Announcements

- Reading: Chapter 16
- Project #6 Due on Tuesday 5:00 pm
- Final is a week from Sat at 8:00 am (this room)

Sending Data

- Data is split into *packets*
  - limited size units of sending information
  - can be
    - fixed sized (ATM)
    - variable size (Ethernet)
- Need to provide a destination for the packet
  - need to identify two levels of information
    - machine to send data to
    - comm abstraction (e.g. process) to get data
  - address may be:
    - a globally unique destination
      - for example every host has a unique id
    - may unique between hops
      - unique id between two switches

TCP/IP Protocol

- Name for a family of Network and Transport layers
  - can run over many link layers:
    - Arpanet, Ethernet, Token Ring, SLIP/PPP, T1/T3, etc.
- IP - Internet Protocol
  - network level packet oriented protocol
  - 32 bit host addresses (dotted quad 128.8.128.84)
  - 8 bit protocol field (e.g. TCP, UDP, ICMP)
- TCP - Transmission Control Protocol
  - transport protocol
  - end-to-end reliable byte streams
  - provides ports for application specific end-points
- UDP- user datagram protocol
  - transport protocol
  - unreliable packet service
  - provides ports for application specific end-points

TCP/IP History

- Arpanet was the origin of today's Internet
  - started in 1969 to connect universities and DoD sites
  - early example of packet switched network
  - original links were 64kbps and 9.6kpbs
- TCP/IP v4
  - started in use Jan 1, 1983
  - This was a *flag day*
    - all systems had to change to the new protocol at once
    - with the modern Internet this would be **hard** to do
- TCP/IP v6
  - Moves to 128 bit addresses
  - Simplified packet header

Subnet Addressing

- Single site which has many physical networks
  - Only local routers know about all the physical nets
  - Site chooses part of address that distinguishes between physical networks
- subnet mask: splits the IP address into two parts
  - /xx notation defines boundary where xx is the number of bits in part 1
  - First part is network mask
  - Second part is address within that network
- Common /24 site mask 255.255.255.0
  - use 24 bits represent physical net
  - Final 8 bits represent host

Routing

- How does a packet find its destination?
  - problem is called routing
- Several options:
  - source routing
    - end points know how to get everywhere
    - each packet is given a list of hops before it is sent
  - hop-by-hop
    - each host knows for each destination how to get one more hop in the right direction
- Can route packets:
  - per session
    - each packet in a connection takes same path
  - per packet
    - packets may take different routes
    - possible to have out of order delivery

Routing IP Datagrams

- Direct Delivery:
  - a machine on a physical network can send a physical frame directly to another
  - transmission of an IP datagram between two machines on a single physical network does not involve routers.
    - Sender encapsulates datagram into a physical frame, maps destination IP address to a physical address and sends frame directly to destination
  - Sender knows that a machine is on a directly connected network
    - compare network portion of destination ID with own ID - if these match, the datagram can be sent directly
  - Direct delivery can be viewed as the final step in any datagram transmission

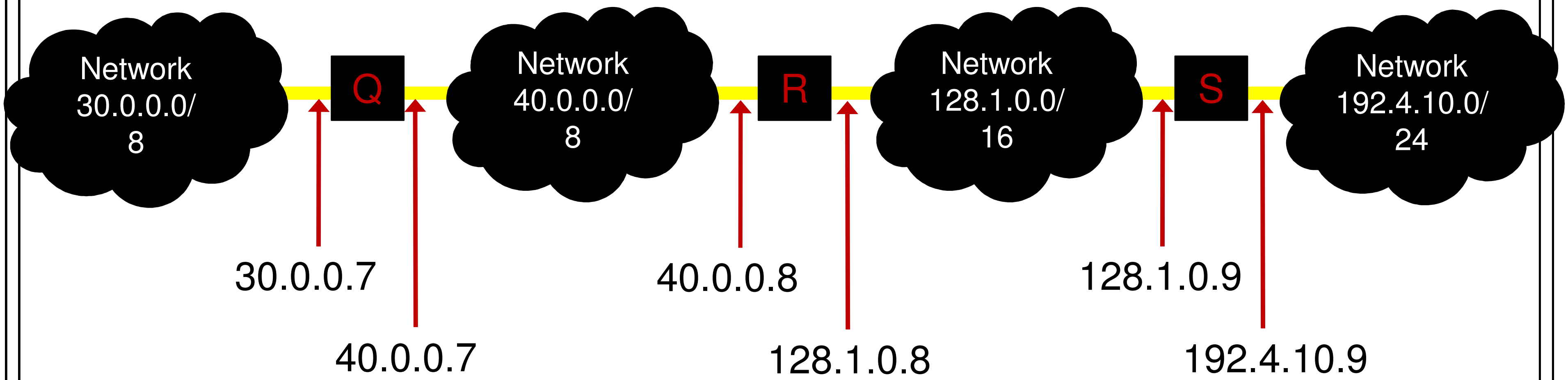
Routing Datagrams (cont.)

