Computer Networks

Lecture 23

Inter-AS routing using BGP, Network Forwarding

11/21/2013

Admin

- Assignment 5 to be posted later today
- Samples of Exam 2 will be posted by Friday
Recap: Basic Routing Protocols

- Distance vector protocols
  - Basic DV protocol
    - take away: use monotonicity as a technique to understand liveness
  - DSDV, EIGRP
    - take away: use global invariants to understand safety

- Link state protocols
  - OSPF, IS-IS

Recap: Routing in the Internet

- AS A (OSPF intra-routing)
- AS C (RIP intra-routing)
- Inter-AS routers form an overlay
- Gateway routers participate in intradomain to learn internal routes.
- Border Gateway routers exchange routes using BGP
Recap: BGP

- The de facto Inter-AS standard protocol
- A Path Vector protocol:
  - Derived from Distance Vector protocol
  - Send the *entire path* (i.e., a sequence of ASNs)
- Why path vector:
  - Path vector prevents counting-to-infinity problem
  - Path vector allows an AS to define local policies on the selection of AS paths for each destination

Recap: BGP Routing Decision Process
BGP Example 1

Export to E: i->e I can reach hosts in D; path: IAD

No Export to F (why?)

Route selection policy:
- Shortest AS Path policy
- Choose AD using a1

AS A (OSPF)

AS B (OSPF intra routing)

AS D

AS C

AS I

Selection policy:
- Low local pref for A
- Shortest AS Path
- Prefer eBGP (hot potato)

Choose BCD using b

BGP Example 2

Export to E: i->e I can reach hosts in D; path: IAD

No Export to F (why?)

Route selection policy:
- Shortest AS Path policy
- Choose AD using a1

AS A (OSPF)

AS B (OSPF intra routing)

AS D

AS C

AS I

Selection policy:
- Low local pref for A
- Shortest AS Path
- Prefer eBGP (hot potato)

Choose BCD using b
**Routing: Example 3**

- **AS A (OSPF)**
- **AS B (OSPF intra-routing)**
- **AS C**
- **AS D**

**Selection policy:**
- Low local-preference
- Shortest AS Path
- Prefer iBGP (cold potato)

**Export to E:**
- i->e: I can reach hosts in D; path: IBCD
- i2->i: I can reach hosts in D; path: BCD

**Choose BCD using i2**

**Observing BGP Paths**

- Using one of the looking glass servers:
  http://www.bgp4.as/looking-glasses
Summary of Interdomain Routing

Key features
- Hierarchical routing
- Policy routing

Benefits of Internet Hierarchical Routing
- ASes have flexibility to choose their own intra-AS routing protocols
  - allows autonomy
- Only a small # of routers (gateways) from each AS in the inter-AS level
  - improves scalability
- Inter-AS route represented as a list of ASNs instead of detailed routers
  - improves scalability/privacy
**Issue 1: Hierarchical Routing May Pay a Price for Path Quality**

The diagram illustrates a network of Autonomous Systems (AS) with routing decisions made based on hierarchical routing policies. AS 1, 2, 3, and 4 are interconnected, with AS 1 being the destination. The router selection policy of each AS is to prefer its counter-clockwise neighbor, which can lead to issues if there is a conflict in preference between AS 2 and AS 3.

**Issue 2: BGP Instability**

The BAD GADGET example:
- 0 is the destination
- The route selection policy of each AS is to prefer its counter-clockwise neighbor

The policy interaction causes routing instability!
Understanding Instability: P-Graph

- Nodes in P-graph are feasible paths
- Edges represent priority (low to high)
  - A directed edge from path $N_1P_1$ to $P_1$
    - Intuition: to let $N_1$ choose $N_1P_1$, $P_1$ must be chosen and exported to $N_1$
  - A directed edge from a lower ranked path to a higher ranked path
    - Intuition: the higher ranked path should be considered first

Partial Order Graph and BGP Convergence

- If the P-graph has no loop, then BGP policy converges.
  - Intuition: choose the path node from the partial order graph with no out-going edge to non-fixed path nodes, fix the path node, eliminate all no longer feasible; continue
- Example: suppose we swap the order of 30 and 320
**Question**

Do we often see path instability in the Internet?

