Last Time

- Bridge: Level 2 switch
  - Switches directly at level 2
  - Unscalable routing: (e.g., learning bridges with spanning tree)
  - Addresses that are not spatially aggregatable
  - Limited in bridging heterogeneity

- Packet switching Approaches: datagram, Virtual circuit, source routing

- Examples
  - IEEE 802.1 for datagram
  - ATM for VCS
  - Myrinet for Source Routing

---

Spanning Tree Algorithm in IEEE 802.1

- IEEE 802.1 is the standard for "LAN/MAN Bridging and Management"
- Idea: bridges disable some ports to eliminate cycles

  General Algorithm:
  - Bridges elect a "tree root" bridge
  - Each bridge calculates shortest path to root
  - Bridges on each LAN elect a "designated bridge" such that
    - It is the closest bridge to the root
    - Break ties using bridge id

ATM as an example of VCS L2 switching

- Goal, one network to carry both voice and data

  Shape of design
  - VCS to support voice (QoS/reservation)
  - Fixed size cells to enable faster switching
  - Small size cells to support voice traffic and to enable low grain sharing of link
  - But how to support data on top of tiny 48-byte cells?
    * Solution: Add a software layer (Application adaptation layer) that supports larger variable size frames
    * Segmentation into ATM cells and Reassembly at the other side

- LAN Emulation (LANE) for ATM in LAN settings
Virtual LANs

- Logically configure parts of the extended LAN to form a virtual LAN
- Packets sent on one VLAN segment will be visible to all segments of the VLAN (but nowhere else)
- When a packet arrives at a bridge port with a host id as sender, the bridge inserts the VLAN id associated with that port
- Advantage: Enhances scalability – why?
- Disadvantage: Requires administrator intervention

Internetworking

- We know how to build directly connected networks
- Switches have the potential of scaling to large networks
- Bridges (link-level switching) provided valuable lessons
  - Not scalable (consider spanning tree, or addressing scheme)
  - Not heterogeneous (same MAC address family; compatible payloads, etc.)
- Scalability: must scale indefinitely
  - Addressing and Routing
  - Multicast and broadcast?
- Heterogeneity:
  - Users on different networks must be able to talk
  - Might need to cross several other networks on the way

Internet Protocol (IP)

- Internet Protocol (IP; due to Karn and Cerf)
- Runs on all hosts (remember the “hour-glass” shape of the Internet Protocol Suite)
- Provides isolation from the networking technology
- Must provide one service model that is common to all possible underlying technologies – what should we use?
  - Connectionless best-effort delivery

IP Packet Format

- Version: 4 for IPv4
- Hlen: number of 32-bit words in the header
- TOS: Type of Service; can be used for QoS
- Length: Number of bytes in the datagram
- Ident/Flags/Offset: used for fragmentation and reassembly
- TTL: Time to Live (maximum hop count allowed; why?)
- Protocol: key to identify higher level protocol (e.g., TCP)
- Checksum: applied to header (why not CRC?)
- Source and destination addresses (what addresses?)
- Options
- Payload
- Where does the “link-layer” header fit?
### Fragmentation and Reassembly – Why?

- **Why is this needed?**
- Each network has some Maximum Transmission Unit (e.g., ethernet 1500 bytes; PPP 532 bytes)
  - Restrict IP payload to the smallest MTU; or
  - Use fragmentation and reassembly if necessary

- **Strategy:**
  - Fragment when necessary (MTU < Datagram)
  - Fragments are self-contained IP packets
  - Try to avoid fragmentation at the source
  - Refragmentation is possible
  - Delay reassembly until destination
  - What if a fragment is lost?

### Fragmentation and Reassembly – How?

- Ident is the packet sequence number
- M-bit (flags field) is 1 in all but the last fragment
- How would refragmentation be implemented?
- “path MTU discovery” to minimize fragmentation

### Internet Addresses – Pre 1993

- **Class D [1110— - - - multicast group id - - -]**
- Address broken into a network number and a host number
- Why different address classes?
  - Class A: 126 networks, each $2^{24}$ hosts
  - Class B: $2^{14}$ networks, each $2^{16}$ hosts
  - Class C: $2^{21}$ networks, each $2^{8}$ hosts
- What is an address associated with?
- Is this scalable? How many maximum hosts?
Address Translation

- Need to map the IP address to a physical MAC address for:
  - Destination host (for a direct connection)
  - Next hop router

- Possible approaches:
  - Encode the Physical address as part of the IP address?
  - Statically construct translation tables (e.g., administrator does it)
  - Dynamically construct translations (ARP)

Address Resolution Protocol (ARP)

- Maintain a table of IP to physical address mappings; another network cache
- If translation not in table, broadcast asking for it
- Target hears broadcast, and replies with physical address
- Table entries expire; TTL usually a few minutes
- What happens on switched LANs? Proxy ARP
Back to the Routing Table

