Project 2 - BGP Hijacking Attacks

Goal

In this project, using an interactive Mininet demo [1], we will explore some of the vulnerabilities of Border Gateway Protocol (BGP). In particular, we will see how BGP is vulnerable to abuse and manipulation through a class of attacks called BGP hijacking attacks. A malicious Autonomous System (AS) can mount these attacks through false BGP announcements from a rogue AS, causing victim ASes to route their traffic bound for another AS through the malicious AS. This attack succeeds because the false advertisement exploits BGP routing behavior by advertising a shorter path to reach a particular prefix, which causes victim ASes to attempt to use the newly advertised (and seemingly better!) route.

Instructions

Part 1: Background reading, resources and example BGP router configurations

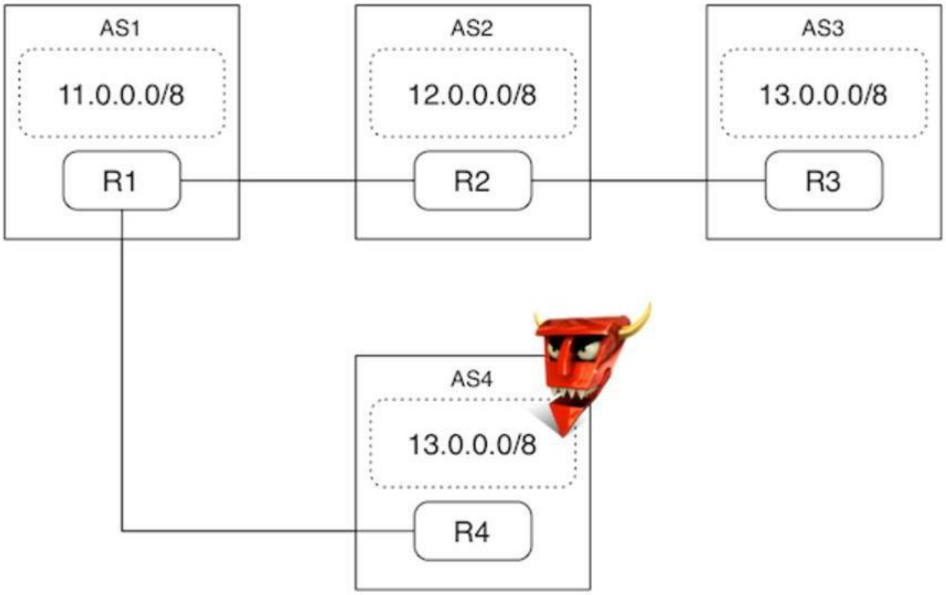
A. Browse this paper as a reference for subsequent tasks and for some important background on Prefix Hijack Attacks. Title of the paper: “Understanding Resiliency of Internet Topology Against Prefix Hijack Attacks”
B. Refer to this resource on configuring a BGP router with Quagga.

C. Project Intro Slides Appended to Project File.

NOTE: Use the same VM that was provided from Project 1 to complete this project!!!

Part 2: Interactive Demonstration using a Mininet Topology and simulated prefixes/paths

The demo creates the network topology shown below, consisting of four ASes and their peering relationships. AS4 is the malicious AS that will mount the attack. Once again, we will be simulating this network in Mininet, however there are some important distinctions to make from our previous projects. In this set up, each container is not a host, but an entire autonomous system. Each host runs a routing daemon (quagga), communicates with other ASes using BGP (bgpd), and configures its own isolated set of routing entries in the kernel (zebra). Each AS has an IP address, which is the IP address of its border router.

