

Deploy a Botnet and DDoS Attack Lab

Lab Description

Distributed Denial of Service (DDoS) attacks pose a significant threat to online services, leveraging multiple compromised devices to overwhelm targeted systems with traffic, rendering them inaccessible to legitimate users. This lab will simulate a Distributed Denial of Service (DDoS) attack orchestrated by multiple bots within a network, providing users with a real-time visual representation of the network traffic.

Learning Outcomes

- Understand what a Distributed Denial of Service attack is and visualize such attack.
- Understand Botnets and how they operate.
- Understand mitigation that could be placed in order to prevent a DDoS attack.

Lab Environment

- Use the SEED VM

Start-up Procedures

- Start-up Procedures will begin in Task 1.

Lab Instructions

Task 1: Set Up

1.1 Create & Move Files

To start this lab, you are going to create a new folder called **botnet-ddosLab**. We are going to copy the contents of B00-mini-interent folder and B05-botnet folder, you should be able to find these folders within the **lab-content/examples** directory.

```
seed@VM: ~/.../examples
[03/10/24]seed@VM:~$ cd Documents/lab-content/examples/
[03/10/24]seed@VM:~/.../examples$ ls
A00-simple-peering      A21-shadow-internet      B08-Remix-Connection
A01-transit-as          B00-mini-internet        B09-Smart-Contract-Attacks
A02-transit-as-mpls     B01-dns-component        B10-dhcp
A03-real-world          B02-mini-internet-with-dns C00-hybrid-internet
A04-visualization       B03-ip-anycast           C01-hybrid-dns-component
A05-components          B04-bgp-prefix-hijacking C02-hybrid-internet-with-dns
A06-merge-emulation     B05-botnet               C03-bring-your-own-internet
A07-compilers           B06-blockchain           not-ready-examples
A20-nano-internet       B07-darknet-tor
```

Figure 1: Contents of lab-content/examples directory.

Store the contents of these two folders in the new **botnet-ddosLab**. Also be sure to include **seedemu** folder in the **botnet-ddosLab** folder. Your folder should look something like this:

```
seed@VM: ~/.../botnet-ddosLab
[03/06/24]seed@VM:~/.../botnet-ddosLab$ ll
total 148
-rw-rw-r-- 1 seed seed 122674 Feb 29 18:56 base-component.bin
-rw-rw-r-- 1 seed seed 1954 Nov 3 18:34 botnet-basic.py
-rw-rw-r-- 1 seed seed 230 Nov 3 18:34 ddos.py
-rw-rw-r-- 1 seed seed 5323 Nov 3 18:34 mini-internet.py
drwxrwxr-x 2 seed seed 4096 Feb 23 19:19 not_ready
-rw-rw-r-- 1 seed seed 2336 Nov 3 18:34 README.md
drwxrwxr-x 13 seed seed 4096 Jan 25 19:46 seedemu
```

Figure 2: Contents of bot-ddosLab folder.

Use the font Courier Prime in 12 pt bold for terminal and code commands.

```
$ ls -l
```

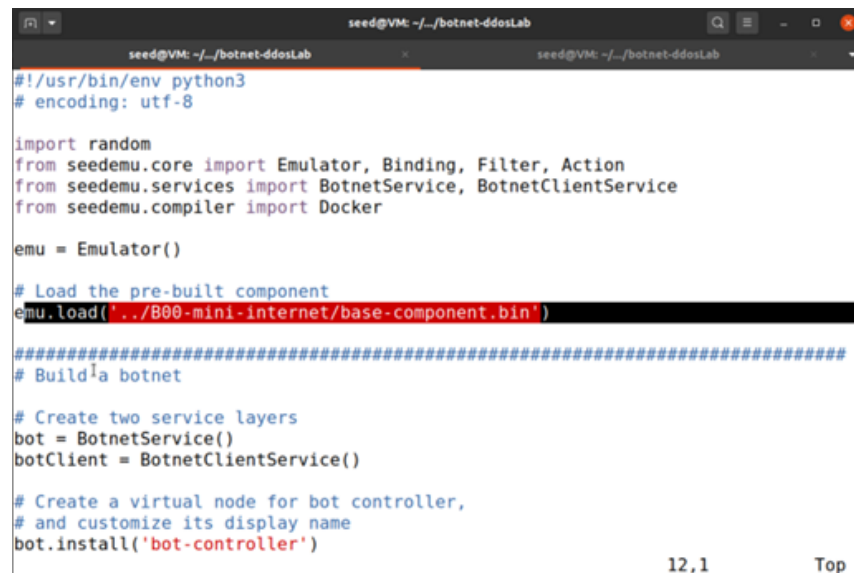
Make sure text and headers after the 1x1 table have a space before the paragraph in the paragraph settings.

