been written out? (lose pointer to inode!)

Review: RAID 5+: High I/O Rate Parity

- Data stripped across multiple disks
  - Successive blocks stored on successive (non-parity) disks
  - Increased bandwidth over single disk
- Parity block (in green) constructed by XORing data bocks in stripe
  - $P0=D0\oplus D1\oplus D2\oplus D3$
  - Can destroy any one disk and still reconstruct data
  - Suppose D3 fails, then can reconstruct: $D3=D0\oplus D1\oplus D2\oplus P0$



- Later in term: talk about spreading information widely across internet for durability.

Goals for Today

- Authorization
- Networking
  - Broadcast
  - Point-to-Point Networking
  - Routing
  - Internet Protocol (IP)

Note: Some slides and/or pictures in the following are adapted from slides ©2005 Silberschatz, Galvin, and Gagne. Many slides generated from my lecture notes by Kubiatowicz.

Authorization: Who Can Do What?

- How do we decide who is authorized to do actions in the system?
- **Access Control Matrix:** contains all permissions in the system
  - Resources across top
    - » Files, Devices, etc…
  - Domains in columns
    - » A domain might be a user or a group of users
    - » E.g. above: User D3 can read F2 or execute F3
  - In practice, table would be huge and sparse!

domain \ object	F_1	F_2	F_3	printer
D_1	read		read	
D_2				print
D_3		read	execute	
D_4	read write		read write	

Authorization: Two Implementation Choices

- **Access Control Lists:** store permissions with object
  - Still might be lots of users!
  - UNIX limits each file to: r,w,x for owner, group, world
    - » More recent systems allow definition of groups of users and permissions for each group
  - ACLs allow easy changing of an object's permissions
    - » Example: add Users C, D, and F with rw permissions
  - *Requires mechanisms to prove identity*
- **Capability List:** each process tracks which objects it has permission to touch
  - Consider page table: Each process has list of pages it has access to, not each page has list of processes …
    - » Capability list easy to change/augment permissions
    - » E.g.: you are promoted to system administrator and should be given access to all system files
  - Implementation: Capability like a "Key" for access
    - » Example: cryptographically secure (non-forgeable) chunk of data that can be exchanged for access

Authorization: Combination Approach

- Users have capabilities, called "groups" or "roles"
  - Everyone with particular group access is "equivalent" when accessing group resource
  - Like passport (which gives access to country of origin)
- Objects have ACLs
  - ACLs can refer to users or groups
  - Change object permissions object by modifying ACL
  - Change broad user permissions via changes in group membership
  - Possessors of proper credentials get access

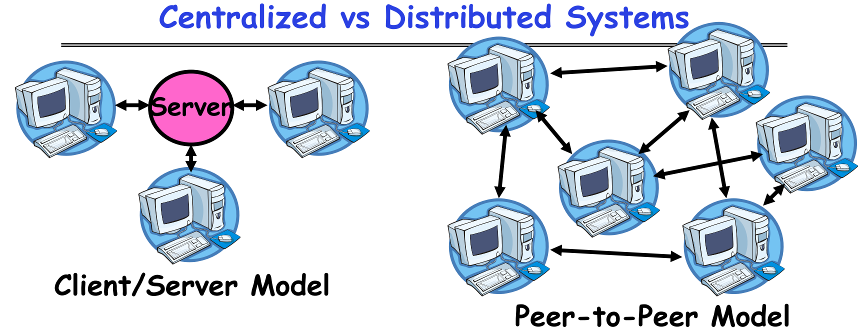
Authorization: How to Revoke?

- How does one revoke someone's access rights to a particular object?
  - Easy with ACLs: just remove entry from the list
  - Takes effect immediately since the ACL is checked on each object access
- Harder to do with capabilities since they aren't stored with the object being controlled:
  - Not so bad in a single machine: could keep all capability lists in a well-known place (e.g., the OS capability table).
  - Very hard in distributed system, where remote hosts may have crashed or may not cooperate (more in a future lecture)

Revoking Capabilities

- Various approaches to revoking capabilities:
  - Put expiration dates on capabilities and force reacquisition
  - Put epoch numbers on capabilities and revoke all capabilities by bumping the epoch number (which gets checked on each access attempt)
  - Maintain back pointers to all capabilities that have been handed out (Tough if capabilities can be copied)
  - Maintain a revocation list that gets checked on every access attempt

