CS6250 Computer Networks

Link to Presentation: https://youtu.be/Ptq02wuj1xI

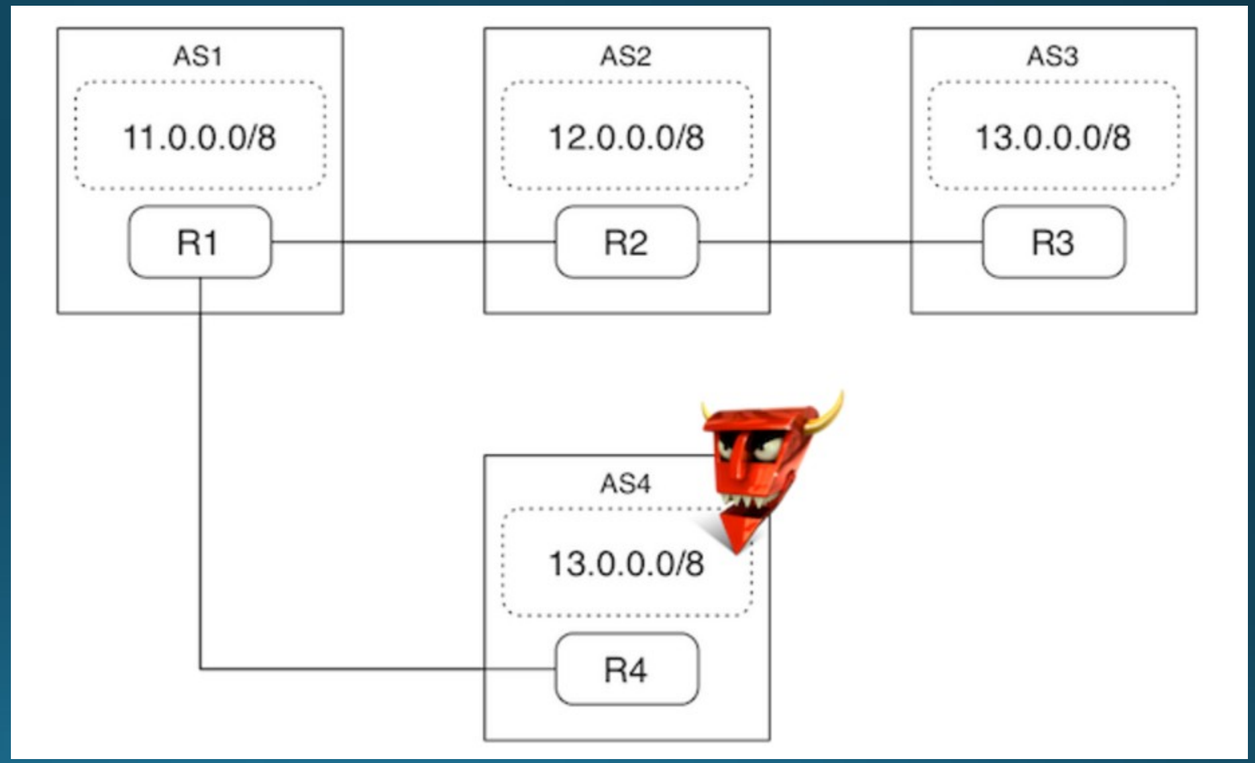
BGP Hijacking Project

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.
- 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



Files in the Demo

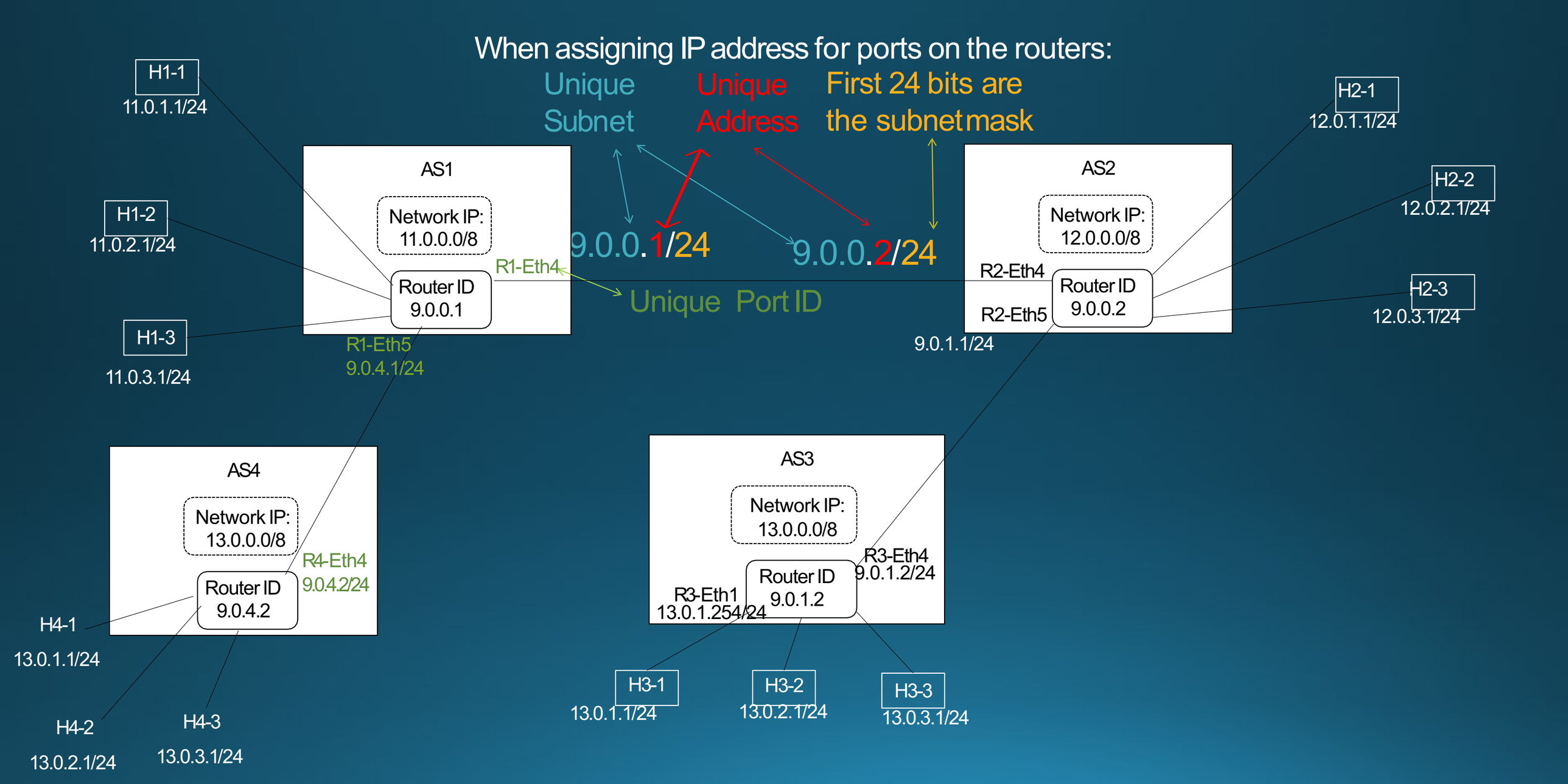
- bgp.py
  - Main file, defines the topology
- connect.sh
  - Connects the routers' bgpd shells
- run.py
  - Runs the simulation
- start_rogue.sh
  - Starts the rogue AS, begins the BGP hijacking
- stop_rogue.sh
  - Stops the rogue AS, stops the BGP hijacking
- webserver.py
- website.sh
  - Initiates 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

