CS4450

Computer Networks:
Architecture and Protocols

Lecture 14
Border-Gateway Protocol

Rachit Agarwal
Announcements

• Graded copies and regrades
  • Graded copies available during office hours
  • We will release them by Saturday
    • Finding ways to make notes on your solutions/grades

• Course Grading
  • I am not in favor of changing policies mid-semester
  • However, as I have acknowledged, prelim 1 was harder than usual
  • Here is a possible improvement
    • We compute the two grades as in current grading mechanism
    • We compute a third grade based on “curved” grading
      • You get the best of the three grades
  • Your grade can only get better using this improvement
Goals for today’s lecture

• Dive deep into Border Gateway protocol
  • One of the most non-intuitive protocols in the course
  • Driven by “business goals”, rather than “performance goals”
    • I will try to provide as much intuition as possible
    • But, for the above reasons, BGP is one of the harder protocols

• Understanding BGP
  • Do a lot of examples
  • We will focus on a synchronous version: one node acts at a time
    • In practice, BGP implementations are fully asynchronous
Recap from last lecture
Recap: Three requirements for addressing

• Scalable routing
  • How must state must be stored to forward packets?
  • How much state needs to be updated upon host arrival/departure?

• Efficient forwarding
  • How quickly can one locate items in routing table?

• Host must be able to recognize packet is for them
Recap: Today’s Addressing: CIDR

• Addressing based on L2 names does not enable scalable routing

• Today’s addressing mechanism: Classless Inter-domain Routing
  • Idea: Flexible division between network and host addresses
    • Prefix is network address
    • Suffix is host address

• Example:
  • 128.84.139.5/23 is a 23 bit prefix with:
    • First 23 bits for network address
    • Next 9 bits for host addresses: maximum 2^9 hosts

• Terminology: “Slash 23”
Recap: What does a computer network look like?

“Autonomous System (AS)” or “Domain”
Region of a network under a single administrative entity

“Border Routers”

An “end-to-end” route

“Interior Routers”
Recap: What does a computer network look like?

“Autonomous System (AS)” or “Domain”
Region of a network under a single administrative entity

“Autorimous System” or “Domain”
Region of a network under a single administrative entity

“Border Routers”

An “end-to-end” route

“Interior Routers”
Recap: IP addressing => Scalable Routing via “aggregation”
Recap: IP addressing => Scalable Routing via “aggregation”

Can add new hosts/networks without updating the routing entries at France Telecom

France Telecom  a.*.*.* is this way

AT&T  a.0.0.0/8

LBL  a.b.0.0/16

foo.com  a.d.0.0/16

Cornell  a.c.0.0/16
Recap: Administrative Structure Shapes Inter-domain Routing

• ASes want freedom to pick routes based on policy
  • “My traffic can’t be carried over my competitor’s network!”
  • “I don’t want to carry A’s traffic through my network!”
  • Cannot be expressed as Internet-wide “least cost”

• ASes want autonomy
  • Want to choose their own internal routing protocol
  • Want to choose their own policy

• ASes want privacy
  • Choice of network topology, routing policies, etc.
Recap: Business Relationships Shape Topology and Policy

- Three basic kinds of relationships between ASes
  - AS A can be AS B’s *customer*
  - AS A can be AS B’s *provider*
  - AS A can be AS B’s *peer*

- Business implications
  - Customer *pays* provider
  - Peers *don’t pay* each other
    - Exchange roughly equal traffic
Recap: Business Relationships

Relations between ASes

provider ←→ customer
peer → peer

Business Implications

• Customers pay provider
• Peers don’t pay each other
Questions?
Why Peer?

Relations between ASes

provider ↔ peer ↔ customer ↔ peer

Business Implications

- Customers pay provider
- Peers don’t pay each other

E.g., D and E talk a lot

Peering saves B and C money
ASes provide “transit” between their customers

Peers do not provide transit between other peers
Routing Follows the Money

- ASes provide “transit” between their customers and providers
- Peers do not provide transit between other peers and providers
Recap: Administrative Structure Shapes Inter-domain Routing

- ASes want freedom to pick routes based on **policy**
  - “My traffic can’t be carried over my competitor’s network!”
  - “I don’t want to carry A’s traffic through my network!”
  - Cannot be expressed as Internet-wide “least cost”

- ASes want **autonomy**
  - Want to choose their own internal routing protocol
  - Want to choose their own policy

