BGP

Brad Smith
Administrativia

• How are the labs going?
• This week
  – STP quiz Thursday, 5/9
• Next week
  – STP lab due Wednesday (in BE 301a!), 5/15
  – BGP quiz Thursday (remember required reading), 5/16
• Following week
  – Project status report due Tuesday, 5/21
  – BGP lab due Wednesday, 5/22
  – Multicast quiz Thursday, 5/23
• Projects due
  – Presentations last week of class and final slot (I’ll schedule with random assignments)
    • Alex Lowe, John, Jeff, Dennis, Erik, David, Jeff
  – Write-up, lab, and answer key bye last day of quarter (June 12th)
Functional Classification: IGP vs. EGP

- **An autonomous system** (AS) or **routing domain** is a region of the Internet that is administered by a single entity
  - UCSC’s network
  - IBM’s corporate network
  - AT&T’s ISP network

- Routing inside an AS
  - Focus is on performance
  - Popular protocols: RIP, OSPF
  - Called *intra-domain* or *internal gateway (IGP)* routing

- Routing between ASs
  - Focus is on policy
  - Popular protocol: BGP
  - Called *inter-domain* or *external gateway (EGP)* routing
Functional Classification: IGP vs. EGP

- An **autonomous system** (AS) or **routing domain** is a region of the Internet that is administered by a single entity
  - UCSC’s network
  - IBM’s corporate network
  - AT&T’s ISP network

- Routing inside an AS
  - Focus is on performance
  - Popular protocols: RIP, OSPF
  - Called *intra-domain* or *internal gateway (IGP)* routing

- Routing between ASs
  - Focus is on policy
  - Popular protocol: BGP
  - Called *inter-domain* or *external gateway (EGP)* routing
How ensure *correct* routes?

• Recall requirement for correctness of routing protocol
  – Loop-free
  – Desired path characteristics

• Two strategies for ensuring correctness
  – Use identical algorithm for selecting paths
    • Share minimal topology information
    • Use identical path selection algorithm at all nodes
    • Used for IGP/Intra-domain routing
    • Use link-state or distance vector protocol
  – Use custom (private) algorithm for selecting paths
    • Share full path information
    • Use policy-specific path selection algorithm at each node
    • Used for EGP/Inter-domain routing
    • Use path-vector protocol
How ensure *correct* routes?

- Recall requirement for correctness of routing protocol
  - Loop-free
  - Desired path characteristics
- Two strategies for ensuring correctness
  - Use identical algorithm for selecting paths
    - Share minimal topology information
    - Use identical path selection algorithm at all nodes
    - Used for IGP/Intra-domain routing
    - Use link-state or distance vector protocol
  - Use custom (private) algorithm for selecting paths
    - Share full path information
    - Use policy-specific path selection algorithm at each node
    - Used for EGP/Inter-domain routing
    - Use path-vector protocol
Routing Algorithms

• **Distance-Vector**
  - Vectors of destination and distance sent to neighbors
    - “Tell your neighbors about the rest of the network”
  - Destination in terms of a network prefix
  - Distance in terms of a metric: hop count, delay, bandwidth
  - Use Distributed Bellman-Ford path selection algorithm
  - Popular protocol: Routing Information Protocol (RIP)

• **Link-State**
  - Flood description of your links (link state)
    - “Tell the rest of the network about your neighbors”
  - Links described by
    - End-point routers of subnet in internet
    - Cost of subnet: delay, bandwidth
  - Use Dijkstra path selection algorithm
  - Popular protocol: Open Shortest Path First (OSPF)

• **Path-Vector**
  - Routes advertised as full-paths
  - Paths described by sequence of ASs
  - Popular protocol is Border Gateway Routing Protocol (BGP)
Routing Algorithms

• **Distance-Vector**
  - Vectors of destination and distance sent to neighbors
    • “Tell your neighbors about the rest of the network”
  - Destination in terms of a network prefix
  - Distance in terms of a metric: hop count, delay, bandwidth
  - Use Distributed Bellman-Ford path selection algorithm
  - Popular protocol: Routing Information Protocol (RIP)