1.2 Alter botnet-basic.py Script

In order for this lab to run properly, you are going to need to change part of the botnet-basic.py code. Use a text editor of your preference to edit this part of the code to the current directory of the folder in which you are in (Edit line 12).

NOTE: Be sure to include **“/base-component.bin”** at the end.

Before:



```
seed@VM: ~/botnet-ddosLab
#!/usr/bin/env python3
# encoding: utf-8

import random
from seedemu.core import Emulator, Binding, Filter, Action
from seedemu.services import BotnetService, BotnetClientService
from seedemu.compiler import Docker

emu = Emulator()

# Load the pre-built component
emu.load('../B00-mini-internet/base-component.bin')

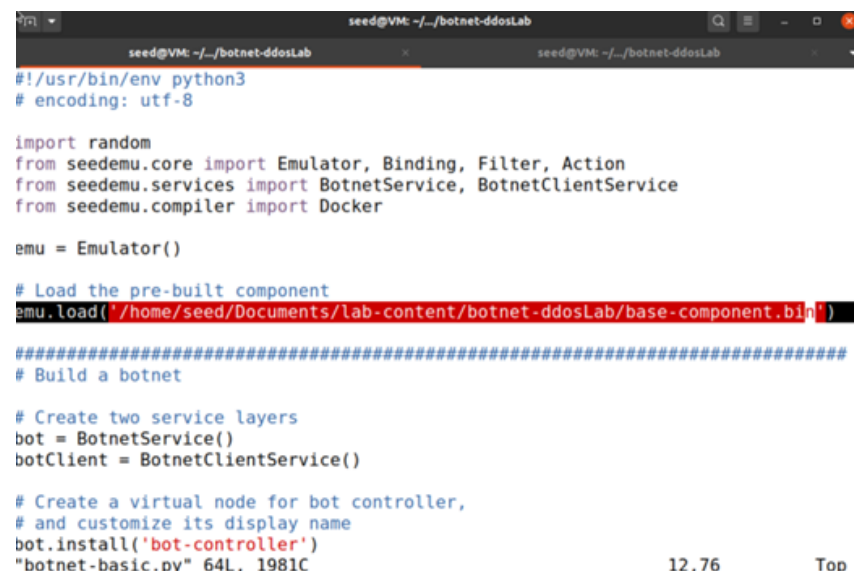
#####
# Build a botnet

# Create two service layers
bot = BotnetService()
botClient = BotnetClientService()

# Create a virtual node for bot controller,
# and customize its display name
bot.install('bot-controller')
```

12,1 Top

After:



```
seed@VM: ~/botnet-ddosLab
#!/usr/bin/env python3
# encoding: utf-8

import random
from seedemu.core import Emulator, Binding, Filter, Action
from seedemu.services import BotnetService, BotnetClientService
from seedemu.compiler import Docker

emu = Emulator()

# Load the pre-built component
emu.load('/home/seed/Documents/lab-content/botnet-ddosLab/base-component.bin')

#####
# Build a botnet

# Create two service layers
bot = BotnetService()
botClient = BotnetClientService()

# Create a virtual node for bot controller,
# and customize its display name
bot.install('bot-controller')
"botnet-basic.py" 64L, 1981C
```