Preview: the current Internet ISP economy implies no loop in P-graph!

---

**Internet Economy: Two Types of Business Relationship**

- **Customer provider relationship**
  - a provider is an AS that connects the customer to the rest of the Internet
  - customer pays the provider for the transit service
  - e.g., Yale is a customer of AT&T and QWEST

- **Peer-to-peer relationship**
  - mutually agree to exchange traffic between their respective customers
  - there is no payment between peers

![Diagram showing provider to customer and peer-to-peer relationships]
Implication of Business Relationship: Route Selection Policies

- Route selection (ranking) policy:
  - The typical route selection policy is to prefer customers over peers/providers to reach a destination, i.e., Customer > Peer/Provider

Implication of Business Relationship: Export Policies

- Case 1: Routes learned from a customer are sent to all other neighbors.
- Case 2: Routes learned from a provider are sent only to customers.
- Case 3: Routes learned from a peer are sent only to customers.
Example: Typical Export -> No-Valley Routing

Suppose $P_1$ and $P_2$ are providers of $A$; $A$ is a provider of $C$

Typical Export Policies Imply Patterns of Routes

- Assume a BGP path $SABCD$ to destination $ASD$. Consider the business relationship between each pair:

- Three types of business relationships:
  - PC (provider-customer)
  - CP (customer-provider)
  - PP (peer-peer)
Typical Export Policies Imply Patterns of Routes

Invariant 1 of valid BGP routes (with labels representing business relationship)

```
P       C       P       C       P       C       ?P       C       Dest
```

Reasoning: only route learned from customer is sent to provider; thus after a PC, it is always PC to the destination

Invariant 2 of valid BGP routes (with labels representing business relationship)

```
CP       ?       CP       CP/PP       Dest
```

Reasoning: routes learned from peer or provider are sent to only customers; thus all relationship before is CP
Stability of BGP Routing

- Suppose
  1. there is no loop formed by provider-customer relationship in the Internet
  2. each AS uses typical route selection policy: $C > E/P$
  3. each AS uses the typical export policies

- Then BGP policy routing always converges.

Case 1: A Link is PC

Proof by contradiction. Assume a loop in P-graph. Consider a fixed link in the loop.

AS 1 ➔ PC ➔ AS 2 ➔ PC ➔ AS 3 ➔ PC ➔ AS 4 ➔
Case 2: Link is CP/PP

AS 1 AS 2 AS 3 AS 4

CP/PP CP CP/PP CP

Summary: BGP Policy Routing

- **Advantage**
  - satisfies real demand

- **Issue**
  - policy dispute can lead to instability
    - current Internet economy provides a stability framework, but if the framework changes, we may see instability

- **Comment:**
  - a policy routing framework is a preference (ranking) aggregation framework
  - a fundamental negative result is Arrow’s Impossibility Theorem (http://en.wikipedia.org/wiki/Arrow’s_impossibility_theorem)
Routing: Remaining Issue

How to specify?

AS I

AS A (OSPF)

AS B (OSPF intra routing)

AS C

AS D

d

d1

d2

i

How to specify?

