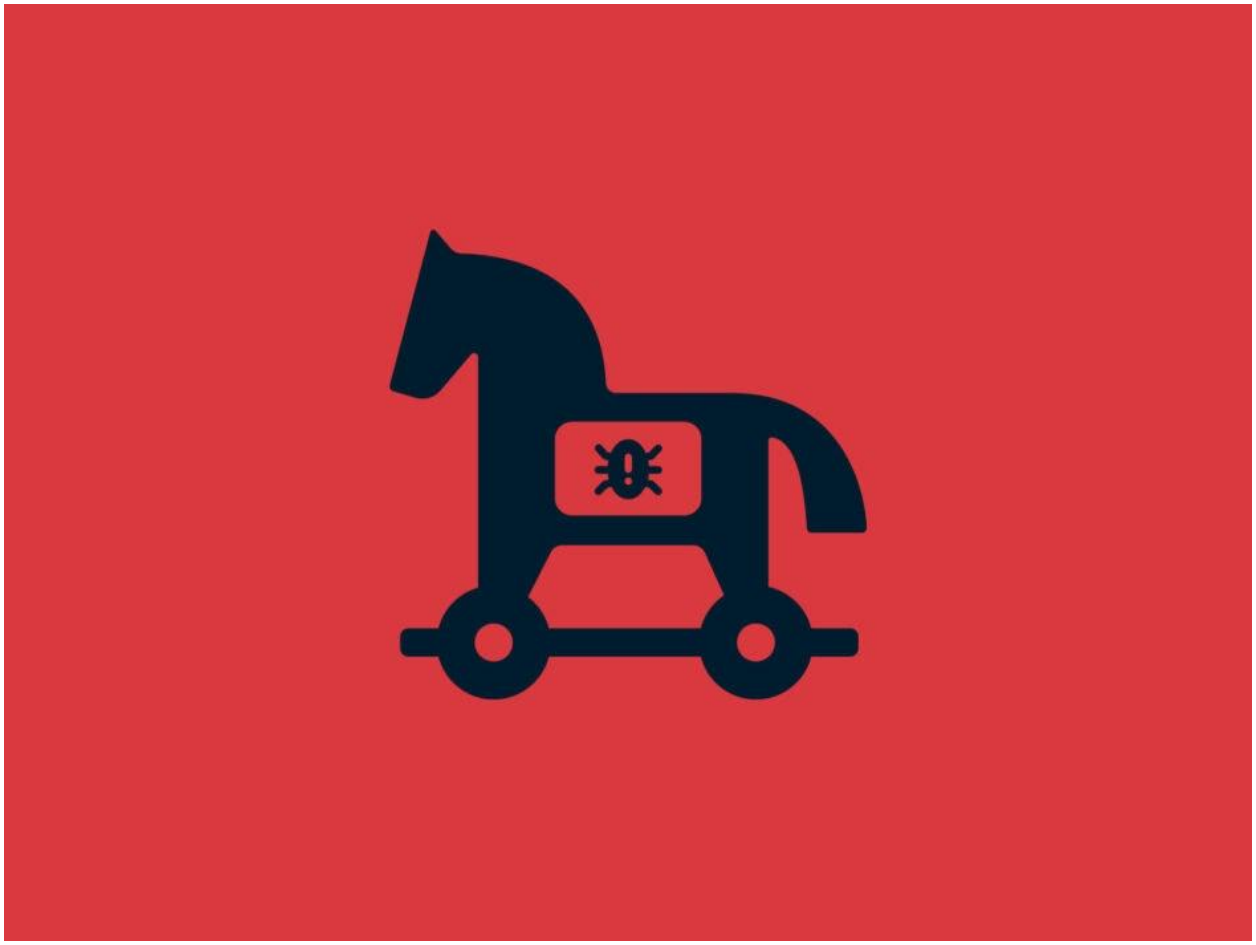


Md. Shadmim Hasan Sifat
1705021

Malware Design : Morris Worm

CSE 406 : Computer Security Sessional

Assignment 2

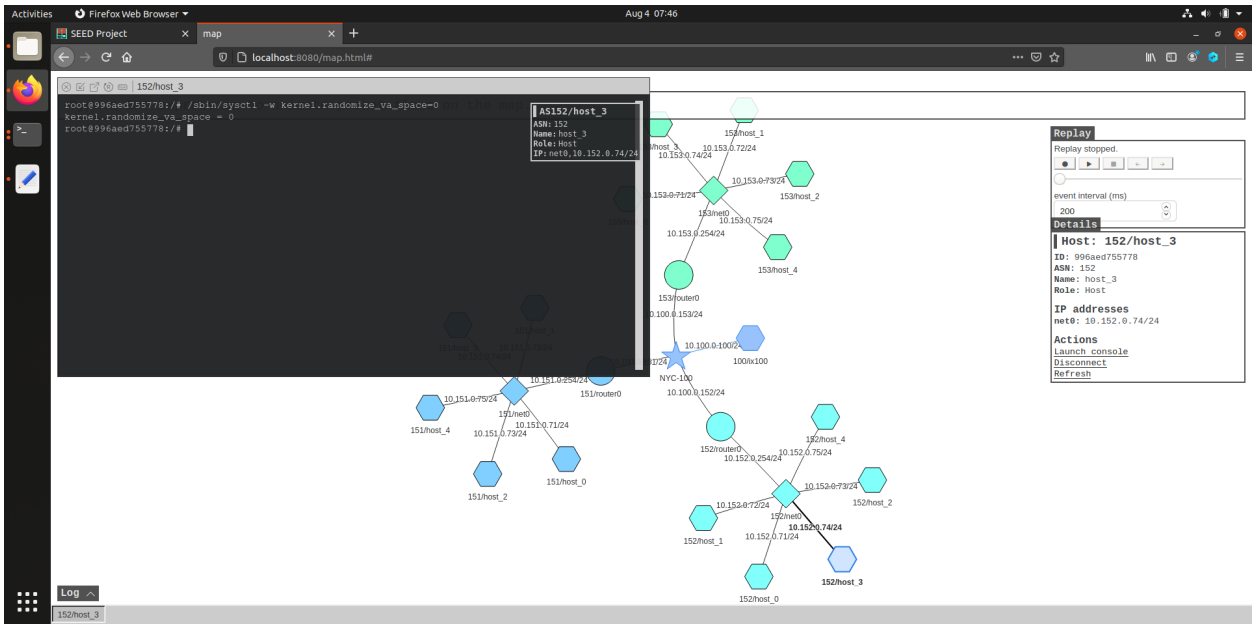
CSE,BUET

Task 1: Attack Any Target Machine

Assignment Task: Please modify the provided skeleton code in worm.py to launch the attack against any server.

In this first task, we focused on the attacking part of the worm. The Morris worm exploited several vulnerabilities to gain entry to targeted systems, including a buffer-overflow vulnerability.So we exploited that buffer-overflow vulnerability here.

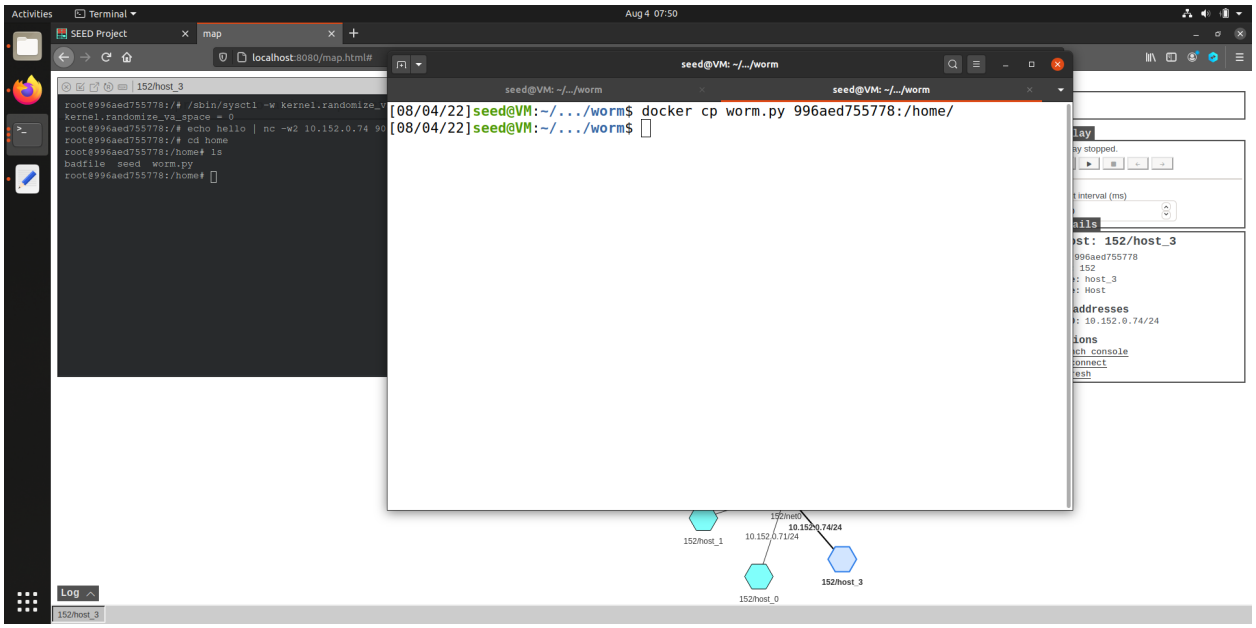
A vulnerable server was installed on all the containers, and they all have a buffer-overflow vulnerability.The goal of this task was to exploit this vulnerability, so that we could run our malicious code on the server.



