Routing: Network Layer Part II

- Routing Algorithms:
  - Link state vs. Distance Vector
- Routing in the Internet:
  - Intra-AS vs. Inter-AS routing
  - Intra-AS: RIP and OSPF
  - Inter-AS: BGP and Policy Routing
- MPLS

Readings: Textbook: Chapter 4: Sections 4.2-4.3, 4.5-4.6

IP Forwarding Process

1. Remove a packet from an input queue
2. Check for sanity, decrement TTL field
3. Match packet's destination to a table entry
4. Place packet on correct output queue

IP Forwarding Table

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net A</td>
<td>Router 1</td>
<td>INT 7</td>
</tr>
<tr>
<td>Net B</td>
<td>Direct</td>
<td>INT 4</td>
</tr>
<tr>
<td>Net C, Host</td>
<td>Router 2</td>
<td>INT 3</td>
</tr>
<tr>
<td>Net C</td>
<td>Router 1</td>
<td>INT 7</td>
</tr>
</tbody>
</table>

A destination is usually a network. May also be a host, or a “gateway of last resort” (default)

The next hop is either a directly connected network or a router on a directly connected network

A physical interface

How Are Forwarding Tables Populated to Implement Routing?

<table>
<thead>
<tr>
<th>Statically</th>
<th>Dynamically</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator manually configures forwarding table entries</td>
<td></td>
</tr>
<tr>
<td>Routers exchange network reachability information using ROUTING PROTOCOLS: Routers use this to compute best routes</td>
<td></td>
</tr>
<tr>
<td>+ More control</td>
<td></td>
</tr>
<tr>
<td>+ Not restricted to destination-based forwarding</td>
<td></td>
</tr>
<tr>
<td>- Doesn’t scale</td>
<td></td>
</tr>
<tr>
<td>- Slow to adapt to network failures</td>
<td></td>
</tr>
</tbody>
</table>

In practice: a mix of these, Static routing mostly at the “edge”

Dynamic Routing: Intra- vs. Inter-AS

IGP = Interior Gateway Protocol
Metric based: OSPF, IS-IS, RIP, EIGRP (Cisco)

EIGP = Exterior Gateway Protocol
Policy based: BGP

The Routing Domain of BGP is the entire Internet
Internet AS Hierarchy

border (exterior gateway) routers

interior routers

Intra-AS vs. Inter-AS Routing

Host

Intra-AS routing within AS A

Inter-AS routing between A and B

Host

Intra-AS routing within AS B

Intra-AS and Inter-AS Routing

"Gateways":
- perform inter-AS routing amongst themselves
- perform intra-AS routing with other routers in their AS network layer
- link layer
- physical layer

Where Does Forwarding Table Come From?

RIP Process
- RIP Routing tables

BGP Process
- BGP Routing tables

OSPF Process
- OSPF Routing tables

Forwarding Table Manager

Where Does Forwarding Table Come From?

Basic Routing Problem

- Assume
  - A network with N nodes, where each edge is associated a cost
  - A node knows only its neighbors and the cost to reach them

- How does each node learn how to reach every other node along the shortest path?
Routing: Issues

- How are routing tables determined?
- Who determines table entries?
- What info is used in determining table entries?
- When do routing table entries change?
- Where is routing info stored?
- How to control routing table size?

Answer these questions, we are done!

Routing Paradigms

- Hop-by-hop Routing
  - Each packet contains destination address
  - Each router chooses next-hop to destination
  - Routing decision made at each (intermediate) hop!
  - Packets to same destination may take different paths!
  - Example: IP's default datagram routing

- Source Routing
  - Sender selects the path to destination precisely
  - Routers forward packet to next-hop as specified
  - Problem: if specified path no longer valid due to link failure!
  - Example:
    - IP's loose/strict source route option
    - Virtual circuit setup phase in ATM (or MPLS)

Routing Algorithms/Protocols

Issues Need to Be Addressed:

- Route selection may depend on different criteria
  - Performance: choose route with the smallest delay
  - Policy: choose a route that doesn't cross .gov network
- Adapt to changes in network topology or condition
  - Self-healing: little or no human intervention
- Scalability
  - Must be able to support a large number of hosts, routers

Centralized vs. Distributed Routing Algorithms

Centralized:

- A centralized route server collects routing information and network topology, makes route selection decisions, then distributes them to routers

Distributed:

- Routers cooperate using a distributed protocol
  - To create mutually consistent routing tables
- Two standard distributed routing algorithms
  - Link State (LS) routing
  - Distance Vector (DV) routing

Link State vs Distance Vector

- Both assume that
  - The address of each neighbor is known
  - The cost of reaching each neighbor is known
- Both find global information
  - By exchanging routing info among neighbors
- Differ in the information exchanged and route computation
  - LS: tells every other node its distances to neighbors
  - DV: tells neighbors its distance to every other node

Link State Algorithm

- Basic idea: Distribute link state packet to all routers
  - Topology of the network
  - Cost of each link in the network
- Each router independently computes optimal paths
  - From itself to every destination
  - Routes are guaranteed to be loop free if
  - Each router sees the same cost for each link
  - Uses the same algorithm to compute the best path
Link State: Control Traffic
- Each node floods its local information to every other node in the network
- Each node ends up knowing the entire network topology - use Dijkstra to compute the shortest path to every other node

Link State: Node State

Topology Dissemination
- Each router creates a set of link state packets (LSPs)
  - Describing its links to neighbors
  - LSP contains: Router id, neighbor’s id, and cost to its neighbor
- Copies of LSPs are distributed to all routers
  - Using controlled flooding
- Each router maintains a topology database
  - Database containing all LSPs

Topology Database: Example

Constructing Routing Table: Dijkstra's Algorithm
- Given the network topology
  - How to compute the shortest path to each destination?
- Some notation
  - X: source node
  - N: set of nodes to which shortest paths are known so far
    - N is initially empty
  - D(V): the cost of the known shortest path from source X to V
  - C(U,V): cost of link U to V
  - C(U,V) = ∞ if not neighbors

Algorithm (at Node X)
- Initialization
  - N = {X}
  - For all nodes V
    - If V adjacent to X, D(V) = C(X,V) else D(V) = ∞

- Loop
  - Find U not in N such that D(U) is the smallest
  - Add U into set N
  - Update D(V) for all V not in N
    - D(V) = min(D(V), D(U) + C(U,V))
  - Until all nodes in N
Example: Dijkstra's Algorithm

Initialization:
1. \( N = \{A\} \)
2. for all nodes \( v \) adjacent to \( A \)
   3. \( D(v) = c(A,v) \)
   4. else \( D(v) = \infty \)

Loop
9. find \( w \) not in \( N \) s.t. \( D(w) \) is a minimum;
10. add \( w \) to \( N \);
11. update \( D(v) \) for all \( v \) adjacent to \( w \) and not in \( N \):
   12. \( D(v) = \min(D(v), D(w) + c(w,v)) \);
13. until all nodes in \( N \);

\[\begin{array}{c|c|c|c|c|c|c}
\text{Step} & \text{start} & D(B),p(B) & D(C),p(C) & D(D),p(D) & D(E),p(E) & D(F),p(F) \\
\hline
0 & A & 2, A & 5, A & 1, A & \infty & \infty \\
1 & AD & 4, D & 2, D & \infty & & \\
2 & ADE & 3, E & 4, E & & & \\
3 & & & & & & \\
4 & & & & & & \\
5 & & & & & & \\
\end{array}\]
**Distance Vector Routing**

- A router tells neighbors its distance to every router
  - Communication between neighbors only
- Based on Bellman-Ford algorithm
  - Computes “shortest paths”
- Each router maintains a distance table
  - A row for each possible destination
  - A column for each neighbor
    - \( D(X, Y, Z) \): distance from \( X \) to \( Y \) via \( Z \)
- Exchanges distance vector (the table) with neighbors
  - Distance vector: current least cost to each destination