- Indirect Delivery
  - sender must identify a router to which a datagram can be sent
  - sending processor can reach a router on the sending processor's physical network (otherwise the network is isolated!)
  - when frame reaches router, router extracts encapsulated datagram and IP software selects the next router
    - datagram is placed in a frame and sent off to the next router

Table Driven Routing

- Routing tables on each machine store information about possible destinations and how to reach them
- Routing tables only need to contain network prefixes, not full IP addresses
  - No need to include information about specific hosts
- Each entry in a routing table points to a router that can be reached across a single network
- Hosts and routers decide
  - can packet be directly sent?
  - which router should be responsible for a packet (if there is more than one on physical net)

Routing (w/ subnets)



To reach hosts on network	Mask*	Next Hop
30.0.0.0	255.0.0.0	40.0.0.7
40.0.0.0	255.0.0.0	<DIRECT>
128.1.0.0	255.255.0.0	<DIRECT>
192.4.10.0	255.255.255.0	128.1.0.9

Mask field is used to extract the network part of an address during lookup.

If((Mask[i] & D) == Destination[i]) forward to nextHop[i]

Consider a datagram destined for address 192.4.10.3 and the datagram arrives at router R

Extract destination IP address, D from datagram and compute network prefix N

255.0.0.0&192.4.10.3 is not equal to 30.0.0.0

<same for entry 2 and 3>

255.255.255.0&192.4.10.3=192.4.10.0
→ send to 128.1.0.9

Example from Comer book: Internetworking with TCP/IP: volume 1 [Third Edition]

10

Algorithm: RouteDatagram (Datagram, RoutingTable)

Extract destination IP address, D, from datagram and compute network prefix N

If N matches any directly connected network address
  [Direct delivery]

Else if the table contains a host-specific route for D
  [send datagram to next-hop specified in table]

Else if the table contains a route for network N
  [send datagram to next-hop specified in table]

Else if the table contains a default route
  [send the datagram to the default route]

Else *declare a routing error*

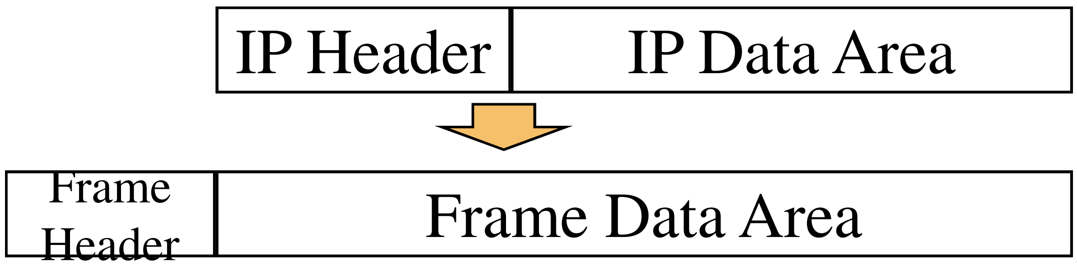
Algorithm from Comer book: Internetworking with TCP/IP: volume 1 [Third Edition]

11

Encapsulation

How do we send higher layer packets over lower layers?

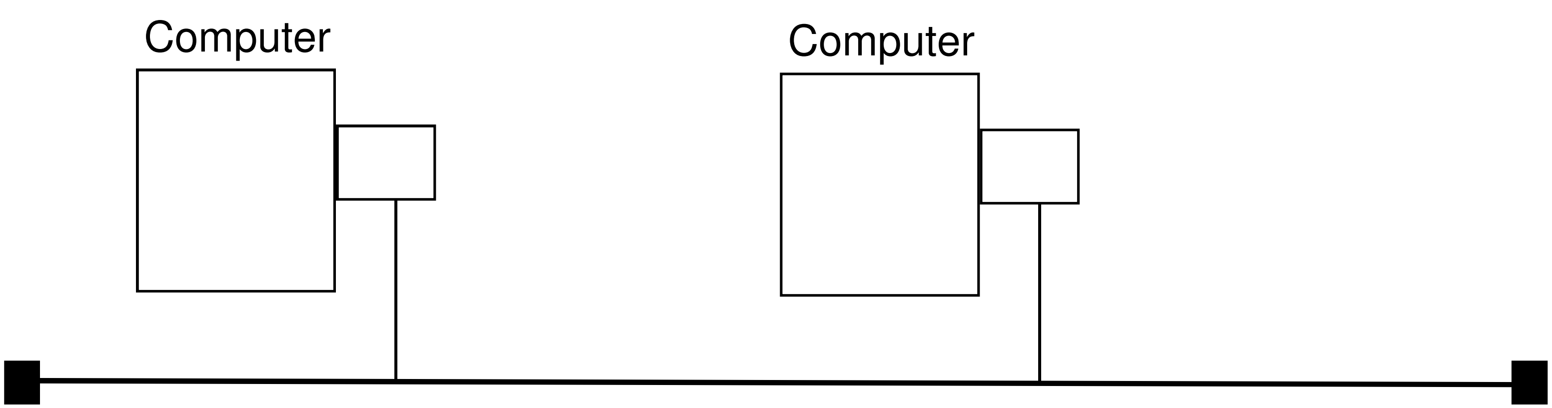
- Higher level info is opaque to lower layers
  - it's just data to be moved from one point to another



