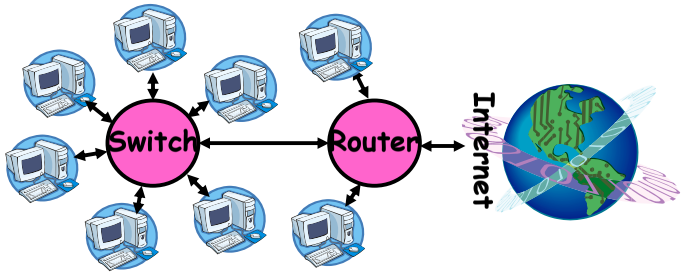
CS162
Operating Systems and Systems Programming
Lecture 22

Networking II

November 17, 2008
Prof. John Kubiatowicz
http://inst.eecs.berkeley.edu/~cs162

Review: Point-to-point networks



- **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.
- **Hub:** a multiport device that acts like a repeater broadcasting from each input to every output
- **Router:** a device that acts as a junction between two networks to transfer data packets among them.

Review: 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)

Goals for Today

- Networking
  - Routing
  - Naming
  - Protocols
  - Reliable Messaging

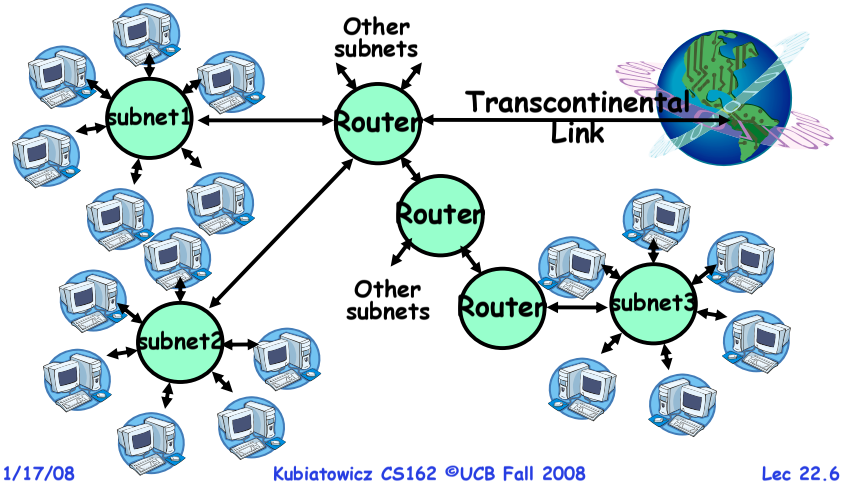
Note: Some slides and/or pictures in the following are adapted from slides ©2005 Silberschatz, Galvin, and Gagne. Many slides generated from my lecture notes by Kubiatowicz.

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

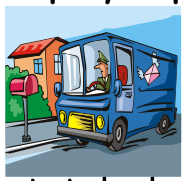
Hierarchical Networking (The Internet)

- 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

- 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

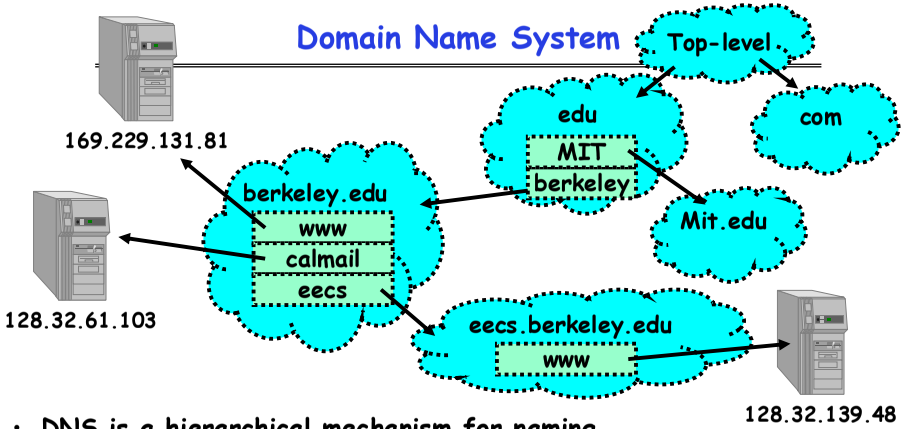
- How do you set up routing tables?
  - Internet has no centralized state!
    - » No single machine knows entire topology
    - » Topology constantly changing (faults, reconfiguration, etc)
  - Need dynamic algorithm that acquires routing tables
    - » Ideally, have one entry per subnet or portion 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
    - » Global scale: BGP (Border Gateway Protocol), others
    - » Local scale: OSPF (Open Shortest Path First), others

