CS244a: An Introduction to Computer Networks

Handout 5: Internetworking and Routing

Nick McKeown
Professor of Electrical Engineering and Computer Science, Stanford University
nickm@stanford.edu
http://www.stanford.edu/~nickm

Outline

Techniques
- Naïve: Flooding
- Distance vector: Distributed Bellman Ford Algorithm
- Link state: Dijkstra's Shortest Path First-based Algorithm

Routing in the Internet
- Hierarchy and Autonomous Systems
- Interior Routing Protocols: RIP, OSPF
- Exterior Routing Protocol: BGP

Multicast Routing
Routing is a very complex subject, and has many aspects. Here, we will concentrate on the basics.

Nick McKeown
Professor of Electrical Engineering and Computer Science, Stanford University
nickm@stanford.edu
http://www.stanford.edu/~nickm
The Problem

How does R₁ choose a next-hop on the path towards host B?

Routing Metrics

- Metrics
  - Delay to send an average size packet (Make high speed links attractive, but closeness counts)
  - Bandwidth
  - Link utilization
  - Stability: Is a link (or path) up or down?

- Today: about 1/3 of Internet routes are asymmetric
Example network

Objective: Determine the route from A to B that minimizes the path cost.

Examples of link cost: Distance, data rate, price, congestion/delay, ...

Example network

In this simple case, solution is clear from inspection.
So what about this network...!?  
*The public Internet in 1999*

---

**Technique 1: Naïve Approach**

**Flood!** -- Routers forward packets to all ports except the ingress port.

---

**Advantages:**
- Simple.
- Every destination in the network is reachable.

**Disadvantages:**
- Some routers receive a packet multiple times.
- Packets can go round in loops forever.
- Inefficient.
Spanning Trees

**Objective:** Find the lowest cost route from each of 
$(R_1, ..., R_7)$ to $R_8$.

A Spanning Tree

- The solution is a spanning tree with $R_8$ as the root of the tree.
- Tree: There are no loops.
- Spanning: All nodes included.
- We’ll see two algorithms that build spanning trees automatically:
  - The distributed Bellman-Ford algorithm
  - Dijkstra’s shortest path first algorithm
Technique 2: Distance Vector
The Distributed Bellman-Ford Algorithm

1. Let $X_n = (C_1, C_2, ..., C_7)$ where: $C_i =$ cost from $R_i$ to $R_n$.
2. Set $X_0 = (\infty, \infty, \infty, \infty, \infty)$.
3. Every $T$ seconds, router $i$ sends $C_i$, for all $i$, to its neighbors. This is the "Distance vector".
4. If router $i$ is told of a lower cost path to $R_n$, it updates $C_i$. Hence, $X_{n+1} = f(X_n)$
   where $f(.)$ determines the next step improvement.
5. If $X_{n+1} \neq X_n$ then goto step (3).
6. Stop.

Bellman-Ford Algorithm

Example
Bellman-Ford Algorithm

Questions:
1. How long can the algorithm take to run?
2. How do we know that the algorithm always converges?
3. What happens when link costs change, or when routers/links fail?

Topology changes make life hard for the Bellman-Ford algorithm...
A Problem with Bellman-Ford

“Bad news travels slowly”

Consider the calculation of distances to $R_4$:

<table>
<thead>
<tr>
<th>Time</th>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3,$R_2$</td>
<td>2,$R_3$</td>
<td>1,$R_4$</td>
</tr>
<tr>
<td>1</td>
<td>3,$R_2$</td>
<td>2,$R_3$</td>
<td>3,$R_2$</td>
</tr>
<tr>
<td>2</td>
<td>3,$R_2$</td>
<td>4,$R_3$</td>
<td>3,$R_2$</td>
</tr>
<tr>
<td>3</td>
<td>5,$R_2$</td>
<td>4,$R_3$</td>
<td>5,$R_2$</td>
</tr>
</tbody>
</table>

... “Counting to infinity” ...