**Distance Vector: Example**

<table>
<thead>
<tr>
<th>Destination</th>
<th>( D^E )</th>
<th>Cost to Destination via</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>14, 11</td>
</tr>
<tr>
<td>B</td>
<td>7</td>
<td>8, 5</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>9, 4</td>
</tr>
<tr>
<td>D</td>
<td>10</td>
<td>11, 2</td>
</tr>
</tbody>
</table>

**Distance Table to Routing Table**

<table>
<thead>
<tr>
<th>Destination</th>
<th>Cost to Destination via</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>14, 11</td>
</tr>
<tr>
<td>B</td>
<td>8, 5</td>
</tr>
<tr>
<td>C</td>
<td>9, 4</td>
</tr>
<tr>
<td>D</td>
<td>11, 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outgoing link to use, cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
</tbody>
</table>

**Dijkstra’s Algorithm: In a Nutshell**

<table>
<thead>
<tr>
<th>Step</th>
<th>start N</th>
<th>( D(B), p(B) )</th>
<th>( D(C), p(C) )</th>
<th>( D(D), p(D) )</th>
<th>( D(E), p(E) )</th>
<th>( D(F), p(F) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>2.0</td>
<td>3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
</tr>
<tr>
<td>1</td>
<td>AD</td>
<td>2.0, 1.0</td>
<td>3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
</tr>
<tr>
<td>2</td>
<td>ADE</td>
<td>2.0, 3.0</td>
<td>3.0, 1.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
</tr>
<tr>
<td>3</td>
<td>ADEB</td>
<td>3.0, 1.0</td>
<td>3.0, 4.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
</tr>
<tr>
<td>4</td>
<td>ADEBC</td>
<td>4.0, 1.0</td>
<td>4.0, 3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
</tr>
<tr>
<td>5</td>
<td>ADEBCF</td>
<td>4.0, 1.0</td>
<td>4.0, 3.0</td>
<td>4.0, 2.0</td>
<td>5.0</td>
<td>6.0</td>
</tr>
</tbody>
</table>

**Routing Table Computation**
Distance Vector Routing Algorithm

Iterative:
- continues until no nodes exchange info.
- self-terminating: no "signal" to stop

Asynchronous:
- nodes need not exchange info/iterate in lock step

Distributed:
- each node talks only with directly-attached neighbors

Distance Table data structure:
- each node has its own
- row for each possible destination
- column for each directly-attached neighbor to node
- example: in node X, for dest. Y via neighbor Z:

\[
D(Y,Z) = \text{distance from } X \text{ to } Y, \text{ via } Z \text{ as next hop}
\]

\[
c(X,Z) + \min_{W} \{D(Y,W)\}
\]

Distance Vector Routing: Overview

Iterative, asynchronous:
- each iteration caused by:
  - local link cost change
  - message from neighbor: its least cost path change from neighbor
- distributed:
  - each node notifies neighbors only when its least cost path to any destination changes
  - neighbors then notify their neighbors if necessary

Each node:
- wait for (change in local link cost or msg from neighbor)
- recompute distance table
- if least cost path to any dest has changed, notify neighbors

Distance Vector Algorithm: Example

\[
D(Y,Z) = c(X,Z) + \min_{W} \{D(Y,W)\} = 7 + 1 = 8
\]

\[
D(Z,Y) = c(X,Y) + \min_{W} \{D(Z,W)\} = 2 + 1 = 3
\]

Convergence of DV Routing

- router detects local link cost change
- updates distance table
- if cost change in least cost path, notify neighbors

"good news travels fast"

Problems with DV Routing

Link cost changes:
- good news travels fast
- bad news travels slow
- "count to infinity" problem
Count-to-Infinity Problem