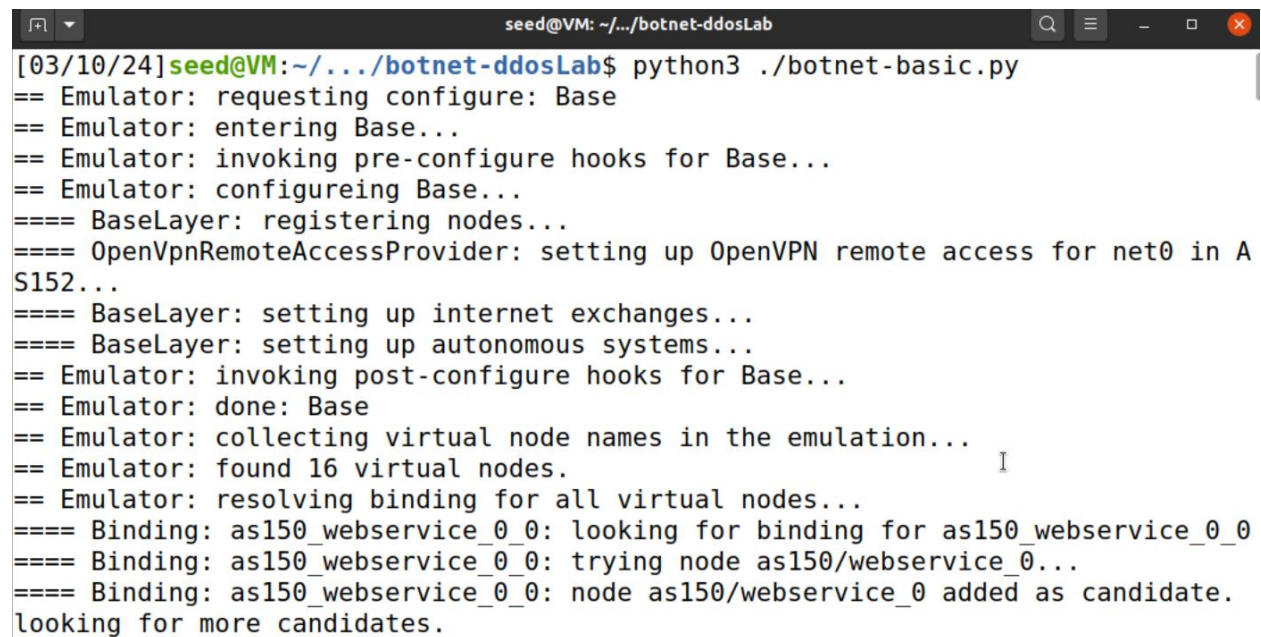
12,76 Top

1.3 Bring Up the Network

We are going to have to set up the network by running the python script.

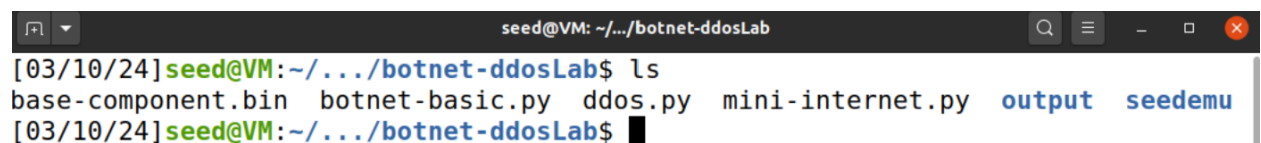
Run this command in the **/botnet-ddosLab** directory.

```
$ python3 ./botnet-basic.py
```



```
seed@VM: ~/.../botnet-ddosLab
[03/10/24]seed@VM:~/.../botnet-ddosLab$ python3 ./botnet-basic.py
== Emulator: requesting configure: Base
== Emulator: entering Base...
== Emulator: invoking pre-configure hooks for Base...
== Emulator: configureing Base...
==== BaseLayer: registering nodes...
==== OpenVpnRemoteAccessProvider: setting up OpenVPN remote access for net0 in A
S152...
==== BaseLayer: setting up internet exchanges...
==== BaseLayer: setting up autonomous systems...
== Emulator: invoking post-configure hooks for Base...
== Emulator: done: Base
== Emulator: collecting virtual node names in the emulation...
== Emulator: found 16 virtual nodes.
== Emulator: resolving binding for all virtual nodes...
==== Binding: as150_webservice_0_0: looking for binding for as150_webservice_0_0
==== Binding: as150_webservice_0_0: trying node as150/webservice_0...
==== Binding: as150_webservice_0_0: node as150/webservice_0 added as candidate.
looking for more candidates.
```

An output folder should be created after entering this command.



```
seed@VM: ~/.../botnet-ddosLab
[03/10/24]seed@VM:~/.../botnet-ddosLab$ ls
base-component.bin  botnet-basic.py  ddos.py  mini-internet.py  output  seedemu
[03/10/24]seed@VM:~/.../botnet-ddosLab$
```

Figure 3: output folder created.