R$_3$ → R$_4$ fails

Counting to Infinity Problem

Solutions

1. Set infinity = “some small integer” (e.g. 16). Stop when count = 16.
2. Split Horizon: Because $R_2$ received lowest cost path from $R_3$, it does not advertise cost to $R_3$.
4. There are many problems with (and fixes for) the Bellman-Ford algorithm.
Technique 3: Link State
Dijkstra’s Shortest Path First Algorithm

- Routers send out update messages whenever the state of an incident link changes.
  - Called “Link State Updates”

- Based on all link state updates received each router calculates lowest cost path to all others, starting from itself.
  - Use Dijkstra’s single-source shortest path algorithm
  - Assume all updates are consistent

- At each step of the algorithm, router adds the next shortest (i.e. lowest-cost) path to the tree.

- Finds spanning tree rooted at the router.

Reliable Flooding of LSP

- The Link State Packet:
  - The ID of the router that created the LSP
  - List of directly connected neighbors, and cost
  - Sequence number
  - TTL

- Reliable Flooding
  - Resend LSP over all links other than incident link, if the sequence number is newer. Otherwise drop it.

- Link State Detection:
  - Link layer failure
  - Loss of “hello” packets
**Dijkstra's Shortest Path First Algorithm**

*Example*

**Step 1:** Shortest path set, $S = \{R_9\}$. Candidate set, $C = \{R_3, R_5, R_7, R_6\}$

**Step 2:**
- $S = \{R_6, R_8\}$
- $C = \{R_3, R_5, R_7, R_6\}$

**Step 3:**
- $S = \{R_6, R_7, R_9\}$
- $C = \{R_3, R_5, R_7, R_6\}$

**Step 4:**
- $S = \{R_6, R_7, R_9, R_8\}$
- $C = \{R_3, R_5, R_7, R_6\}$

---

**Dijkstra’s SPF Algorithm**

Step 8: $S = \{R_8, R_5, R_6, R_7, R_2, R_1, R_3, R_4\}$,
- $C = \{\}.}$
Distance Vector vs Link State

- **Messages**
  - Size: small with LS; potentially large with DV
  - Exchange: LS \(\rightarrow\) flood; DV \(\rightarrow\) only to neighbors

- **Space requirements**
  - LS maintains entire topology
  - DV maintains only neighbor state

- **Robustness:**
  - LS can broadcast incorrect/corrupted LSP
  - DV can advertise incorrect paths to all destinations
  - Incorrect calculation can spread to entire network

- **Examples (coming up later):**
  - LS: OSPF
  - DV: RIP, RIP2
Outline

Techniques
- Flooding
- Distributed Bellman Ford Algorithm
- Dijkstra's Shortest Path First Algorithm

Routing in the Internet
- Hierarchy and Autonomous Systems
- Interior Routing Protocols: RIP, OSPF
- Exterior Routing Protocol: BGP

Multicast Routing

Routing in the Internet

The Internet uses hierarchical routing
- The Internet is split into Autonomous Systems (AS's)
  - Examples of AS's: Stanford (32), HP (71), MCI Worldcom (17373)
  - Try: `whois -h whois.arin.net "MCI Worldcom"`
- Within an AS, the administrator chooses an Interior Gateway Protocol (IGP)
  - Examples of IGPs: RIP (rfc 1058), OSPF (rfc 1247)
- Between AS's, the Internet uses an Exterior Gateway Protocol
  - AS's today use the Border Gateway Protocol, BGP-4 (rfc 1771)
Routing in the Internet

Routing within a Stub AS

- There is only one exit point, so routers within the AS can use default routing.
  - Each router knows all Network IDs within AS.
  - Packets destined to another AS are sent to the default router.
  - Default router is the border gateway to the next AS.
- Routing tables in Stub AS's tend to be small.
Interior Routing Protocols

- **RIP**
  - Uses distance vector (distributed Bellman-Ford algorithm).
  - Updates sent every 30 seconds.
  - No authentication.
  - Originally in BSD UNIX.
  - Widely used for many years; not used much anymore.

- **OSPF**
  - Link-state updates sent (using flooding) as and when required.
  - Every router runs Dijkstra's algorithm.
  - Authenticated updates.
  - Autonomous system may be partitioned into "areas".
  - Widely used.

