CS6250 Computer Networks
TA Matthew Kalita

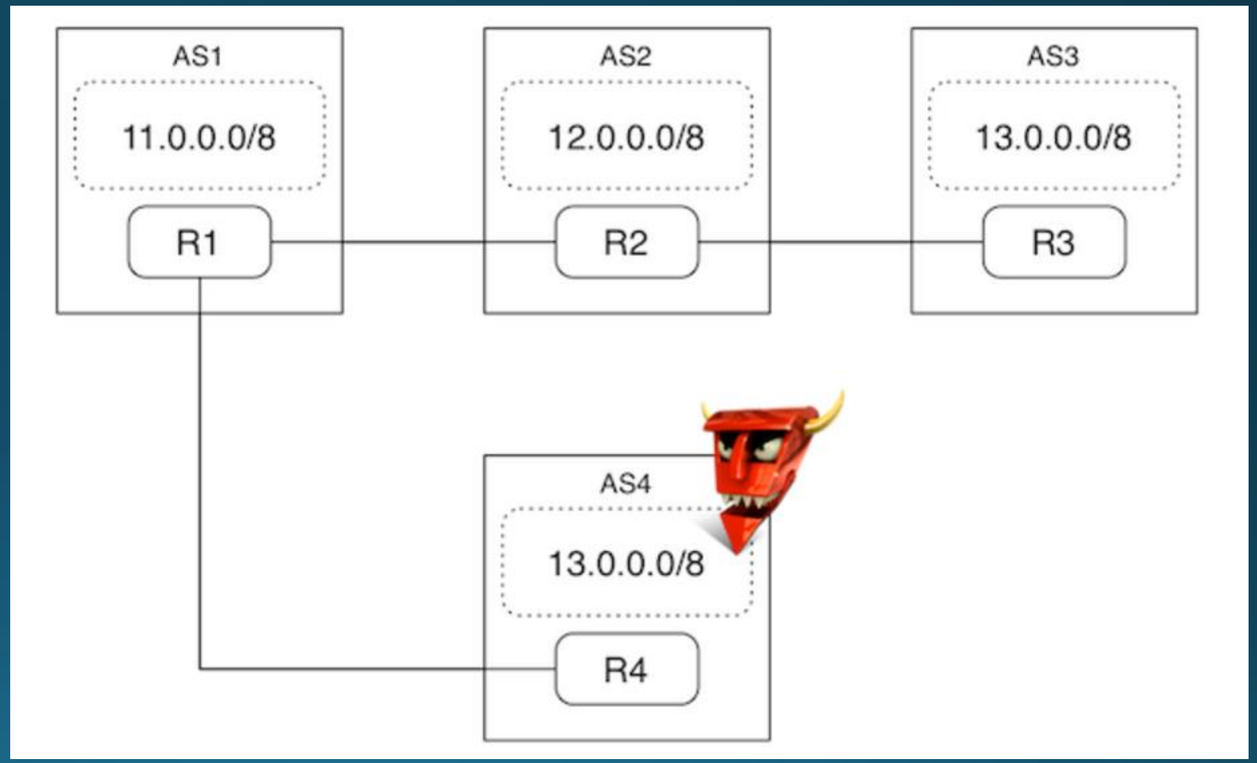
BGP Hijacking Project

Introduction

- In this project, using an interactive Mininet demo, we will explore some of the vulnerabilities of Border Gateway Protocol (BGP)
- 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
- Quagga is an advanced routing software package that provides a suite of TCP/IP based routing protocols.
  - Example: 11.2 BGP router

Demo Network Topology

- The demo creates the network topology shown here, consisting of four ASes and their peering relationships.
- AS_4 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