Centralized vs Distributed Systems



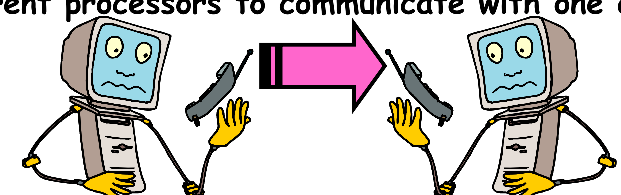
- **Centralized System:** System in which major functions are performed by a single physical computer
  - Originally, everything on single computer
  - Later: client/server model
- **Distributed System:** physically separate computers working together on some task
  - Early model: multiple servers working together
    - » Probably in the same room or building
    - » Often called a "cluster"
  - Later models: peer-to-peer/wide-spread collaboration

Distributed Systems: Motivation/Issues

- Why do we want distributed systems?
  - Cheaper and easier to build lots of simple computers
  - Easier to add power incrementally
  - Users can have complete control over some components
  - Collaboration: Much easier for users to collaborate through network resources (such as network file systems)
- The *promise* of distributed systems:
  - Higher availability: one machine goes down, use another
  - Better durability: store data in multiple locations
  - More security: each piece easier to make secure
- Reality has been disappointing
  - Worse availability: depend on every machine being up
    - » Lamport: "a distributed system is one where I can't do work because some machine I've never heard of isn't working!"
  - Worse reliability: can lose data if any machine crashes
  - Worse security: anyone in world can break into system
- Coordination is more difficult
  - Must coordinate multiple copies of shared state information (using only a network)
  - What would be easy in a centralized system becomes a lot more difficult

Distributed Systems: Goals/Requirements

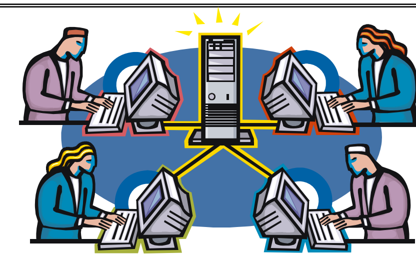
- **Transparency:** the ability of the system to mask its complexity behind a simple interface
- Possible transparencies:
  - **Location:** Can't tell where resources are located
  - **Migration:** Resources may move without the user knowing
  - **Replication:** Can't tell how many copies of resource exist
  - **Concurrency:** Can't tell how many users there are
  - **Parallelism:** System may speed up large jobs by spliting them into smaller pieces
  - **Fault Tolerance**: System may hide varoius things that go wrong in the system
- Transparency and collaboration require some way for different processors to communicate with one another



Administrivia

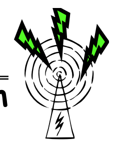
- Exam reminders:
  - MIDTERM II: Wednesday Dec 3rd
    - » All material from last midterm and up to Monday 12/1
    - » Lectures #13 – 27
  - Final Exam
    - » Thurs Dec 18th, 5:00-8:00pm, Bechtel Auditorium
    - » All Material
- Moved deadline for Project 3 to Friday, 11/14
  - Now due Friday at Midnight
  - Project 3 is the trickiest to get right
- Note: There will *NOT* be an extension for Project 4
  - Timing is too tight
  - Plan accordingly

Networking Definitions

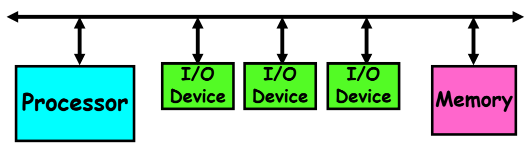


- **Network:** physical connection that allows two computers to communicate
- **Packet:** unit of transfer, sequence of bits carried over the network
  - Network carries packets from one CPU to another
  - Destination gets interrupt when packet arrives
- **Protocol:** agreement between two parties as to how information is to be transmitted

