Lecture 6: Intra-Domain Routing
Overview

- Internet structure, ASes
- Forwarding vs. Routing
- Distance vector and link state
- Example distance vector: RIP
- Example link state: OSPF
The Internet, 1990

- Hierarchical structure w. single backbone
Address allocation, 1990

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

Class B address

11111111111111111111111100000000

Subnet mask (255.255.255.0)

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

Subnetted address

- **Hierarchical IP addresses**
  - Class A (8-bit prefix), B (16-bit), C (24-bit)

- **Subnetting adds another level within organizations**
  - Subnet masks define variable partition of host part
  - Subnets visible only within site
The Internet, today

- Multiple “backbones”
Address allocation, today

• **Class system makes inefficient use of addresses**
  - class C with 2 hosts \((2/255 = 0.78\%\) efficient)
  - class B with 256 hosts \((256/65535 = 0.39\%\) efficient)
  - Causes shortage of IP addresses (esp. class B)
  - Makes address authorities reluctant to give out class Bs

• **Still Too Many Networks**
  - routing tables do not scale
  - route propagation protocols do not scale
Supernetting

• Assign block of contiguous network numbers to nearby networks

• Called CIDR: Classless Inter-Domain Routing

• Represent blocks with a single pair
  \((\text{first network address}, \text{count})\)

• Restrict block sizes to powers of 2
  - Represent length of network in bits w. slash
  - E.g.: 128.96.34.0/25 means netmask has 25 1 bits, followed by 7 0 bits, or \(0xffffffff80 = 255.255.255.128\)
  - E.g.: 128.96.33.0/24 means netmask 255.255.255.0

• All routers must understand CIDR addressing
IP Connectivity

- For each destination address, must either:
  1. Have prefix mapped to next hop in forwarding table, or
  2. know “smarter router”—default for unknown prefixes

- Route using longest prefix match, default is prefix 0.0.0.0/0

- Core routers know everything—no default

- Manage using notion of *Autonomous System* (AS)

- Two-level route propagation hierarchy
  - interior gateway protocol (each AS selects its own)
  - exterior gateway protocol (Internet-wide standard)
Autonomous systems

• Correspond to an administrative domain
  - Internet is not a single network
  - ASes reflect organization of the Internet
  - E.g., Stanford, large company, etc.

• Goals:
  - ASes want to choose their own local routing algorithm
  - ASes want to set policies about non-local routing

• Each AS assigned unique 16-bit number
Types of traffic & AS

- **Local traffic** – packets with src or dst in local AS
- **Transit traffic** – passes through an AS
- **Stub AS**
  - Connects to only a single other AS
- **Multihomed AS**
  - Connects to multiple ASes
  - Carries no transit traffic
- **Transit AS**
  - Connects to multiple ASes and carries transit traffic
Intra-domain routing

- Intra-domain routing: within an AS
- Single administrative control: optimality is important
  - Contrast with inter-AS routing, where policy dominates
  - Next lecture will cover inter-domain routing (BGP)
Forwarding vs. Routing
What is routing?

- **forwarding** – moving packets between ports
  - Look up destination address in forwarding table
  - Find `out-port` or `<out-port, MAC addr>` pair

- **Routing** is process of populating forwarding table
  - Routers exchange messages about nets they can reach
  - Goal: Find optimal route for every destination
  - …or maybe good route, or just any route (depending on scale)
Stability

- Stable routes are often preferred over rapidly changing ones

- **Reason 1: management**
  - Hard to debug a problem if it’s transient

- **Reason 2: higher layer optimizations**
  - E.g., TCP RTT estimation
  - Imagine alternating over 500ms and 50ms routes

- **Tension between optimality and stability**
Routing algorithm properties

• **Global vs. decentralized**
  - Global: All routers have complete topology
  - Decentralized: Only know neighbors & what they tell you

• **Intra-domain vs. Inter-domain routing**
  - Intra-: All routers under same administrative control
  - Intra-: Scale to ~100 networks (e.g., campus like Stanford)
  - Inter-: Decentralized, scale to Internet