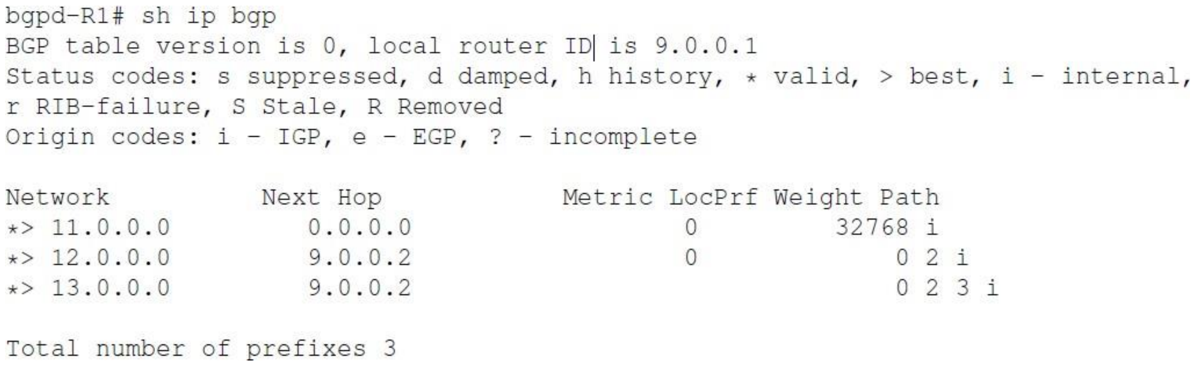


NOTE: In this topology solid lines indicate peering relationships and the dotted boxes indicate the prefix advertised by that AS.

1. First, download and unzip the Project-2 files (modify permissions if necessary).
2. Next, in the `Project-2` directory, start the demo using the following command:
   - `sudo python bgp.py`
3. After loading the topology, the Mininet CLI should be visible. Keep this terminal open throughout the experiment.
4. Start another terminal and navigate to the `Project-2` directory. We will use this terminal to start a remote session with AS1's routing daemon:
   - `./connect.sh`
5. This script will start quagga, which will require access verification. The password is:
   - `en`
6. Next, use the following commands to start the admin shell and view the routing table entries for AS1:
   - `en`
   - You will be prompted for the password again, retype `en`

- `sh ip bgp`

7. You should see output very much like the screen grab below. In particular, notice that AS1 has chosen the path via AS2 and AS3 to reach the prefix 13.0.0.0/8:



```
bgpd-R1# sh ip bgp
BGP table version is 0, local router ID is 9.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
r RIB-failure, S Stale, R Removed
Origin codes: i - IGP, e - EGP, ? - incomplete

Network          Next Hop            Metric LocPrf Weight Path
*> 11.0.0.0         0.0.0.0                  0         32768 i
*> 12.0.0.0         9.0.0.2                  0             0 2 i
*> 13.0.0.0         9.0.0.2                                0 2 3 i

Total number of prefixes 3
```

9. Next, let’s verify that network traffic is traversing this path. Open a third terminal and navigate to the `Project-2` directory. In this terminal you will start a script that continuously makes web requests from a host within AS1 to a web server in AS3:

```
./website.sh
```

10. Leave this terminal running as well, and open a fourth terminal, also in the `Project-2` directory. Now, we will start a rogue AS (AS4) that will connect directly to AS1 and advertise the same 13.0.0.0/8 prefix. This will allow AS4 to hijack the prefix due to the shorter AS Path Length:

```
./start_rogue.sh
```

11. Return to the third terminal window and observe the continuous web requests. After the BGP routing tables converge on this simple network, you should eventually see the attacker start responding to requests from AS1, rather than AS3.

12. Additionally, return to the second terminal and rerun the command to print the routing table. You may need to repeat the steps to establish the remote session if it closes due to inactivity. You should now see the fraudulent advertisement for the 13.0.0.0/8 prefix in the routing table, in addition to the longer unused path to the legitimate owner.

13. Finally, let’s stop the attack by switching to the fourth terminal and using the following command:

```
./stop_rogue.sh
```

14. You should notice a fairly quick re-convergence to the original legitimate route in the third terminal window, which should now be delivering the original traffic. Additionally, you can check the BGP routing table again to see the original path is being traversed.

Part 3: Creating a more complex topology and attack scenario

As demonstrated in Part 2, network virtualization can be very useful in demonstrating and analyzing network attacks that would otherwise require a large amount of physical hardware to accomplish. In Part 3, you are tasked with replicating a different topology and attack scenario to demonstrate the effects of a different instance of a Prefix Hijack Attack.