Run these commands within the newly created **/output** directory.

```
$ docker-compose build
$ docker-compose up
$ yes | docker network prune
$ docker-compose up
```

1.4 Debugging Task (1.3)

There is a high chance that you may run into errors.

If unsuccessful in bringing up the network with the commands above, try these commands:

```
$ docker stop $(docker ps -aq)
$ docker rm $(docker ps -aq)
```

Then start again from task 1.3 and skip task 1.4.

1.5 Bring up Client Side of the Network

Now that the network is up in the '**output**' directory, we must now bring the network up on the client side. Go into the **/client** directory.

Run these commands within the **botnet-ddosLab** directory.

```
$ docker-compose build
$ docker-compose down
$ docker-compose up
```

```

Successfully built 6d16e6dca4
Successfully tagged client_seedemu-client:latest
[03/07/24]seed@VM:~/.../client$ docker-compose down
Removing network client_default
WARNING: Network client_default not found.
[03/07/24]seed@VM:~/.../client$ docker-compose up
Creating network "client_default" with the default driver
Creating seedemu_client ... done
Attaching to seedemu_client

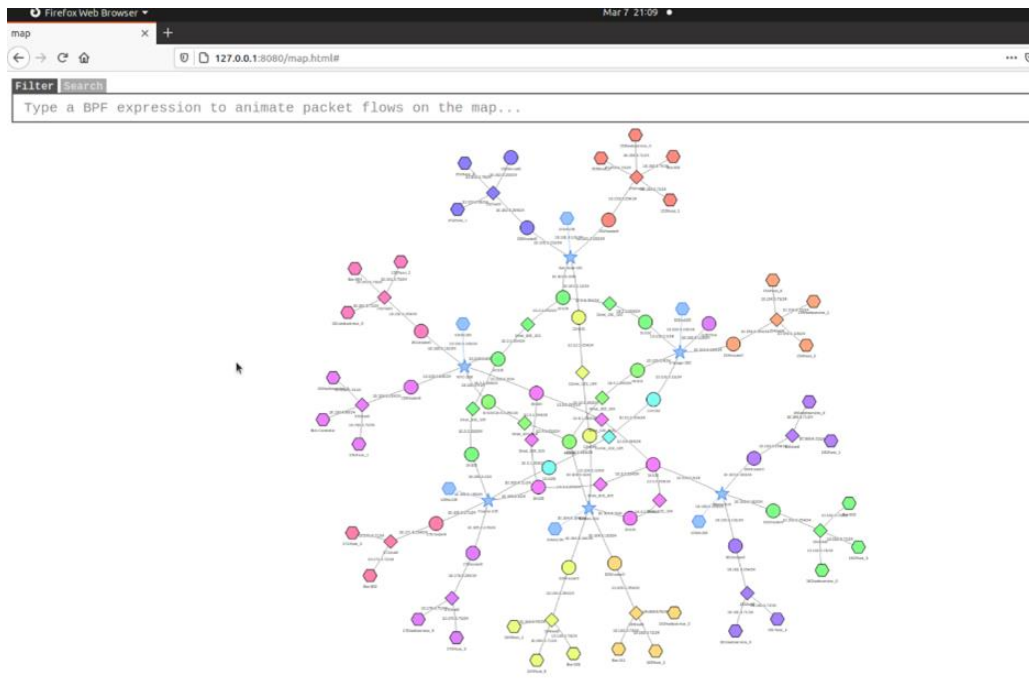
```

Figure 4: Client successfully launched.

1.6 Opening up the Visual Map

Open up Firefox web browser and go to this link:

- <http://127.0.0.1:8080/map.html>



Task 2 – Locating Bots and Commencing DDoS Attack

2.1 Locate Bots

There should be 6 bots located on this map, locate all 6 of them, take a screenshot and place them here (Hint: use the search bar to locate them):

Filter

Search

bot

bot

Press enter to show all matches on the map...