I can reach hosts in D via my path: AD

AS Addressing Scheme: Requirements

- We need an address to uniquely identify each destination

- Routing scalability needs flexibility in aggregation of destination addresses
  - we should be able to aggregate a set of destinations as a single routing unit

- Preview: the unit of routing in the Internet is a network---the destinations in the routing protocols are networks, not individual IP addresses
IP Address: An IP Address Identifies an Interface

- **IPv4 address**: 32-bit identifier for an interface

- **interface**:
  - Routers typically have multiple interfaces
  - Host may have multiple interfaces

```bash
%sbin/ifconfig -a
```

223.1.1.1
223.1.1.2
223.1.1.3
223.1.1.4
223.1.1.9
223.1.2.2
223.1.2.1
223.1.3.2
223.1.3.1
223.1.3.27

IP Addressing

- **IP address**:
  - Network part
  - Host part

- **What’s a network?** (from IP address perspective)
  - Is a unit of routing: can be routed together (depend on the routing protocol)
Specifying Network Address in IP

“class-ful” addressing in the original IP design:

<table>
<thead>
<tr>
<th>Class</th>
<th>Address Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.0.0.0 to 127.255.255.255</td>
</tr>
<tr>
<td>B</td>
<td>128.0.0.0 to 191.255.255.255</td>
</tr>
<tr>
<td>C</td>
<td>192.0.0.0 to 223.255.255.255</td>
</tr>
<tr>
<td>D</td>
<td>224.0.0.0 to 239.255.255.255</td>
</tr>
<tr>
<td>E</td>
<td>240.0.0.0 to 255.255.255.255</td>
</tr>
</tbody>
</table>

Problem of class-ful addressing?

IP Addressing: CIDR

- (Static)classful addressing:
  - inefficient use of address space, address space exhaustion
    - e.g., a class A net allocated enough addresses for 16 million hosts; a class B address may also be too big
  - not flexible for aggregation

- CIDR: Classless InterDomain Routing
  - network portion of address of arbitrary length
  - address format: a.b.c.d/x, where x is # bits in network portion of address

```
11001000  00010111  00010000  00000000

200.23.16.0/23
```
CIDR Address Aggregation

Q: how to do IP addr lookup: at S?

Routing Table Size of BGP (number of globally advertised, aggregated networks)

Active BGP Entries (http://bgp.potaroo.net/as1221/bgp-active.html)
Internet Growth (http://www.caida.org/research/topology/as_core_network/historical.xml)
Routing Table Prefix Length Distr.

IP Addressing: How to Get One?

Q: How does an ISP get its block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers
   - allocates addresses
   - manages DNS
   - assigns domain names, resolves disputes

Example
% whois -h whois.arin.net 130.132.1.1
to check the organization who owns an address
IP addresses: How to Get One?

Q: How does a host get an IP address?

- Static configured
  - wintel: control-panel->network->configuration->tcp/ip->properties
  - unix: 
    ```bash
    %/sbin/ifconfig eth0 inet 192.168.0.10 netmask 255.255.255.0
    ```

- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - “plug-and-play”

Outline

- Admin. and recap
- BGP
- IP addressing
  - IP forwarding
What A Router Looks Like

Cisco GSR 12416

Juniper M160

Basic Router Structure

run routing algorithms/protocol (RIP, OSPF, BGP)
**Input Port Functions**

- **Physical layer:**
  - Bit-level reception

- **Data link layer:**
  - E.g., Ethernet

- **Network layer:**
  - Lookup output port using forwarding table

**IP Datagram Format**

- IP protocol version number (32 bits)
- Header length (bytes)
- "type" of data (16-bit identifier)
- Max number remaining hops (decremented at each router)
- Upper layer protocol to deliver payload to

**Data Forwarding: Steps**

- Error checking, e.g., check header checksum; if error, set up error flag
- Decrement TTL; if TTL == 0, set error flag
- If error, drop the packet, and generate ICMP report

- How much overhead with TCP?
  - 20 bytes of TCP
  - 20 bytes of IP
  - = 40 bytes + app layer overhead

E.g. timestamp, record route taken, specify list of routers to visit.
The Network Layer

Host, router network layer functions:

- Routing protocols
  - path selection
  - RIP, OSPF, BGP
- The IP protocol
  - addressing
  - datagram format
- ICMP protocol
  - error reporting
  - router “signaling”

