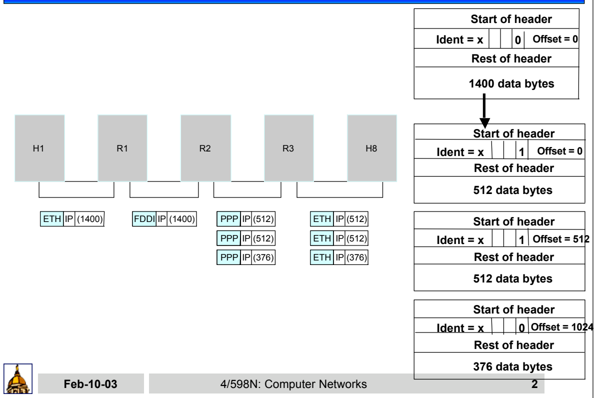
2. Fragmentation and Reassembly

- Each network has some MTU
- Strategy
  - fragment when necessary (MTU < Datagram)
  - try to avoid fragmentation at source host
  - re-fragmentation is possible
  - fragments are self-contained datagrams
  - use CS-PDU (not cells) for ATM
  - delay reassembly until destination host
  - do not recover from lost fragments

Example



3. Global Addresses

- Properties
  - globally unique
  - hierarchical: network + host
- Dot Notation
  - 10.3.2.4
  - 129.74.33.81
  - 192.12.69.77

A: | 0 | Network (7) | Host (24) |

B: | 1 | 0 | Network (14) | Host (16) |

C: | 1 | 1 | 0 | Network (21) | Host (8) |

Datagram Forwarding

- Strategy
  - every datagram contains destination's address
  - if directly connected to destination network, then forward to host
  - if not directly connected to destination network, then forward to some router
  - forwarding table maps network number into next hop
  - each host has a default router
  - each router maintains a forwarding table
- Example (R2)

Network number	Next
1	R3
2	R1
3	Interface 1
4	Interface 0

Address Translation

- Map IP addresses into physical addresses
  - destination host
  - next hop router
- Techniques
  - encode physical address in host part of IP address
  - table-based
- ARP
  - table of IP to physical address bindings
  - broadcast request if IP address not in table
  - target machine responds with its physical address
  - table entries are discarded if not refreshed

ARP Details

- Request Format
  - HardwareType: type of physical network (e.g., Ethernet)
  - ProtocolType: type of higher layer protocol (e.g., IP)
  - HLEN & PLEN: length of physical and protocol addresses
  - Operation: request or response
  - Source/Target-Physical/Protocol addresses
- Notes
  - table entries timeout in about 10 minutes
  - update table with source when you are the target
  - update table if already have an entry
  - do not refresh table entries upon reference

ARP Packet Format

0	8	16	31
Hardware type = 1		ProtocolType = 0x0800	
HLen = 48	PLen = 32	Operation	
SourceHardwareAddr (bytes 0 –3)			
SourceHardwareAddr (bytes 4 –5)		SourceProtocolAddr (bytes 0 –1)	
SourceProtocolAddr (bytes 2 –3)		TargetHardwareAddr (bytes 0 –1)	
TargetHardwareAddr (bytes 2 -5)			
TargetProtocolAddr (bytes 0 -3)			

Sample arp table in darwin

- arp -a

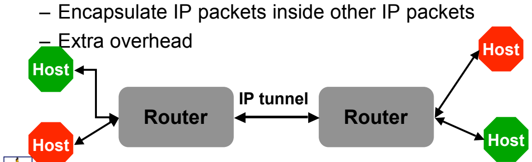
```
Net to Media Table: IPv4
Device  IP Address            Mask             Flags  Phys Addr
------ -------------------- --------------- ----- ---------------
eri0   eafs-e06.gw.nd.edu   255.255.255.255       00:d0:d3:3c:72:1c
eri0   mailspool.helios.nd.edu 255.255.255.255    08:00:20:a1:8d:b6
eri0   penny.helios.nd.edu  255.255.255.255       08:00:20:9d:64:22
eri0   bubba.helios.nd.edu  255.255.255.255       08:00:20:92:5a:e4
eri0   yaya.helios.nd.edu   255.255.255.255       08:00:20:c4:05:c4
eri0   pinky.helios.nd.edu  255.255.255.255       08:00:20:a9:88:30
eri0   dagger.nd.edu        255.255.255.255       08:00:20:7a:4a:3e
eri0   bind.nd.edu          255.255.255.255       08:00:20:8a:5f:cf
eri0   license.nd.edu       255.255.255.255       08:00:20:86:2b:52
eri0   admissions.nd.edu    255.255.255.255       00:90:27:d5:f2:2f
eri0   casper.helios.nd.edu 255.255.255.255       08:00:20:b1:f8:e1
```