Host: Bot-005

10.164.0.73/24

Host: Bot-003

10.171.0.72/24

Host: Bot-004

10.151.0.73/24

Host: Bot-002

10.162.0.73/24

Host: Bot-000

10.153.0.74/24

Host: Bot-001

10.163.0.73/24



Find the IP address of each bot and include a screenshot of each individual location:

Bot-00:

Bot-01:

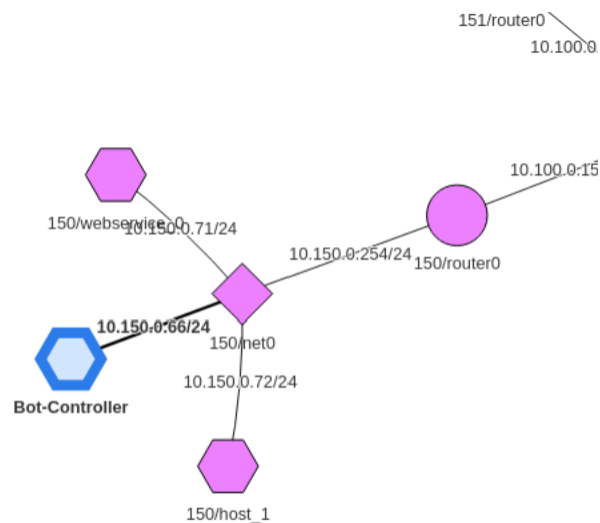
Bot-02:

Bot-03:

Bot-04:

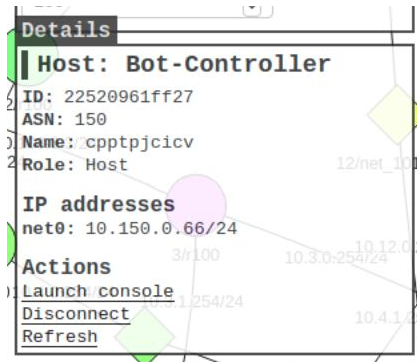
Bot-05:

Example:



2.2 Launch Console Controller

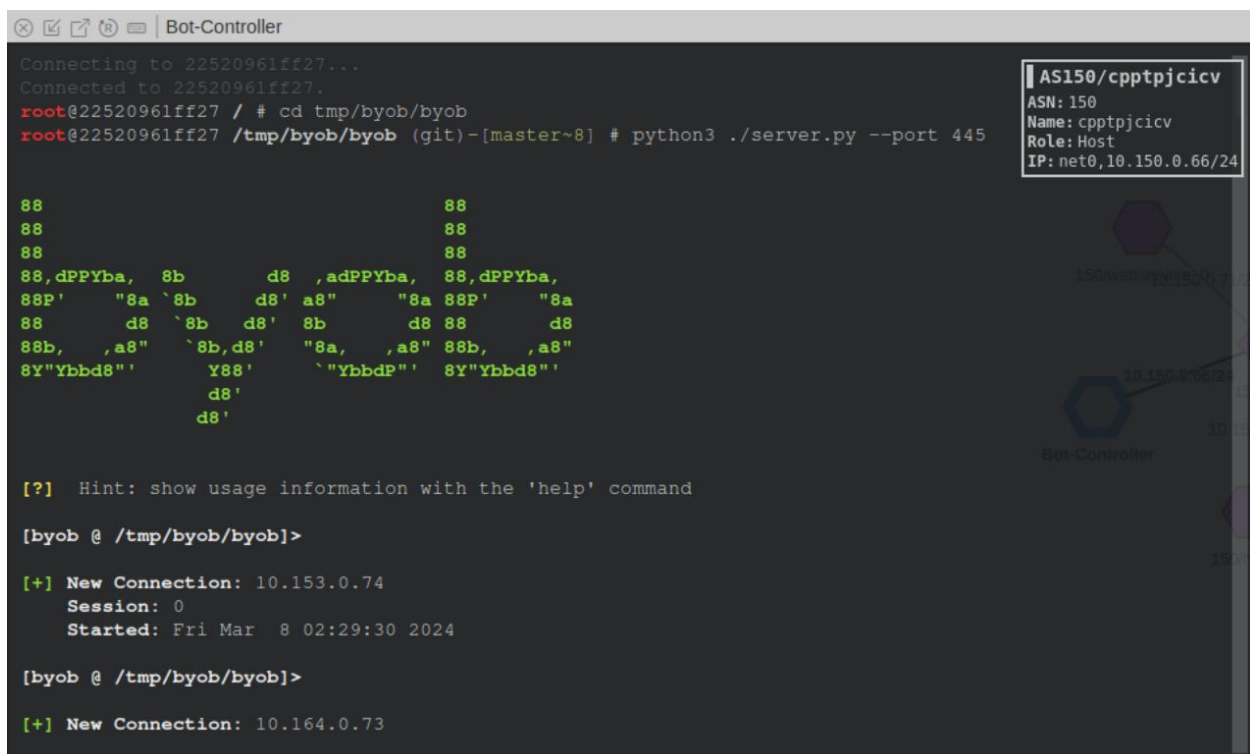
Click on the Bot-controller that is located on the network diagram and you should be able to see the details of that specific host on the right of the web page. Under 'Actions' you should be able to see 3 options, click on '**Launch Console**'.



