CS162
Operating Systems and Systems Programming
Lecture 21

Networking

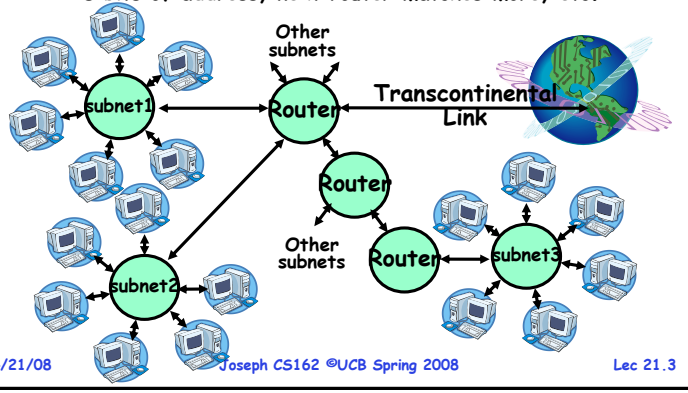
April 21, 2008
Prof. Anthony D. Joseph
http://inst.eecs.berkeley.edu/~cs162

Review: 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 353 million computers as of July 2005
  - 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

Review: 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.

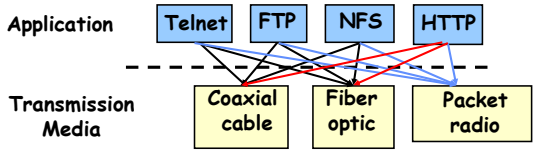


Goals for Today

- 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.

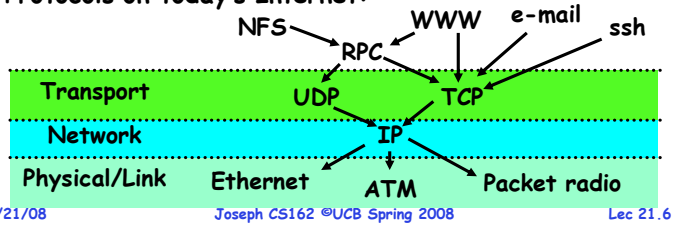
The Problem



- Re-implement every application for every technology?
- No! But how does the Internet architecture avoid this?

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:

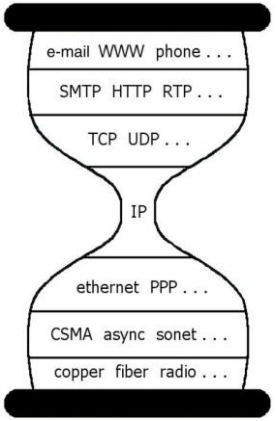


Network Layering

- 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

Hourglass – Single Internet Layer



- Allows networks to interoperate
  - Any network technology that supports IP can exchange packets
- Allows applications to function on all networks
  - Applications that can run on IP can use any network
- Simultaneous developments above and below IP

Back to Reality

- Layering is a convenient way to think about networks
- But layering is often violated
  - Firewalls
  - Transparent caches
  - NAT boxes
  - .....

Administrivia

- Midterm #2 returned in sections (Thursday/Friday)
  - Mean: 68.0 Std dev: 12.1
  - 30.0 - 35.0: 1 *
  - 35.0 - 40.0: 1 *
  - 40.0 - 45.0: 2 **
  - 45.0 - 50.0: 4 ****
  - 50.0 - 55.0: 9 *********
  - 55.0 - 60.0: 9 *********
  - 60.0 - 65.0: 7 *******
  - 65.0 - 70.0: 19 *******************
  - 70.0 - 75.0: 21 *********************
  - 75.0 - 80.0: 17 *****************
  - 80.0 - 85.0: 9 *********
  - 85.0 - 90.0: 4 ****
  - 90.0 - 95.0: 3 ***
- Final Exam – May 21st, 12:30-3:30pm
  - Email conflicts to cs162@cory by Wed 4/23 at 5pm
- Project #3 code deadline Tue 4/22 at 11:59pm
- Project #4 uses Google/IBM Cloud Computing Cluster
  - 40 64-bit dual-core AMD Opterons (80 cores)
  - You need to create an account (ASAP!)