- Routes in the table map an IP address to the address of the next hop
- Three different types of routes can exist in the routing table
  - Host-specific: an entry giving the route to a specific host
  - Network specific: an entry giving the route for all hosts in a network
  - Default: an entry for routes to hosts we know nothing about

Example Forwarding Table

```
[root@garnet] netstat -r
Kernel IP routing table
Destination Gateway Genmask Iface
128.226.123.0 * 255.255.255.0 eth0
127.0.0.0 * 255.0.0.0 lo
default 128.226.123.1 0.0.0.0 eth0
```

- Go through entries in order
- Logically AND destination IP with Mask. If output matches destination, use the entry.

Route Examples

- Masks are actually used instead of *
IP Addresses

- Problem: severe underutilization of address space if we follow the address classes strictly.
  - Example, using a class C network for a LAN with 3 machines, 252 addresses are wasted.
  - Example, if you have a network of 400 machines, need a class B address.
- Need a bigger address space (IPv6, NAT).
- Until then, need to develop more efficient ways to use the available space.
  - Use subnetting to fit multiple sub-networks within a single network.
  - Use supernetting to assign multiple networks a single ID.

Subnetting

- Allow multiple networks to share the same “physical network” space.
- Example, Computer science department has 100 machines. Electrical Engineering department has 100 machines. We need two networks that can each be managed and routed independently.
  - Assign 2 class C networks?
  - Use subnetting to share one class C network.
    - CS gets 128.226.123.0 to 128.226.123.127
    - EE gets 128.226.123.128 to 128.226.123.255
  - How?
    - Address mask is used to figure out what network you are on.
      - Address Mask is 255.255.255.128
      - Any address in CS ANDed with the mask produces 128.226.123.0
      - Any address in EE ANDed with the mask produces 128.226.123.128

Another Subnetting Example

<table>
<thead>
<tr>
<th>Network number</th>
<th>Host number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class B address</td>
<td>00000000</td>
</tr>
<tr>
<td>Subnet mask (255.255.255.0)</td>
<td></td>
</tr>
</tbody>
</table>

Subnetted address

<table>
<thead>
<tr>
<th>Network number</th>
<th>Subnet ID</th>
<th>Host ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subnet mask: 255.255.255.128</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subnet number: 128.96.34.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.96.34.15 H1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.96.34.1  R1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.96.34.130</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.96.34.129</td>
<td>R2</td>
<td></td>
</tr>
<tr>
<td>128.96.33.14 H3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Supernetting and CIDR

- Instead of assigning a corporation that has 400 hosts a class B address, assign it two class C addresses.

Generalizing subnetting and supernetting gives us CIDR: Classless Internet Domain Routing.

- Idea: allow the network portion of the address to be dynamic in size.
- Use masks to sort things out.
- As you get closer to the destination, more details become available.
CIDR – after 1993

Mask is the number of significant bits represented by the prefix length: 24 prefix is 255.255.255.0

- Arbitrary sized network field
- Routes represented by tuple: address and prefix size
- e.g., 128.226.123.1/16, 128.226 is network
- Why CIDR vs. Static Networks?
  - Good: Fragmentation, Route aggregation
  - Bad?

CIDR Routing

- Multi-regional 192.0.0.0 - 193.255.255.255
- Europe 194.0.0.0 - 195.255.255.255
- Others 196.0.0.0 - 197.255.255.255
- North America 198.0.0.0 - 199.255.255.255
- Central/South America 200.0.0.0 - 201.255.255.255
- Pacific Rim 202.0.0.0 - 203.255.255.255
- Others 204.0.0.0 - 207.255.255.255
- How can we make use of this information?

Assigning Addresses

- Don't have to worry about hardcoded network class boundaries
- Company kludge.com wants 1024 addresses
  - Assign it 198.15.15.255 with mask 255.255.254.0
  - Router entry for this company can be 198.15.0.0/24
- Company bob.com wants 4096 addresses
  - Assign it 198.20.16.0 to 198.20.33.255 with mask 255.255.240.0 (20 bit mask, must start at 4096 boundary)
- Company oops.com wants 2048 addresses, assign it what?

Addressing – summary/discussion

- Two issues:
  1. Assigning addresses efficiently so we don't run out of addresses
  2. Scalability of routing
- What makes routing scalable?
  - Routing table size
  - Routing protocol overhead
- Classless addresses
  - any power of two size network
  - use masks to sort things out
  - Helps both efficient address assignment and routing scalability (how?)
Aside–Configuration: Who assigns IP addresses to hosts?