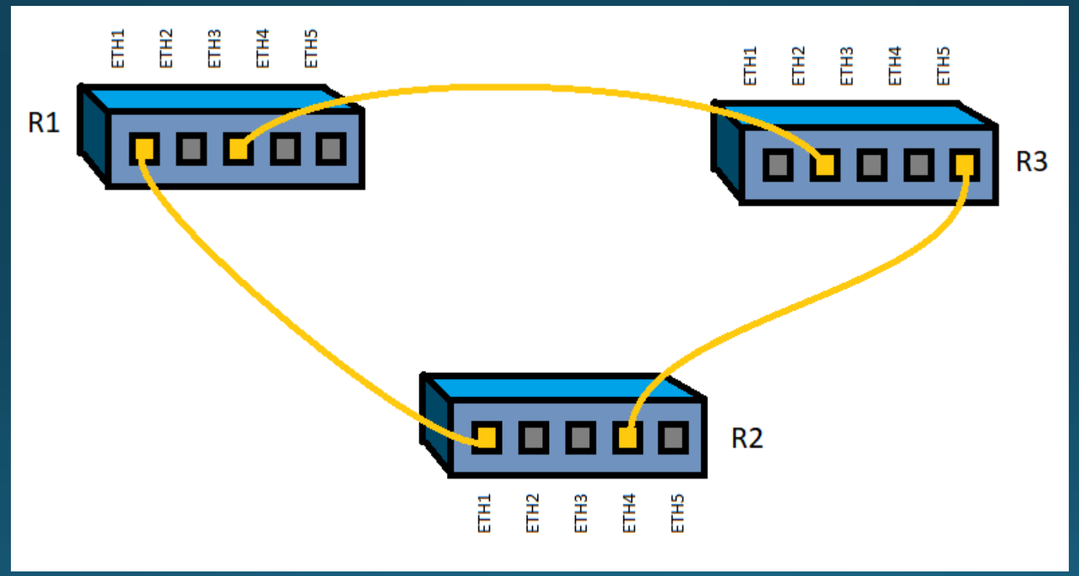
- Update-source: Allows iBGP sessions to use any operational interface for TCP connections.
- Multihop is for when a BGP message needs to go through an intermediate router and not a border router.
- bgp router-id: is not an IP address but is an ID in quad dot notation A.B.C.D, can be anything so long 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.
- Don't change the password for bgpd-R3

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.


When assigning IP address for ports on the routers:
Unique Subnet
Unique Address
First 24 bits are the subnet mask
H1-1
11.0.1.1/24
H1-2
11.0.2.1/24
H1-3
11.0.3.1/24
AS1
Network IP:
11.0.0.0/8
Router ID
9.0.0.1
R1-Eth4
9.0.0.1/24
R1-Eth5
9.0.4.1/24
9.0.0.2/24
Unique Port ID
AS2
Network IP:
12.0.0.0/8
R2-Eth4
R2-Eth5
Router ID
9.0.0.2
9.0.1.1/24
H2-1
12.0.1.1/24
H2-2
12.0.2.1/24
H2-3
12.0.3.1/24
AS4
Network IP:
13.0.0.0/8
R4-Eth4
9.0.4.2/24
Router ID
9.0.4.2
H4-1
13.0.1.1/24
H4-2
13.0.2.1/24
H4-3
13.0.3.1/24
AS3
Network IP:
13.0.0.0/8
R3-Eth4
9.0.1.2/24
Router ID
9.0.1.2
R3-Eth1
13.0.1.254/24
H3-1
13.0.1.1/24
H3-2
13.0.2.1/24
H3-3
13.0.3.1/24

Visual of Network Connections Between Routers



This is just for sample reference – not exact replica of any topology

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 rouge.