• **We’ll cover basic algorithms for intra-domain routing and two protocols: RIP and OSPF**
Optimality

- View network as a graph
- Assign cost to each edge
  - Can be based on latency, b/w, utilization, queue length, …
- Problem: Find lowest cost path between two nodes
  - Each node individually computes the cost (some recent research argues against doing this, more on this later)
Scaling issues

• Every router must be able to forward based on any destination IP address
  - Given address, it needs to know “next hop” (table)
  - Naïve: Have an entry for each address
  - There would be $10^8$ entries!

• Solution: Entry covers range of addresses
  - Can’t do this if addresses are assigned randomly! (e.g., Ethernet addresses)
  - This is why address aggregation is important
  - Addresses allocation should be based on network structure

• What is structure of the Internet?
Distance Vector and Link State
Basic Algorithms

• Two classes of intra-domain routing algorithms

• Link state
  - Have a global view of the network
  - Simpler to debug
  - Require global state

• Distance vector
  - Require only local state (less overhead, smaller footprint)
  - Harder to debug
  - Can suffer from loops
Link State

• **Strategy**
  - Send to all nodes (not just neighbors)
  - Send only 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
Reliable flooding

- 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
  - If you hear your own packet w. SEQNO = n, set your next SEQNO to n + 1
- Decrement TTL of each stored LSP
  - discard when TTL= 0
Calculating best path

• Dijkstra’s shortest path algorithm

• Let:
  - \( N \) denote set of nodes in the graph
  - \( l(i, j) \) denotes non-negative cost (weight) for edge \((i, j)\)
  - \( s \) denotes yourself (node computing paths)

• Initialize variables
  - \( M \leftarrow \{s\} \) (set of nodes “incorporated” so far)
  - \( C_n \leftarrow l(s, n) \) (cost of the path from \( s \) to \( n \))
  - \( R_n \leftarrow \bot \) (next hop on path to \( n \))
Dijkstra’s algorithm

• While $N \neq M$
  - Let $w \in (N - M)$ be node with lowest $C_w$
  - $M \leftarrow M \cup \{w\}$
  - Foreach $n \in (N - M)$, if $C_w + l(w, n) < C_n$
    then $C_n \leftarrow C_w + l(w, n), R_n \leftarrow w$

• Example: $D (D, 0, \perp)(C, 2, C)(B, 5, C)(A, 10, C)$
Distance Vector

• *Local* routing algorithm

• Each node maintains a set of triples
  - \((\text{Destination}, \text{Cost}, \text{NextHop})\)

• Exchange updates with direct neighbors
  - periodically (on the order of several seconds to minutes)
  - whenever table changes (called triggered update)

• Each update is a list of pairs:
  - \((\text{Destination}, \text{Cost})\)

• Update local table if receive a “better” route
  - smaller cost
  - from newly connected/available neighbor

• Refresh existing routes; delete if they time out
Calculating best path

• Bellman-Ford equation

• Let:
  - $D_a(b)$ denote the distance from $a$ to $b$
  - $c(a, b)$ denote the cost of a link from $a$ to $b$

• Then $D_x(y) = \min_z (c(x, z) + D_z(y))$

• Routing messages contain $D$; when does a node transmit a message?
• B’s routing table:

<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>
- 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
Danger: Loops

- 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…
How to avoid loops

• IP TTL field prevents a packet from living forever
  - Does not break or repair a loop

• Simple approach: consider small cost $n$ (e.g., 16) to be infinity
  - After $n$ rounds will decide node is unavailable
  - But rounds can be long: this takes time

• Distance vector based only on local information
Better loop avoidance

**Split horizon**

- When sending updates to node $A$, don’t include destinations you route to through $A$
- Prevents $B$ and $C$ from sending cost 2 to $A$

**Split horizon with poison reverse**

- When sending updates to node $A$, explicitly include very high cost ("poison") for destinations you route to through $A$
- When does poison reverse help?
Poison Reverse example
Poison Reverse example