• **Link-State**
  - Flood description of your links (link state)
    • “Tell the rest of the network about your neighbors”
  - Links described by
    • End-point routers of subnet in internet
    • Cost of subnet: delay, bandwidth
  - Use Dijkstra path selection algorithm
  - Popular protocol: Open Shortest Path First (OSPF)

• **Path-Vector**
  - Routes advertised as full-paths
  - Paths described by sequence of ASs
  - Popular protocol is Border Gateway Routing Protocol (BGP)
Policies

• Each AS selects paths based on *it’s own policies*

• Called “independent route selection”
  – See paper “Persistent route oscillations in inter-domain routing”
  – “…domains independently choose their route preference functions.”

• Policies reflect many issues
  – Business relationships
  – Traffic engineering
  – Scalability
  – Security

• *Is a very different world!*
Policies

• **Business relationships** - policy arising from economic or political relationships
  – Customer-provider – customer pays provider to forward traffic
  – Peer-to-peer – mutually beneficial traffic exchange with no payments
    • “Settlement-free peering”
  – Backup – peer-to-peer but for backup

• **Traffic engineering** - managing traffic to achieve performance requirements
  – Manage outbound traffic to balance load or control congestion
  – Manage inbound traffic with similar goals

• **Scalability**
  – Limit routing table size
  – Limit rate of route changes
Policies

- **Security**
  - Discard invalid routes (e.g. private prefixes, unallocated prefixes, etc.)
  - Enforce routing peering policies
  - Protect internal services with route filtering
  - Block denial-of-service attacks (e.g. limit number of prefixes allowed)
Review

• BGP routing enforces policies
  – Business relationships: e.g. customers, providers, peers.
  – Traffic engineering
  – Scalability/resource management
  – Security

• “Independent route selection”
  – Private algorithm (determined by policy)... “domains independently choose their route preference functions.”
  – Exchange full-paths to ensure loop freedom
  – Path-Vector routing
Policy-Based, Path-Vector Algorithm
Purpose is Policies

• Largely ignores distance.

• Primary purpose is to implement policies on how traffic should be handled

• When should I use BGP?
  – Dual- or multi-homed
  – Providing partial or full Internet routing to a downstream customer
  – Anytime the AS path information is required
  – *When you need to make a decision based on policy considerations!*
  – Or if you’re bored and want to read the Internet BGP table😊
Single-Homed AS

- A single homed AS does not need BGP!!
- AS 100 is only connected to one AS
- Use static routes
Multi-Homed AS

- AS 200 is multi-homed
- AS 200 needs to run BGP
Policy mechanisms

- **Import transformation** – $I_{ij}$
  - Applied to new route learned at AS “$i$” from AS “$j$”.
  - Applies local policy to determine if route accepted.
  - If so, applies transformations defined by policy.

- **Export transformation** – $E_{ij}$
  - Applied to new routes selected at AS “$i$” for export to AS “$j$”.
  - Applies local policy to determine if route should be exported
  - If so, applies transformation defined by policy.

- **Preference function** – $\lambda_i$
  - Selects best route for a given destination of those learned from neighbors
Updates composed of Path Attributes

- **NLRI**
  - Network layer reachability information
  - The IP prefix this update applies to
- **AS_PATH**
  - List of AS’s a route has traversed
  - Used to ensure loop freedom, and influence decision process
- **LOCAL_PREF**
  - Local to an AS
  - Used to coordinate route processing
- **MED**
  - Multi-exit discriminator
  - Conveys preference of multiple entry points to neighboring AS’s
- **Others**
  - **CLUSTER_LIST**
  - **ORIGINATOR_ID**
  - **AGGREGATOR**
  - **ATOMIC-AGGREGATE**
- **Community attributes**
  - Variable-length string used to control route processing in remote routers...
\( \lambda_i \) – “decision process”