Exterior Routing Protocols

**Problems:**

- **Topology:** The Internet is a complex mesh of different AS's with very little structure.
- **Autonomy of AS's:** Each AS defines link costs in different ways, so not possible to find lowest cost paths.
- **Trust:** Some AS's can't trust others to advertise good routes (e.g. two competing backbone providers), or to protect the privacy of their traffic (e.g. two warring nations).
- **Policies:** Different AS's have different objectives (e.g. route over fewest hops; use one provider rather than another).
Border Gateway Protocol (BGP-4)

- BGP is not a link-state or distance-vector routing protocol.
  - Instead, BGP uses "Path vector".
- BGP advertises complete paths (a list of AS's).
  - Also called AS_PATH (this is the path vector)
  - Example of path advertisement:
    "The network 171.64/16 can be reached via the path (AS1, AS5, AS13)."
- Paths with loops are detected locally and ignored.
- Local policies pick the preferred path among options.
- When a link/router fails, the path is "withdrawn".

Customers and Providers

Customer pays provider for access to the Internet.
Customer may not always need BGP.
Customer-Provider Hierarchy

The Peering Relationship

Peers provide transit between their respective customers
Peers do not provide transit between peers
Peers (often) do not exchange $$$
**BGP Messages**

- **Open**: Establish a BGP session.
- **Keep Alive**: Handshake at regular intervals.
- **Notification**: Shuts down a peering session.
- **Update**: Announcing new routes or withdrawing previously announced routes.

**BGP announcement = prefix + path attributes**

- Attributes include: Next hop, AS Path, local preference, Multi-exit discriminator, ...
  - Used to select among multiple options for paths

---

**BGP Route Selection Summary**

<table>
<thead>
<tr>
<th>Highest Local Preference</th>
<th>Enforce relationships</th>
<th>E.g. prefer customer routes over peer routes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shortest AS PATH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lowest MED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>i-BGP &lt; e-BGP</td>
<td></td>
<td>traffic engineering</td>
</tr>
<tr>
<td>Lowest IGP cost to BGP egress</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lowest router ID</td>
<td></td>
<td>Throw up hands and break ties</td>
</tr>
</tbody>
</table>
**ASPATH Attribute**

AS 1239
135.207.0.0/16
AS Path = 1239 7018 6341

AS 1755
135.207.0.0/16
AS Path = 1239 7018 6341

AS 7018
135.207.0.0/16
AS Path = 7018 6341

AS 1129
Global Access

AS 12654
RIPE NCC RIS project

135.207.0.0/16
Pick shorter AS path

AS 3549
Global Crossing

135.207.0.0/16

So Many Choices...

Frank’s Internet Barn

AS 4

AS 3

AS 2

AS 1

Which route should Frank pick to 13.13.0.0/16?
Frank's Choices...

- Route learned from customer preferred over route learned from peer, preferred over route learned from provider.

- Set appropriate "local pref" to reflect preferences: Higher Local preference values are preferred.

