Scalable Routing

Outline
Routing 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: traffic load and link failure
Distance Vector Algorithm

- Each node maintains a set of triples
  - (Destination, Cost, NextHop)
- Directly connected neighbors exchange updates
  - 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
  - higher cost from the current NextHop (e.g. link failures)
- Refresh existing routes; delete if they time out

Example

<table>
<thead>
<tr>
<th>Destination</th>
<th>Cost</th>
<th>NextHop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>C</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>E</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>F</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>G</td>
<td>3</td>
<td>A</td>
</tr>
</tbody>
</table>

pp. 275 ~ 277, Tables 4.5 – 4.8
Routing Loops

- **Example 1: Fast Convergence**
  - 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” due to the loop A-B-C**
  - link from A to E fails
  - A advertises distance of infinity to E
  - B and C still advertise a distance of 2 to E *periodically*
    - NextHop is not in updates
    - Timing: sent before B, C receive (E, ∞) from A, received after (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...

Loop-Breaking Heuristics

- **Set infinity to 16**
  - Nodes can be reached beyond 16 links.
  - RIP (Routing Information Protocol)

- **Split horizon**
  - For the triple (dest, cost, X), don’t include (dest, cost) in the update sent to X.
  - with poison reverse: for the triple (dest, cost, X), include (dest, ∞) in the update sent to X.
  - Solve loops involving two nodes (e.g. G x A ↔ B)
  - Cannot but solve loops of three or more nodes
Link State

• 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 (see seqno) 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
  - See example on pp. 286 - 287
- 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
  - \(C(n)\) denotes cost of the path from \(s\) to node \(n\)
  - \(M\) denotes the set of nodes incorporated so far

\[
\begin{align*}
M &= \{s\} \\
\text{for each } n \text{ in } N - \{s\} & \quad C(n) = l(s, n) \\
\text{while } (N \neq M) & \quad M = M \cup \{w\} \text{ such that } C(w) \text{ is the minimum for all } w \text{ in } (N - M) \\
\text{for each } n \text{ in } (N - M) & \quad C(n) = \min(C(n), C(w) + l(w, n))
\end{align*}
\]

Metrics

- Assigning “1” to each link is inefficient
  - Satellite links have higher propagation delays.
  - Links have different capacities
    - OSPF uses \(100Mbps / C\)
  - Links have dynamic traffic loads
- New ARPANET metric
  - for each packet, record its arrival time \((AT)\) and record departure time \((DT)\)
  - when link-level ACK arrives, compute
    \[
    \text{Delay} = (DT - AT) + \text{Transmit} + \text{Latency}
    \]
  - link cost = average delay over some time period
  - if timeout (link-level ACK used), reset \(DT\) to departure time for retransmission
    - \(DT - AT\) captures not only queuing delay, but also the link reliability
**Metrics (cont)**

- **Problems:**
  - under heavy load, DT – AT dominates the delay. Traffic moves back and forth between links
  - A 56 kbps looks too costly than a 9.6 kbps terrestrial link (due to the long delay), making its bandwidth underutilized.

- **Revised Metrics (p.293)**

![Graph showing different link utilizations](image)

**DV vs. LS**

- LS is more stable and robust
  - With DV, incorrect computation can spread to entire network.
- LS avoids loops better
- LS converges faster than DV
- LS reveals the complete topology.
- DV requires less memory and CPU time
  - maintains neighbor states only
  - no Dijkstra’s algorithm
  - Since LS floods LSP to entire network, *seqno* and hence *checksum* are introduced to guarantee consistency
Internet Structure Today

- Large corporation networks can be connected to Backbone.
- Consumers can be connected to ISPs
- Many providers arrange to interconnect to each other at a single “peering” point.