- ASes want **privacy**
  - Choice of network topology, routing policies, etc.
Choice of Routing Algorithm

- Link State (LS) vs. Distance Vector (DV)
  - LS offers no privacy — broadcasts all network information
  - LS limits autonomy — need agreement on metric, algorithm

- DV is a decent starting point
  - Per-destination updates by intermediate nodes give us a hook
  - But, wasn’t designed to implement policy
  - ... and is vulnerable to loops if shortest paths not taken

The “Border Gateway Protocol” (BGP) extends Distance-Vector ideas to accommodate policy
Inter-domain Routing: Setup

- Destinations are IP prefixes (12.0.0.0/8)

- Nodes are Autonomous Systems (ASes)
  - Internals of each AS are hidden

- Links represent both physical links and business relationships

- BGP (Border Gateway Protocol) is the Interdomain routing protocol
  - Implemented by AS border routers
Each AS selects the "best" route it hears advertised for a prefix.

An AS advertises its best routes to one or more IP prefixes.

Sound familiar?
BGP Inspired by Distance Vector

- Per-destination route advertisements
- No global sharing of network topology
- Iterative and distributed convergence on paths
- But, four key differences
BGP vs. DV

(1) BGP does not pick the shortest path routes!

- Distance vector: choose shortest paths/routes
- BGP: choose selects route based on policy, not shortest distance/least cost

Node 2 may prefer 2, 3, 1 over 2, 1

- How do we avoid loops?
BGP vs. DV

(2) Path-vector Routing

- Idea: advertise the entire path
  - Distance vector: send *distance metric* per dest. d
  - Path vector: send the *entire path* for each dest. d
Loop Detection with Path-Vector

- Node can easily detect a loop
- Look for its **own node identifier** in the path
- Node can simply **discard** paths with loops
- e.g. node 1 sees itself in the path 3, 2, 1

```
3
```

```
2
```

```
1
```

```
d: path (2,1)
```

```
d: path (1)
```

```
d: path (3,2,1)
```

```
d
```
BGP vs. DV

(2) Path-vector Routing

- Idea: advertise the entire path
  - Distance vector: send *distance metric* per dest. d
  - Path vector: send the *entire path* for each dest. d

- Benefits
  - Loop avoidance is easy
  - Flexible policies based on entire path
(3) Selective Route Advertisement

• For policy reasons, an AS may choose not to advertise a route to a destination.

• As a result, reachability is not guaranteed even if the graph is connected.

Example: AS#2 does not want to carry traffic between AS#1 and AS#3.
For scalability, BGP may aggregate routes for different prefixes.
The stable paths problem can be illustrated in Fig. 1(a). If we arrive at node 4, we produce a specification called SPVP that assigns paths. This assignment implicitly defines a tree rooted at the origin. Note, however, that this is not always a spanning tree. The stable path assignment prefers longer paths to shorter paths. The stable path assignment is a stable path assignment for the solution for the problem called SHORTEST PATH. If no such assignment exists, then the solution for the problem is NAUGHTY GADGET. So far, our examples each has had at most one solution. This is not always the case. The simplest instance, called BAD GADGET, is illustrated in Fig. 3(a). The solution for this problem can diverge for this problem. Finally, by reordering the ranking of paths at node 4, we produce a specification called DISAGREE, presented in Fig. 2(d). This specification has no unique solution to this problem.
The stable path assignment is shown in Fig. 2(c). The stable path assignment implicitly defines a tree rooted at the origin. Note, however, that this is not always a spanning tree. A stable path assignment is a stable path assignment for the solution for the problem called the shortest path tree. This is the solution for paths of equal length. This results in one shortest path tree as illustrated in Fig. 2(b). An alternative solution is illustrated in Fig. 3(b). An alternative solution is depicted in Fig. 3(c). No other path assignments are stable for this problem. Finally, by reordering the ranking of paths at node 4, we produce a specification called the SPVP protocol will always diverge.