Traceroute with ASNs

- TTLLEFT trace to 216.35.221.77:80/tcp
  - [AS7011] [ELI-NETWORK-ELIX] eli-gw.home.micronet.net (65.73.254.1) 20.2ms 2
  - [AS5650] [ELI-NETBLK98] 209.210.114.245 20.2ms 3
  - [AS5650] [ELI-NETBLK99] s3-1-0-136.gw01.phnx.eli-net (216.190.111.161) 20.3ms 4
  - [AS5650] [ELI-2-NETBLK99] asp-2-0.cr01.phnx.eli-net (208.186.20.118) 20.3ms 5
  - [AS5650] [ELI-NETBLK99] p6-0.cr01.lsan.eli.net (207.173.114.29) 40.3ms 6
  - [AS5650] [ELI-NETBLK99] p0-0.cr02.sntd.eli.net (207.173.114.54) 40.3ms 7
  - [AS5650] [ELI-2-NETBLK99] aps-2-0.cr01.sntd.eli.net (208.186.21.133) 40.3ms 8
  - [AS5650] [ELI-NETBLK99] s0-0-0-0-0.sntd.eli.net (207.173.114.138) 40.3ms 9
  - [AS5650] [SAVVIS] bpr2-ge-5-3-0.pascaltpaix.savvis.net (206.24.241.229) 40.2ms 10
  - [AS707] [SAVVIS] dcr2-ao-3-3-0.sanfrancisco.savvis.net (208.172.147.93) 40.3ms 11
  - [AS827] [SAVVIS] dcr1-loopback.washington.savvis.net (206.24.226.99) 100.4ms 12
  - [AS707] [SAVVIS] bhr1-pos-10-0.sterlingsc2.savvis.net (206.24.227.106) 100.5ms 13
  - [AS827] [SAVVIS] crr1-w240.sterlingsc2.savvis.net (216.33.96.58) 100.5ms
  - [neglected] no reply packets received from TTL 14 15
  - [AS707] [SAVVIS] [target] 216.35.221.77:80 100.5ms
Who owns an address block?

Prompt: `whois 216.35.221.77`

| OrgName:    | Savvis          |
| OrgID:      | SAVVI-2         |
| Address:    | 3300 Regency Parkway |
| City:       | Cary           |
| StateProv:  | NC             |
| PostalCode: | 27511          |
| Country:    | US             |
| ReferralServer: | whois://whois.exodus.net:4321/ |
| NetRange:   | 216.32.0.0 - 216.35.255.255 |
| CIDR:       | 216.32.0.0/14   |
| NetName:    | SAVVIS          |
| NetHandle:  | NET-216-32-0-0-1 |
| Parent:     | NET-216-0-0-0-0 |
| NetType:    | Direct Allocation |
| NameServer: | DNS01.SAVVIS.NET |
| NameServer: | DNS02.SAVVIS.NET |
| NameServer: | DNS03.SAVVIS.NET |
| NameServer: | DNS04.SAVVIS.NET |
| Comment:    | RegDate: 1998-07-30 |
|             | Updated: 2004-10-07 |

# ARIN WHOIS database, last updated 2005-01-17 19:10
# Enter ? for additional hints on searching ARIN's WHOIS database.

Prompt: `whois SU-NET`

| OrgName:     | Stanford University |
| OrgID:       | STANFORD           |
| Address:     | Pine Hall 115      |
| City:        | Stanford           |
| StateProv:   | CA                |
| PostalCode:  | 94305             |
| Country:     | US                |
| NetRange:    | 128.12.0.0 - 128.12.255.255 |
| CIDR:        | 128.12.0.0/16      |
| NetName:     | SU-NET             |
| NetHandle:   | NET-128-12-0-0-1   |
| Parent:      | NET-128-0-0-0-0    |
| NetType:     | Direct Assignment  |
| NameServer:  | ARGUS.STANFORD.EDU |
| NameServer:  | AVALLONE.STANFORD.EDU |
| NameServer:  | ATALANTE.STANFORD.EDU |
| Comment:     | RegDate: 2000-05-01 |
|             | Updated: 2000-05-01 |
| TechHandle:  | JK535-ARIN        |
| TechName:    | Kahn, Jay         |
| TechPhone:   | +1-650-723-7515   |
| TechEmail:   | security@stanford.edu |

# ARIN WHOIS database, last updated 2005-01-17 19:10

To receive AS from a particular route arbiter: 
`Whois -h whois.ra.net 128.125.0.0`

North American AS Numbers and Addresses

DNS Top level domains and delegates IP Address blocks

Organizations

DNS Top level domains and delegates IP Address blocks
Multicast Routing

- Applications that benefit from multicast.
- Trees, addressing and forwarding.
- Multicast routing
  - Distance Vector-based (DVMRP, PIM-DM)
  - Link-state based (MOSPF)
  - Rendezvous-based (PIM-SM, CBT)
- Some interesting questions...