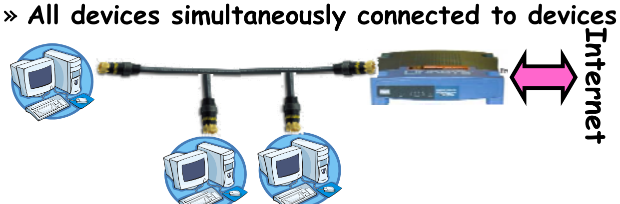
Broadcast Networks



- **Broadcast Network:** Shared Communication Medium

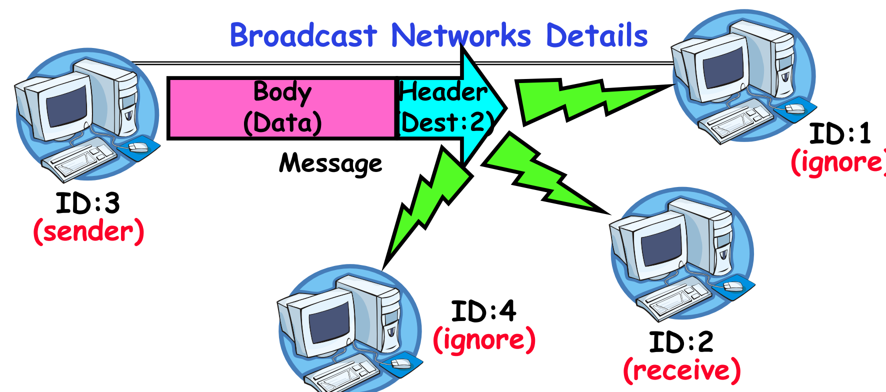


  - Shared Medium can be a set of wires
    - » Inside a computer, this is called a bus
    - » All devices simultaneously connected to devices



  - Originally, Ethernet was a broadcast network
    - » All computers on local subnet connected to one another
  - More examples (wireless: medium is air): cellular phones, GSM GPRS, EDGE, CDMA 1xRTT, and 1EvDO

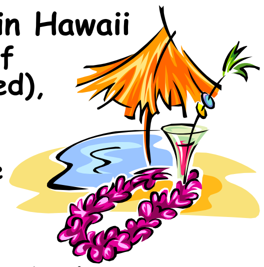
Broadcast Networks Details



- **Delivery:** When you broadcast a packet, how does a receiver know who it is for? (packet goes to everyone!)
  - Put header on front of packet: [Destination | Packet]
  - Everyone gets packet, discards if not the target
  - In Ethernet, this check is done in hardware
    - » No OS interrupt if not for particular destination
  - This is layering: we're going to build complex network protocols by layering on top of the packet

Broadcast Network Arbitration

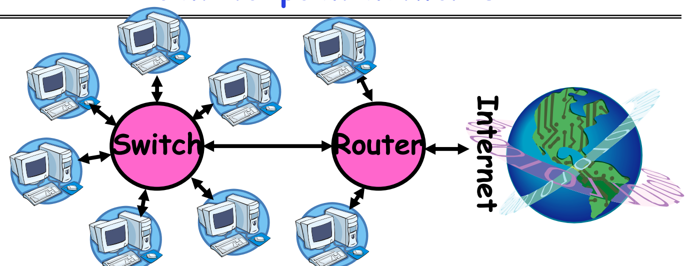
- **Arbitration:** Act of negotiating use of shared medium
  - What if two senders try to broadcast at same time?
  - Concurrent activity but can't use shared memory to coordinate!
- Aloha network (70's): packet radio within Hawaii
  - Blind broadcast, with checksum at end of packet. If received correctly (not garbled), send back an acknowledgement. If not received correctly, discard.
    - » Need checksum anyway – in case airplane flies overhead
  - Sender waits for a while, and if doesn't get an acknowledgement, re-transmits.
  - If two senders try to send at same time, both get garbled, both simply re-send later.
  - Problem: Stability: what if load increases?
    - » More collisions ⇒ less gets through ⇒more resent ⇒ more load… ⇒ More collisions…
    - » Unfortunately: some sender may have started in clear, get scrambled without finishing



Carrier Sense, Multiple Access/Collision Detection

- Ethernet (early 80's): first practical local area network
  - It is the most common LAN for UNIX, PC, and Mac
  - Use wire instead of radio, but still broadcast medium
- Key advance was in arbitration called CSMA/CD: Carrier sense, multiple access/collision detection
  - **Carrier Sense:** don't send unless idle
    - » Don't mess up communications already in process
  - **Collision Detect:** sender checks if packet trampled.
    - » If so, abort, wait, and retry.
  - **Backoff Scheme:** Choose wait time before trying again
- How long to wait after trying to send and failing?
  - What if everyone waits the same length of time? Then, they all collide again at some time!
  - Must find way to break up shared behavior with nothing more than shared communication channel
- Adaptive randomized waiting strategy:
  - **Adaptive and Random:** First time, pick random wait time with some initial mean. If collide again, pick random value from bigger mean wait time. Etc.
  - Randomness is important to decouple colliding senders
  - Scheme figures out how many people are trying to send!