"Fixes" to Count-to-Infinity Problem

• Split horizon
  - A router never advertises the cost of a destination to a neighbor
    • If this neighbor is the next hop to that destination
• Split horizon with poisonous reverse
  - If X routes traffic to Z via Y, then
    • X tells Y that its distance to Z is infinity
    • Instead of not telling anything at all
    • Accelerates convergence

Split Horizon with Poisoned Reverse

If Z routes through Y to get to X:

• Z tells Y its (Z's) distance to X is infinite (so Y won’t route to X via Z)

Count-to-Infinity Problem Revisited

Link State vs Distance Vector

• Tells everyone about neighbors
• Controlled flooding to exchange link state
• Dijkstra’s algorithm
• Each router computes its own table
• May have oscillations
• Open Shortest Path First (OSPF)
• Tells neighbors about everyone
• Exchanges distance vectors with neighbors
• Bellman-Ford algorithm
• Each router’s table is used by others
• May have routing loops
• Routing Information Protocol (RIP)

Link State vs. Distance Vector (cont’d)

Message complexity:
  • LS: $O(n^2e)$ messages
  • n: number of nodes
  • e: number of edges
  • DV: $O(dn^{k+1})$ messages
  • d: node’s degree
  • k: number of rounds

Time complexity:
  • LS: $O(n \log n)$
  • DV: $O(n)$

Robustness: what happens if router malfunctions?
  • LS:
    • node can advertise incorrect link cost
    • each node computes only its own table
  • DV:
    • node can advertise incorrect path cost
    • each node’s table used by others; error propagate through network
Routing in the Real World
Our routing study thus far - idealization
• all routers identical
• network "flat"

How to do routing in the Internet
• scalability and policy issues
scale: with 200 million destinations
• can't store all dest's in routing tables!
• routing table exchange would swamp links!

administrative autonomy
• internet = network of networks
• each network admin may want to control routing in its own network

Routing in the Internet
• The Global Internet consists of Autonomous Systems (AS) interconnected with each other hierarchically:
  - Stub AS: small corporation: one connection to other AS's
  - Multihomed AS: large corporation (no transit): multiple connections to other AS's
  - Transit AS: provider, hooking many AS's together

• Two-level routing:
  - Inter-AS: administrator responsible for choice of routing algorithm within network
  - Inter-AS: unique standard for inter-AS routing: BGP

Internet Architecture
Internet: "networks of networks"!

Internet AS Hierarchy
Intra-AS border (exterior gateway) routers
Intra-AS interior (gateway) routers

Intra-AS vs. Inter-AS Routing
Intra-AS routing within AS A
Intra-AS routing within AS B
Inter-AS routing between A and B

Why Different Intra- and Inter-AS Routing?
Policy:
• Inter-AS: admin wants control over how its traffic routed, who routes through its net.
• Inter-AS: single admin, so no policy decisions needed
Scale:
• hierarchical routing saves table size, update traffic
Performance:
• Inter-AS: can focus on performance
• Inter-AS: policy may dominate over performance
Intra-AS and Inter-AS Routing

"Gateways":
• perform inter-AS routing amongst themselves
• perform intra-AS routers with other routers in their AS

Intra-AS Routing

• Also known as Interior Gateway Protocols (IGP)
• Most common Intra-AS routing protocols:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IS-IS: Intermediate System to Intermediate System (OSI Standard)
  - EIGRP: Extended Interior Gateway Routing Protocol (Cisco proprietary)

RIP (Routing Information Protocol)

• Distance vector algorithm
• Included in BSD-UNIX Distribution in 1982
• Distance metric: # of hops (max = 15 hops)

RIP advertisements

• Distance vectors: exchanged among neighbors every 30 sec via Response Message (also called advertisement)
• Each advertisement: list of up to 25 destination nets within AS

RIP: Example
RIP: Link Failure and Recovery
If no advertisement heard after 180 sec →
neighbor/link declared dead
- routes via neighbor invalidated
- new advertisements sent to neighbors
- neighbors in turn send out new advertisements (if tables changed)
- link failure info quickly propagates to entire net
- poison reverse used to prevent ping-pong loops (infinite distance = 16 hops)