- Two approaches:
  - Static: parameters are hard wired
    - Conceptually easy, but difficult to administer
    - Necessary for servers to have a fixed point of attachment to the Internet
  - Dynamic:
    - Parameters obtained from a server
    - Easy to administer
    - Dynamic use of available addresses
    - How to do it?

DHCP

- Goals:
  - Provide all the information necessary to configure the host
    - IP address
    - Subnet Mask (to be explained soon)
    - DNS server (to be explained later)
    - Routing information
  - Allow management and sharing of IP addresses
    - Servers manage a finite number of IP addresses
    - Addresses are leased to clients for finite leases
      - Renew lease if you need IP address longer

- Reverse ARP (RARP – RFC 903):
  - When machine is booted, it announces its hardware address and asks if anyone knows its IP address
  - RARP server replies and assigns the machine an IP address

- Boot Protocol (BOOTP 951):
  - Newer protocol, layered on top of UDP
  - New capability is to supply also the name of a machine that will serve the boot image
  - This allows diskless workstations to boot via the network

- Dynamic Host Configuration Protocol (DHCP – RFC 2131 and 2132):
  - Successor to BOOTP
  - Provides all the information necessary to configure the host
    - IP address
    - Subnet Mask (to be explained soon)
    - DNS server (to be explained later)
    - Routing information
  - Allows management and sharing of IP addresses
    - Servers manage a finite number of IP addresses
    - Addresses are leased to clients for finite leases
      - Renew lease if you need IP address longer

- When a host "wakes up," it sends a DHCP–DISCOVER message (IP address 255.255.255.255)
  - The DHCP server or a DHCP relay agent receives the message
  - Server replies with an IP address and configuration information
  - Relay agent forwards the message to the DHCP server
  - Server replies with an IP number and configuration information

- What happens if a host turns itself off without releasing the address?
  - Reverse ARP (RARP – RFC 903):
    - When machine is booted, it announces its hardware address and asks if anyone knows its IP address
  - Boot Protocol (BOOTP 951):
    - Newer protocol, layered on top of UDP
    - New capability is to supply also the name of a machine that will serve the boot image
      - This allows diskless workstations to boot via the network
  - Dynamic Host Configuration Protocol (DHCP – RFC 2131 and 2132):
    - Successor to BOOTP
    - Provides all the information necessary to configure the host
      - IP address
      - Subnet Mask (to be explained soon)
      - DNS server (to be explained later)
      - Routing information
    - Allows management and sharing of IP addresses
      - Servers manage a finite number of IP addresses
      - Addresses are leased to clients for finite leases
        - Renew lease if you need IP address longer
ICMP

- ICMP message types include:
  - Echo request/reply (with or without a timestamp)
  - Address mask request/reply
  - Parameter problem
  - Source quench (to control congestion)
  - Redirect (router knows a better route)
  - Destination unreachable (port, protocol, host)
  - TTL exceeded
  - Checksum failed ...
- What is the idea? I thought IP was unreliable??
- Several applications use ICMP (ping, traceroute...)

Traceroute

- Implemented by sending UDP datagrams to the destination
- First packet has TTL=1
  - Next hop replies that destination is unreachable (ICMP message);
    we learn what the next hop is
  - Increase TTL by 1, and repeat

Internet Control Message Protocol (ICMP)

- A protocol for signalling/feedback among routers and hosts
  - Not intended for use by applications; used to send information about problems
  - No multiplexing is provided (i.e., to a specific process), it is intended purely as a host-to-host mechanism
  - Built on top of IP, but considered "at the same layer"
- To prevent message explosion, ICMP messages are not generated in response to errors experienced by other ICMP messages
- Several types of information can be exchanged (bad packets, congestion, failed routes, etc.)
Traceroute Example

[tracergarnet] /usr/sbin/traceroute syracuse.edu
traceroute to syracuse.edu (128.230.18.35), 30 hops max, 38 byte packet
1 128.226.123.1 (128.226.123.1) 1.394 ms 0.772 ms 0.748 ms
2 128.226.100.1 (128.226.100.1) 6.363 ms 6.919 ms 5.027 ms
3 * 128.226.100.30 (128.226.100.30) 3.908 ms 4.659 ms
4 149.125.1.1 (149.125.1.1) 1.488 ms 2.788 ms 3.857 ms
5 199.109.4.38 (199.109.4.38) 14.012 ms 12.792 ms 11.748 ms
6 syru-vbns1.nysernet.net (199.109.4.13) 26.954 ms 21.26 ms 20.432 ms
7 128.230.249.2 (128.230.249.2) 22.655 ms 20.399 ms 20.445 ms
8 128.230.93.1 (128.230.93.1) 20.780 ms 20.514 ms 22.336 ms
9 cwis01.syr.edu (128.230.18.35) 20.704 ms * 24.706 ms