Point-to-point networks

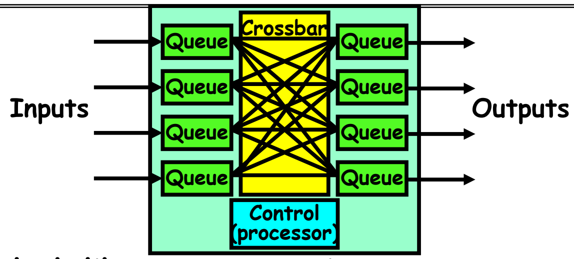


- Why have a shared bus at all? Why not simplify and only have point-to-point links + routers/switches?
  - Originally wasn't cost-effective
  - Now, easy to make high-speed switches and routers that can forward packets from a sender to a receiver.
- **Point-to-point network:** a network in which every physical wire is connected to only two computers
- **Switch:** a bridge that transforms a shared-bus (broadcast) configuration into a point-to-point network.
- **Router:** a device that acts as a junction between two networks to transfer data packets among them.

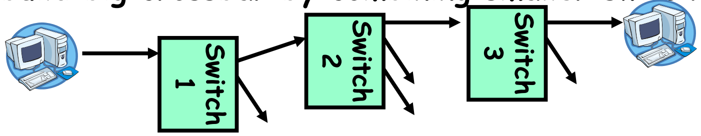
Point-to-Point Networks Discussion

- Advantages:
  - Higher link performance
    - » Can drive point-to-point link faster than broadcast link since less capacitance/less echoes (from impedance mismatches)
  - Greater aggregate bandwidth than broadcast link
    - » Can have multiple senders at once
  - Can add capacity incrementally
    - » Add more links/switches to get more capacity
  - Better fault tolerance (as in the Internet)
  - Lower Latency
    - » No arbitration to send, although need buffer in the switch
- Disadvantages:
  - More expensive than having everyone share broadcast link
  - However, technology costs now much cheaper
- Examples
  - ATM (asynchronous transfer mode)
    - » The first commercial point-to-point LAN
    - » Inspiration taken from telephone network
  - Switched Ethernet
    - » Same packet format and signaling as broadcast Ethernet, but only two machines on each ethernet.

Point-to-Point Network design

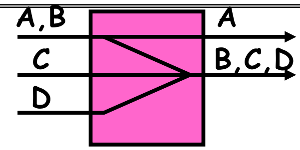


- Switches look like computers: inputs, memory, outputs
  - In fact probably contains a processor
- Function of switch is to forward packet to output that gets it closer to destination
- Can build big crossbar by combining smaller switches



- Can perform broadcast if necessary

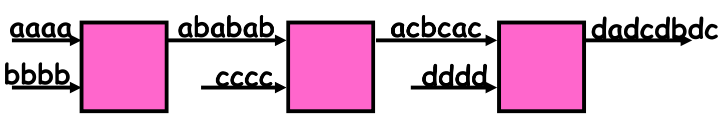
Flow control options



- What if everyone sends to the same output?
  - Congestion—packets don't flow at full rate
- In general, what if buffers fill up?
  - Need flow control policy
- Option 1: no flow control. Packets get dropped if they arrive and there's no space
  - If someone sends a lot, they are given buffers and packets from other senders are dropped
  - Internet actually works this way
- Option 2: Flow control between switches
  - When buffer fills, stop inflow of packets
  - Problem: what if path from source to destination is completely unused, but goes through some switch that has buffers filled up with unrelated traffic?