Administrivia – Creating a CCC Account

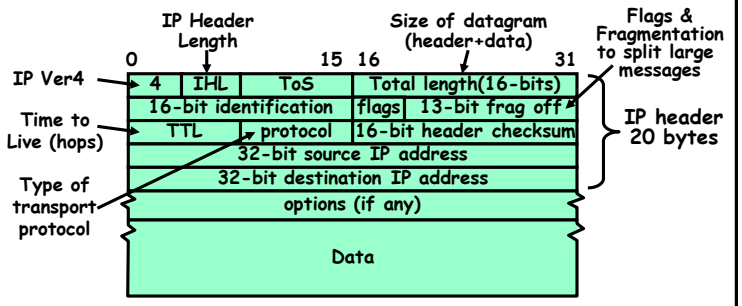
1. Login to http://inst.eecs.berkeley.edu/webacct/aws and select: "Request Cloud Cluster account"
2. Review the Usage Agreement and a form with your "token" will be displayed
3. Go to http://univsupport.hipods.ihost.com and select "Sign Up" (not "Sign In")
4. Enter the token on the "New User Registration" page and request an account (a login name and password)
   - This login and password will let you login to the forums at http://univsupport.hipods.ihost.com
5. Within 24 hours, your account will be enabled on the Cloud Cluster

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)

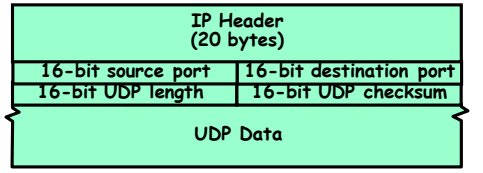
IP Packet Format

- 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]

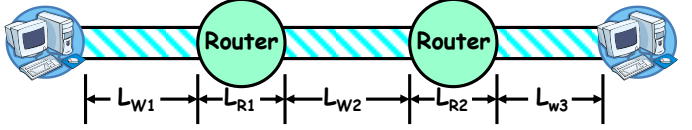


Building a messaging service (con't)

- 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 (e.g., Joost)
    - » Many uses of UDP considered "anti-social" – none of the "well-behaved" aspects of (say) TCP/IP
- But we need ordered messages
  - Create ordered messages on top of unordered ones
    - » IP can reorder packets! P_0,P_1 might arrive as P_1,P_0
  - How to fix this? 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

Performance Considerations

- Before 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!

BREAK

Sequence Numbers

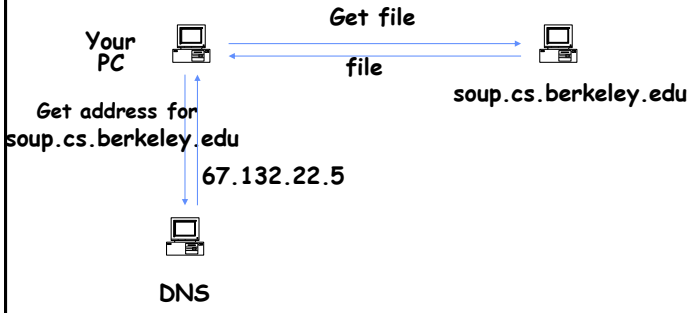
- Ordered Messages
  - Several network services are best constructed by ordered messaging
    - » Ask remote machine to first do x, then do y, etc.
  - Unfortunately, underlying network is packet based:
    - » Packets are routed one at a time through the network
    - » Can take different paths or be 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…

Domain Name Service (DNS)

- Humans/applications use machine names
  - e.g., www.cs.berkeley.edu
- Network (IP) uses IP addresses
  - e.g., 67.114.112.23
- DNS translates between the two
  - An overlay service in its own right
  - Global distribution of name-to-IP address mappings—a kind of content distribution system as well
  - Unsung hero of the Internet

File Transfer (FTP, SCP, etc.)

Get file from soup.cs.berkeley.edu



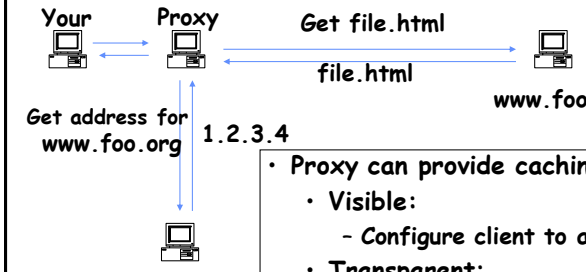
Email

Email message exchange is similar to previous example, except

- Exchange is between mail servers
- DNS gives name of mail server for domain
  - E.g., smtp.berkeley.edu

Web

Get www.foo.org/file.html



- Proxy can provide caching:
  - Visible:
    - Configure client to ask cache
  - Transparent:
    - Cache intercepts packet on its way to web server
    - An "application-aware middlebox"
    - Violates architectural purity, but are prevalent...

Content Distribution Network (CDN)

How to get closest copy of replicated content?

- CDNs have mirror servers distributed globally
- CDN customers allow CDN to run their DNS
- "Smart" DNS server returns results based on requester's IP address
- "Anycast IP address" routes to nearest server
  - Used for DNS top-level servers

Conclusion

- **Internet Protocol (IP):** Layering used to abstract details
  - Used to route messages through routes across globe
  - 32-bit addresses, 16-bit ports
- **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