So far, our examples each has had at most one solution. This is not always the case. The simplest instance, called the NAUGHTY GADGET, presented in Fig. 2(d). This specification has no unique solution, yet it has the same unique solution as its two solutions. The ranking at node 4 breaks ties between preferring longer paths to shorter paths. The stable path assignment also called a GRIFFIN adds one permitted path if there is no such assignment exists, then the ranking at node 4 breaks ties between preferring shorter paths to longer paths and the solutions are shortest good gadget.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>430</td>
</tr>
<tr>
<td>R3</td>
<td>130</td>
<td>20</td>
<td>30</td>
<td>430</td>
</tr>
</tbody>
</table>
BGP Outline

- BGP Policy
  - Typical policies and implementation
- BGP protocol details
- Issues with BGP
Policy:

Imposed in how routes are selected and exported

- **Selection**: Which path to use
  - Controls whether / how traffic *leaves* the network
- **Export**: Which path to advertise
  - Controls whether / how traffic *enters* the network
Typical Selection Policy

- In decreasing order of priority:
  1. Make or save money (send to customer > peer > provider)
  2. Maximize performance (smallest AS path length)
  3. Minimize use of my network bandwidth (“hot potato”)
  4. ...
## Typical Export Policy

<table>
<thead>
<tr>
<th>Destination prefix advertised by...</th>
<th>Export route to...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>Everyone (providers, peers, other customers)</td>
</tr>
<tr>
<td>Peer</td>
<td>Customers</td>
</tr>
<tr>
<td>Provider</td>
<td>Customers</td>
</tr>
</tbody>
</table>

Known as the “Gao-Rexford” rules. Capture common *(but not required!)* practice.
With Gao-Rexford, the AS policy graph is a DAG (directed acyclic graph) and routes are “valley free”
BGP Outline

- BGP Policy
  - Typical policies and implementation

- BGP protocol details

- Issues with BGP
Who speaks BGP?

Border routers at an Autonomous System
What Does “speak BGP” Mean?

- Implement the **BGP Protocol Standard**
  - Internet Engineering Task Force (IETF) RFC 4271

- Specifies what messages to exchange with other BGP “speakers”
  - Message *types* (e.g. route advertisements, updates)
  - Message *syntax*

- Specifies how to process these messages
  - When you receive a BGP update, do x
  - Follows BGP state machine in the protocol spec and policy decisions, etc.
A border router speaks BGP with border routers in other ASes
A border router speaks BGP with other (interior and border) routers in its own AS
**eBGP, iBGP, IGP**

- **eBGP**: BGP sessions between border routers in **different** ASes
  - Learn routes to external destinations

- **iBGP**: BGP sessions between border routers and other routers within the **same** AS
  - Distribute externally learned routes internally

- **IGP**: Interior Gateway Protocol = **Intradomain routing protocol**
  - Provides internal reachability
  - e.g. Link state, distance vector, etc.
1. Provide internal reachability (IGP)
2. Learn routes to external destinations (eBGP)
3. Distribute externally learned routes internally (iBGP)
4. Travel shortest path to egress (IGP)
Basic Messages in BGP

- **Open**
  - Establishes BGP session

- **Update**
  - **Announcements**: Inform neighbor of new routes
  - **Withdrawals**: Inform neighbor of old routes that become inactive

- **Keepalive**
  - Inform neighbor that connection is still viable
Route Updates

- Format: `<IP prefix: route attributes>`
- Two kinds of updates:
  - **Announcements**: new routes or changes to existing routes
  - **Withdrawals**: remove routes that no longer exist
- Route Attributes
  - Describe routes, used in *selection/export* decisions
  - Some attributes are *local*
    - i.e. private within an AS, not included in announcements
  - Some attributes are *propagated* with eBGP route announcements
  - Many standardized attributes in BGP
Route Attributes (1): ASPATH

- Carried in route announcements
- Vector that lists all the ASes a route advertisement has traversed (in reverse order)
Route Attributes (2): **LOCAL PREF**

- “Local Preference”
- Used to choose between different AS paths
- The higher the value, the more preferred
- Local to an AS; carried only in iBGP messages

![BGP table at AS4:](image)

<table>
<thead>
<tr>
<th>Destination</th>
<th>AS Path</th>
<th>Local Pref</th>
</tr>
</thead>
<tbody>
<tr>
<td>140.20.1.0/24</td>
<td>AS3 AS1</td>
<td>300</td>
</tr>
<tr>
<td>140.20.1.0/24</td>
<td>AS2 AS1</td>
<td>100</td>
</tr>
</tbody>
</table>
Route Attributes (3) : MED

- “Multi-Exit Discriminator”
- Used when ASes are interconnected via two or more links
- Specifies how close a prefix is to the link it is announced on
- Lower is better
- AS announcing prefix sets MED
- AS receiving prefix (optionally!) uses MED to select link
Route Attributes (4): IGP Cost

- Used for hot-potato routing
- Each router selects the closest egress point based on the path cost in intra-domain protocol
Using Attributes

