Goals of congestion control

- Use allocated bandwidth efficiently
- Avoid sending so quickly that the network has to drop packets
- Avoid sending so slowly that the network is underutilized

Approach taken by TCP

- Congestion window limits outstanding packets
- Adjust congestion window in response to packet losses (AIMD)
- Slow start
- Fast retransmit/fast recovery

New topic: routing

How do I get there from here?

Routing within a network/organization
- A single administrative domain

Overall goals
- Provide intra-network connectivity
- Adapt quickly to failures or topology changes
- Optimize use of network resources

Problem statement
- Network is a directed graph G=(V,E)
- Routers are vertices, links are edges labeled with some metric
  - For simplicity ignore hosts, they are part of each V
- For each V, find the shortest path to every other V

Routing is essentially a problem in graph theory
- Find “best” path between every pair of vertices

Network as a Graph

- X = router
- = link
- 1 = cost

Intra-domain routing

Overview

- Intro & Design choices
- Intra-domain routing
  - Distance vector
  - Link state
- Inter-domain routing
  - Policy
Routing Questions

- How to choose best path?
  - Defining “best” can be slippery
- How to scale to millions of users?
  - Minimize control messages and routing table size
- How to adapt to failures or changes?
  - Node and link failures, plus message loss

What does a router do?

- Forwarding
  - Move packet from input link to the appropriate output link
  - Purely local computation
  - Must go be very fast (executed for ever packet)
- Routing
  - Doing work so you’re sure that the “next hop” actually leads to the destination
  - Global decisions; distributed computation and communication
  - Can go slower (only important when topology changes)

What’s in a Routing Table?

- The routing table at A, for example, lists at a minimum the next hops for the different destinations

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

Kinds of routing/forwarding

- Source routing
  - Complete path in packet
  - Host computes path
  - Must know global topology and detect failures
  - Packet contains complete ordered path information
  - i.e. node A then D then X then J...
  - Requires variable length path header
- Forwarding
  - Router looks up next hop in packet header, strips it off and forwards remaining packet
  - Very quick forwarding, no lookup required
  - Very flexible
  - In practice
    - ad hoc networks (DSR), SANs (Myrinet), and for debugging on the Internet (LSR, SSR)

- Virtual circuits
  - Complete path in packet
  - Set up path out-of-band and store path identifier in routers
  - Local path identifier in packet
  - Router looks up address in forwarding table
  - Forwarding table contains (address, next-hop) tuples

- Destination-based routing
  - Router looks up address in forwarding table
  - Forwarding table contains (address, next-hop) tuples

In practice
- ATM: fixed VC identifiers and separate signaling code
- MPLS: ATM meets the IP world (why? traffic engineering)
**Destination-based routing**

- Routing
  - All addresses are globally known
  - No connection setup
  - Host sends packet with destination address in header
  - No path state; only routers need to worry about failure
  - Distributed routing protocol used to routing tables
- Forwarding
  - Router looks up destination in table
  - Must keep state proportional to destinations rather than connections
  - (Address, next-hop) tuple
  - Lookup address, send packet to next-hop link
  - All packets follow same path to destination
- In Practice: IP routing

**Three approaches to routing**

- Static
  - Type in the right answers and hope they are always true
- Distance vector
  - Tell your neighbors when you know about everyone
- Link state
  - Tell everyone what you know about your neighbors

**Distance Vector routing**

- Assume
  - Each router knows own address & cost to reach neighbors
- Goal
  - Calculate routing table containing next-hop information for every destination at each router
- Distributed Bellman-Ford algorithm
  - Each router maintains a vector of costs to all destinations
  - Initialize neighbors with known cost, others with infinity
  - Periodically send copy of distance vector to neighbors
  - On reception of a vector
  - If neighbor’s path to a destination is shorter, switch to it

**Initial conditions**

<table>
<thead>
<tr>
<th>Node</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>7</td>
<td>∞</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>∞</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>∞</td>
<td>∞</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>8</td>
<td>∞</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

**E receives D’s vector**

- I’m 2 from C, 0 from D and 2 from E
- D is 2 away, 2+2<∞, so best path to C is 4...---

**A receives B’s vector**

- I’m 7 from A, 8 from B, 1 from X & 8 from D
- B is 7 away, 7+1<∞, so best path to C is 8...
A receives E’s vector

I’m 1 from A, 8 from B, 4 from C, 2 from D & 0 from E

E is 1 away, 4+1<8 so
C is 5 away, 1+2<∞ so D is 3 away

Final state

View from a node (B)

Link failure

Problems: Count to Infinity

Why?

- Updates don’t contain enough information
- Can’t totally order bad news above good news
- B’s accepts A’s path to C that is implicitly through B!
- Aside: this also causes delays in convergence
**Solutions**

- **Hold downs**
  - As metric increases, delay propagating information
  - Limitation: Delays convergence

- **Split horizon**
  - Never advertise a destination through its next hop
  - Poison reverse: Send negative information when advertising a destination through its next hop
    - A advertises C to B
  - Limitation: Only works for "loop"’s of size 2

- **Loop avoidance**
  - Full path information in route advertisement
  - Explicit queries for loops (e.g. DUAL)

---

**How split horizon/pv fails**

![Diagram of network topology]

- A tells B & C that D is unreachable
- B tells C that D is unreachable
- B tells A that D is unreachable with cost=3 (since route is through C, split horizon doesn’t apply)
- A tells C that D is reachable through A (cost=4)
- Etc…

---

**Other issues**

- When to send route updates?

- **Periodically**
  - Limits granularity of failure recovery
  - Global synchronization can cause packet loss

- **Jittered**
  - Random offset from periodic deals with synchronization problem

- **Triggered**
  - Send updates immediately when metric changes
  - Converges more quickly, but causes flood of packets

---

**Routing Information Protocol (RIP)**

- **DV protocol with hop count as metric**
  - Infinity value is 16 hops; limits network size
  - Includes split horizon with poison reverse

- **Routers send vectors every 30 seconds**
  - With triggered updates for link failures
  - Time-out in 180 seconds to detect failures

- **RIPv1** specified in RFC1058
  - www.ietf.org/rfc/rfc1058.txt

- **RIPv2** (adds authentication etc.) in RFC1388
  - www.ietf.org/rfc/rfc1388.txt

---

**Key Concepts**

- Routing is a global process, forwarding is local one

- **The Distance Vector algorithm and RIP**
  - Simple and distributed exchange of shortest paths.
  - Weak at adapting to changes (loops, count to infinity)

---

**For next time...**

- No new reading… although review the section about link state routin

- Finally, projects 😊