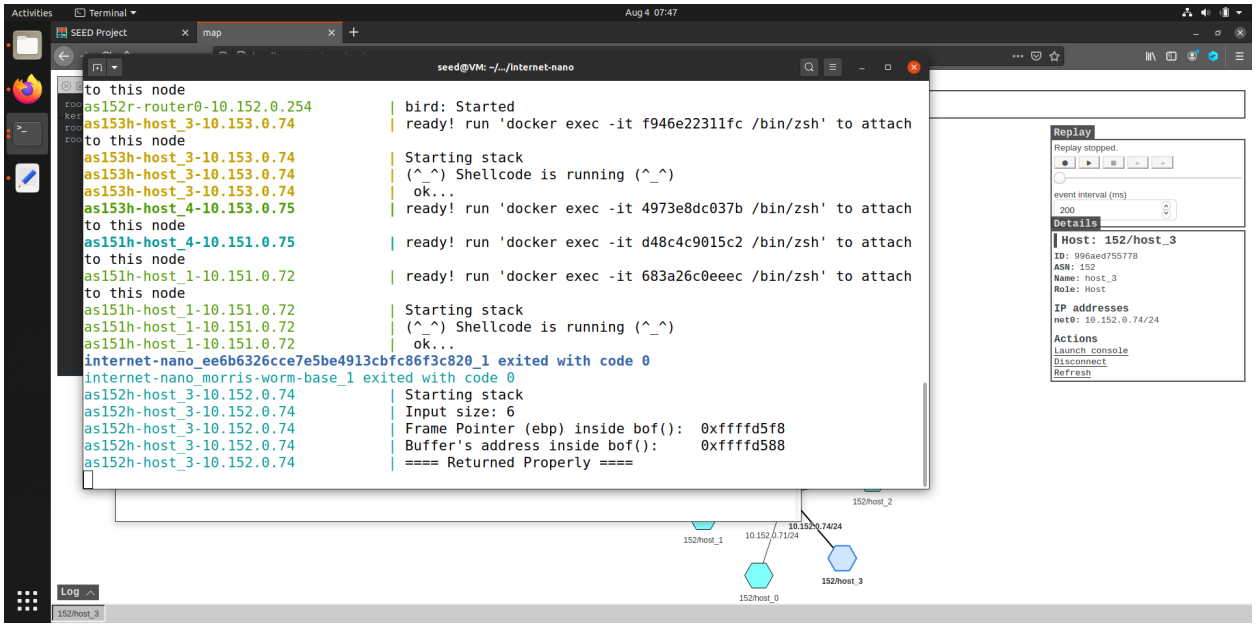
Firstly we needed to turn off the address randomization (shown in console above).This kernel parameter is global, so once we turned it off from the host machine, all the containers were affected.

All the non-router containers in the emulator run the same vulnerable server. With the address randomization disabled, all the servers will have the identical parameters, the addresses of the buffer and the value of the frame pointers will be the same across all the containers which makes our attack easier.

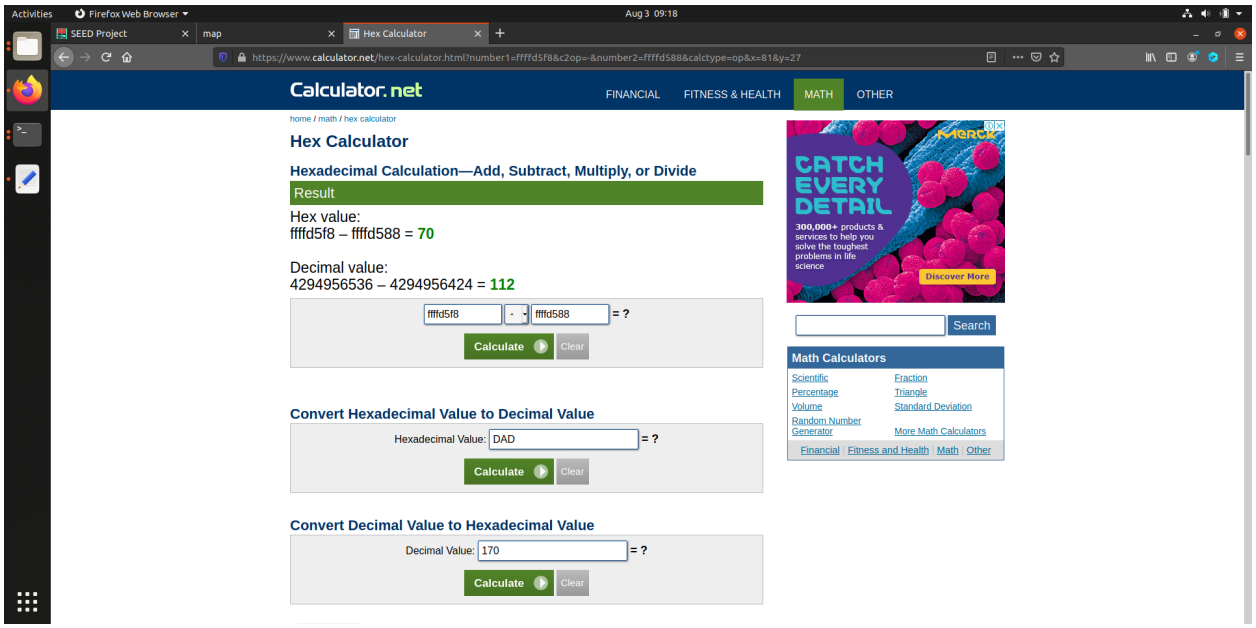
A skeleton code was provided in the Labsetup/worm folder.In this task, we focused on completing the createBadfile() function, which was to generate the malicious payload for the buffer-overflow attack. We launched this attack against our first target. We chose 10.152.0.74 host as our first target. In the code, we had hard-coded this target.



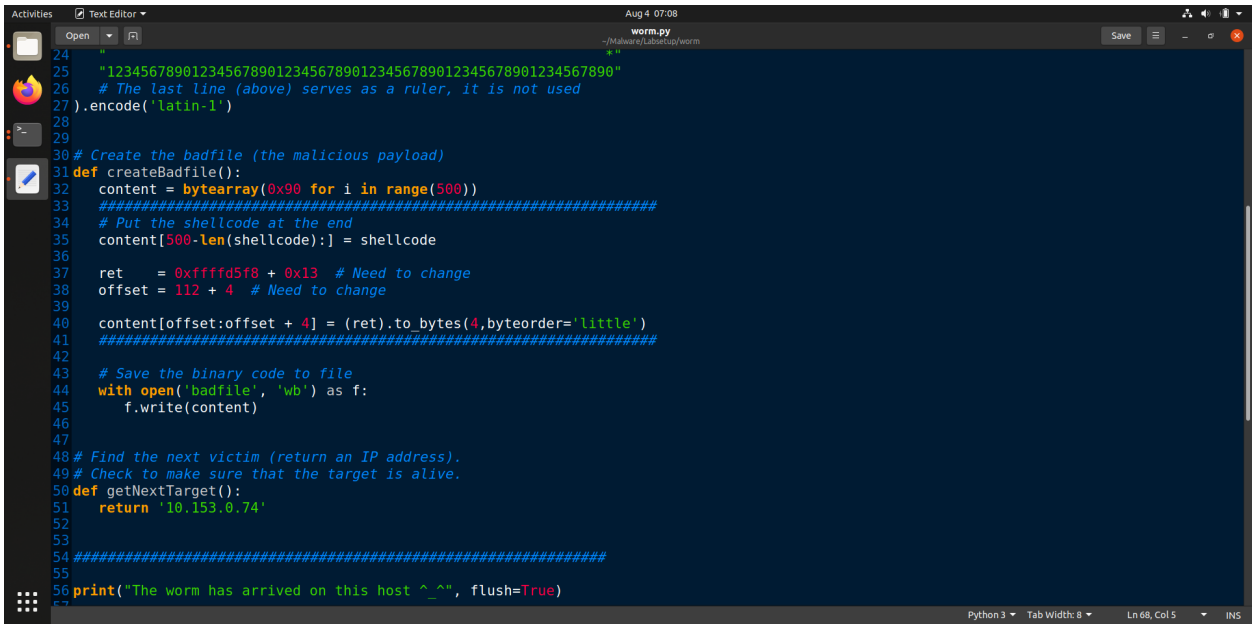
Secondly we sent a benign message (echo hello | nc -w2 10.152.0.74 9090) to our server. We saw the following messages printed out by the server container .



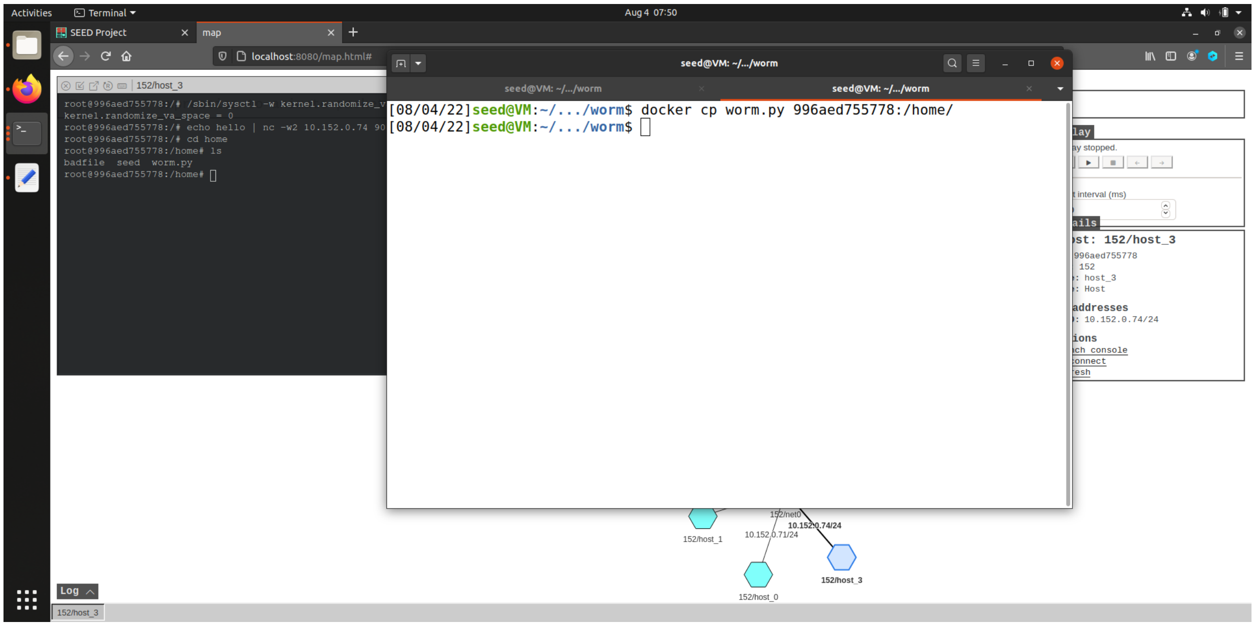
We saw the server printed out some of its internal parameters like Frame pointer (ebp) inside bof() and Buffer address inside bof().



Thirdly, We measured the offset (by taking difference) in decimal which was 112.