Flow Control (con't)

- Option 3: Per-flow flow control.
  - Allocate a separate set of buffers to each end-to-end stream and use separate "don't send me more" control on each end-to-end stream



- Problem: fairness
  - Throughput of each stream is entirely dependent on topology, and relationship to bottleneck
- Automobile Analogy
  - At traffic jam, one strategy is merge closest to the bottleneck
    - » Why people get off at one exit, drive 50 feet, merge back into flow
    - » Ends up slowing everybody else a huge emount
  - Also why have control lights at on-ramps
    - » Try to keep from injecting more cars than capacity of road (and thus avoid congestion)

The Internet Protocol: "IP"

- The Internet is a large network of computers spread across the globe
  - According to the Internet Systems Consortium, there were over 490 million computers as of July 2007
  - In principle, every host can speak with every other one under the right circumstances
- IP Packet: a network packet on the internet
- IP Address: a 32-bit integer used as the destination of an IP packet
  - Often written as four dot-separated integers, with each integer from 0—255 (thus representing 8x4=32 bits)
  - Example CS file server is: 169.229.60.83 ≡ 0xA9E53C53
- Internet Host: a computer connected to the Internet
  - Host has one or more IP addresses used for routing
    - » Some of these may be private and unavailable for routing
  - Not every computer has a unique IP address
    - » Groups of machines may share a single IP address
    - » In this case, machines have private addresses behind a "Network Address Translation" (NAT) gateway

Address Subnets

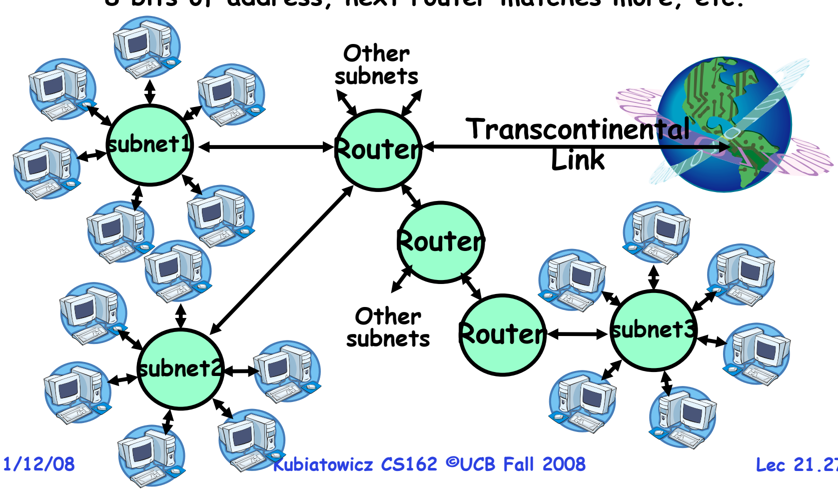
- Subnet: A network connecting a set of hosts with related destination addresses
- With IP, all the addresses in subnet are related by a prefix of bits
  - Mask: The number of matching prefix bits
    - » Expressed as a single value (e.g., 24) or a set of ones in a 32-bit value (e.g., 255.255.255.0)
- A subnet is identified by 32-bit value, with the bits which differ set to zero, followed by a slash and a mask
  - Example: 128.32.131.0/24 designates a subnet in which all the addresses look like 128.32.131.XX
  - Same subnet: 128.32.131.0/255.255.255.0
- Difference between subnet and complete network range
  - Subnet is always a subset of address range
  - Once, subnet meant single physical broadcast wire; now, less clear exactly what it means (virtualized by switches)

Address Ranges in IP

- IP address space divided into prefix-delimited ranges:
  - Class A: NN.0.0.0/8
    - » NN is 1–126 (126 of these networks)
    - » 16,777,214 IP addresses per network
    - » 10.xx.yy.zz is private
    - » 127.xx.yy.zz is loopback
  - Class B: NN.MM.0.0/16
    - » NN is 128–191, MM is 0-255 (16,384 of these networks)
    - » 65,534 IP addresses per network
    - » 172.[16-31].xx.yy are private
  - Class C: NN.MM.LL.0/24
    - » NN is 192–223, MM and LL 0-255 (2,097,151 of these networks)
    - » 254 IP addresses per networks
    - » 192.168.xx.yy are private
- Address ranges are often owned by organizations
  - Can be further divided into subnets

Hierarchical Networking: The Internet

- How can we build a network with millions of hosts?
  - Hierarchy! Not every host connected to every other one
  - Use a network of Routers to connect subnets together
    - » Routing is often by prefix: e.g. first router matches first 8 bits of address, next router matches more, etc.