4. Internet Control Message Protocol (ICMP)

- Echo (ping)
- Redirect (from router to source host)
- Destination unreachable (protocol, port, or host)
- TTL exceeded (so datagrams don't cycle forever)
- Checksum failed
- Reassembly failed
- Cannot fragment

5. Virtual Private Networks (VPN) and tunnel

- Create a virtual network connecting different networks across the general Internet
  - Connect ND campus in South Bend and London to make them look like a single LAN even though packets traverse general IP network
- Use IP tunneling or IP over IP
  - Encapsulate IP packets inside other IP packets
  - Extra overhead

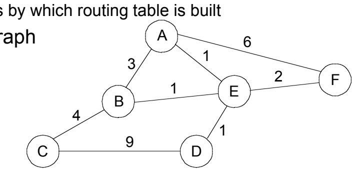


Routing

- Outline
  - Algorithms
  - Scalability

Overview

- Forwarding vs Routing
  - forwarding: to select an output port based on destination address and routing table
  - routing: process by which routing table is built
- Network as a Graph

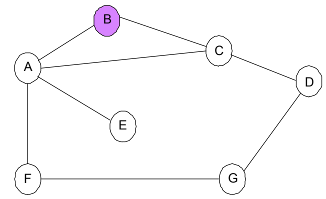


- Problem: Find lowest cost path between two nodes
- Factors
  - static: topology
  - dynamic: load

Technique 1: Distance Vector (RIP)

- Each node maintains a set of triples
  - (Destination, Cost, NextHop)
- Exchange updates directly connected neighbors
  - periodically (on the order of several seconds)
  - whenever table changes (called triggered update)
- Each update is a list of pairs:
  - (Destination, Cost)
- Update local table if receive a "better" route
  - smaller cost
  - came from next-hop
- Refresh existing routes; delete if they time out

Example



Destination	Cost	NextHop
A	1	A
C	1	C
D	2	C
E	2	A
F	2	A
G	3	A

Routing Loops

- Example 1
  - F detects that link to G has failed
  - F sets distance to G to infinity and sends update to A
  - A sets distance to G to infinity since it uses F to reach G
  - A receives periodic update from C with 2-hop path to G
  - A sets distance to G to 3 and sends update to F
  - F decides it can reach G in 4 hops via A
- Example 2: count to infinity problem
  - link from A to E fails
  - A advertises distance of infinity to E
  - B and C advertise a distance of 2 to E
  - B decides it can reach E in 3 hops; advertises this to A
  - A decides it can read E in 4 hops; advertises this to C
  - C decides that it can reach E in 5 hops…

Loop-Breaking Heuristics

- Set infinity to 16
- Split horizon
  - Don't send route updates to neighbor where your learnt the previous update from
- Split horizon with poison reverse
  - Send a negative cost back to the neighbor
  - Does not solve large loops
  - Waiting can help, but increases convergence delay

Technique 2: Link State (OSPF)

- Strategy
  - send to all nodes (not just neighbors) information about directly connected links (not entire routing table)
- Link State Packet (LSP)
  - id of the node that created the LSP
  - cost of link to each directly connected neighbor
  - sequence number (SEQNO)
  - time-to-live (TTL) for this packet

Link State (cont)

- Reliable flooding
  - store most recent LSP from each node
  - forward LSP to all nodes but one that sent it
  - generate new LSP periodically
    - increment SEQNO
  - start SEQNO at 0 when reboot
  - decrement TTL of each stored LSP
    - discard when TTL=0

Route Calculation

- Dijkstra's shortest path algorithm
- Let
  - *N* denotes set of nodes in the graph
  - *l* (*i*, *j*) denotes non-negative cost (weight) for edge (*i*, *j*)
  - *s* denotes this node
  - *M* denotes the set of nodes incorporated so far
  - *C*(*n*) denotes cost of the path from *s* to node *n*

```
M = {s}
for each n in N - {s}
   C(n) = l(s, n)
while (N != M)
   M = M union {w} such that C(w) is the minimum for
       all w in (N - M)
   for each n in (N - M)
      C(n) = MIN(C(n), C (w) + l(w, n ))
```


Metrics

- Original ARPANET metric
  - measures number of packets enqueued on each link
  - took neither latency or bandwidth into consideration
- New ARPANET metric
  - stamp each incoming packet with its arrival time (AT)
  - record departure time (DT)
  - when link-level ACK arrives, compute
    - Delay = (DT - AT) + Transmit + Latency
  - if timeout, reset DT to departure time for retransmission
  - link cost = average delay over some time period
- Fine Tuning
  - compressed dynamic range
  - replaced Delay with link utilization