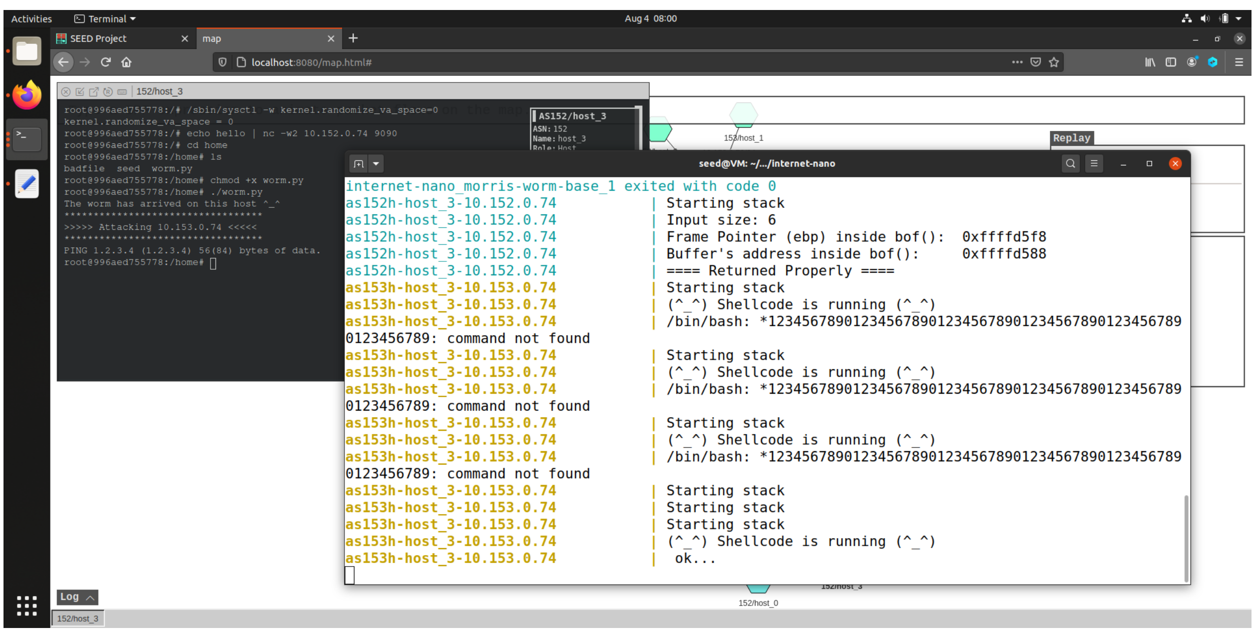


Fourthly, we used these internal parameters in constructing our attacks. In particular, we needed to modify createBadfile() function in Lines 37 where we used ebp value which needed extra offset to fill the required gap and thus execute properly and in Line 38 where we put our measured offset (differnce between the bof function's return address position and start of the buffer).

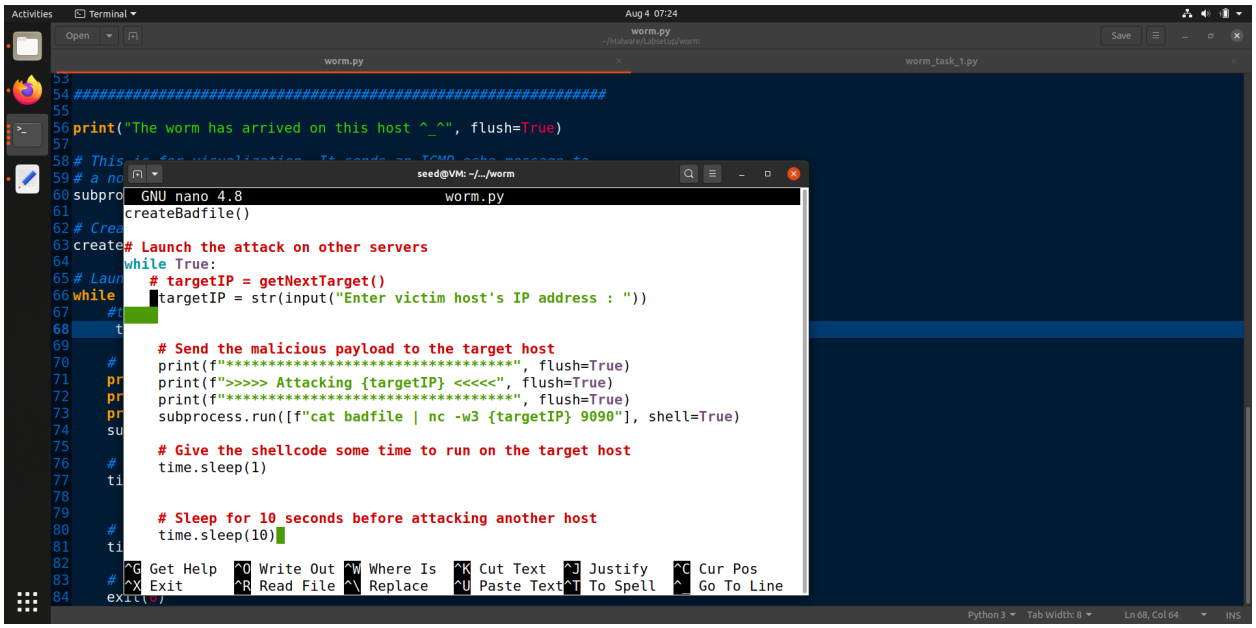


Fifthly, we copied our modified worm.py into our attacker machine (container).

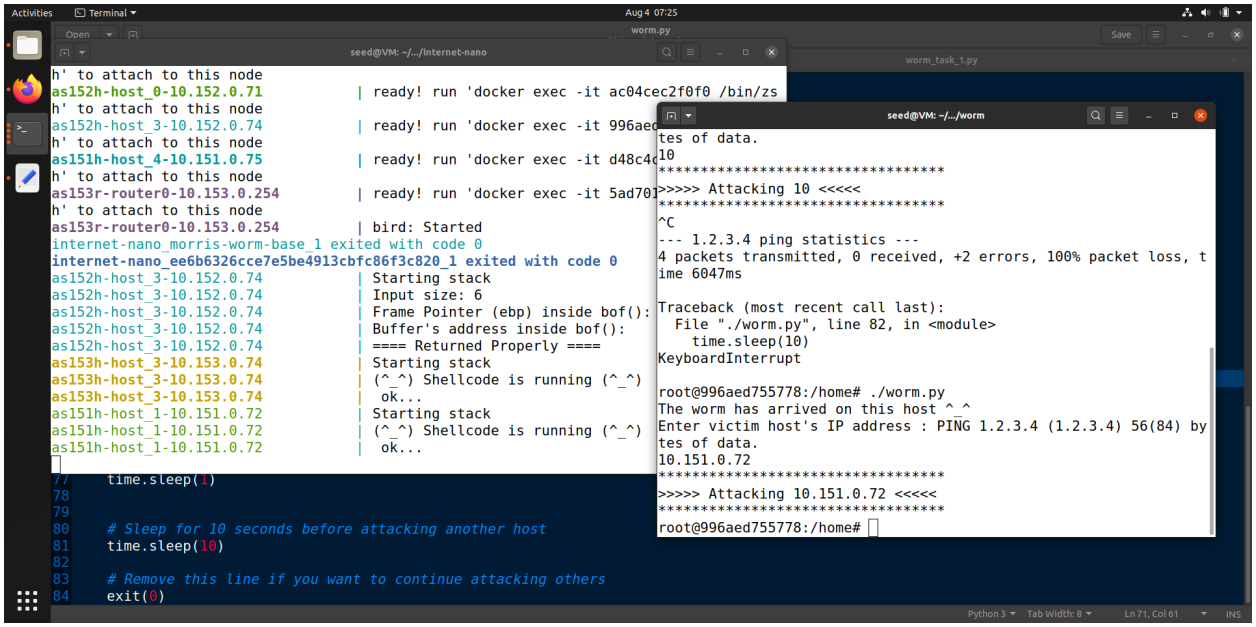
To test the attack, we then simply run the attack program worm.py. It generated the badfile, and then sent its malicious content to the target machine (victim container).



As we saw a smiley face printed out on the target machine (the message shown in the terminal), that meant that **eventually**, we successfully made our attack and our injected malicious code had been executed inside victim machine.



As we are told to modify the provided skeleton code in worm.py to launch the attack against any server. So we modified it like above (In Line 67) taking targeted victim's IP (here we used 10.151.0.72 as victim) and then made the attack.

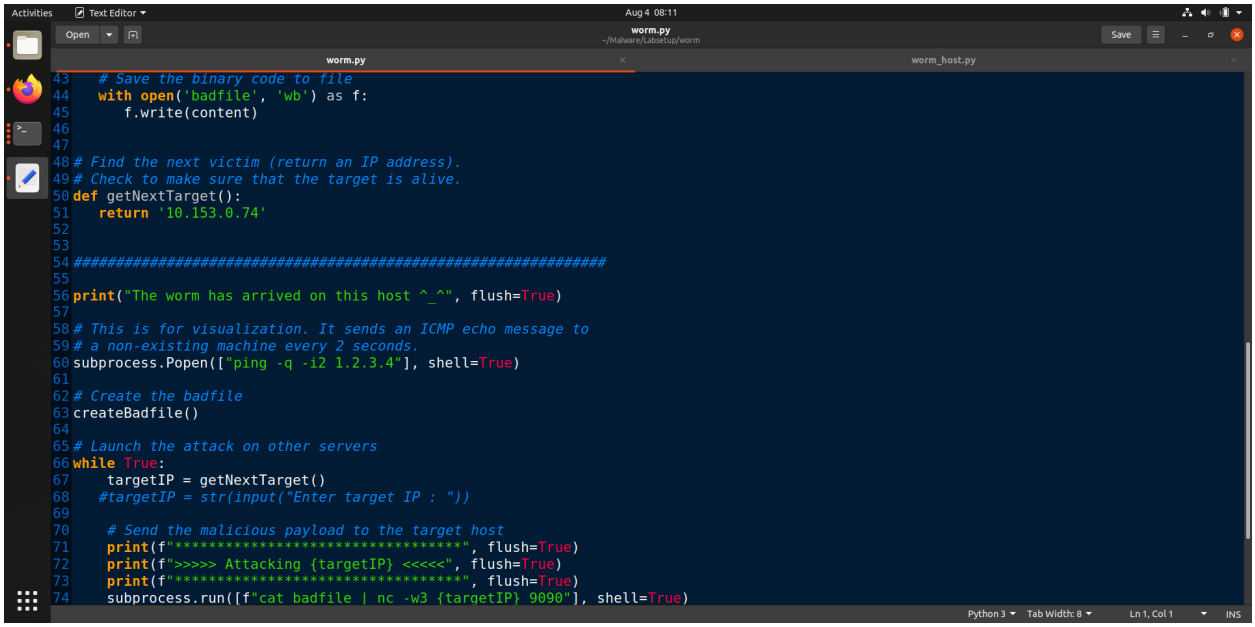


Thus our targeted victim 10.151.0.72 got attacked successfully.