Transport layer: TCP, UDP

Network layer

Link layer

Physical layer

ICMP: Internet Control Message Protocol

- network-layer “above”
  - IP:
    - ICMP msgs carried in IP datagrams
  - ICMP message: type, code plus first 8 bytes of IP datagram causing error

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>echo reply (ping)</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>dest network unreachable</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>dest host unreachable</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>dest protocol unreachable</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>dest port unreachable</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>dest network unknown</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>dest host unknown</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>source quench (congestion control - not used)</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>echo request (ping)</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>route advertisement</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>router discovery</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>TTL expired</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>bad IP header</td>
</tr>
</tbody>
</table>

traceroute is developed by a clever use of ICMP
Data Forwarding: Steps

- If no error, look up packet destination address in forwarding table:
  - if datagram for a host on directly attached network, it is the job of the link layer now
  - otherwise,
    - lookup: find next-hop router using longest-prefix matching, and its outgoing interface
    - if needed, do fragmentation
    - forward packet to outgoing interface (to the next hop neighbor)

try `%netstat -rn` to see the forwarding table

Forwarding Look up using Patricia Trie

<table>
<thead>
<tr>
<th></th>
<th>prefix</th>
<th>interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>a)</td>
<td>00001</td>
<td></td>
</tr>
<tr>
<td>b)</td>
<td>00010</td>
<td></td>
</tr>
<tr>
<td>c)</td>
<td>00011</td>
<td></td>
</tr>
<tr>
<td>d)</td>
<td>001</td>
<td></td>
</tr>
<tr>
<td>e)</td>
<td>0101</td>
<td></td>
</tr>
<tr>
<td>f)</td>
<td>011</td>
<td></td>
</tr>
<tr>
<td>g)</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>h)</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>i)</td>
<td>1010</td>
<td></td>
</tr>
<tr>
<td>j)</td>
<td>1011</td>
<td></td>
</tr>
<tr>
<td>k)</td>
<td>1100</td>
<td></td>
</tr>
</tbody>
</table>

The networks are represented by a decision tree, e.g., a Patricia Trie to look for the longest match of the destination address.
Example 1 (same network): A->B

- Look up dest address
- Find dest is on same net
- Link layer will send the datagram directly inside a link-layer frame

<table>
<thead>
<tr>
<th>src</th>
<th>dst</th>
</tr>
</thead>
<tbody>
<tr>
<td>misc fields</td>
<td>223.1.1.1</td>
</tr>
<tr>
<td></td>
<td>223.1.1.3</td>
</tr>
<tr>
<td>data</td>
<td></td>
</tr>
</tbody>
</table>

Example 2 (Different Networks): A->E

- Look up dest address in forwarding table
- Routing table: next hop router to dest is 223.1.1.4
- Link layer sends datagram to router 223.1.1.4 inside a link-layer frame
  - The dest. of the link layer frame is 223.1.1.4
Example 2 (Different Networks): A→E

Arriving at 223.1.1.4, destined for 223.1.2.2

- look up dest address in router’s forwarding table
- E on same network as router’s interface 223.1.2.9
  - router, E directly attached
- link layer sends datagram to 223.1.2.2 inside link-layer frame via interface 223.1.2.9
- datagram arrives at 223.1.2.2!! (hooray!)

Backup Slides
Backup: Switching Fabric

Switching: Low End
Switching Via An Interconnection Network

- Overcome bus bandwidth limitations
- Fragmenting datagram into fixed length cells, switch cells through the fabric.
- Crossbar, Banyan networks, and others

- Cisco 12416: switches 320 Gbps (upgradeable to 1.28 Tbps) with 16 slots (each 10G full-duplex) through the crossbar interconnection network

New Potential Bottleneck: Output Ports

- Due to output port contention and head-of-the-Line (HOL) blocking (i.e., queued datagram at front of queue prevents others in queue from moving forward)
Head-of-Line Blocking Limits Thrput

- Due to output-port contention and HOL blocking, the stable throughput is only around $2 - \sqrt{2} = 0.586$ of line speed!