- Higher levels may support larger sizes than lower
  - could need to *fragment* a higher level packet
    - split into several lower level packets
    - need to re-assemble at the end
  - examples:
    - ATM cells are 48 bytes, but IP packets can be 64K
    - IP packets are 64K, but files are megabytes

Ethernet

- 10 Mbps (to 100 Gbps)
- mili-second latency
- limited to several kilometers in distance
- variable sized units of transmission
- Conceptually a bus based protocol
  - requests to use the network can collide
- addresses are 48 bits
  - unique to each interface



Switched Ethernet

- Logically it is still a bus
- Physically, it is a star configuration
  - the hub is at the center of the network
- Switches provide:
  - better control of hosts
    - possible to restrict traffic to only the desired target
    - can shutdown a host's connection at the hub if its Ethernet device is misbehaving
  - easier wiring
    - can use twisted pair wiring
- 100 Mbps/1Gbps Ethernet
  - is only available with switches
- 10Gbps Ethernet
  - Requires cat-6 (to 100 feet) or cat-7 wiring (to 100 meters)

Ethernet Collisions

- If one host is sending, other hosts must wait
  - called Carrier Sense with Multiple Access (CSMA)
- Possible for two hosts to try to send at once
  - each host can detect this event (cd- Collision Detection)
  - both hosts must re-send information
    - if they both try immediately, will collide again
    - instead each waits a random interval then tries again
- Only provides statistical guarantee of transmission
  - however, the probability of success if higher than the probability of hardware failures and other events

My Research Interests

- Parallel Computing
  - There are limits to how fast one processor can run
  - solution: use more than one processor
- Issues in parallel computing design
  - do the processors share memory?
    - is the memory “uniform”?
    - how do processors cache memory?
  - if not how do they communicate?
    - message passing
    - what is the latency of message passing

Parallel Processing

- What happens in parallel?
- Several different processing steps
  - pipeline
  - simple example: grep foo | sort > out
  - called: *multiple instruction multiple data* (MIMD)
- The same operation
  - every processor runs the same instruction (or no-instruction)
  - called: *single instruction multiple data* (SIMD)
  - good for image processing
- The same program
  - every processor runs the same program, but not “lock step”
  - called: *single program multiple data* (SPMD)
  - most common model

Issues in effective Parallel Computation

- Getting enough parallelism
  - Limited by what is left serial
  - Even 10% serial limited to a speedup of 10x even with infinite numbers of processors
- Load balancing
  - every processor should to have some work to do.
- Latency hiding/avoidance
  - getting data from other processors (or other disks) is slow
  - need to either:
    - hide the latency
      - processes can “pre-fetch” data before they need it
      - block and do something else while waiting
    - avoid the latency
      - use local memory (or cache)
      - use local disk (of file buffer cache)
- Limit communication bandwidth
  - use local data
  - use “near” data (i.e. neighbors)

My Research:

- Given a parallel program and a machine
- Try to answer performance related questions
  - Why is the programming running so slowly?
  - How do I fix it?
- Issues:
  - how to measure a program without changing it?
  - how do you find (and then present) the performance problem, not tons of statistics?
- Techniques:
  - dynamic data collection
  - automated search
  - analysis of process interactions

Introduction

- Software today
  - makes extensive use of libraries and re-usable components
  - Libraries used by an application may not be tuned to the application's need
- Fast software development/distribution with built-in (default) configurations
  - Applications may not run well in all environments
  - There may be no single configuration good for all environments

Large Scale Computing

- Today (11/2014)
  - 29 systems with more than 128k processors
  - More than 50 systems >= 16k processors
  - World's fastest computer (Tianhe-2i n China)
    - 3,120,000 cores
    - Uses 17.8 MW of electricity