A
B
C
D
E
F
G

B:E:2
Poison Reverse example
Poison Reverse example
Warning

- **Note:** Split horizon/split horizon with poison reverse only help between two nodes
  - Can still get loop with three nodes involved
  - Might need to delay advertising routes after changes, but will affect convergence time
Distance Vector vs. Link State

• # of messages
  - DV: convergence time varies, but $\Omega(d)$ where $d$ is # of neighbors of node
  - LS: $O(n \cdot d)$ for $n$ nodes in system

• Computation
  - DV: Could count all the way to $\infty$ if loop
  - LS: $O(n^2)$

• Robustness – what happens with malfunctioning router?
  - DV: Node can advertise incorrect path cost
  - DV: Costs used by others, errors propagate through net
  - LS: Node can advertise incorrect link cost
Metrics

• Original ARPANET metric
  - measures number of packets enqueued 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
    \[ Delay = (DT - AT) + Transmit + Latency \]
  - if timeout, reset DT to departure time for retransmission
  - link cost = average delay over some time period

• Fine Tuning
  - compressed dynamic range
  - replaced \( Delay \) with link utilization

• Today: policy often trumps performance [more later]
Intradomain routing protocols

- **RIP (routing information protocol)**
  - Fairly simple implementation of DV
  - RFC 2453 (38 pages)

- **OSPF (open shortest path first)**
  - More complex link-state protocol
  - Adds notion of *areas* for scalability
  - RFC 2328
2-minute stretch
Example Distance Vector: RIP
RIPv2 [RFC 2453]

- Runs over UDP port 520
- Limits networks to 15 hops (16 = \( \infty \))
- Depends on count to infinity for loops
- Supports split horizon, poison reverse
- RFC 1812 specifies what options routers should or must have
RIPv2 packet format

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| command (1) | version (1) | must be zero (2) | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| |
~ RIP Entry (20) ~
| |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
RIPv2 Entry

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

+-----------------------------------------------+
| address family identifier (2) | Route Tag (2) |
+-----------------------------------------------+
| IP address (4) |
+-----------------------------------------------+
| Subnet Mask (4) |
+-----------------------------------------------+
| Next Hop (4) |
+-----------------------------------------------+
| Metric (4) |
+-----------------------------------------------+
Route Tag Field

- Allows RIP nodes to distinguish internal and external routes
- Must persist across announcements
- E.g., encode AS
Next Hop Field

- Allows one router to advertise routes for multiple routers on same subnet

- Suppose only XR1 talks RIP2:

```
-----------
|IR1| |IR2| |IR3| |XR1| |XR2| |XR3|
-----------
```

```
----- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- 
|   |   |   |   |   |   |   |   |   |   |   |   |
```

```
<-----------------RIP-2----------------->
```
Example Link State: OSPF
OSPFv2 [RFC 2328]

- Link state protocol
- RFC 2328 (244 pages)
- Runs directly over IP (protocol 89)
  - Has to provide its own reliability
- All exchanges are authenticated
- Adds notion of areas for scalability
OSPF Areas

- Area 0 is “backbone” area (includes all boundary routers)
- Traffic between two areas must always go through area 0
- Only need to know how to route exactly within area
- Else, just route to appropriate area
- (Virtual links can allow distant routers to be in area 0)
OSPF areas

Figure 4.40 ♦ Hierarchically structured OSPF AS with four areas
# OSPF Packet Header

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</th>
<th>Version #</th>
<th>Type</th>
<th>Packet length</th>
</tr>
</thead>
<tbody>
<tr>
<td>+-----------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Router ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Area ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Checksum</td>
<td></td>
<td>AuType</td>
</tr>
<tr>
<td>+-----------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Authentication</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Authentication</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------------------------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
OSPF Packet Types

• 1 - Hello packet
• 2 - Database Description
• 3 - Link State Request
• 4 - Link State Update
• 5 - Link State Acknowledgment
# Database Description