OSPF (Open Shortest Path First)
- "open": publicly available
- Uses Link State algorithm
  - LS packet dissemination
  - Topology map at each node
  - Route computation using Dijkstra’s algorithm
- OSPF advertisement carries one entry per neighbor router
- Advertisements disseminated to entire AS (via flooding)
  - Carried in OSPF messages directly over IP (rather than TCP or UDP)

Hierarchical OSPF
- Two-level hierarchy: local area, backbone.
  - Link-state advertisements only in area
  - Each node has detailed area topology; only know direction (shortest path) to nets in other areas.
  - Communications between areas via backbone
- Area border routers: "summarize" distances to nets in own area, advertise to other Area Border routers.
- Backbone routers: run OSPF routing limited to backbone.
- Boundary routers: connect to other AS's.
Inter-AS Routing in the Internet: BGP

BGP (Border Gateway Protocol): the de facto standard
BGP provides each AS a means to:
1. Obtain subnet reachability information from neighboring ASs.
2. Propagate the reachability information to all routers internal to the AS.
3. Determine "good" routes to subnets based on reachability information and policy.
4. Allows a subnet to advertise its existence to rest of the Internet: "I am here"

BGP basics
- Pairs of routers (BGP peers) exchange routing info over semi-permanent TCP connections: BGP sessions
- Note that BGP sessions do not correspond to physical links.
- When AS2 advertises a prefix to AS1, AS2 is promising it will forward any datagrams destined to that prefix towards the prefix.
  - AS2 can aggregate prefixes in its advertisement
- With eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
- 1c can then use iBGP to distribute this new prefix reach info to all routers in AS1.
- 1b can then re-advertise the new reach info to AS2 over the 1b-to-2a eBGP session.
- When router learns about a new prefix, it creates an entry for the prefix in its forwarding table.

Path attributes & BGP routes
- When advertising a prefix, advert includes BGP attributes.
  - prefix + attributes = "route"
- Two important attributes:
  - AS-PATH: contains the ASes through which the advert for the prefix passed: AS 67 AS 17
  - NEXT-HOP: Indicates the specific internal-AS router to next-hop AS. (There may be multiple links from current AS to next-hop AS.)
- When gateway router receives route advert, uses import policy to accept/decline.
BGP messages

- BGP messages exchanged using TCP.
- BGP messages:
  - OPEN: opens TCP connection to peer and authenticates sender
  - UPDATE: advertises new path (or withdraws old)
  - KEEPALIVE keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - NOTIFICATION: reports errors in previous msg; also used to close connection

BGP routing policy

- A, B, C are provider networks
- X, W, Y are customer (of provider networks)
- X is dual-homed: attached to two networks
  - X does not want to route from B via X to C
  - ... so X will not advertise to B a route to C

BGP routing policy (2)

- A advertises to B the path AW
- B advertises to X the path BAW
- Should B advertise to C the path BAW?
  - No way! B gets no “revenue” for routing CBAW since neither W nor C are B’s customers
  - B wants to force C to route to W via A
  - B wants to route only to/from its customers!

Why different Intra- and Inter-AS routing?

Policy:
- Inter-AS: admin wants control over how its traffic routed, who routes through its net.
- Intra-AS: single admin, so no policy decisions needed

Scale:
- hierarchical routing saves table size, reduced update traffic

Performance:
- Intra-AS: can focus on performance
- Inter-AS: policy may dominate over performance

Multi-Protocol Label Switching (MPLS)

- initial goal: speed up IP forwarding by using fixed length label (instead of IP address) to do forwarding
  - borrowing ideas from Virtual Circuit (VC) approach
  - but IP datagram still keeps IP address!

