Today's Lecture

- Routing
  - Distance Vector
  - Link State
- Global Routing

Routing (Revisited)

Outline
- Distance Vector
- Link State

Distance Vector
- Each node maintains a set of triples
  - (Destination, Cost, NextHop)

<table>
<thead>
<tr>
<th>Destination</th>
<th>Cost</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>Local</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>E</td>
</tr>
<tr>
<td>C</td>
<td>7</td>
<td>B</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>E</td>
</tr>
</tbody>
</table>

Exchange the vector (the table) of my current routing information.

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
  - How much do you know?
  - Distance vector: Just my neighbors and who they can reach
  - Link state: Everything on my network
Time t=0, Start Time

Perspective of A

A’s Routing Table

A : C=0 NH: *  
B : C=3 NH: B  
E : C=1 NH: E  
F : C=6 NH: F

Assume know our neighbors and link cost

Symmetry

Cost in either direction is the same

A sends to its neighbors

Routing Table

A : C=0 NH: Me  
B : C=3 NH: B  
E : C=1 NH: E  
F : C=6 NH: F

Vector of reachability

B,1  
F,1

Send this to our neighbors

E broadcasts to its neighbors

Distance Vector

A,1  
B,1  
D,1  
F,2

Vector = Here is my best cost to get to these locations if you come through me

Gather round….

• Send our distance vector
  – Cost to reach nodes through us
  – Two ways
    • Cost in the vector
    • Let external node put in cost
• When we receive a DV
  – Are any routes better than our own?
  – Could we take the cost of using that link and navigate via that node?

What happens at A?

Perspective of A

A’s Routing Table

A : C=0 NH: *  
B : C=3 NH: B  
E : C=1 NH: E  
F : C=6 NH: F

DV from E

A,1  
B,1  
D,1  
F,2

Compare new vs. current

That’s us, duh

3 vs. 1+1 via E
Inf vs. 1+1 via E
6 vs. 1+2 via E
What happens at A?

**A’s Routing Table**

<table>
<thead>
<tr>
<th>Node</th>
<th>Distance</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>C=0</td>
<td>NH: *</td>
</tr>
<tr>
<td>B</td>
<td>C=3</td>
<td>NH: B</td>
</tr>
<tr>
<td>E</td>
<td>C=1</td>
<td>NH: E</td>
</tr>
<tr>
<td>F</td>
<td>C=6</td>
<td>NH: F</td>
</tr>
</tbody>
</table>

**Perspective of A**

Convergence

- **Convergence**
  - Stop when no more changes
  - Non-convergence bad
    - We’ll see that in a moment
- **How?**
  - Only send on change of local table
  - Stop seeing broadcast, stop

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**
  - 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 reach E in 4 hops; advertises this to C
  - C decides that it can reach E in 5 hops…

Is DV used?

- **Yes**
  - RIP – Routing Information Protocol
  - BGP – Border Gateway Protocol
    - Sort of; BGP is kind of a hybrid
    - Various mobile routing protocols

Routing Information Protocol (RIP)

- Distributed along with BSD Unix
- Straightforward implementation of DV
- Updates sent every 30 seconds
- Link costs constant at 1 (16 = infinity)
Link State

• Strategy
  - Two-fold
  • Exchange HELLO msgs with neighbors
  • Broadcast changes to all of neighbor state

• Link State Packet (LSP)
  - ID of the node that created the LSP
  - Cost of link
  - Sequence number
  - TTL

Link State (cont)

• Changes → Reliably flood
  - 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 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 \( M \neq N \)
  
  \[
  M = M \cup \{w\} \text{ such that } C(w) \text{ is the minimum for all } w \text{ in } (N - M)
  \]
  for each \( n \) in \( (N - M) \)
  \[
  C(n) = \min(C(n), C(w) + l(w,n))
  \]

Flooding

Step 0: Set everyone’s weight to infinity
Add ourselves as the lowest cost node

Link State Example

Assume that A gets updates from everyone else via reliably flooded updates, i.e. A knows the topology of the graph
Link State Example

Who has the lowest total reachable cost from our covered net?
Lowest cost total reachable cost → E

E wins, add it to our covered graph

Who has the lowest total reachable cost from our covered net?
Lowest cost total reachable cost → B or D tied at 2

B wins, add it to our covered graph

Who has the lowest total reachable cost from our covered net?
Lowest cost total reachable cost → D at 2

D wins, add it to our covered graph

Who has the lowest total reachable cost from our covered net?
Lowest cost total reachable cost → F with a cost of 3

F wins, add it to our covered graph

Who has the lowest total reachable cost from our covered net?
Lowest cost total reachable cost → C with a cost of 6

C wins, all done
What does it mean?