2.3 Start the byob Server

To start the byob server, you must:

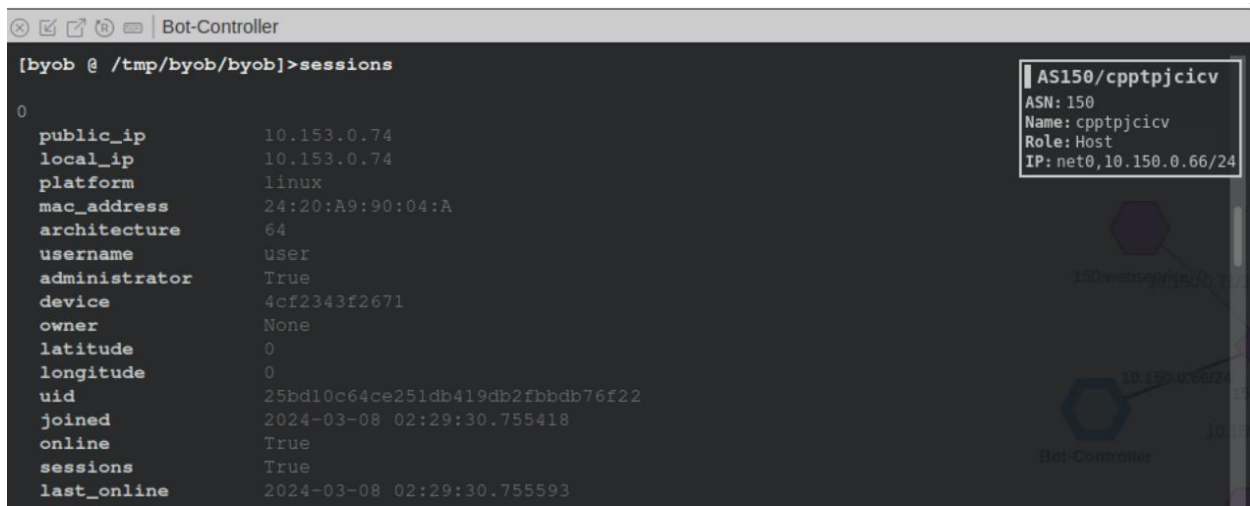
- cd into the **temp/byob/byob** folder
- run the server.py script on port 445 (hint: **python3 ./server.py --port 445**)



Enter Screenshot of your own byob server launching:

2.4 Wait for the all the Bot Nodes to Connect

There should be 6 bot nodes (takes about 15-20 seconds). From this we can type in 'sessions' in the console and gather information about the bot nodes.



```
[byob @ /tmp/byob/byob]>sessions
0
public_ip      10.153.0.74
local_ip      10.153.0.74
platform      linux
mac_address    24:20:A9:90:04:A
architecture   64
username      user
administrator  True
device        4cf2343f2671
owner         None
latitude       0
longitude      0
uid           25bd10c64ce251db419db2fbbdb76f22
joined        2024-03-08 02:29:30.755418
online        True
sessions      True
last_online    2024-03-08 02:29:30.755593
```