MPLS Capable Routers

- a.k.a. label-switched router
- forwards packets to outgoing interface based only on label value (don’t inspect IP address)
  - MPLS forwarding table distinct from IP forwarding tables
- signaling protocol needed to set up forwarding
  - RSVP-TE, LDP
  - forwarding possible along paths that IP alone would not allow (e.g., least cost path routing)!!
  - use MPLS for traffic engineering
- must co-exist with IP-only routers
**MPLS Forwarding Tables**

<table>
<thead>
<tr>
<th>In label</th>
<th>Out label</th>
<th>Dest</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>D</td>
<td>0</td>
</tr>
</tbody>
</table>

**Why Mobile IP?**

- Need a protocol which allows network connectivity across host movement
- Protocol to enable mobility must not require massive changes to router software, etc.
- Must be compatible with large installed base of IPv4 networks/hosts
- Confine changes to mobile hosts and a few support hosts which enable mobility

**Internet Protocol (IP)**

- Network layer, "best-effort" packet delivery
- Supports UDP and TCP (transport layer protocols)
- IP host addresses consist of two parts
  - network id + host id
- By design, IP host address is tied to home network address
  - Hosts are assumed to be wired, immobile
  - Intermediate routers look only at network address
  - Mobility without a change in IP address results in unroute-able packets

**IP Routing Breaks Under Mobility**

Why this hierarchical approach? Answer: Scalability! Millions of network addresses, billions of hosts!

**Mobile IP: Basics**

- Proposed by IETF (Internet Engineering Task Force)
  - Standards development body for the Internet
- Mobile IP allows a mobile host to move about without changing its permanent IP address
- Each mobile host has a home agent on its home network
- Mobile host establishes a care-of address when it’s away from home

**Mobile IP: Basics, Cont.**

- Correspondent host is a host that wants to send packets to the mobile host
- Correspondent host sends packets to the mobile host’s IP permanent address
- These packets are routed to the mobile host’s home network
- Home agent forwards IP packets for mobile host to current care-of address
- Mobile host sends packets directly to correspondent, using permanent home IP as source IP
**Mobile IP: Basics, Cont.**

Whenever a mobile host connects to a remote network, two choices:
- care-of can be the address of a foreign agent on the remote network
- foreign agent delivers packets forwarded from home agent to mobile host
- care-of can be a temporary, foreign IP address obtained through, e.g., DHCP
- home agent tunnels packets directly to the temporary IP address
- Regardless, care-of address must be registered with home agent

---

**Mobile IP: Care-of Addresses**

- Packet to be forwarded is encapsulated in a new IP packet
- In the new header:
  - Destination = care-of-address
  - Source = address of home agent
  - Protocol number = IP-in-IP

---

**At the Other End...**

- Depending on type of care-of address:
  - Foreign agent or
  - Mobile host
- ... strips outer IP header of tunneled packet, which is then fed to the mobile host
- Aside: Any thoughts on advantages of foreign agent vs. co-located (foreign IP) address?

---

**Routing Inefficiency**

Mobile host and correspondent host might even be on the same network!!

---

**Route Optimizations**

- Possible Solution:
  - Home agent sends current care-of address to correspondent host
  - Correspondent host caches care-of address
  - Future packets tunneled directly to care-of-address
- But:
  - An instance of the cache consistency problem arises...
  - Cached care-of-address becomes stale when the mobile host moves
  - Potential security issues with providing care-of-address to correspondent
Possible Route Optimization

Network Layer Part II Summary

- Network Layer Routing
  - Basic Issues
  - Distributed Routing Algorithms: LS vs. DV
  - Link State (LS): How does it work?
  - Distance Vector (DV): How does it work? Issues?
  - Mobile IP: how does it work? Issues?
  - MPLS

- Routing in the Internet
  - Intra-AS vs. Inter-AS routing
  - Intra-AS: RIP and OSPF
  - Inter-AS: BGP and Policy Routing

- Things we didn’t cover: VPN, IP Multicast, IPv6 (but please read by yourself!)