<table>
<thead>
<tr>
<th>Target</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>E</td>
</tr>
<tr>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>F</td>
<td>E</td>
</tr>
</tbody>
</table>

OSPF

- Open Shortest Path First Protocol
- Authentication
- Additional hierarchy
- Load balancing

Quintessential link state protocol

Metrics

- Original ARPANET metric
  - measures number of packets queued on each link
  - took neither latency nor 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
    
    $\text{Delay} = (\text{DT} - \text{AT}) + \text{Transmit} + \text{Latency}$
  - if timeout, reset DT to departure time for retransmission
  - link cost = average delay over some time period
- Revised ARPANET metric
  - compressed dynamic range
  - replaced Delay with link utilization
- Practice
  - static metrics (e.g., 1/bandwidth)

Why might variable weights be bad?

Global Internet

Outline

- Subnetting
- Supernetting

How to Make Routing Scale

- Flat versus Hierarchical Addresses
- Inefficient use of Hierarchical Address Space
  - Class C with 2 hosts (2/254 = 0.78% efficient)
  - Class B with 256 hosts (256/65534 = 0.39% efficient)
- Too many networks
  - Routing tables do not scale
  - Route propagation protocols do not scale

Current routers – IPv4 – core $\rightarrow$ 160k entries

Internet Structure

Recent Past
Internet Structure

Today

Abilene – aka Internet 2

Subnetting

• Add another level to address/routing hierarchy: subnet
• Subnet masks define variable partition of host part
• Subnets visible only within site

<table>
<thead>
<tr>
<th>Class B address</th>
<th>Network number</th>
<th>Host number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1111111111111111</td>
<td>100000000</td>
<td></td>
</tr>
<tr>
<td>Subnet mask (255.255.255.0)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network number</th>
<th>Subnet ID</th>
<th>Host ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subnetted address</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Forwarding Algorithm

D = destination IP address
for each entry (SubnetNum, SubnetMask, NextHop)
  D1 = SubnetMask & D
  if D1 = SubnetNum
    if NextHop is an interface
      deliver datagram directly to D
    else
      deliver datagram to NextHop
  else
    use a default router if nothing matches
• Not necessary for all 1s in subnet mask to be contiguous
  - More likely than not, it will be
• Can put multiple subnets on one physical network
• Subnets not visible from the rest of the Internet

Supernetting

• Assign block of contiguous network numbers to nearby networks
• Called CIDR: Classless Inter-Domain Routing
• Represent blocks with a single pair (first_network_address, count)
• Restrict block sizes to powers of 2
• Use a bit mask (CIDR mask) to identify block size
• All routers must understand CIDR addressing

CIDR

Instead of advertising two networks upstream, just advertise one
Route Propagation

- Know a smarter router
  - hosts know local router
  - local routers know site routers
  - site routers know core router
  - core routers know everything
- Autonomous System (AS)
  - corresponds to an administrative domain
  - examples: University, company, backbone network
  - assign each AS a 16-bit number
- Two-level route propagation hierarchy
  - interior gateway protocol (each AS selects its own)
  - exterior gateway protocol (Internet-wide standard)

Popular Interior Gateway Protocols

- RIP: Route Information Protocol
  - developed for XNS
  - distributed with Unix
  - distance-vector algorithm
  - based on hop-count
- OSPF: Open Shortest Path First
  - recent Internet standard
  - uses link-state algorithm
  - supports load balancing
  - supports authentication

BGP-4: Border Gateway Protocol

- AS Types
  - Stub AS: has a single connection to one other AS
    - carries local traffic only
  - Multihomed AS: has connections to more than one AS
    - refuses to carry transit traffic
  - Transit AS: has connections to more than one AS
    - carries both transit and local traffic
  - Each AS has:
    - one or more border routers
    - one BGP speaker that advertises:
      - local networks
      - other reachable networks (transit AS only)
      - gives path information

Inter-Domain/Intra-Domain

- Edge router
- Core router

EGP: Exterior Gateway Protocol

- Overview
  - designed for tree-structured Internet
  - concerned with reachability, not optimal routes
- Protocol messages
  - Neighbor acquisition: one router requests that another be its peer; peers exchange reachability information
  - Neighbor reachability: one router periodically tests if the another is still reachable; exchange HELLO/ACK messages; uses a k-out-of-n rule
  - Routing updates: peers periodically exchange their routing tables (distance-vector)

Multi-Backbone Internet

- Large corporation
- Consumer ISP
- Small corporation
- Backbone service provider
- Peering point
BGP Example

- Speaker for AS2 advertises reachability to P and Q
  - network 128.96, 192.4.153, 192.4.32, and 192.4.3, can be reached directly from AS2
- Speaker for backbone advertises
  - networks 128.96, 192.4.153, 192.4.32, and 192.4.3 can be reached along the path (AS1, AS2).
- Speaker can cancel previously advertised paths

IP Version 6

- Features
  - 128-bit addresses (classless)
  - multicast
  - real-time service
  - authentication and security
  - autoconfiguration
  - end-to-end fragmentation
  - protocol extensions
- Header
  - 40-byte “base” header
  - extension headers (fixed order, mostly fixed length)
    - fragmentation
    - source routing
    - authentication and security
    - other options

Addresses