- Policy implemented in first 4 steps using update attributes
  - LOCAL_PREF at step 1 allows operator to override other steps
  - First 4 steps identify set of equally good paths
- Last three steps are tie-breakers
  - Step 5 – always prefer someone else’s bandwidth:)
  - Step 6 – use as little of our bandwidth as possible
- Vendors may (do) augment (but not reorder) this function

<table>
<thead>
<tr>
<th>Step</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Highest LOCAL_PREF</td>
</tr>
<tr>
<td>2</td>
<td>Lowest AS_PATH length</td>
</tr>
<tr>
<td>3</td>
<td>Lowest ORIGIN type</td>
</tr>
<tr>
<td>4</td>
<td>Lowest MED</td>
</tr>
<tr>
<td>5</td>
<td>External over Internal path</td>
</tr>
<tr>
<td>6</td>
<td>Lowest IGP cost to border router</td>
</tr>
<tr>
<td>7</td>
<td>Lowest router ID</td>
</tr>
</tbody>
</table>
Import and Export transformations

- **Filter** routes for import/export from/to neighbor ASs
- **Modify route attributes** to influence preference function
- **Tag** route with *community attribute* to coordinate actions among a group of routers
- Implemented using a *route-map* in Cisco IOS (and similar for other vendors)
  - Set of conditions for routes it applies to
  - Actions (reject or modify)
- Examples later...

- Much of the following from “BGP Routing Policies in ISP Networks” by Caesar and Rexford (see class web)
Pseudo-code is my approximation

Import transformation applied on receipt of update.

Export transformations applied before SendUpdate().

Preference function invoked to select new route.
Review

• Use BGP when need to make routing decision not based on distance.
  – Multi-homed
  – When you need to make a decision based on policy considerations!

• Policies implemented with three mechanisms
  – Import and export transformations
  – Preference function (BGP decision process)

• Data for filters and decision process carried in path attributes
  – NLRI, AS_PATH, LOCAL_PREF, MED, community attributes, etc.
  – Filters modify path attributes
  – Preference function is defined in terms of path attributes

• BGP path selection is composed of the following steps
  – highest LOCAL_PREF
  – lowest AS_PATH length
  – lowest ORIGIN type
  – lowest MED
  – external over internal path
  – lowest IGP cost to border router
  – lowest router ID
BGP
Overview

• BGP = Border Gateway Protocol

• Currently in version 4

• Uses TCP to send routing messages

• Network administrators can specify routing policies

• BGP’s goal is to find any path (not an optimal one) that meets the policies of all the ASes it transits.
Big Picture

AS 1

Router

AS 2

Router

Router

AS 3

Router

AS 4

Router
Peer Establishment

• Both peers send an OPEN message to TCP port 179
  – IP addresses must be configured correctly
  – update-source must be configured correctly

• If OPENs are exchanged at the exact same time then two TCP sessions will be established but the TCP session from the Rtr with the highest router-ID will be kept and the other torn down

• If RtrA’s OPEN to RtrB is the OPEN that sets up the session, RtrA is said to have “Actively” opened the session and RtrB is said to have “Passively” opened the session

• R4 Actively opened this session:
  r4# show ip bgp neighbors 7.7.7.7
  [snip]
  Local host: 4.4.4.4, Local port: 12916
  Foreign host: 7.7.7.7, Foreign port: 179
External BGP

- BGP peer in different AS
- Usually directly connected
- If not directly connected, use `ebgp-multihopself`

Router A
- `router bgp 100`
- `neighbor 1.1.1.2 remote-as 200`

Router B
- `router bgp 200`
- `neighbor 1.1.1.1 remote-as 100`
Internal BGP

- BGP peer in same AS
- May be several hops away
- iBGP must have a logical full mesh!
- iBGP allows multiple routers to implement BGP in an AS; these routers collectively implement the desired routing policy
Internal BGP

- Peer with loopback addresses
- iBGP session is not dependent on a single interface
- Loopback interface does not go down
- Provides stability!!
- Use `update-source` keyword