Naming in the Internet

Name ⇒ Address

- How to map human-readable names to IP addresses?
  - E.g. www.berkeley.edu ⇒ 128.32.139.48
  - E.g. www.google.com ⇒ different addresses depending on location, and load
- Why is this necessary?
  - IP addresses are hard to remember
  - IP addresses change:
    - » Say, Server 1 crashes gets replaced by Server 2
    - » Or – google.com handled by different servers
- Mechanism: Domain Naming System (DNS)

Domain Name System



- DNS is a hierarchical mechanism for naming
  - Name divided in domains, right to left: www.eecs.berkeley.edu
- Each domain owned by a particular organization
  - Top level handled by ICANN (Internet Corporation for Assigned Numbers and Names)
  - Subsequent levels owned by organizations
- Resolution: series of queries to successive servers
- Caching: queries take time, so results cached for period of time

How Important is Correct Resolution?

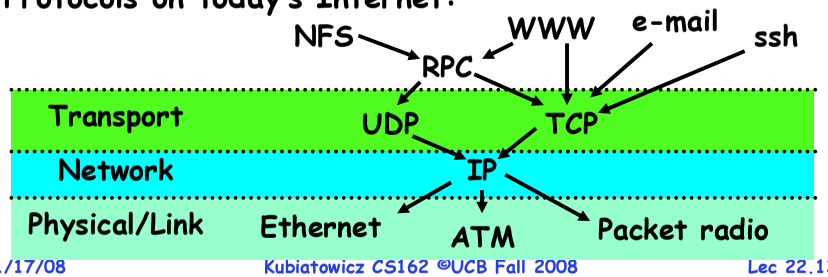
- If attacker manages to give incorrect mapping:
  - Can get someone to route to server, thinking that they are routing to a different server
    - » Get them to log into "bank" – give up username and password
- Is DNS Secure?
  - Definitely a weak link
    - » What if "response" returned from different server than original query?
    - » Get person to use incorrect IP address!
  - Attempt to avoid substitution attacks:
    - » Query includes random number which must be returned
- This summer (July 2008), hole in DNS security located!
  - Dan Kaminsky (security researcher) discovered an attack that broke DNS globally
    - » One person in an ISP convinced to load particular web page, then *all* users of that ISP end up pointing at wrong address
  - High profile, highly advertised need for patching DNS
    - » Big press release, lots of mystery
    - » Security researchers told no speculation until patches applied

Administrivia

- Projects:
  - Project 4 design document due Monday, November 24th
- MIDTERM II: Monday Dec 3rd
  - Location: 10 Evans, 5:30pm – 8:30pm
  - Topics:
    - » All material from last midterm and up to Monday 12/1
    - » Lectures #13 – 26
    - » One cheat sheet (both sides)
- Final Exam
  - Thursday, Dec 18th, 8:00-11:00am
  - Topics: All Material except last lecture (freebie)
  - Two Cheat sheets.
- Final Topics: Any suggestions?
  - Please send them to me…

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:



Network Layering

- **Layering:** building complex services from simpler ones
  - Each layer provides services needed by higher layers by utilizing services provided by lower layers
- The physical/link layer is pretty limited
  - Packets are of limited size (called the "Maximum Transfer Unit or MTU: often 200-1500 bytes in size)
  - Routing is limited to within a physical link (wire) or perhaps through a switch
- Our goal in the following is to show how to construct a secure, ordered, message service routed to anywhere:

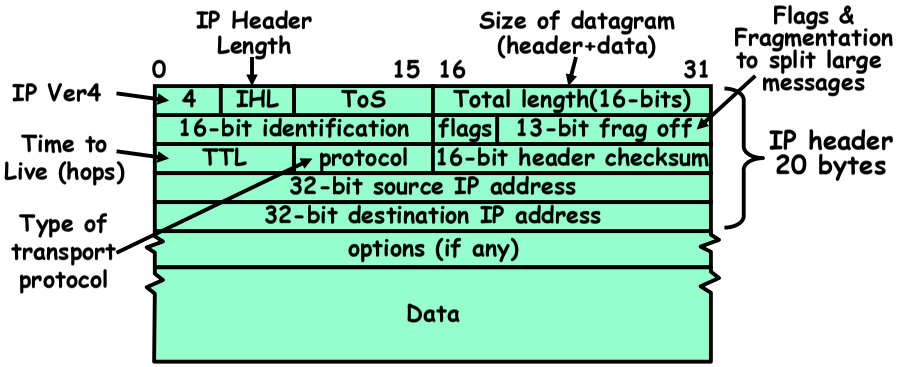
Physical Reality: Packets	Abstraction: Messages
Limited Size	Arbitrary Size
Unordered (sometimes)	Ordered
Unreliable	Reliable
Machine-to-machine	Process-to-process
Only on local area net	Routed anywhere
Asynchronous	Synchronous
Insecure	Secure

Building a messaging service

- Handling Arbitrary Sized Messages:
  - Must deal with limited physical packet size
  - Split big message into smaller ones (called fragments)
    - » Must be reassembled at destination
  - Checksum computed on each fragment or whole message
- Internet Protocol (IP): Must find way to send packets to arbitrary destination in network
  - Deliver messages unreliably ("best effort") from one machine in Internet to another
  - Since intermediate links may have limited size, must be able to fragment/reassemble packets on demand
  - Includes 256 different "sub-protocols" build on top of IP
    - » Examples: ICMP(1), TCP(6), UDP (17), IPSEC(50,51)

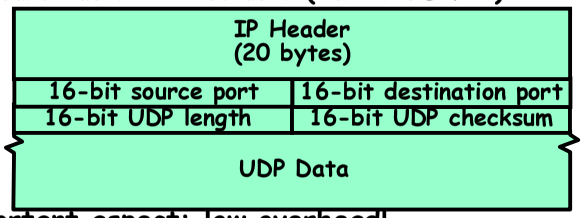
IP Packet Format

- IP Packet Format:



Building a messaging service

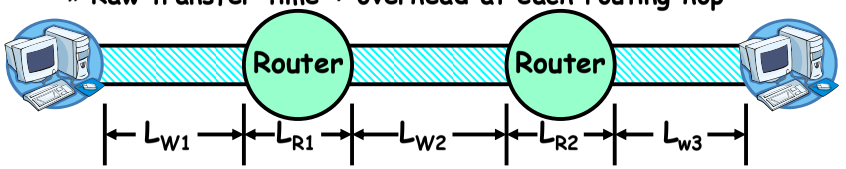
- Process to process communication
  - Basic routing gets packets from machine→machine
  - What we really want is routing from process→process
    - » Add "ports", which are 16-bit identifiers
    - » A communication channel (connection) defined by 5 items: [source addr, source port, dest addr, dest port, protocol]
- UDP: The Unreliable Datagram Protocol
  - Layered on top of basic IP (IP Protocol 17)
    - » Datagram: an unreliable, unordered, packet sent from source user → dest user (Call it UDP/IP)



  - Important aspect: low overhead!
    - » Often used for high-bandwidth video streams
    - » Many uses of UDP considered "anti-social" – none of the "well-behaved" aspects of (say) TCP/IP

Performance Considerations

- Before we continue, need some performance metrics
  - Overhead: CPU time to put packet on wire
  - Throughput: Maximum number of bytes per second
    - » Depends on "wire speed", but also limited by slowest router (routing delay) or by congestion at routers
  - Latency: time until first bit of packet arrives at receiver
    - » Raw transfer time + overhead at each routing hop



- Contributions to Latency
  - Wire latency: depends on speed of light on wire
    - » about 1–1.5 ns/foot
  - Router latency: depends on internals of router
    - » Could be < 1 ms (for a good router)
    - » Question: can router handle full wire throughput?