How to Make Routing Scale

- Flat versus Hierarchical Addresses
- Inefficient use of Hierarchical Address Space
  - class C with 2 hosts \( \frac{2}{255} = 0.78\% \) efficient
  - class B with 256 hosts \( \frac{256}{65535} = 0.39\% \) efficient
- Still Too Many Networks
  - routing tables and route propagation protocols do not scale
- Subnetting
  - divide a “large” network number (e.g. class B) into smaller network spaces for physical networks with small numbers (< 65535) of hosts.
- Supernetting
  - aggregate “small” network numbers (e.g. class C) into a “larger” network number for a physical network with more than 255 hosts
Subnetting

- Add another level to address/routing hierarchy: *subnet*
- *Subnet masks* define variable partition of host part
  - For networks with small number of hosts.
  - Do not have to align with byte boundary
- Subnets visible only within site
  - routing scalability

<table>
<thead>
<tr>
<th>Network number</th>
<th>Host number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Class B address

<table>
<thead>
<tr>
<th>111111111111111111111111</th>
<th>00000000</th>
</tr>
</thead>
</table>

Subnet mask (255.255.255.0)

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

Subnetted address

Subnet Example

Subnet mask: 255.255.255.128
Subnet number: 128.96.34.0

<table>
<thead>
<tr>
<th>Subnet number</th>
<th>Subnet Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.96.34.128</td>
<td>255.255.255.128</td>
</tr>
<tr>
<td>128.96.33.14</td>
<td>255.255.255.0</td>
</tr>
</tbody>
</table>

Forwarding table at router R1

<table>
<thead>
<tr>
<th>Subnet Number</th>
<th>Subnet Mask</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.96.34.0</td>
<td>255.255.255.128</td>
<td>Interface 0</td>
</tr>
<tr>
<td>128.96.34.128</td>
<td>255.255.255.128</td>
<td>Interface 1</td>
</tr>
<tr>
<td>128.96.33.0</td>
<td>255.255.255.0</td>
<td>R2</td>
</tr>
</tbody>
</table>
Forwarding Algorithm

- External routers only see class B network number: 128.96
  - One entry is kept for all hosts under 128.96
- Internal routers and hosts use subnet masks:
  - (SubnetNum, SubnetMask, NextHop)
  - Routers search for a match:
    - dest IP & SubnetMask == SubnetNum ?
    - (SubnetNum & SubnetMask == SubnetNum)
  - Sending hosts use the above to see whether the dest IP is in the local subnet (e.g. H1 → H2).
  - Use a default router if nothing matches

Supernetting

- Assign block of contiguous network numbers to nearby networks
  - Called CIDR: Classless Inter-Domain Routing
- Use a bit mask (CIDR mask) to identify block size
- All routers must understand CIDR addressing
- Efficient address allocation and Scalable Routing
- Used by BGP
- Forwarding: longest prefix match based on PATRICIA tree
Example: 2 levels of Supernetting

- Corporation Y: 11000000 00000100 0000
- Corporation X:
  - Class C numbers: 192.4.16, …… 192.4.32 → 11000000 00000100 0001
- ……
- AS: 11000000 000001

Route Propagation

- Know a smarter router
  - hosts know local default 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**
  - distance-vector algorithm
  - based on hop-count
- **OSPF: Open Shortest Path First**
  - recent Internet standard
  - uses link-state algorithm
  - supports authentication
  - supports load balancing
    - Install routes with same costs. Attempt to send approximately the same amount of traffic along each of the routes. (e.g. destination-based)

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 but 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
  - **BGP speakers** that advertise:
    - local networks
    - other reachable networks (transit AS only)
    - gives path information
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

![BGP Diagram]

- 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

- Classless 128-bit addresses
  - Every atom in the universe can have an IP address.
- IPv4-compatible IPV6 address: zero-extend IPv4 address to 128 bits
- 010... Provider-based Unicast Address
  - Similar to CIDR in IPv4
  - A provider with few customers could have a longer
    prefix (less address space)
  - All addresses in Europe, for example, can have a
    common Registry ID, for routing scalability.

<table>
<thead>
<tr>
<th>010</th>
<th>RegistryID</th>
<th>ProviderID</th>
<th>SubscriberID</th>
<th>SubnetID</th>
<th>InterfaceID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**IPv6 Header**

- **Version = 6**
- **QoS with Priority and Flow Label**
- **NextHeader: Protocol or Options**
  - E.g. extension header for IPv6 fragmentations

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>4</th>
<th>12</th>
<th>16</th>
<th>24</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Version</td>
<td>TrafficClass</td>
<td>FlowLabel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PayloadLen</td>
<td>NextHeader</td>
<td>HopLimit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SourceAddress</td>
<td>DestinationAddress</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8</th>
<th>16</th>
<th>29</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NextHeader</td>
<td>Reserved</td>
<td>Offset</td>
</tr>
</tbody>
</table>

Ident