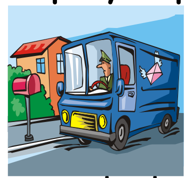


Simple Network Terminology

- Local-Area Network (LAN) – designed to cover small geographical area
  - Multi-access bus, ring, or star network
  - Speed ≈ 10 – 1000 Megabits/second
  - Broadcast is fast and cheap
  - In small organization, a LAN could consist of a single subnet. In large organizations (like UC Berkeley), a LAN contains many subnets
- Wide-Area Network (WAN) – links geographically separated sites
  - Point-to-point connections over long-haul lines (often leased from a phone company)
  - Speed ≈ 1.544 – 45 Megabits/second
  - Broadcast usually requires multiple messages

Routing

- Routing: the process of forwarding packets hop-by-hop through routers to reach their destination
  - Need more than just a destination address!
    - » Need a path
  - Post Office Analogy:
    - » Destination address on each letter is not sufficient to get it to the destination
    - » To get a letter from here to Florida, must route to local post office, sorted and sent on plane to somewhere in Florida, be routed to post office, sorted and sent with carrier who knows where street and house is…



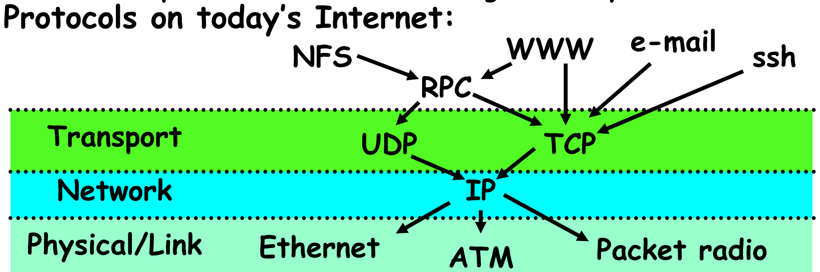
- Internet routing mechanism: routing tables
  - Each router does table lookup to decide which link to use to get packet closer to destination
  - Don't need 4 billion entries in table: routing is by subnet
  - Could packets be sent in a loop? Yes, if tables incorrect
- Routing table contains:
  - Destination address range → output link closer to destination
  - Default entry (for subnets without explicit entries)

Setting up Routing Tables

- How do you set up routing tables?
  - Internet has no centralized state!
    - » No single machine knows entire topology
    - » Topology constantly changing (faults, reconfiguration, etc)
  - Need dynamic algorithm that acquires routing tables
    - » Ideally, have one entry per subnet or portion of address
    - » Could have "default" routes that send packets for unknown subnets to a different router that has more information
- Possible algorithm for acquiring routing table
  - Routing table has "cost" for each entry
    - » Includes number of hops to destination, congestion, etc.
    - » Entries for unknown subnets have infinite cost
  - Neighbors periodically exchange routing tables
    - » If neighbor knows cheaper route to a subnet, replace your entry with neighbors entry (+1 for hop to neighbor)
- In reality:
  - Internet has networks of many different scales
  - Different algorithms run at different scales

Network Protocols

- **Protocol:** Agreement between two parties as to how information is to be transmitted
  - Example: system calls are the protocol between the operating system and application
  - Networking examples: many levels
    - » Physical level: mechanical and electrical network (e.g. how are 0 and 1 represented)
    - » Link level: packet formats/error control (for instance, the CSMA/CD protocol)
    - » Network level: network routing, addressing
    - » Transport Level: reliable message delivery
- Protocols on today's Internet:



Conclusion

- Network: physical connection that allows two computers to communicate
  - Packet: sequence of bits carried over the network
- **Broadcast Network:** Shared Communication Medium
  - Transmitted packets sent to all receivers
  - Arbitration: act of negotiating use of shared medium
    - » Ethernet: Carrier Sense, Multiple Access, Collision Detect
- **Point-to-point network:** a network in which every physical wire is connected to only two computers
  - Switch: a bridge that transforms a shared-bus (broadcast) configuration into a point-to-point network.
- **Protocol:** Agreement between two parties as to how information is to be transmitted
- Internet Protocol (IP)
  - Used to route messages through routes across globe
  - 32-bit addresses, 16-bit ports