<table>
<thead>
<tr>
<th>Interface MTU</th>
<th>Options</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>I</th>
<th>M</th>
<th>MS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DD sequence number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>~ An LSA Header (20+bytes) ~</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Database Description Packet Fields

- Interface MTU
- Options (multicast, external LSAs, etc.)
- Init bit, More bit, Master/Slave bit
- Sequence number: distinguishes DD packets.
- Exchange
  - First packet in an exchange has the I bit sent, all but last have M bit set, sequence number increments on each packet
DD Packet Exchange

- Used to initialize routing state

- Node A sends an empty DD to node B with sequence number $n$, the I, M, and MS bits set

- Node B responds with a DD with the I and MS bits cleared, seq. no $n$
  - Can contain LSAs - if there are more, set the M bit

- Node A $n + 1$, I bit cleared, MS bit set, M bit depends on whether there are more LSAs

- Continues until both send cleared M bit
OSPF areas

Figure 4.40 ♦ Hierarchically structured OSPF AS with four areas
LSA Details

- Many different LSA formats

- Example 1: Router-LSAs, describe a router’s links

- Example 2: Summary-LSAs, generated by area border routers
  - Describes an IP network within the area
  - Includes IP address, network mask, and cost metric

- Example 3: AS-external-LSAs
  - Describes an IP network in another AS
  - Includes IP address, netmask, cost, and forwarding address
Basic Algorithms

- Two classes of intra-domain routing algorithms

- Link state (e.g., OSPF)
  - Have a global view of the network
  - Simpler to debug
  - Require global state (OSPF reduces this through areas)

- Distance vector (e.g. RIP)
  - Require only local state (less overhead, smaller footprint)
  - Harder to debug
  - Can suffer from loops
The Internet, today

- Backbone service provider
- Peering point
- Large corporation
- "Consumer" ISP
- Small corporation

"Consumer" ISP
There were 56 contributors, in 53 different cities, 9 countries, and American Network Operators' Group (NANOG) mailing list. The IPv6 data was collected between January 1st and 8th 2008 seen in Route Views Border Gateway Protocol (BGP) routing toward 48M /24 networks spread across 95% of the prefixes cities, 11 countries, and 3 continents. The monitors probed paths for each routing system. The IPv6 Internet topologies observed during the first week of January 2008 reflected the wider adoption of IPv6 outside the United States. The largest AS in the IPv6 graph is European and that the other IPv4 graph with 18,753 ASes. While the IPv4 graph's central core of Asia were subdivided prefixes into the smallest prefixes that Digital Envoy's Netacuity (R) mapped to a single geographic location in approximately corresponds to an Internet Service Provider (ISP). We counted for Peering: 2,358 IPv6 destinations spread across 822 prefixes or 81% weighted average (by number of IP addresses in each mapped AS) we used the IPv6 equations: (radius, angle) calculated using the following position. More accurate inference of geographic coverage of an AS. Mapped each AS to its set of announced IPv4 prefixes. (IPv4 tables are currently much larger, facilitating more efficient processing.) We used the IPv6 from Route Views and mapped each AS to its set of announced prefixes that were observed accepting our probe traffic from this AS. The outdegree of an AS node is the number of next-hop ASes toward the center of the graph we have manually.
IPv6

This visualization represents macroscopic snapshots of the IPv4 and IPv6 routing systems as of January 2008. The IPv4 data was collected between January 2nd and 17th, 2008 by 13 CAIDA archipelago monitors located in 13 different cities, 11 countries, and 3 continents. The monitors probed paths to determine the longitude of ASes, using the IPv4 BGP table to calculate the best match of this address in BGP routing to it, i.e., to the origin (end-of-path) AS for the IP prefix. For IPv6, 2,358 IPv6 destinations spread across 822 prefixes or 81% were probed, subdivided prefixes into the smallest prefixes that Digital Envoy knows about, and from Route Views and mapped each AS to its set of announced prefixes. To determine the longitude of ASes, we used the IPv4 BGP table to calculate the maximum outdegree and longitude of the AS’s BGP prefixes. We then used the weighted average (by number of IP addresses in each mapped prefix) to find the best location.