Multicast Trees
The basic idea

Multiple unicasts

Single multicast
Applications that need multicast

- **One way, single sender: “one-to-many”**
  - TV
  - Non-interactive learning
  - Database update
  - Information dispersal (e.g. Pointcast)
  - Software updates/patches

- **Two way, interactive, multiple sender: “many-to-many”**
  - Teleconference
  - Interactive learning

**Multicast Routing**

- A multicast tree is a *spanning tree* with the sender at the root, spanning all the members of the group.
Multicast Trees and Addressing

- All members of the group share the same "Class D" Group Address.
- An end station may be the member of multiple groups.
- An end-station "joins" a multicast group by (periodically) telling its nearest router that it wishes to join (uses IGMP - Internet Group Management Protocol).
- Routers maintain “soft-state” indicating which end-stations have subscribed to which groups.
Multicast Trees
Multiple source trees

Sender/Speaker
Multicast Group \((S_2,G)\)

Multicast Forwarding is Sender-specific

<table>
<thead>
<tr>
<th>Group Address</th>
<th>Src Address</th>
<th>Src Interface</th>
<th>Dist Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G)</td>
<td>(S_1)</td>
<td>1</td>
<td>2,3</td>
</tr>
<tr>
<td>(S_2)</td>
<td></td>
<td>2</td>
<td>1,3</td>
</tr>
</tbody>
</table>

Winter 2008
CS244a Handout 5
Outline

- Applications that need multicast.
- Trees, addressing and forwarding.
- Multicast routing
  - Distance Vector-based: DVMRP, PIM-DM
  - Link-state based: MOSPF
  - Rendezvous-based: PIM-SM, CBT
- Some interesting problems...

Distance-vector Multicast

RPB: Reverse-Path Broadcast

- Uses existing unicast shortest path routing table.
  - Computed using Distance vector
- If packet arrived through interface that is the shortest path to the packet’s SA, then forward packet to all interfaces.
- Else drop packet.
Distance-vector Multicast

RPB: Reverse-Path Broadcast

Sender/Speaker
Multicast Group (S,G)

Designated Parent Router:
One parent router picked per LAN (one “closest” to source).

Sender/Speaker
Multicast Group (S,G)

Q: Is it shortest path from source?
Distance-vector Multicast

RPM: Reverse-Path Multicast

- RPM = RPB + Prune
- RPB used when a source starts to send to a new group address.
- Routers that are not interested in a group send prune messages up the tree towards source.
- Prunes sent implicitly by not indicating interest in a group.
- DVMRP works this way.

Protocol Independent Multicast

- PIM-DM (Dense Mode) uses RPM.
- PIM-SM (Sparse Mode) designed to be more efficient than DVMRP
  - Key idea: use a rendezvous point (RP) so multiple sources can share the same tree
  - Routers explicitly join multicast tree by sending unicast Join and Prune messages.
  - Routers join a multicast tree via an RP for each group.
  - Several RPs per domain (picked in a complex way).
  - Provides either:
    - Shared tree for all senders (default)
    - Source-specific tree
**PIM-SM**

Unicast to R3: (S,G)
- Source knows to send all G packets to RP.

Source "tunnels" mcast-in-ucast packets to RP.

Unicast to RP: (*, G)
- RP knows R1 has joined
- R2 learns to send (*, G) packets to R1

RP unwraps mcast pkt and forwards to local tree.

Sender/Source

Optional Source-specific join to bypass RP router:
- Routers along the way learn new path for (S,G)

**PIM-SM**

Sender/Source
Outline

- Applications that need multicast.
- Trees, addressing and forwarding.
- Multicast routing
  - Distance Vector-based: DVMRP, PIM-DM
  - Link-state based: MOSPF
  - Rendezvous-based: PIM-SM, CBT
- Some interesting problems...

Multicast: Interesting Questions

- How to make multicast reliable?
- How to implement flow-control?
- How to support/provide different rates for different end users?
- How to secure a multicast conversation?

- Will multicast become widespread?
  - Several protocols for multicast routing in IP
    - But IP multicast is not enabled in routers!
    - No one uses IP multicast, really
    - End-system based, overlay-based approaches more popular