Task 2: Self Duplication

Assignment Task: Add the self-duplication feature to your worm, so when the buffer-overflow attack is successful, a copy of the worm, i.e., worm.py is copied to the victim machine. You can get a shell on the victim container, and check whether a copy is created there or not.

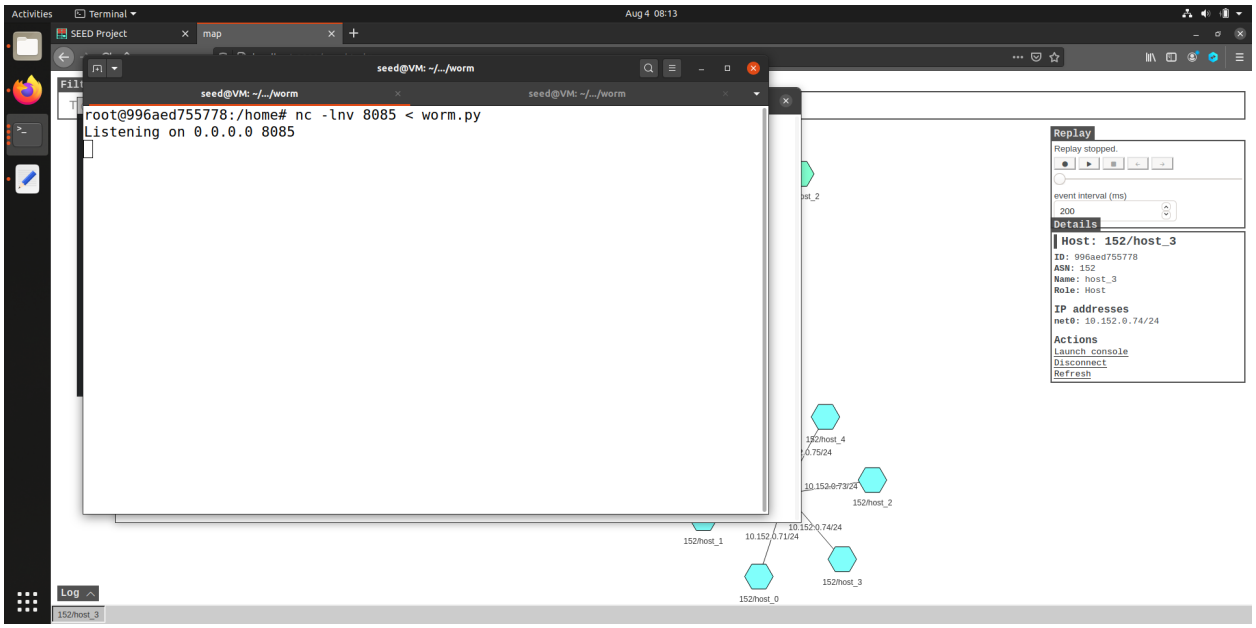
We divided the attack code into two parts, a small payload that contained a simple pilot code, and a larger payload that contained more sophisticated malicious code. The pilot code was the shellcode included in the malicious payload in the buffer-overflow attack. Once the attack was successful and the pilot code run a shell on the target, it used shell commands to fetch the larger payload from the attacker machine, completing the self duplication.



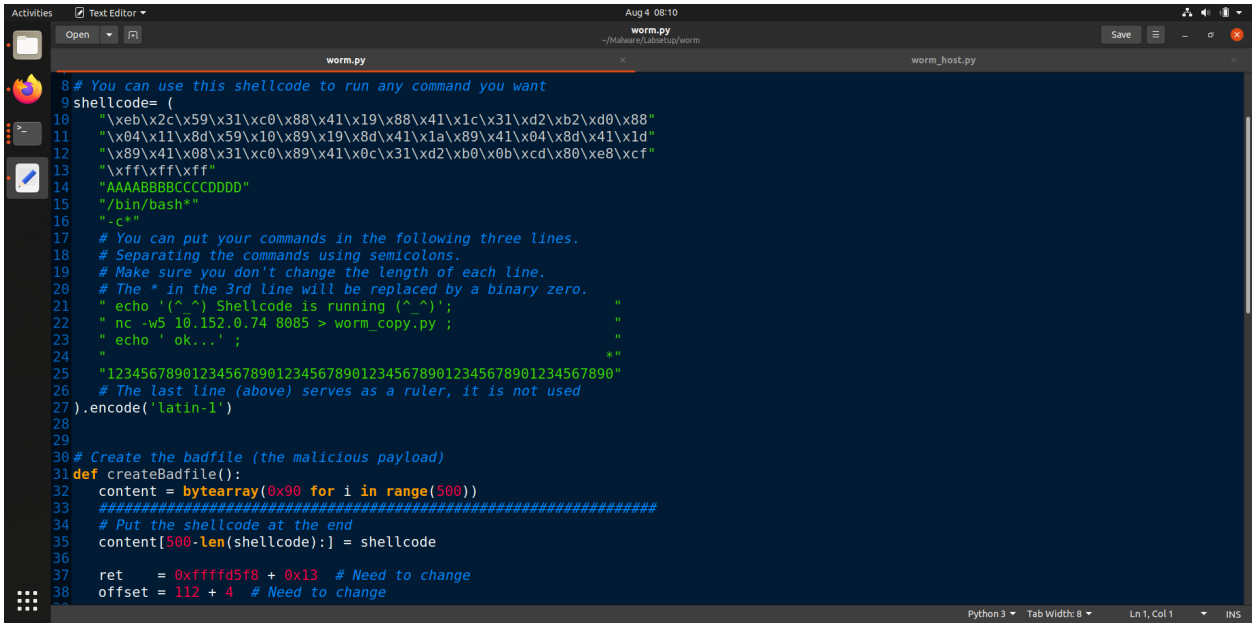
Our targeted victim machine's IP was 10.153.0.74 (can be seen in Line 51).

And attacker machine's IP was 10.152.0.74 (can be seen at right side of next snapshot as 152/host_3 at next page)

To get files from another computer, we used a client and a server program. On the emulators, many client/server programs had already been installed.

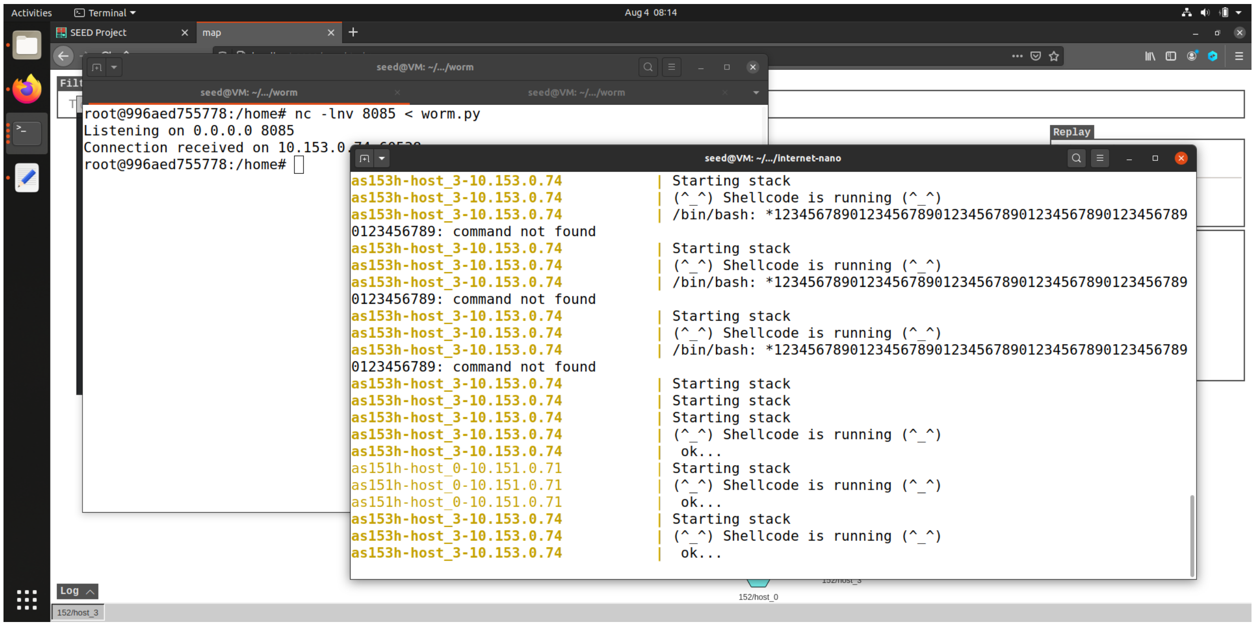


Using **netcat** we opened a tcp port 8085 and started our server in attacker machine who took its input from worm.py, and sent the malicious content to the client / our victim machine.

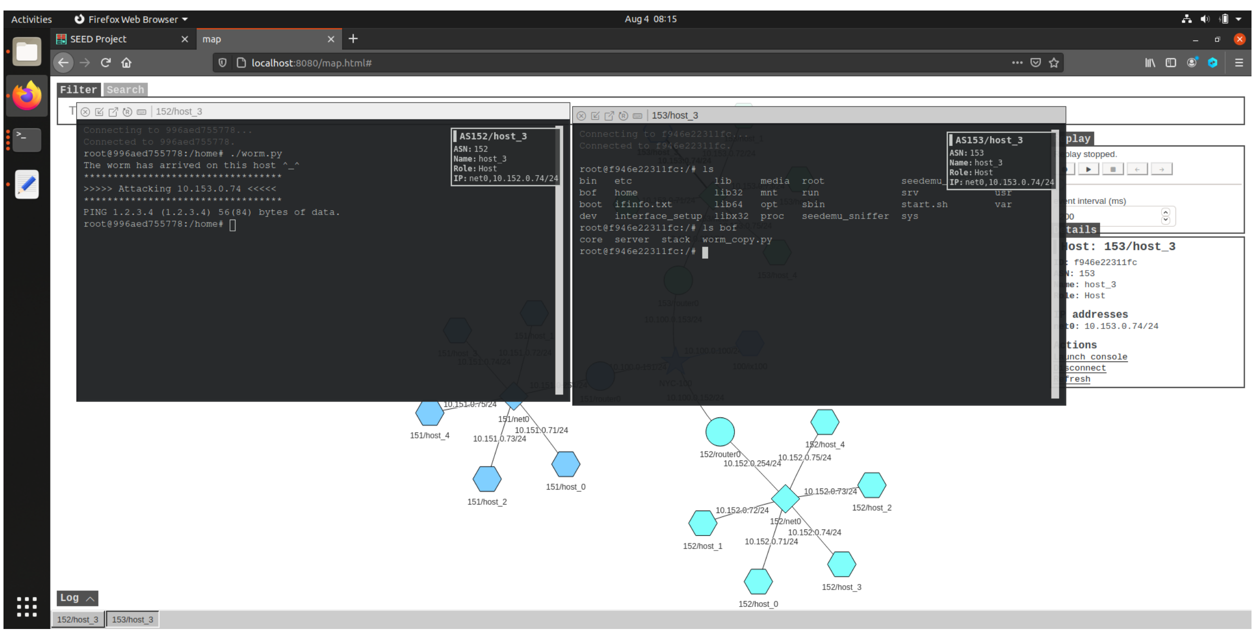


On the other hand , after successful attack we made a shell opened at victim machine so that we could execute our netcat listening code (in Line 22) to recieve data from server or attacker machine with attacker's IP address at a particular port 8085 (can be

anything we want).And then It would receive the payload and then stored it into a copy file named worm.py.