Files in the Demo

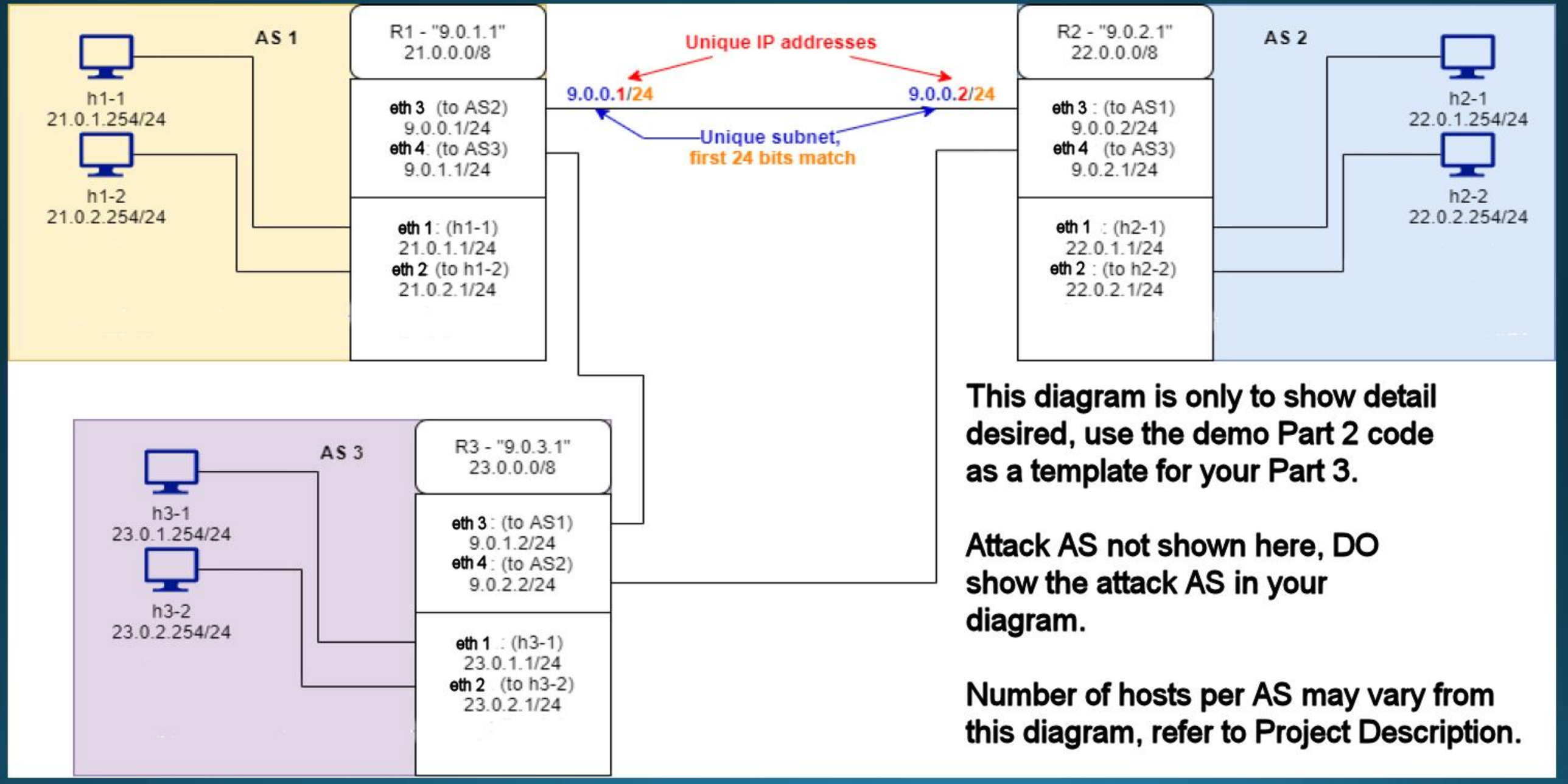
- bgp.py
  - Main file, defines the topology
- connect.sh
  - Connects the routers' bgpd shells
- run.py
  - Runs the simulation
- start_rogue.sh
  - Starts the rogue AS, begins the BGP hijacking
- stop_rogue.sh
  - Stops the rogue AS, stops the BGP hijacking
- webserver.py
- website.sh
  - Initiates the simulation
- bgpd-R(X).conf
  - BGP configuration files for each of the routers
- zebra-R(X).conf
  - zebra is an IP routing manager. It provides kernel routing table updates, interface lookups, and redistribution of routes between different routing protocols.

