CS162
Operating Systems and Systems Programming
Lecture 21

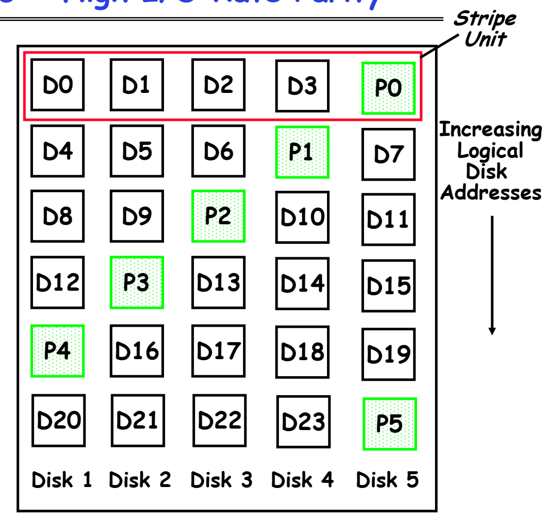
Networking

November 13, 2006

Prof. John Kubiatowicz
http://inst.eecs.berkeley.edu/~cs162

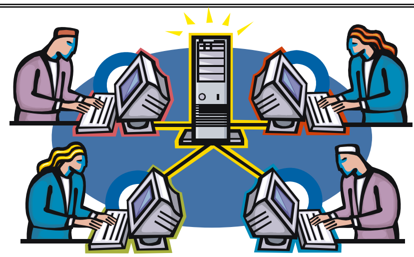
Review: RAID 5+: High I/O Rate Parity

- 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⊕D1⊕D2⊕D3
  - Can destroy any one disk and still reconstruct data
  - Suppose D3 fails, then can reconstruct: D3=D0⊕D1⊕D2⊕P0



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

Review: 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 on CPU to another
  - Destination gets interrupt when packet arrives
- **Protocol:** agreement between two parties as to how information is to be transmitted

Review: Ethernet and CSMA/CD

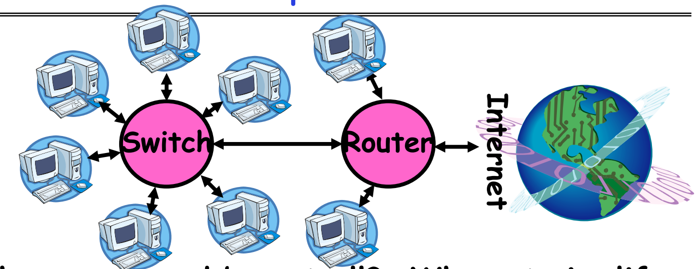
- Ethernet (early 80's): first practical local area network
  - It is the most common LAN for UNIX, PC, and Mac
  - Use wire instead of radio, but still broadcast medium
- Key advance was in arbitration called CSMA/CD: Carrier sense, multiple access/collision detection
  - **Carrier Sense:** don't send 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
- 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!

Goals for Today

- Networking
  - 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.

Point-to-point networks

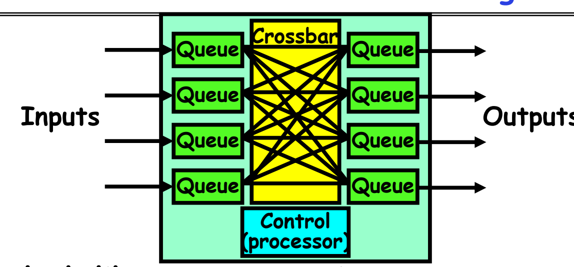


- Why have a shared bus at all? Why not simplify and only have point-to-point links + routers/switches?
  - Didn't used to be cost-effective
  - Now, easy to make high-speed switches and routers that can forward packets from a sender to a receiver.
- **Point-to-point network:** a network in which every 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.

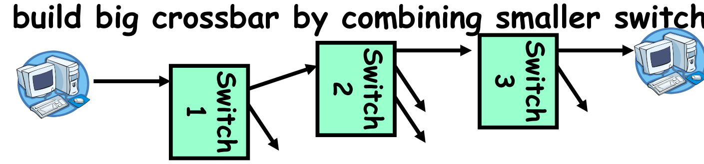
Point-to-Point Networks Discussion

- 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

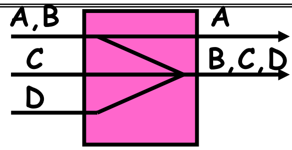


- Switches look like computers: inputs, memory, outputs
  - In fact probably contains a processor
- Function of switch is to forward packet to output that gets it closer to destination
- Can build big crossbar by combining smaller switches