So when we run our worm.py file , then attacked was made and thus successfully victim received the malicious payload as we can see from above. The connection was received and so the file was stored.

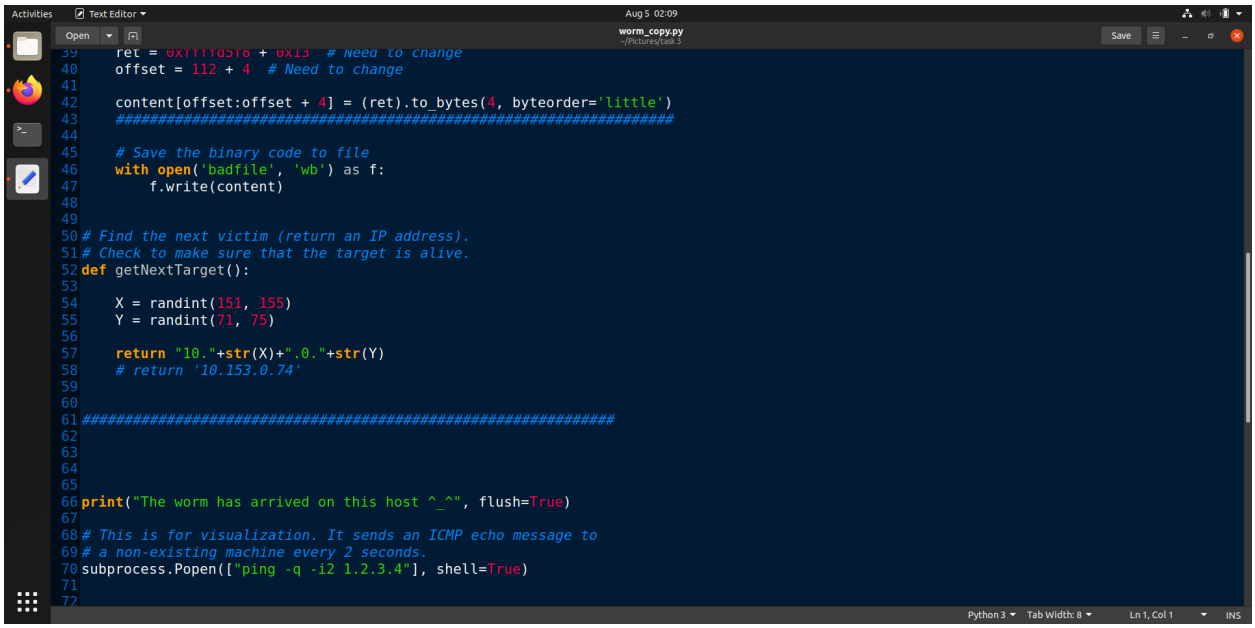


We checked the victim machine whether it got the copied file or not.So we found that it did (can be seen in the right console of victim).

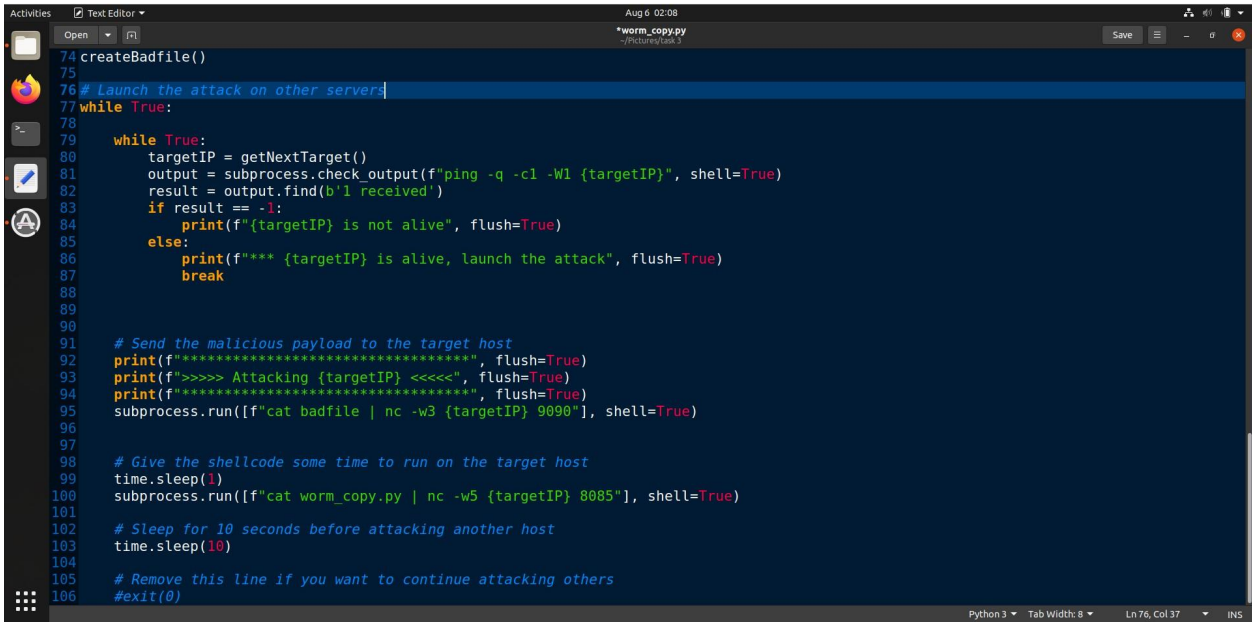
Task 3: Propagation

Assignment Task: After finishing this step, you need to demonstrate that your worms can spread from one computer to another, and eventually reach the entire nano internet. In your initial attack, you should only release the worm on one of the nodes, instead of keep attacking the other nodes from your attack machine.We want the worm to attack others automatically.

After finishing the previous task,we were able to get the worm to crawl from our attacker computer to the first targeted victim machine, but the worm would not keep crawling. So in this task we made some changes to worm.py so that the worm could continue crawling after it arrived on a newly compromised machine.

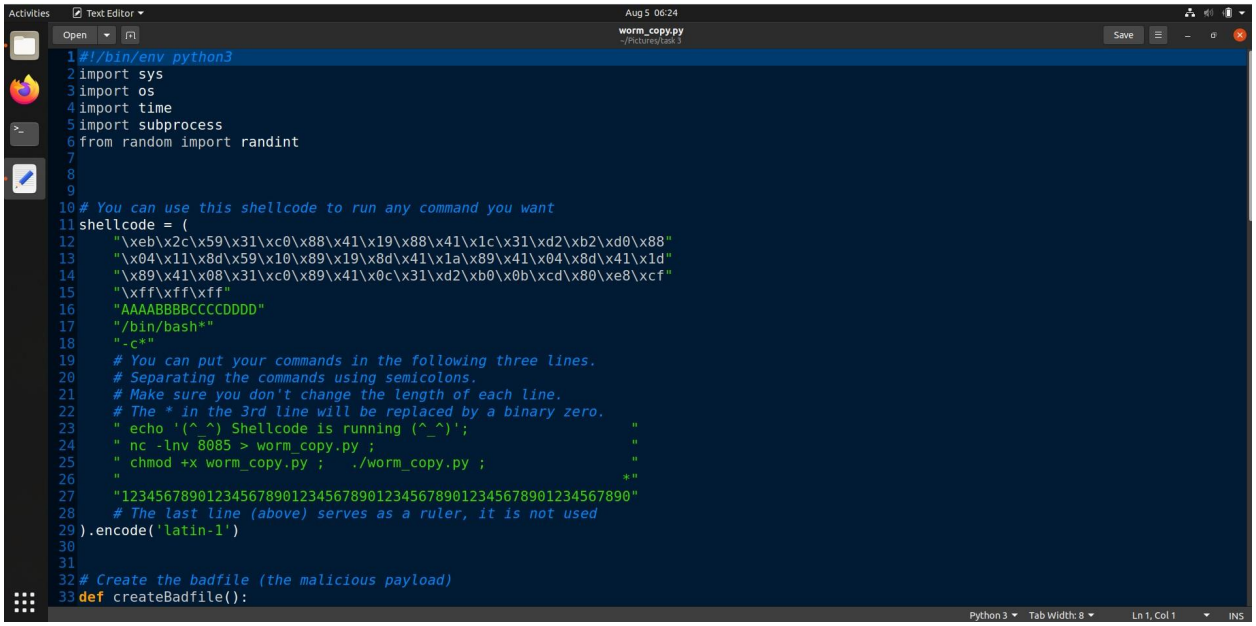


In previous tasks inside getNextTarget(), we hardcoded the IP address of the next target.But here we needed to generate our next target randomly so that we could reach and cover all of the hosts in the network.So we modified getNextTarget() function which gave random IP address of next targeted victim machines.