BGPD Conf. Files

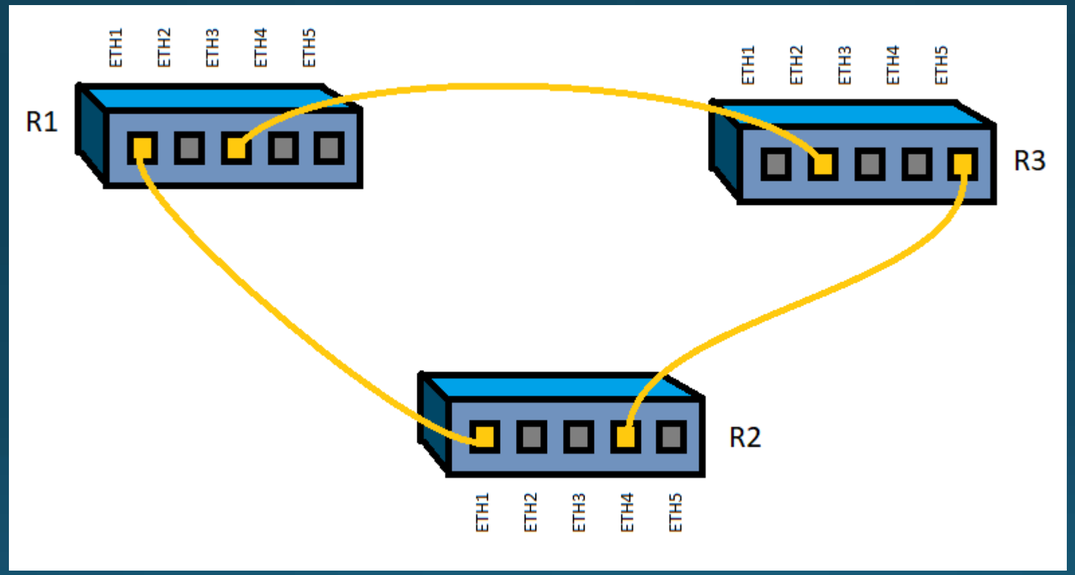
- Update-source: Allows iBGP sessions to use any operational interface for TCP connections.
- Multihop is for when a BGP message needs to go through an intermediate router and not a border router.
- bgp router-id: is not an IP address but is an ID in quad dot notation A.B.C.D, can be anything so long as it is unique
- network: the address of the internal network for each router
- neighbor x.x.x.x remote-as # (the addresses you identified in your network topology)
  - http://www.nongnu.org/quagga/docs/quagga.html#Defining-Peer
  - Example: neighbor 10.0.0.1 remote-as 2 means the router in AS-1 is trying to peer with AS-2 at 10.0.0.1.
- neighbor *peer* next-hop-self [all] BGP: no neighbor *peer* next-hop-self [all]
  - This command specifies an announced route's nexthop as being equivalent to the address of the bgp router if it is learned via eBGP. If the optional keyword all is specified the modification is done also for routes learned via iBGP.
- **debug bgp as4 tells the software to use 32-bit AS identifiers instead of 16-bit. Don't change it.**
- Some of these commands may or may not be necessary. Try to reason out whether they are needed in Part 3 by reading through the documentation linked in the Project Description.
- Leave all router passwords unchanged.

Zebra Conf. Files

- Mininet will add the links in the same order every time. Add your links in ascending router order (R1, R2…) in your zebra conf files to make sure these line up with what's displayed in the links output.
- Define interfaces/assign IP addresses to the ports.


AS 1
h1-1
21.0.1.254/24
h1-2
21.0.2.254/24
R1 - "9.0.1.1"
21.0.0.0/8
eth 3 (to AS2)
9.0.0.1/24
eth 4: (to AS3)
9.0.1.1/24
eth 1: (h1-1)
21.0.1.1/24
eth 2 (to h1-2)
21.0.2.1/24
Unique IP addresses
9.0.0.1/24
9.0.0.2/24
Unique subnet,
first 24 bits match
R2 - "9.0.2.1"
22.0.0.0/8
eth 3 : (to AS1)
9.0.0.2/24
eth 4 (to AS3)
9.0.2.1/24
eth 1 : (h2-1)
22.0.1.1/24
eth 2 : (to h2-2)
22.0.2.1/24
AS 2
h2-1
22.0.1.254/24
h2-2
22.0.2.254/24
AS 3
h3-1
23.0.1.254/24
h3-2
23.0.2.254/24
R3 - "9.0.3.1"
23.0.0.0/8
eth 3 : (to AS1)
9.0.1.2/24
eth 4 : (to AS2)
9.0.2.2/24
eth 1 : (h3-1)
23.0.1.1/24
eth 2 (to h3-2)
23.0.2.1/24
This diagram is only to show detail desired, use the demo Part 2 code as a template for your Part 3.
Attack AS not shown here, DO show the attack AS in your diagram.
Number of hosts per AS may vary from this diagram, refer to Project Description.

Visual of Network Connections Between Routers



Note: This is just a demonstration of router connections, not intended to match the previous slide.

Notes on Previous 2 Slides

- The previous slide (slide 8) on the walkthrough is an unrelated diagram. The intent of this diagram was to help students visualize how the router interfaces (eth1, eth2, etc.) represent physical locations where you can plug in a cable. This diagram doesn't have anything to do with the other diagram on slide 7.
- So let's try to revisit what the diagram originally on slide 7 from the walkthrough was intended to do. There are a few key ideas about how IP addresses, subnets, and routers work that are critical to understand in order to get Part 3 right in the project.