- **Can perform broadcast if necessary**

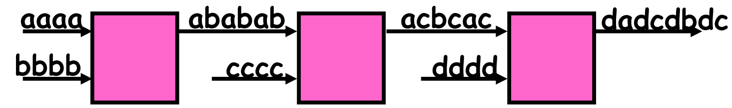
Flow control options



- 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 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

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

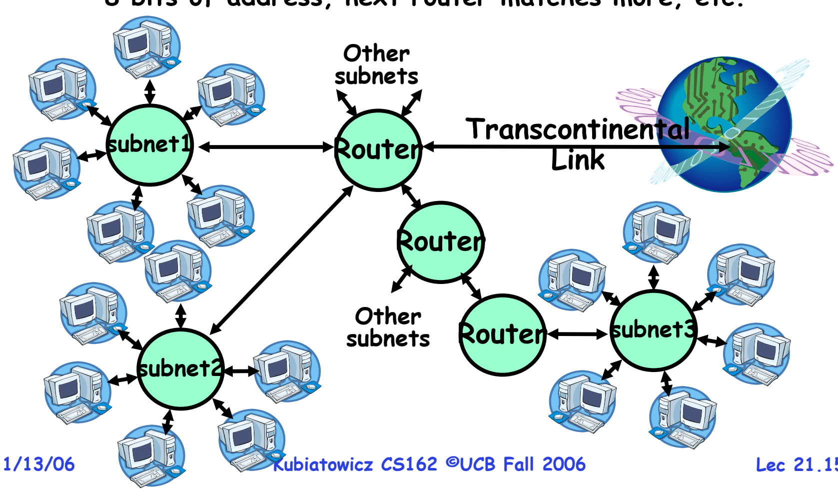
- IP address space divided into prefix-delimited ranges:
  - Class A: NN.0.0.0/8
    - » NN is 1–126 (126 of these networks)
    - » 16,777,214 IP addresses per network
    - » 10.xx.yy.zz is private
    - » 127.xx.yy.zz is loopback
  - Class B: NN.MM.0.0/16
    - » NN is 128–191, MM is 0-255 (16,384 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

Administrivia

- Exam reminders:
  - MIDTERM II: Dec 4th
    - » All material from last midterm and up to Wednesday 11/29
    - » Lectures #13 – 26
  - Final Exam
    - » Sat Dec 16th, 8:00am-11:00am, Bechtel Auditorium
    - » All Material
- Project 3 due Thursday (11/13)

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.

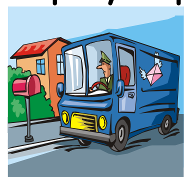


Simple Network Terminology

- 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…



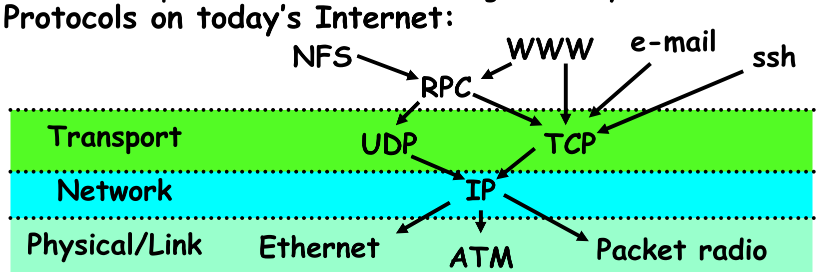
- Internet routing mechanism: routing tables
  - Each router does table lookup to decide which link to use to get packet closer to destination
  - Don't need 4 billion entries in table: routing is by subnet
  - Could packets be sent in a loop? Yes, if tables incorrect
- Routing table contains:
  - Destination 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:



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:

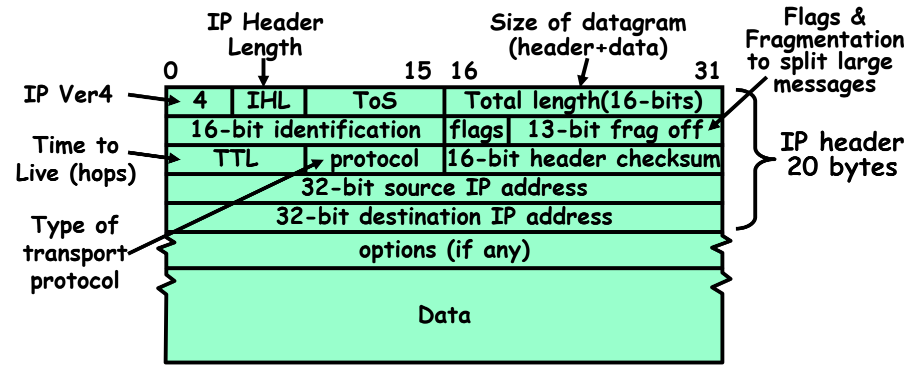
Physical Reality: Packets	Abstraction: Messages
Limited Size	Arbitrary Size
Unordered (sometimes)	Ordered
Unreliable	Reliable
Machine-to-machine	Process-to-process
Only on local area net	Routed anywhere
Asynchronous	Synchronous
Insecure	Secure

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)

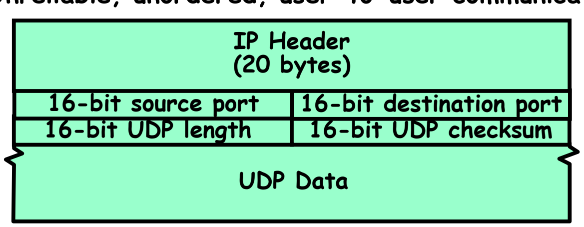
IP Packet Format

- 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
    - » Example: ssh, email, ftp, web browsing
  - Several IP protocols include notion of a "port", which is a 16-bit identifiers used in addition to IP addresses
    - » A communication channel (connection) defined by 5 items: [source address, source port, dest address, dest port, protocol]
- UDP: The User Datagram Protocol
  - UDP layered on top of basic IP (IP Protocol 17)
    - » Unreliable, unordered, user-to-user communication

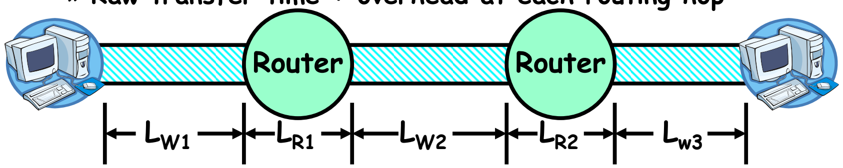


Building a messaging service (con't)

- UDP: The Unreliable Datagram Protocol
  - 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
- 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?

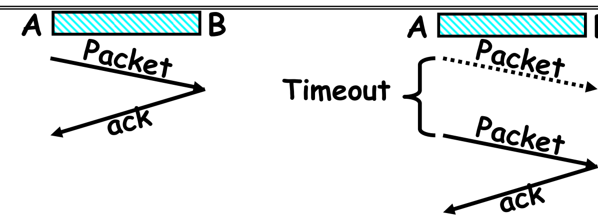
Sample Computations

- 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!

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
  - Reliable messages 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 only once
  - Can combine with ordering: every packet received by process at destination exactly once and in order

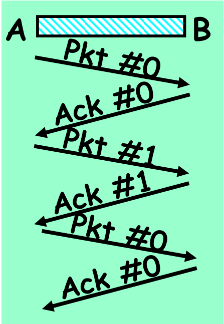
Using Acknowledgements



- How to ensure transmission of packets?
  - Detect garbling at receiver via checksum, discard if bad
  - Receiver acknowledges (by sending "ack") when packet received properly at destination
  - Timeout at sender: if no ack, retransmit
- Some questions:
  - If the sender doesn't get an ack, does that mean the receiver didn't get the original message?
    - » No
  - What it ack gets dropped? Or if message gets delayed?
    - » Sender doesn't get ack, retransmits. Receiver gets message twice, acks each.

How to deal with message duplication

- Solution: put sequence number in message to identify re-transmitted packets
  - Receiver checks for duplicate #'s; Discard if detected
- Requirements:
  - Sender keeps copy of unack'ed messages
    - » Easy: only need to buffer messages
  - Receiver tracks possible duplicate messages
    - » Hard: when ok to forget about received message?
- Simple solution: Alternating-bit protocol
  - Send one message at a time; don't send next message until ack received
  - Sender keeps last message; receiver tracks sequence # of last message received
- Pros: simple, small overhead
- Con: Poor performance
  - Wire can hold multiple messages; want to fill up at (wire latency × throughput)
- Con: doesn't work if network can delay or duplicate messages arbitrarily



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
- Reliable, Ordered, Arbitrary-sized Messaging:
  - Built through protocol layering on top of unreliable, limited-sized, non-ordered packet transmission links