Although NTT is a Japanese telecommunication company, the IPv6 graph center is still dominated by American ASes, the IPv4 graph with 18,753 ASes. While the IPv4 graph’s central core is comparable in degree to the European ASes, it is much larger and still dominated by American ASes. The IPv6 graph with 486 ASes remains much smaller than the IPv4 graph with 18,753 ASes. While the IPv4 graph’s central core is comparable in degree to the European ASes, it is much larger and still dominated by American ASes. The IPv6 graph with 486 ASes remains much smaller than the IPv4 graph with 18,753 ASes. The IPv6 data was collected between January 1st and 8th 2008 from 13 CAIDA archipelago monitors located in 13 different cities, 11 countries, and 3 continents. The monitors probed paths to determine the longitude of ASes, using the IPv4 BGP table to calculate the best match of this address in BGP routing to it, i.e., to the origin (end-of-path) AS for the IP prefix. For IPv6, 2,358 IPv6 destinations spread across 822 prefixes or 81% were probed, subdivided prefixes into the smallest prefixes that Digital Envoy knows about, and from Route Views and mapped each AS to its set of announced prefixes. To determine the longitude of ASes, we used the IPv4 BGP table to calculate the maximum outdegree and longitude of the AS’s BGP prefixes. We then used the weighted average (by number of IP addresses in each mapped prefix) to find the best location.

IPv4

The IPv4 data was collected between January 2nd and 17th, 2008 by 13 CAIDA archipelago monitors located in 13 different cities, 11 countries, and 3 continents. The monitors probed paths to determine the longitude of ASes, using the IPv4 BGP table to calculate the best match of this address in BGP routing to it, i.e., to the origin (end-of-path) AS for the IP prefix. For IPv6, 2,358 IPv6 destinations spread across 822 prefixes or 81% were probed, subdivided prefixes into the smallest prefixes that Digital Envoy knows about, and from Route Views and mapped each AS to its set of announced prefixes. To determine the longitude of ASes, we used the IPv4 BGP table to calculate the maximum outdegree and longitude of the AS’s BGP prefixes. We then used the weighted average (by number of IP addresses in each mapped prefix) to find the best location.

IPv4 & IPv6 INTERNET TOPOLOGY MAP JANUARY 2008

IPv4

170W
160W
150W
140W
130W
120W
110W
100W
90W
80W
70W
60W
50W
40W
30W
20W
10W
0W

IPv6

WIDE (2500)
LambdaNet (13237)
OpenCarrier (41692)
Tiscali (3257)
Hurricane (6939)
Sprint (6175)
Global (3549)
Teleglobe (6453)
LavaNet (6435)
Asia Netcom (18084)
IU (2497)
NTT (2914)

IPv4

170E
160E
150E
140E
130E
120E
110E
100E
90E
80E
70E
60E
50E
40E
30E
20E
10E
0E

IPv6

WIDE (2500)
LambdaNet (13237)
OpenCarrier (41692)
Tiscali (3257)
Hurricane (6939)
Sprint (6175)
Global (3549)
Teleglobe (6453)
LavaNet (6435)
Asia Netcom (18084)
IU (2497)
NTT (2914)

IPv4

170E
160E
150E
140E
130E
120E
110E
100E
90E
80E
70E
60E
50E
40E
30E
20E
10E
0E

IPv6

WIDE (2500)
LambdaNet (13237)
OpenCarrier (41692)
Tiscali (3257)
Hurricane (6939)
Sprint (6175)
Global (3549)
Teleglobe (6453)
LavaNet (6435)
Asia Netcom (18084)
IU (2497)
NTT (2914)
Overview

• Internet structure, ASes
• Forwarding vs. Routing
• Distance vector and link state
• Example distance vector: RIP
• Example link state: OSPF
• Next lecture: BGP