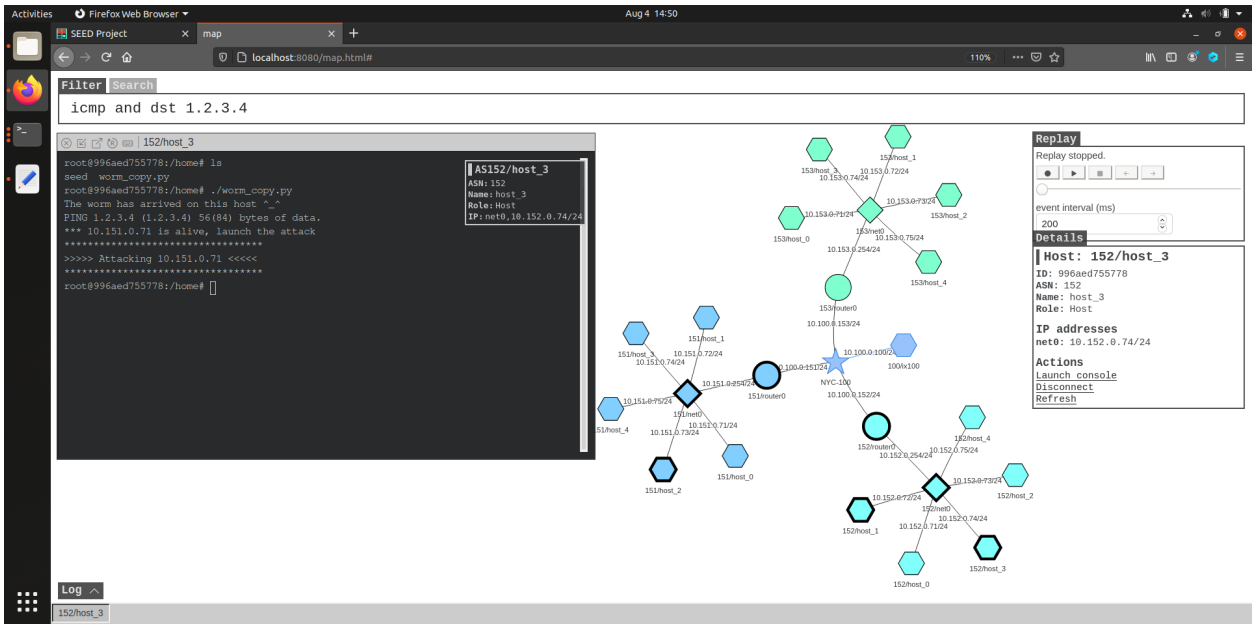
Before attacking a randomly selected target, it was better to check whether the target was alive or not.So we used an approach in which we used the ping command to send an echo request to the target, and checked whether the target had sent back a reply or not.

- As we can see In the above code snippet (started in Line 88), we sent out one ping packet (-c1) to the target victim machine, waited one second (-W1) for the reply, and then checked whether the output of the command contained "1 received", indicating that the reply had been received.If it was found alive only then went forward to launch our attack.
- After that we used a new subprocess (in Line 95) to send our malicious payload to the vulnerable server so that it could execute it (badfile) and got attacked.It could be thought as a simple pilot code to open the possibility of injecting larger more sophisticated malicious code which was actually our **worm.py** file.
- So after opening a shell using that simple pilot code we would run another subprocess to send our sophisticated code to victim side.We did that in Line 100 using target victims IP address (targetIP) and a particular port which was 8085.
- We also commented out the 106th line (the exit 0 command) so that we could facilitate the propagation of infection further.

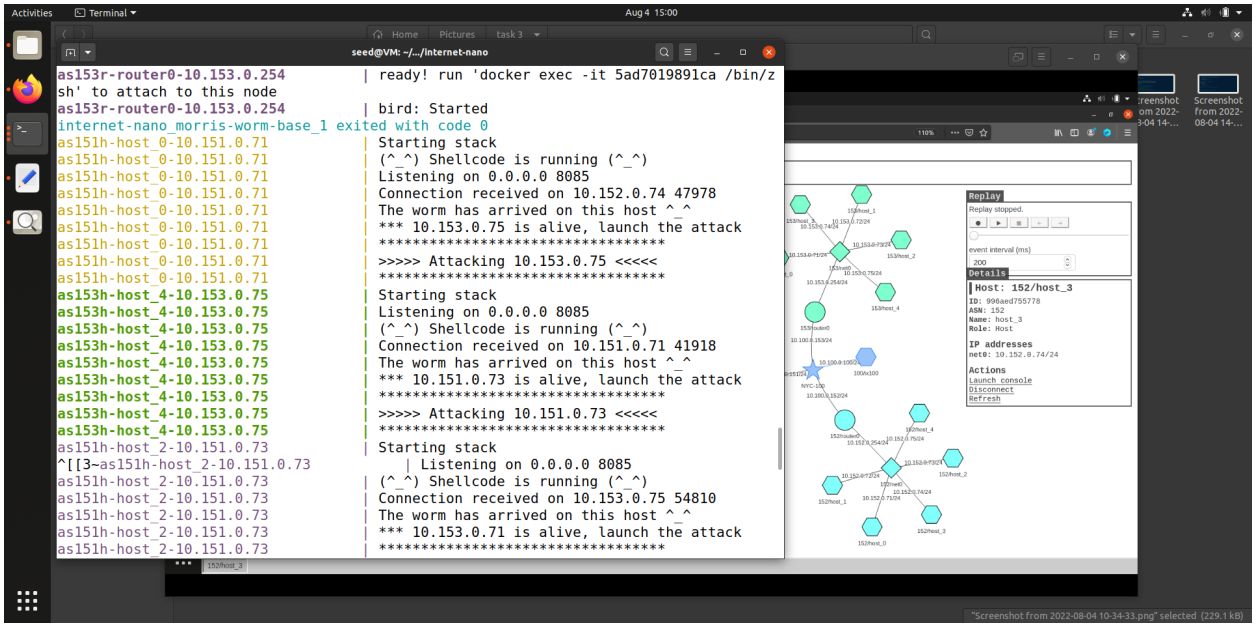


As we made changes in server (attacker) side to change our sophisticated code , we also changed in our shellcode (shown in Line 24) the way we received payload by victim side. Previously (in task 2) the victim itself was listening to attacker port along with attacker's IP address and the attacker was just sending from that port.But this time victim was just listening with a particular port and the attacker instead was sending the file to the victim's IP address(targetIP).

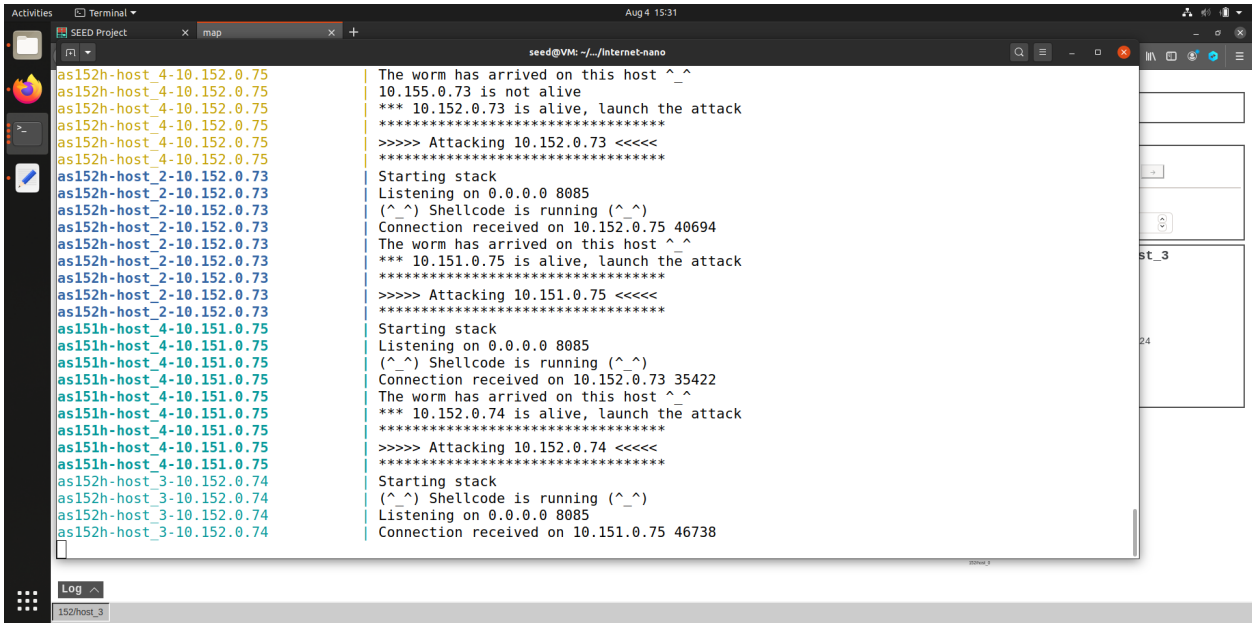
We added the permission (in Line 25) to the victim machine so that it could execute the worm by itself and thus propagated it further.