• Rules for route selection in priority order

1. Make or save money (send to customer > peer > provider)
2. Maximize performance (smallest AS path length)
3. Minimize use of my network bandwidth (“hot potato”)
4. …
## Using Attributes

- Rules for route selection in priority order

<table>
<thead>
<tr>
<th>Priority</th>
<th>Rule</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LOCAL PREF</td>
<td>Pick highest LOCAL PREF</td>
</tr>
<tr>
<td>2</td>
<td>ASPATH</td>
<td>Pick shortest ASPATH length</td>
</tr>
<tr>
<td>3</td>
<td>MED</td>
<td>Lowest MED preferred</td>
</tr>
<tr>
<td>4</td>
<td>eBGP &gt; iBGP</td>
<td>Did AS learn route via eBGP (preferred) or iBGP?</td>
</tr>
<tr>
<td>5</td>
<td>iBGP path</td>
<td>Lowest IGP cost to next hop (egress router)</td>
</tr>
<tr>
<td>6</td>
<td>Router ID</td>
<td>Smallest next-hop router’s IP address as tie-breaker</td>
</tr>
</tbody>
</table>
BGP Outline

- BGP Policy
  - Typical policies and implementation

- BGP protocol details

- Issues with BGP
BGP: Issues

- Reachability
- Security
- Convergence
- Performance
- Anomalies
Reachability

- In normal routing, if graph is connected then reachability is assured
- With policy routing, this doesn’t always hold
Convergence

- If all AS policies follow Gao-Rexford rules,
  - Then BGP is guaranteed to converge (safety)

- For arbitrary policies, BGP may fail to converge!
Example of Policy Oscillation

“1” prefers “1 3 0” over “1 0” to reach “0”
Step-by-step Policy Oscillation

Initially: nodes 1, 2, 3 know only shortest path to 0
Step-by-step Policy Oscillation

1 advertises its path 1 0 to 2
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

3 advertises its path 3 0 to 1
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

1 withdraws its path 1 0 from 2
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

2 advertises its path 2 0 to 3

advertise: 2 0
Step-by-step Policy Oscillation
3 withdraws its path 3 0 from 1
Step-by-step Policy Oscillation
1 advertises its path 1 0 to 2
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

2 withdraws its path 2 0 from 3

withdraw: 2 0
Step-by-step Policy Oscillation

We are back to where we started!
The stable path assignment is shown in Fig. 2(c). The stable path assignment implicitly defines a tree rooted at the origin. Note, we prefer longer paths to shorter paths. The stable path assignment is a solution for the problem called SHORTEST PATHS PROBLEM AND INTERDOMAIN ROUTING.

A modification of the SHORTEST PATHS PROBLEM 1, then we arrive at a unique solution to this problem.

So far, our examples each have at most one solution. This is not always the case. The simplest instance, called BAD GADGET, presented in Fig. 2(d). This specification has no solutions and the SPVP protocol will always diverge.

Therefore, any stable path assignment adds one permitted path. However, as is explained in Section IV, the protocol specifies that we produce a ranking of paths at node 4, we produce a specification called SHORTEST PATHS PROBLEM 2, depicted in Fig. 1(c). The stable path assignment is solvable NAUGHTY GADGET. So far, our examples each have at most one solution. This is not always the case. The simplest instance, called BAD GADGET, presented in Fig. 2(d). This specification has no solutions and the SPVP protocol will always diverge.

Therefore, any stable path assignment adds one permitted path. However, as is explained in Section IV, the protocol specifies that we produce a specification called SHORTEST PATHS PROBLEM 2, depicted in Fig. 1(c). The stable path assignment is solvable NAUGHTY GADGET.

Therefore, any stable path assignment adds one permitted path. However, as is explained in Section IV, the protocol specifies that we produce a ranking of paths at node 4, we produce a specification called SHORTEST PATHS PROBLEM 2, depicted in Fig. 1(c). The stable path assignment is solvable NAUGHTY GADGET.

Therefore, any stable path assignment adds one permitted path. However, as is explained in Section IV, the protocol specifies that we produce a ranking of paths at node 4, we produce a specification called SHORTEST PATHS PROBLEM 2, depicted in Fig. 1(c). The stable path assignment is solvable NAUGHTY GADGET.