Sample Computations

- E.g.: Ethernet within Soda
  - Latency: speed of light in wire is 1.5ns/foot, which implies latency in building < 1 µs (if no routers in path)
  - Throughput: 10-1000Mb/s
  - Throughput delay: packet doesn't arrive until all bits
    - » So: 4KB/100Mb/s = 0.3 milliseconds (same order as disk!)
- E.g.: ATM within Soda
  - Latency (same as above, assuming no routing)
  - Throughput: 155Mb/s
  - Throughput delay: 4KB/155Mb/s = 200µ
- E.g.: ATM cross-country
  - Latency (assuming no routing):
    - » 3000miles * 5000ft/mile ⇒ 15 milliseconds
  - How many bits could be in transit at same time?
    - » 15ms * 155Mb/s = 290KB
  - In fact, Berkeley→MIT Latency ~ 45ms
    - » 872KB in flight if routers have wire-speed throughput
- Requirements for good performance:
  - Local area: minimize overhead/improve bandwidth
  - Wide area: keep pipeline full!

Sequence Numbers

- Ordered Messages
  - Several network services are best constructed by ordered messaging
    - » Ask remote machine to first do x, then do y, etc.
  - Unfortunately, underlying network is packet based:
    - » Packets are routed one at a time through the network
    - » Can take different paths or be delayed individually
  - IP can reorder packets! P_0,P_1 might arrive as P_1,P_0
- Solution requires queuing at destination
  - Need to hold onto packets to undo misordering
  - Total degree of reordering impacts queue size
- Ordered messages on top of unordered ones:
  - Assign sequence numbers to packets
    - » 0,1,2,3,4…..
    - » If packets arrive out of order, reorder before delivering to user application
    - » For instance, hold onto #3 until #2 arrives, etc.
  - Sequence numbers are specific to particular connection
    - » Reordering among connections normally doesn't matter
  - If restart connection, need to make sure use different range of sequence numbers than previously…

Reliable Message Delivery: the Problem

- All physical networks can garble and/or drop packets
  - Physical media: packet not transmitted/received
    - » If transmit close to maximum rate, get more throughput – even if some packets get lost
    - » If transmit at lowest voltage such that error correction just starts correcting errors, get best power/bit
  - Congestion: no place to put incoming packet
    - » Point-to-point network: insufficient queue at switch/router
    - » Broadcast link: two host try to use same link
    - » In any network: insufficient buffer space at destination
    - » Rate mismatch: what if sender send faster than receiver can process?
- Reliable Message Delivery on top of Unreliable Packets
  - Need some way to make sure that packets actually make it to receiver
    - » Every packet received at least once
    - » Every packet received at most once
  - Can combine with ordering: every packet received by process at destination exactly once and in order

Conclusion

- DNS: System for mapping from names⇒IP addresses
  - Hierarchical mapping from authoritative domains
  - Recent flaws discovered
- Layering: building complex services from simpler ones
- Datagram: an independent, self-contained network message whose arrival, arrival time, and content are not guaranteed
- Performance metrics
  - Overhead: CPU time to put packet on wire
  - Throughput: Maximum number of bytes per second
  - Latency: time until first bit of packet arrives at receiver
- Arbitrary Sized messages:
  - Fragment into multiple packets; reassemble at destination
- Ordered messages:
  - Use sequence numbers and reorder at destination
- Reliable messages:
  - Use Acknowledgements
  - Want a window larger than 1 in order to increase throughput