Announcements

- Project #5 is available
- Project #4 is extended until Midnight (7 hours extra)
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

- Common “Class B” netmask mask 255.255.255.0
  - use 3rd byte to represent physical net
  - use 4th byte to represent host

Internet Part  Local Part

vanilla scheme

Internet Part  Physical network  Host

subnet scheme
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
- Common Class B site mask 255.255.255.0
  - use 3rd byte to represent physical net
  - use 4th byte to represent host
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

```
| IP Header | IP Data Area |
```

- **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
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 a machine on another network.
  - Transmission of an IP datagram between two machines on a single physical network does not involve routers.
    - Sender encapsulates datagram into a physical frame, binds destination IP address to a physical hardware 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 deliver 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)?
IP Routing Algorithm
(from Comer)

- RouteDatagram(Datagram, Routing Table)

- Extract destination IP address, D from datagram and compute network prefix N
  - if N matches any directly connected network address
  - else if the table contains a host-specific route for D
  - else if the table contains a route for network N
  - else if the table contains a default route
  - else declare a routing error
How are routing tables obtained?

- **Routing with partial information**
  - Hosts do not need complete knowledge of all possible destination addresses
  - Host sends non-local information to (a) router

- **Routers can also route with partial information**
  - Consider a topology consisting of two completely connected subgraphs A and B
  - Subgraphs A and B share a single link
  - If a router in A sees an address it does not recognize, it sends the packet to B and vice-versa
Early Internet Architecture

- Small central set of routers that kept complete information about all destinations
- Larger set of outlying routers with only local information
- Default route for outlying routers is to a central router
- Local administrators can make changes
  - Local changes need to be propagated locally as well as to the central routers
Internet Core Router System

Backbone

router 1
Local Net 1

router 2
Local Net 2

router 3
Local Net 3

Local Net 4
Internet Core Routing System

• Core routers exchange routing information so each will have complete information about optimal routes to all destinations

• This did not scale:
  – maintaining consistency among core routers became increasingly difficult
  – further difficulties arise when there are several backbones (e.g. ARPAnet and NSFnet)
  – if the core architecture is partitioned so that all routers use default routes, may induce routing loops
    • if routing information is not consistent, it is possible for a packet to be repeatedly routed in a circle until the packet times out
Distributed Systems

- **Provide:**
  - access to remote resources
  - security
  - location independence
  - load balancing

- **Basic Services:**
  - remote login (telnet and rlogin protocols)
    - extends basic access provided by normal login
  - file transfer (ftp, rcp)
    - can support anonymous transfers
  - information services (http)
    - two way protocols (request/response)
Distributed Systems

● A unified view of local and remote access

● Typical Services
  – data migration
    • provide only the data required, not the whole file
    • manage multiple copies as versions of the same object
  – process migration
    • a process can move from one machine to another
    • reasons for migration:
      – load balancing
      – data affinity
      – hardware/software preference (better configuration)
Distributed OS Design Issues

- **Should provide same model as a central system**
  - easy to understand for users

- **Needs to be scaleable**
  - will it work with 100, 1,000, or 10,000 nodes?

- **Failure Modes**
  - avoid a single central failure point
  - can loss performance or functionality with failure
    - but loss should be proportional to size of failure

- **Security**
  - should provide same guarantees on data integrity as a local system