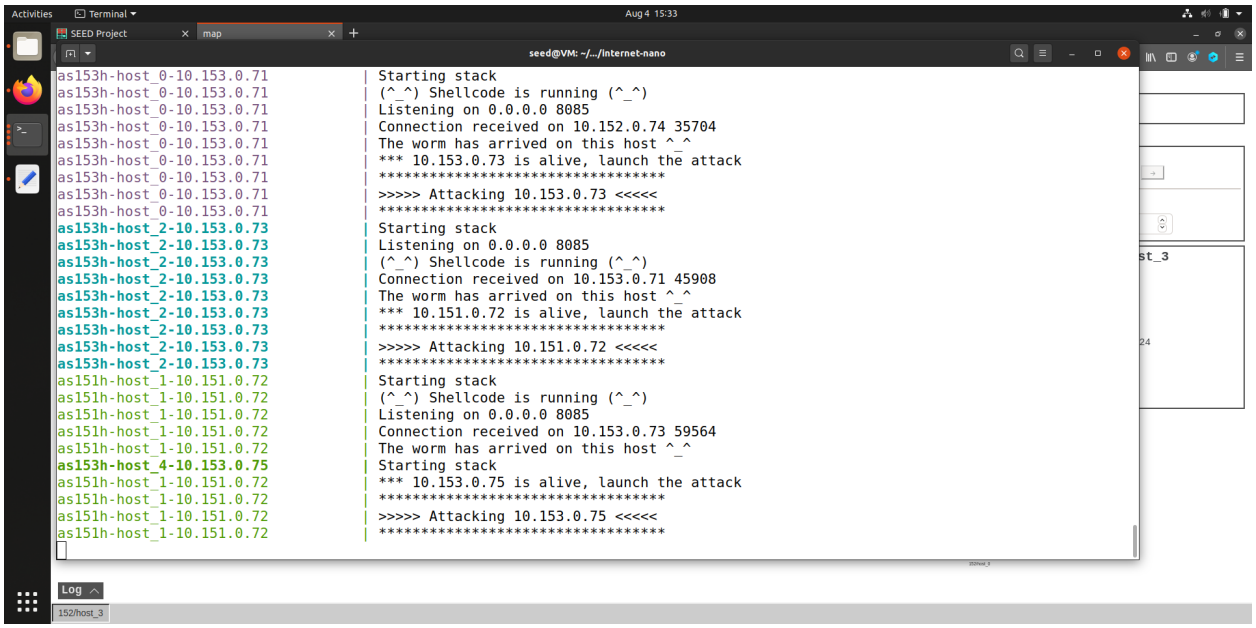


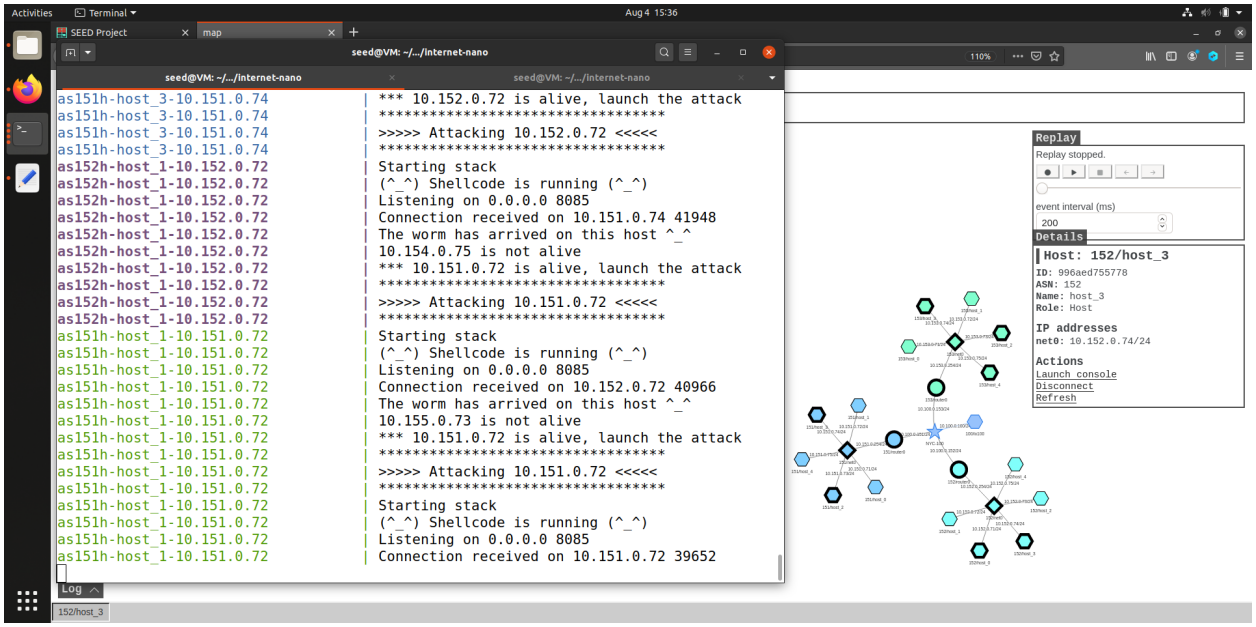
After we run worm.py from the attacker machine , it started attacking and these attacks propagated node by node or machine by machine.We took several snapshots of the attack below.




Activities
Terminal
Aug 4 15:00
seed@VM: ~/.../internet-nano
as152h-host_1-10.152.0.72 | **********
as152h-host_1-10.152.0.72 | >>>>> Attacking 10.152.0.75 <<<<<
as152h-host_1-10.152.0.72 | **********
as152h-host_4-10.152.0.75 | Starting stack
as152h-host_4-10.152.0.75 | Listening on 0.0.0.0 8085
as152h-host_4-10.152.0.75 | (^_^) Shellcode is running (^_^)
as152h-host_4-10.152.0.75 | Connection received on 10.152.0.72 59918
as152h-host_4-10.152.0.75 | The worm has arrived on this host ^_^
as152h-host_4-10.152.0.75 | *** 10.151.0.74 is alive, launch the attack
as152h-host_4-10.152.0.75 | **********
as152h-host_4-10.152.0.75 | >>>>> Attacking 10.151.0.74 <<<<<
as152h-host_4-10.152.0.75 | **********
as151h-host_3-10.151.0.74 | Starting stack
as151h-host_3-10.151.0.74 | Listening on 0.0.0.0 8085
as151h-host_3-10.151.0.74 | (^_^) Shellcode is running (^_^)
as151h-host_3-10.151.0.74 | Connection received on 10.152.0.75 52572
as151h-host_3-10.151.0.74 | The worm has arrived on this host ^_^
as151h-host_3-10.151.0.74 | *** 10.151.0.72 is alive, launch the attack
as151h-host_3-10.151.0.74 | **********
as151h-host_3-10.151.0.74 | >>>>> Attacking 10.151.0.72 <<<<<
as151h-host_1-10.151.0.72 | Starting stack
as151h-host_3-10.151.0.74 | **********
as151h-host_1-10.151.0.72 | Listening on 0.0.0.0 8085
as151h-host_1-10.151.0.72 | (^_^) Shellcode is running (^_^)
as151h-host_1-10.151.0.72 | Connection received on 10.151.0.74 52084
as151h-host_1-10.151.0.72 | The worm has arrived on this host ^_^
as151h-host_1-10.151.0.72 | *** 10.151.0.72 is alive, launch the attack
as151h-host_1-10.151.0.72 | **********
as151h-host_1-10.151.0.72 | >>>>> Attacking 10.151.0.72 <<<<<
Replay
Replay stopped.
event interval (ms)
200
Details
Host: 152/host_3
ID: 996aed755778
ASN: 152
Name: host_3
Role: Host
IP addresses
net0: 10.152.0.74/24
Actions
Launch console
Disconnect
Refresh


Activities
Terminal
Aug 4 15:31
SEED Project
map
seed@VM: ~/.../internet-nano
as152h-host_4-10.152.0.75 | The worm has arrived on this host ^_^
as152h-host_4-10.152.0.75 | 10.155.0.73 is not alive
as152h-host_4-10.152.0.75 | *** 10.152.0.73 is alive, launch the attack
as152h-host_4-10.152.0.75 | **********
as152h-host_4-10.152.0.75 | >>>>> Attacking 10.152.0.73 <<<<<
as152h-host_4-10.152.0.75 | **********
as152h-host_2-10.152.0.73 | Starting stack
as152h-host_2-10.152.0.73 | Listening on 0.0.0.0 8085
as152h-host_2-10.152.0.73 | (^_^) Shellcode is running (^_^)
as152h-host_2-10.152.0.73 | Connection received on 10.152.0.75 40694
as152h-host_2-10.152.0.73 | The worm has arrived on this host ^_^
as152h-host_2-10.152.0.73 | *** 10.151.0.75 is alive, launch the attack
as152h-host_2-10.152.0.73 | **********
as152h-host_2-10.152.0.73 | >>>>> Attacking 10.151.0.75 <<<<<
as152h-host_2-10.152.0.73 | **********
as151h-host_4-10.151.0.75 | Starting stack
as151h-host_4-10.151.0.75 | Listening on 0.0.0.0 8085
as151h-host_4-10.151.0.75 | (^_^) Shellcode is running (^_^)
as151h-host_4-10.151.0.75 | Connection received on 10.152.0.73 35422
as151h-host_4-10.151.0.75 | The worm has arrived on this host ^_^
as151h-host_4-10.151.0.75 | *** 10.152.0.74 is alive, launch the attack
as151h-host_4-10.151.0.75 | **********
as151h-host_4-10.151.0.75 | >>>>> Attacking 10.152.0.74 <<<<<
as151h-host_4-10.151.0.75 | **********
as152h-host_3-10.152.0.74 | Starting stack
as152h-host_3-10.152.0.74 | (^_^) Shellcode is running (^_^)
as152h-host_3-10.152.0.74 | Listening on 0.0.0.0 8085
as152h-host_3-10.152.0.74 | Connection received on 10.151.0.75 46738
Log
152/host_3

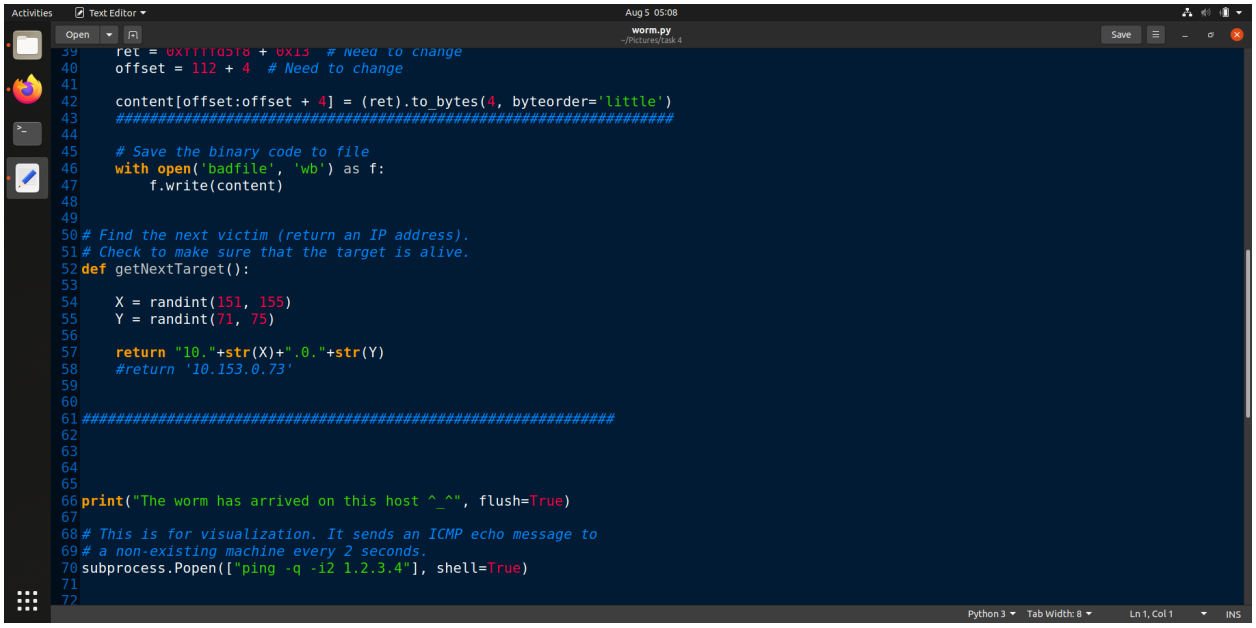

Activities
Terminal
Aug 4 15:33
SEED Project
map
seed@VM: ~/.../internet-nano
as153h-host_0-10.153.0.71 | Starting stack
as153h-host_0-10.153.0.71 | (^_^) Shellcode is running (^_^)
as153h-host_0-10.153.0.71 | Listening on 0.0.0.0 8085
as153h-host_0-10.153.0.71 | Connection received on 10.152.0.74 35704
as153h-host_0-10.153.0.71 | The worm has arrived on this host ^_^
as153h-host_0-10.153.0.71 | *** 10.153.0.73 is alive, launch the attack
as153h-host_0-10.153.0.71 | **********
as153h-host_0-10.153.0.71 | >>>>> Attacking 10.153.0.73 <<<<<
as153h-host_0-10.153.0.71 | **********
as153h-host_2-10.153.0.73 | Starting stack
as153h-host_2-10.153.0.73 | Listening on 0.0.0.0 8085
as153h-host_2-10.153.0.73 | (^_^) Shellcode is running (^_^)
as153h-host_2-10.153.0.73 | Connection received on 10.153.0.71 45908
as153h-host_2-10.153.0.73 | The worm has arrived on this host ^_^
as153h-host_2-10.153.0.73 | *** 10.151.0.72 is alive, launch the attack
as153h-host_2-10.153.0.73 | **********
as153h-host_2-10.153.0.73 | >>>>> Attacking 10.151.0.72 <<<<<
as153h-host_2-10.153.0.73 | **********
as151h-host_1-10.151.0.72 | Starting stack
as151h-host_1-10.151.0.72 | (^_^) Shellcode is running (^_^)
as151h-host_1-10.151.0.72 | Listening on 0.0.0.0 8085
as151h-host_1-10.151.0.72 | Connection received on 10.153.0.73 59564
as151h-host_1-10.151.0.72 | The worm has arrived on this host ^_^
as153h-host_4-10.153.0.75 | Starting stack
as151h-host_1-10.151.0.72 | *** 10.153.0.75 is alive, launch the attack
as151h-host_1-10.151.0.72 | **********
as151h-host_1-10.151.0.72 | >>>>> Attacking 10.153.0.75 <<<<<
as151h-host_1-10.151.0.72 | **********
Log
152/host_3