1. To start, we recommend making a working copy of the code provided to you in the `Project-2` directory. You will likely find this project to be more approachable if you spend time exploring the demo code and fully understanding how each part works rather than immediately trying to edit the code.
2. Next, refer to the referenced paper in Part 1A, and locate Figure 1.
3. Edit the working copy of the demo code you just made to reconstruct the topology in Figure 1. When complete, you should be able to use the commands from Part 2 to explore the routing tables generated by each border router. For our purposes, you can assume:
   a. All links to be bidirectional peering links.
   b. Each AS advertises a single prefix: AS1: 1.0.0.0/8, AS2: 2.0.0.0/8, AS3: 3.0.0.0/8, AS4: 4.0.0.0/8, AS5: 5.0.0.0/8, AS6: 1.0.0.0/8 (Note: We highly recommend using these prefix values in your configuration to simplify grading and for consistency in communication and discussion in Piazza. However, you may use any valid prefix values in your configuration.)
   c. The number of hosts in each AS is the same as in the provided code.

4. Next, locate Figure 2 in the referenced paper. Draw a topology map using any drawing tool of your choice. See Slide 7 of the Project 2 Intro Video and presentation slides for an example of an adequate topology diagram. You may hand-draw your topology with pencil and paper and scan or photograph your drawing. All configuration values drawn on the map must be legible. Save your topology diagram in PDF format with the name **fig2_topo.pdf.** You must use this filename as part of your submission to receive credit for your diagram.

5. Continue to adapt the code in your working copy to simulate this hijack scenario. When complete, you should be able to use the commands from Part 2 to start a Rogue AS and demonstrate a similar change in routing table information as was shown in Part 2.

6. Finally, create a compressed file (zip format) named gtech_username-Part3.zip, i.e. `gburdell3-Part3.zip` containing your entire attack demonstration. You must include all of the files necessary to run your demo in an empty directory - do **NOT** assume that we will provide any of the files necessary to run your demonstration for grading purposes. Include your fig2_topomap.pdf file in your zip.

Part 3 Configuration Debugging Tips

- When viewing the BGP Tables note the “Status codes”. Give your topology enough time to converge before recreating the hijack simulation portion. It may take a minute or so for your topology to fully converge. You may continue to check the BGP Tables to determine whether the topology has converged

- The order that you set up your peering links using addLink() matters. In previous projects, we manually selected which port on the switch to use. There is an optional parameter to the addLink() call which allows you to specify which switch port to use. In this project, you will not use those options. Therefore, the order of the links matters.

- Some of the commands in the boilerplate code may not be necessary to complete Part 3. Some of it is there just so that you know it exists.

- Check for more descriptive errors in the /logs directory. See the zebra files for the location of additional log files.

- Run "links" on the Mininet CLI terminal to see if all links are connected and OK OK.

- Run "net" on the Mininet CLI terminal to see if your ethernet links are connected as you expect.

- Run "ifconfig -a" on all routers and hosts to ensure that all IP addresses are assigned correctly.

- Run "sh ip bgp" and "sh ip bgp summary" on all routers.

- The command `pingall` may not work and that is fine.

- The website.sh may sometimes hang intermittently. If this happens restart the simulation. We are aware of this issue, and we keep this in mind as we grade your submission. You will not lose points if website.sh hangs so long as we are eventually able to run the simulation.

- Read through the **additional debugging tips on the intro slides.**

What to Turn In

For this project you need to turn in two files to Canvas in a Zip file. Please name the zip file based on your Gatech username (i.e., gburdell3-Part3.zip).

You need to make sure the pdf file **fig2_topo.pdf** is inside `Project-2` directory and now you should be outside `Project-2` directory for the following zip command:

```
zip -r gburdell3-Part3.zip Project-2
```

gburdell3-Part3.zip should be submitted in the main assignment on canvas which is named: **Project 2 - BGP Hijacking Attacks (main assignment)**

Please upload `gburdell3-Part3.zip` directly into canvas.

What you can and cannot share

While discussion of the project in general is always permitted on Piazza, you are not permitted to share your code generated for Part 3. You may quote snippets of the unmodified skeleton code provided to you when discussing the Project. You may **not** share your topology diagram you created in Part 3 Step 5.

Rubric (out of 150 points)

5 pts	Submission	for turning in all the correct demo files with the correct names, and significant effort has been made towards completing the project.
5 pts	Fig 2 Topo Diagram	For turning in the correctly named Topology diagram file: **fig2_topo.pdf** with legible configuration values.
140 pts	Attack Demo	for accurately recreating the topology, links, router configuration, and attack per the instructions. Partial credit is available for this rubric item.

[1] This Project inspired by a Mininet Demo originally presented at SIGCOMM 2014