Output Ports

- Buffering required when datagrams arrive from fabric faster than the transmission rate
- Queueing (delay) and loss due to output port buffer overflow!
- Scheduling and queue/buffer management choose among queued datagrams for transmission
**Backup: IP Multicast**

**IP Fragmentation & Reassembly**

- Network links have MTU (max.transfer size) - largest possible link-level frame.
  - Different link types, different MTUs, e.g. Ethernet MTU is 1500 bytes.
- Large IP datagram divided (“fragmented”)
  - One datagram becomes several datagrams
  - “reassembled” only at final destination
  - IP header bits used to identify, order related fragments
IP Fragmentation and Reassembly

Example
- 4000 byte datagram
- MTU = 1500 bytes

One large datagram becomes several smaller datagrams

IP Multicast: Service Model

Multicast group concept: use of indirection
- A group is identified by a location-independent logical address (class D IP address: prefix 1110)

Open group model
- Anyone can send packets to the "logical" group address
- Anyone can join a group and receive packets

Needed: infrastructure to deliver mcast-addressed datagrams to all hosts that have joined that multicast group
Multicast Across LANs

- **Goal:** find a tree (or trees) connecting routers having local mcast group members
  - **source-based:** different tree from sender to each receiver
    - Distance-vector multicast routing protocol (DVMRP)
    - Protocol-independent multicast-dense mode (PIM-DM)
  - **shared-tree:** same tree used by all group members
    - Core-Based Tree (CBT)
    - Protocol-independent multicast-sparse mode (PIM-SM)

Source Tree: Reverse Path Flooding (RPF)

- A router x forwards a packet from source (S) iff it arrives via neighbor y, and y is on the shortest path from x back to S
- A packet is replicated to all but the incoming interface
Reverse Path Forwarding: Improvement

- Basic idea: forward a packet from $S$ only on child links for $S$
- A child link of router $x$ for source $S$
  - a link that has $x$ as parent on the shortest path from the link to $S$
  - a child $x$ notifies its parent $y$ (through the routing protocol) that it has selected $y$ as its parent

Reverse Path Forwarding: Pruning

- No need to forward datagrams down subtree with no mcast group members
- “prune” msgs sent upstream by router with no downstream group members

LEGEND:
- $S$: source
- $R1$, $R2$, $R3$, $R4$, $R5$, $R6$, $R7$: routers
- $P$: prune message
- links with multicast forwarding
Pruning

- Prune (Source, Group) at a leaf router if no members
  - send No-Membership Report (NMR) up tree
- If all children of router R prune (S,G)
  - propagate prune for (S,G) to its parent
- What do you do when a member of a group (re)joins?
  - send a Graft message to upstream parent
- How to deal with failures?
  - prune dropped
  - flow is reinstated
  - down stream routers re-prune
- Note: again a soft-state approach

Implementation of Source Trees in the Internet

- Multicast OSFP (MOSFP)
  - Membership is part of the link state distribution; calculate source specific, pre-pruned trees
- Reverse Path Forwarding
  - Distance Vector Multicast Routing Protocol (DVMRP)
  - Protocol Independent Multicast - Dense Mode (PIM-DM)
    - very similar to DVMRP
  - Difference: PIM uses any unicast routing algorithm to determine the path from a router to the source; DVMRP uses distance vector
  - Question: the state requirement of Reverse Path Forwarding
Building a Shared Tree

- **Steiner Tree**: minimum cost tree connecting all routers with attached group members
- A Steiner tree is not a spanning tree because you do not need to connect all nodes in the network
- Problem is NP-hard
- Excellent heuristics exists
- Not used in practice:
  - computational complexity
  - information about entire network needed
  - monolithic: rerun whenever a router needs to join/leave

Center (Core) based Shared Tree

- Single delivery tree shared by all
- One router identified as “center” of tree
- Tree construction is receiver-based
  - edge router sends unicast `join-msg` addressed to center router
  - `join-msg` “processed” by intermediate routers and forwarded towards center
  - `join-msg` either hits existing tree branch for this center, or arrives at center
  - path taken by `join-msg` becomes new branch of tree for this router
- A sender unicasts a packet to center
  - The packet is distributed on the tree when it hits the tree
Example: M3 Joins

- Group members: M1, M2

Discussion: what is property of the constructed tree?