Router A
router bgp 100
neighbor 2.2.2.2 remote-as 100
neighbor 2.2.2.2 update-source loop0

Router B
router bgp 100
neighbor 1.1.1.1 remote-as 100
neighbor 1.1.1.1 update-source loop0
Review

• Two components to BGP: eBGP and iBGP
  – iBGP assumes full mesh among routers for an AS
  – iBGP allows multiple routers to implement BGP in an AS; these routers collectively implement the desired routing policy

• BGP peering done over TCP connections (unique among routing protocols)
  – Provides reliability
  – Can be multihop

• Peering typically done between loopback interfaces
  – Loopback interface only fails if router fails
  – Fate-Sharing principle!
Examples
Assigned reading

“BGP Routing Policies in ISP Networks”!
Policy examples – business relationship

• Prefer...
  – ...routes learned from **Customers** over
    • Earn as much $ as possible
  – ...routes learned from **Peers** over
    • Peering relationships based on balance of traffic
  – ...routes learned from **Providers**.
    • Spend as little $ as possible
Assume A is a peer of B, C is a provider to B, and R3 is connected to a customer
  - For outbound traffic, favor customers over peers over providers

Modify import filter
  - On R3 for routes from customer: \texttt{LOCAL\_PREF} = 90
  - On R1 and R2 for routes from A: \texttt{LOCAL\_PREF} = 80
  - On R4 for routes from C: \texttt{LOCAL\_PREF} = 70

Traffic I send will prefer customer over peer (A) over provider (C)
Geographical Control

• ISP that spans U.S. and Europe
  – Want to minimize use of expensive trans-Atlantic link

• Modify import filter
  – For European routers
    • Routes from European peers: `LOCAL_PREF = High`
  – For U.S. routers
    • Routes from U.S. peers: `LOCAL_PREF = High`
No Transit to Peer

- Same assumption (A is peer and C is provider)
  - B doesn’t want to provide transit service for traffic between A and C
- Import filters
  - On R1 and R2 for routes from A: \texttt{add community attribute X}_{\text{peer}}
  - On R4 for routes from C: \texttt{add community attribute X}_{\text{provider}}
- Export filters
  - On R4 for routes with \texttt{X}_{\text{peer}}: \texttt{reject for export to C}
  - On R1 and R2 for routes with \texttt{X}_{\text{provider}}: \texttt{reject for export to A}
- I won’t receive traffic from C for A or vice-versa
Load Balancing

- B wants to shift traffic from its link to A to its link to C
  - E.g. because link to A is overloaded
- 2 choices
  - Import filters in R1 and R2
    - Routes from A: lower LOCAL_PREF value below routes learned from C
  - Import filter on R4
    - Routes from C: higher LOCAL_PREF value above routes learned from A
- Can use regular expression for prefix
Control Inbound Traffic

- B wants to shift traffic load from R1 to R2 (same neighbor)
- Export filter on R1
  - All routes: increase value of MED attribute relative to R2
“Remote Control”

- C agrees to allow B to control flow of traffic into B
- When B wants to route inbound traffic via A
  - Export filters on routers R1 and R2: add community attribute $X_{\text{high}}$
  - Export filters on router R4: add community attribute $X_{\text{low}}$
- And vice-versa when B wants to route inbound traffic via C...
- Import filters on C
  - Routes for prefixes in B with $X_{\text{high}}$: LOCAL_PREF = 75
  - Routes for prefixes in B with $X_{\text{low}}$: LOCAL_PREF = 60
Review

• You can do impressive things with BGP policy mechanisms!

• A simple example is for implementing the classic business relationship of preferring routes learned from **Customers** over those learned from **Peers** over those learned from **Providers**.
  – In the import filter for **Customers**, set `LOCAL_PREF` high (e.g. 90)
  – In the import filter for **Peers**, set `LOCAL_PREF` high (e.g. 90)
  – In the import filter for **Providers**, set `LOCAL_PREF` high (e.g. 70)