AS150/cpptpjicv
ASN: 150
Name: cpptpjicv
Role: Host
IP: net0,10.150.0.66/24

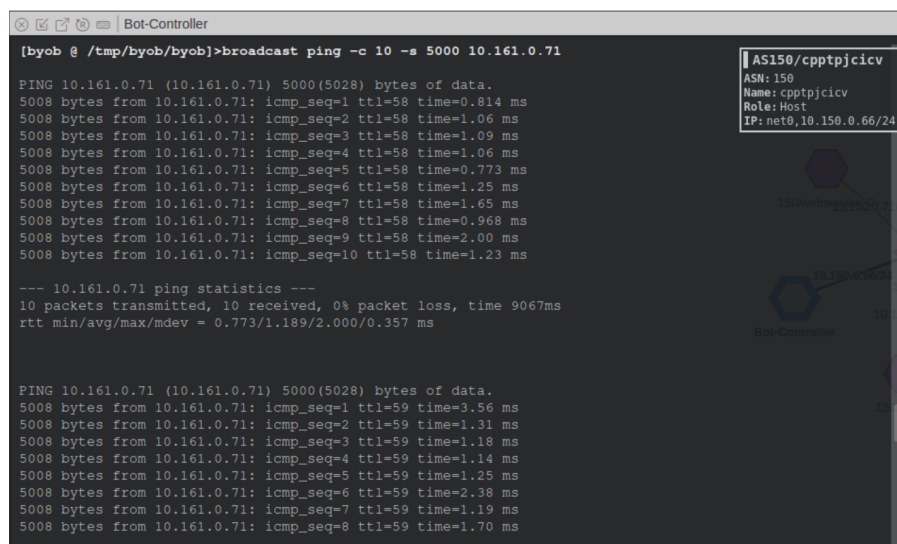
Place screenshot here of all 6 bot nodes sessions:

2.5 Launch DDoS Attack

Now it's time to launch the attack.

To launch the DDoS, use this command:

```
> broadcast ping -c 10 -s 5000 10.161.0.71
```



```
[byob @ /tmp/byob/byob]>broadcast ping -c 10 -s 5000 10.161.0.71

PING 10.161.0.71 (10.161.0.71) 5000(5028) bytes of data.
5008 bytes from 10.161.0.71: icmp_seq=1 ttl=58 time=0.814 ms
5008 bytes from 10.161.0.71: icmp_seq=2 ttl=58 time=1.06 ms
5008 bytes from 10.161.0.71: icmp_seq=3 ttl=58 time=1.09 ms
5008 bytes from 10.161.0.71: icmp_seq=4 ttl=58 time=1.06 ms
5008 bytes from 10.161.0.71: icmp_seq=5 ttl=58 time=0.773 ms
5008 bytes from 10.161.0.71: icmp_seq=6 ttl=58 time=1.25 ms
5008 bytes from 10.161.0.71: icmp_seq=7 ttl=58 time=1.65 ms
5008 bytes from 10.161.0.71: icmp_seq=8 ttl=58 time=0.968 ms
5008 bytes from 10.161.0.71: icmp_seq=9 ttl=58 time=2.00 ms
5008 bytes from 10.161.0.71: icmp_seq=10 ttl=58 time=1.23 ms

--- 10.161.0.71 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9067ms
rtt min/avg/max/mdev = 0.773/1.189/2.000/0.357 ms

PING 10.161.0.71 (10.161.0.71) 5000(5028) bytes of data.
5008 bytes from 10.161.0.71: icmp_seq=1 ttl=59 time=3.56 ms
5008 bytes from 10.161.0.71: icmp_seq=2 ttl=59 time=1.31 ms
5008 bytes from 10.161.0.71: icmp_seq=3 ttl=59 time=1.18 ms
5008 bytes from 10.161.0.71: icmp_seq=4 ttl=59 time=1.14 ms
5008 bytes from 10.161.0.71: icmp_seq=5 ttl=59 time=1.25 ms
5008 bytes from 10.161.0.71: icmp_seq=6 ttl=59 time=2.38 ms
5008 bytes from 10.161.0.71: icmp_seq=7 ttl=59 time=1.19 ms
5008 bytes from 10.161.0.71: icmp_seq=8 ttl=59 time=1.70 ms
```

AS150/cpptpjicv
ASN: 150
Name: cpptpjicv
Role: Host
IP: net0,10.150.0.66/24

In your own words, explain each step of this command here:

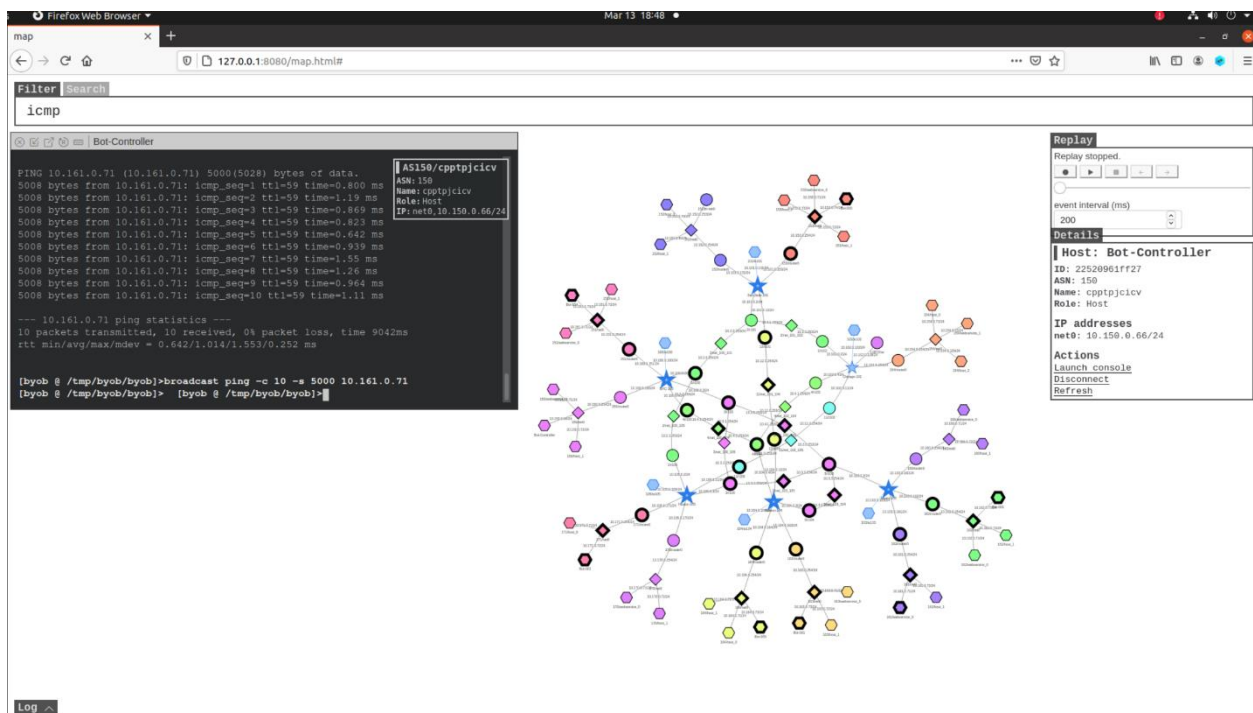
-Broadcast ping:

-c 10:

-s 5000 10.161.0.71:

2.6 Turn on ICMP Traffic

In order to visualize the attack, go back into the filter portion of the search bar and input ICMP. This will turn on ICMP traffic so it can give you a better visual of the pings happening in real time. Repeat step 2.6 to visualize traffic.



Submit your own screenshot of DDoS Visual:

2.6 Mitigations

Traffic Monitoring and Analysis: Continuously monitor network traffic for anomalies and suspicious patterns. Implement intrusion detection systems (IDS) and intrusion prevention systems (IPS) to automatically detect and block malicious traffic.

CAPTCHA and Challenge-Response Mechanisms: Implement CAPTCHA challenges or other challenge-response mechanisms to differentiate between human users and bots. This can help filter out malicious bot traffic while allowing legitimate users to access the service.

Rate Limiting and Traffic Shaping: Implement rate limiting and traffic shaping mechanisms to throttle or prioritize traffic based on predefined rules. This can help mitigate the impact of the DDoS attack by limiting the rate of incoming requests from suspicious sources.

Conclusion

The deployment of a botnet and the execution of a DDoS attack in this lab have provided valuable insights into the complexities of cyber threats. Participants have learned to control a botnet and visualize the impact of a DDoS attack in real-time. This practical experience emphasizes the importance of understanding cyber threats to develop effective security strategies. As cyber threats continue to evolve, ongoing education and hands-on training remain essential in preparing for and mitigating future attacks.