Therefore, any stable path assignment adds one permitted path. However, as is explained in Section IV, the protocol specifies that we produce a ranking of paths at node 4, we produce a specification called SHORTEST PATHS PROBLEM 2, depicted in Fig. 1(c). The stable path assignment is solvable NAUGHTY GADGET.

Therefore, any stable path assignment adds one permitted path. However, as is explained in Section IV, the protocol specifies that we produce a ranking of paths at node 4, we produce a specification called SHORTEST PATHS PROBLEM 2, depicted in Fig. 1(c). The stable path assignment is solvable NAUGHTY GADGET.

Therefore, any stable path assignment adds one permitted path. However, as is explained in Section IV, the protocol specifies that we produce a ranking of paths at node 4, we produce a specification called SHORTEST PATHS PROBLEM 2, depicted in Fig. 1(c). The stable path assignment is solvable NAUGHTY GADGET.
A modification of the work of et al. (2014) gives us the following method: STABLE PATHS PROBLEM AND INTERDOMAIN ROUTING.

Shoest path solutions are unique for node 1 and 2, depicted in Fig. 1(c). The stable path assignment is illustrated in Fig. 2(b) is not a shortest path tree. This is the solution for the stable path assignment for node 4, called the specification of Fig. 3(c). No other path assignments are stable for node 4, yet it has the same unique solution as Fig. 3.

This problem can be solved for node 2, depicted in Fig. 1(d). In both cases, the ranking functions prefer shorter paths to longer paths and the solutions are shortest path trees. Note that the ranking at node 4 breaks ties between paths of equal length. This results in one shortest path tree as illustrated in Fig. 2(a) presents a stable paths of equal length. This results in one shortest path tree as shown in Fig. 2(c). However, that this is not always a spanning tree.

For example, Fig. 2(a) presents a stable path trees. Note that the ranking at node 4 breaks ties between paths of equal length. This results in one shortest path tree as shown in Fig. 2(c). However, that this is not always a spanning tree.

So far, our examples each has had at most one solution. This is not the case for node 3, presented in Fig. 2(d). This specification has no solution and the SPVP protocol will always diverge.
Convergence

- If all AS policies follow Gao-Rexford rules,
  - Then BGP is guaranteed to converge (safety)
- For arbitrary policies, BGP may fail to converge!
- Why should this trouble us?
Performance Non-Issues

- Internal Routing
  - Domains typically use “hot potato” routing
  - Not always optimal, but economically expedient

- Policy **not about performance**
  - So policy-chosen paths aren’t shortest

- AS path length can be misleading
  - 20% of paths **inflated** by at least 5 router hops
Performance (example)

- AS path length can be misleading
  - An AS may have many router-level hops

BGP says that path 4 1 is better than path 3 2 1
Performance: Real Issue

Slow Convergence

- BGP outages are biggest source of Internet problems

- Labovitz et al. *SIGCOMM'97*
  - 10% of routes available less than 95% of the time
  - Less than 35% of routes available 99.99% of the time

- Labovitz et al. *SIGCOMM 2000*
  - 40% of path outages take 30+ minutes to repair

- But most popular paths are very stable
BGP Misconfigurations

- BGP protocol is both **bloated** and **underspecified**
  - Lots of attributes
  - Lots of leeway in how to set and interpret attributes
  - Necessary to allow autonomy, diverse policies
  - … But also gives operators plenty of rope

- Much of this configuration is **manual** and **ad hoc**

- And the core abstraction is **fundamentally flawed**
  - Disjoint per-router configuration to effect AS-wide policy
  - Now strong industry interest in changing this!
Security

- An AS can claim to serve a prefix that they actually don’t have a route to (blackholing traffic)
  - Problem not specific to policy or path vector
  - Important because of AS autonomy
  - *Fixable*: make ASes prove they have a path

- But…

- AS may forward packets along a route different from what is advertised
  - Tell customers about a fictitious short path…
  - Much harder to fix!
BGP: How did we get here?

- BGP was designed for a different time
  - Before commercial ISPs and their needs
  - Before address aggregation
  - Before multi-homing

- We don’t get a second chance: ‘clean slate’ designs virtually impossible to deploy today

- Thought experiment: how would you design a policy-driven interdomain routing solution?
  - How would you deploy it?

- 1989: BGP-1 [RFC 1105]
  - Replacement for EGP (1984, RFC 904)
- 1990: BGP-2 [RFC 1163]
- 1991: BGP-3 [RFC 1267]
- 1995: BGP-4 [RFC 1771]
  - Support for Classless Interdomain Routing (CIDR)