- Only one "cable" or connection per router interface. Again, eth1 means ethernet interface 1 – you can only plug one cable into it.
- Each assigned IP address needs to be unique. Notice that no two hosts/interfaces have the same IP address.
- Each interface of a router needs to be on a different subnet. For instance, look at R1. There's 5 interfaces in use, and each one of them is on a different subnet. You can tell because they all use a /24 subnet mask, which is 3 octets, and each of the first three octets are different.
- For two devices to talk to each other, they must be on the same subnet. Notice that for each link, both IP addresses on either side of the link belong to the same subnet. Understand the difference between /8 and /24
- Each router is advertising a /8 prefix. This is something for BGP. The subnets used between the routers aren't getting advertised this way, which is fine. By advertising the prefix, the other routers will know where to send that traffic. Notice that the hosts may be on smaller subnets (/24), but their prefixes match the /8 prefix advertised by their router.
- It really doesn't matter what IP address the host has and what IP address its default gateway (the connecting interface on the router) have. Here, it's .254 for the host, and .1 for the gateway. It could be another value. Main thing here is that it's consistently configured for both, and don't use .0 or .255.

Final Tips

- As long as your network responds to the commands in Part 2, you will be fine. So if ping all doesn't work or it hangs, that's fine. It won't be part of the grading.
  - Some bugs make Mininet so angry that you need to shut down the whole environment and restart it
- Cisco has forums with lots of great information.
- Note, some subnets are reserved, so check that before picking a subnet. Specifically, I think 1.0.0.x/24 is a bad idea.
- Router ID is independent from the IP address of the router (hint: visit that link). It can be any arbitrary address, but usually will get set to one of the IPs of the router for easier identification.

Final Tips - Debugging

- Carefully go through all of bgp config files and understand exactly what each line means. Most will need modification. Go through all files (including the website shell script, for example). Most will need modification.
- For example: Where AS 5 isn’t showing up:
  - Double check that your IP's for each router's connecting ethernet link are on the same subnet
  - Double check that your IP’s for each router’s connecting ethernet link are not on a conflicting subnet on the network.
- Ensure that your zebra files correspond to their peers’ bgp files (don’t mix up the src and dst ip on the bgp neighbor line).
- Make sure you are assigning valid IP's and gateways in bgp.py for AS 1-5. Check that you have R6 as the rogue.

mininet useful commands

- Just used few commands pretty helpful thought to share if it can help
- `mininet> R5 route` (this will give routing table on R5 router, can use for any router just by replacing R5)
- `mininet>R5 ifconfig -a` (will give all interface config of router R5)
- `mininet> R5 ping 13.0.1.254` (will check ping test, since pingall is not reliable for this project, this command can be very handy)
- To test host to host connectivity, for example, use these commands:

  `xterm h1-1` – this will pop up a new xterm window "logged into" `h1-1`

  `ping 13.0.1.254` -- this pings from `h1-1` to `h3-1`

What to turn in, what to share, rubric

- **What to Turn In**
  - For this project you need to turn in the Part3.zip file you created in Part 3. Also, if you chose to pursue the extra credit, also turn in Part4.zip file and Countermeasure.pdf files you created in Part 4. (see description for proper formatting of files)
- **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 or Part 4. You may quote snippets of the unmodified skeleton code provided to you when discussing the this Project.
- 10 points for turning in all the correct files with the correct names, and significant effort has been made towards completing the project
- 140 points for accurately recreating the topology, links, router configuration, and attack per the instructions. Partial credit is available for this rubric item, see the rubric in the Project Description
- 50 points Extra Credit -- For correctly designing and implementing a countermeasure to the attack from Part 3. Submissions MUST include both the code and documentation - extra credit will not be considered for code without accompanying documentation. Some partial credit may be provided for thorough Countermeasures.pdf identifying a viable solution without accompanying code or with non-working code if the documentation acknowledges the lack of code or the failing code

Extra Credit : Tips and suggestions

- FINISH PART 3, FIRST !! Then duplicate the entire project and place in new folder and start to work the extra credit from there so that you don't ruin your working part 3
- remember sometimes trusted people are the rogue BGP (keep this in mind when working on solution)
- In write-up try to explain concisely on the methods you are proposing show knowledge of the countermeasure you are proposing
- We are looking for a dynamic way to prevent hijacking not hardcoded ways

Helpful links related to Extra credit

- These links illustrate the problem with some events from the past
- https://www.cnet.com/news/how-pakistan-knocked-youtube-offline-and-how-to-make-sure-it-never-happens-again/
- https://www.wired.com/2008/02/pakistans-accid/
- https://www.networkworld.com/article/2272520/six-worst-internet-routing-attacks.html
- https://en.wikipedia.org/wiki/BGP_hijacking

Have fun with this

- Good luck !
- Always check PIAZZA for you questions
- Please do not publicly discuss solutions for the extra credit!