Example: M1 Sends Data

- Group members: M1, M2, M3
- M1 sends data
Shared Tree Protocols in the Internet

- Core Based Tree
- Protocol Independent Multicast (PIM)  
  Sparse mode
- The catch: how do you know the center?  
  - session announcement

Mbone: Tunneling

Q: How to connect “islands” of multicast routers in a “sea” of unicast routers?

- mcast datagram encapsulated inside “normal” (non-multicast-addressed) datagram
- normal IP datagram sent thru “tunnel” via regular IP unicast to receiving mcast router
- receiving mcast router unencapsulates to get mcast datagram
Backup: NAT

Network Address Translation: Motivation

- A local network uses just one public IP address as far as outside world is concerned
- Each device on the local network is assigned a private IP address

All datagrams leaving local network have same single source NAT IP address: 138.76.29.7, different source port numbers

Datagrams with source or destination in this network have 192.168.1/24 address for source, destination (as usual)
Private IP

NAT: Network Address Translation

Implementation: NAT router must:

- **outgoing datagrams**: replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  
  ... remote clients/servers will respond using (NAT IP address, new port #) as destination addr.

- **remember (in NAT translation table)** every (source IP address, port #) to (NAT IP address, new port #) translation pair

- **incoming datagrams**: replace (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

---

<table>
<thead>
<tr>
<th>RFC1518 name</th>
<th>IP address range</th>
<th>number of addresses</th>
<th>classful description</th>
<th>largest CIDR block (subnet mask)</th>
<th>host ID size</th>
</tr>
</thead>
<tbody>
<tr>
<td>24-bit block</td>
<td>10.0.0.0 – 16.255.255.255</td>
<td>16,777,216</td>
<td>single class A</td>
<td>10.0.0.0/8 (255.0.0.0)</td>
<td>24 bits</td>
</tr>
<tr>
<td>20-bit block</td>
<td>172.16.0.0 – 172.31.255.255</td>
<td>1,048,576</td>
<td>16 contiguous class Bs</td>
<td>172.16.0.0/12 (255.240.0.0)</td>
<td>20 bits</td>
</tr>
<tr>
<td>16-bit block</td>
<td>192.168.0.0 – 192.168.255.255</td>
<td>65,536</td>
<td>256 contiguous class Cs</td>
<td>192.168.0.0/16 (255.255.0.0)</td>
<td>16 bits</td>
</tr>
</tbody>
</table>
NAT: Network Address Translation

NAT translation table

<table>
<thead>
<tr>
<th>WAN side addr</th>
<th>LAN side addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>138.76.29.7, 5001</td>
<td>192.168.1.2, 3345</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

1: host 192.168.1.2 sends datagram to 128.119.40.186, 80

2: NAT router changes datagram source addr from 192.168.1.2, 3345 to 138.76.29.7, 5001, updates table

3: Reply arrives dest. address: 138.76.29.7, 5001

4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 192.168.1.2, 3345

Network Address Translation: Advantages

- No need to be allocated range of addresses from ISP: - just one public IP address is used for all devices
  - 16-bit port-number field allows 60,000 simultaneous connections with a single LAN-side address!
  - can change ISP without changing addresses of devices in local network
  - can change addresses of devices in local network without notifying outside world

- Devices inside local net not explicitly addressable, visible by outside world (a security plus)
Network Address Translation: Problems

- If both hosts are behind NAT, they will have difficulty establishing connection

- NAT is controversial:
  - routers should process up to only layer 3
  - violates end-to-end argument
    - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - address shortage should instead be solved by having more addresses --- IPv6!