Activities
Terminal
Aug 4 15:36
SEED Project
map
seed@VM: ~/.../internet-nano
seed@VM: ~/.../internet-nano
seed@VM: ~/.../internet-nano
as151h-host_3-10.151.0.74 | *** 10.152.0.72 is alive, launch the attack
as151h-host_3-10.151.0.74 | **********
as151h-host_3-10.151.0.74 | >>>>> Attacking 10.152.0.72 <<<<<
as151h-host_3-10.151.0.74 | **********
as152h-host_1-10.152.0.72 | Starting stack
as152h-host_1-10.152.0.72 | (^_^) Shellcode is running (^_^)
as152h-host_1-10.152.0.72 | Listening on 0.0.0.0 8085
as152h-host_1-10.152.0.72 | Connection received on 10.151.0.74 41948
as152h-host_1-10.152.0.72 | The worm has arrived on this host ^_^
as152h-host_1-10.152.0.72 | 10.154.0.75 is not alive
as152h-host_1-10.152.0.72 | *** 10.151.0.72 is alive, launch the attack
as152h-host_1-10.152.0.72 | **********
as152h-host_1-10.152.0.72 | >>>>> Attacking 10.151.0.72 <<<<<
as152h-host_1-10.152.0.72 | **********
as151h-host_1-10.151.0.72 | Starting stack
as151h-host_1-10.151.0.72 | (^_^) Shellcode is running (^_^)
as151h-host_1-10.151.0.72 | Listening on 0.0.0.0 8085
as151h-host_1-10.151.0.72 | Connection received on 10.152.0.72 40966
as151h-host_1-10.151.0.72 | The worm has arrived on this host ^_^
as151h-host_1-10.151.0.72 | 10.155.0.73 is not alive
as151h-host_1-10.151.0.72 | *** 10.151.0.72 is alive, launch the attack
as151h-host_1-10.151.0.72 | **********
as151h-host_1-10.151.0.72 | >>>>> Attacking 10.151.0.72 <<<<<
as151h-host_1-10.151.0.72 | **********
as151h-host_1-10.151.0.72 | Starting stack
as151h-host_1-10.151.0.72 | (^_^) Shellcode is running (^_^)
as151h-host_1-10.151.0.72 | Listening on 0.0.0.0 8085
as151h-host_1-10.151.0.72 | Connection received on 10.151.0.72 39652
Log
152/host_3
Replay
Replay stopped.
event interval (ms)
200
Details
Host: 152/host_3
ID: 996aed755778
ASN: 152
Name: host_3
Role: Host
IP addresses
net0: 10.152.0.74/24
Actions
Launch console
Disconnect
Refresh

Task 4: Preventing Self Infection

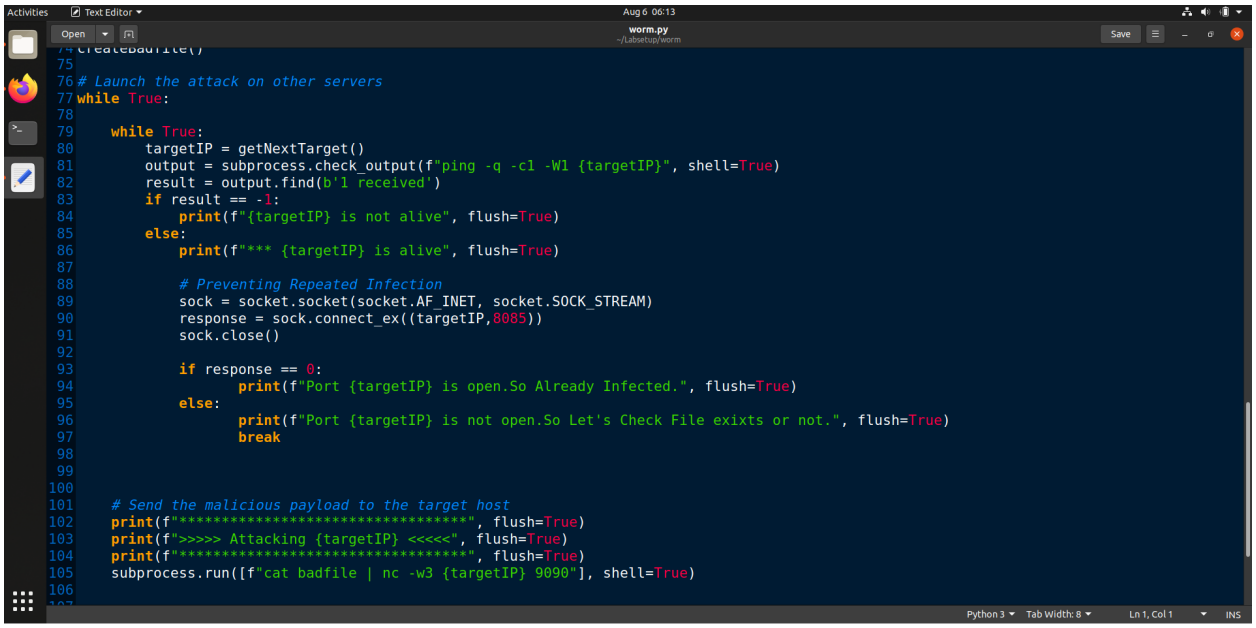
Assignment Task: In this task, you need to add such a checking mechanism to their worm code to ensure that only one instance of the worm can run on a compromised computer. Once this is implemented, during the attack, the CPU usage will unlikely reach the 100 percent. Also, you need to ensure that if a worm file is already present in a victim machine. If so, then do not copy the worm file from the source again.

Demonstration of Final Task



Generating random IP addresses of victims was as usual like task 3 we did.

In this task our target was to add such a checking mechanism to the worm code so that it can be ensured that only one instance of the worm can be run on a compromised computer.Because if any computer gets compromised again, a new instance of the worm will start running. If the worm does not have a mechanism to check whether a computer has already been infected or not, many new instances of the worms will be spawn, consuming more and more resources, and eventually bring the target machines to their knees or, in many cases, crash them.So we found a mechanism of our own.

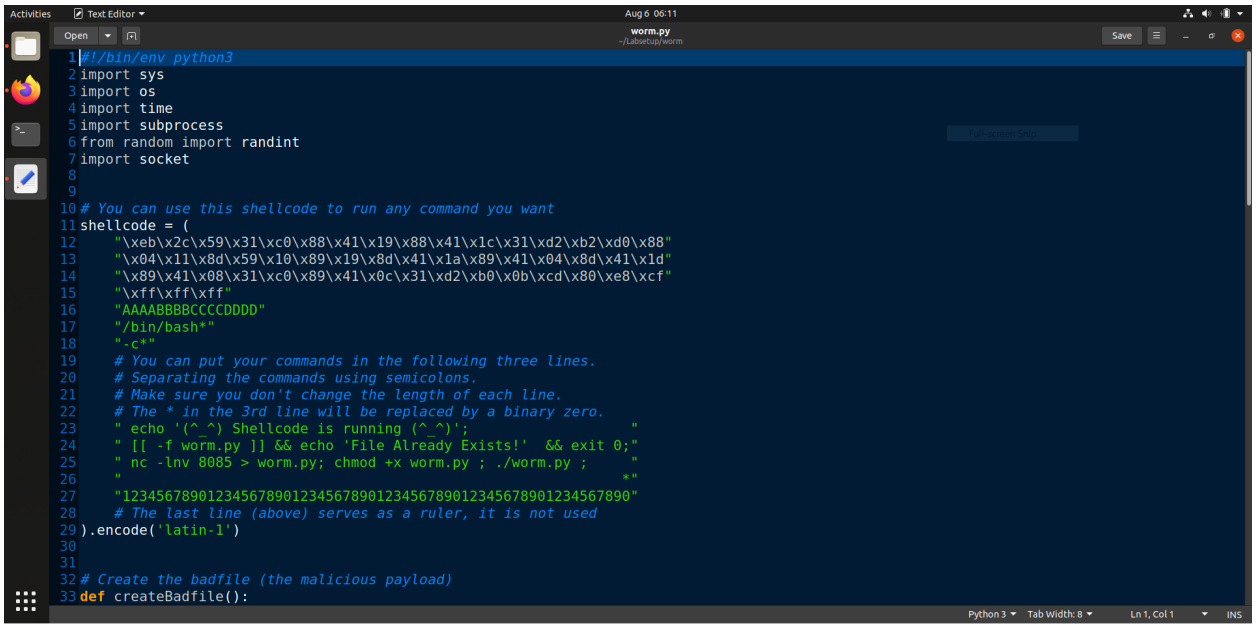


The extra thing we added from task 3 was checking whether the port of our targeted victim active or not.If it was, then it meant it was already infected by another host.Because while attacking , attacker opened a port to send our sophisticated code to victim and then that victim started playing the part of attacker.So once the port started , it wasn't closed.That is why that port was open.

But if it wasn't open then there could be possibly three cases :-

1. That host wasn't attacked before.
2. In some cases,as we forcefully stopped the hosts so that it couldn't execute further it's already executed malicious worm code.
3. Assuming the port was closed accidentally.

Only then we launched our attack.

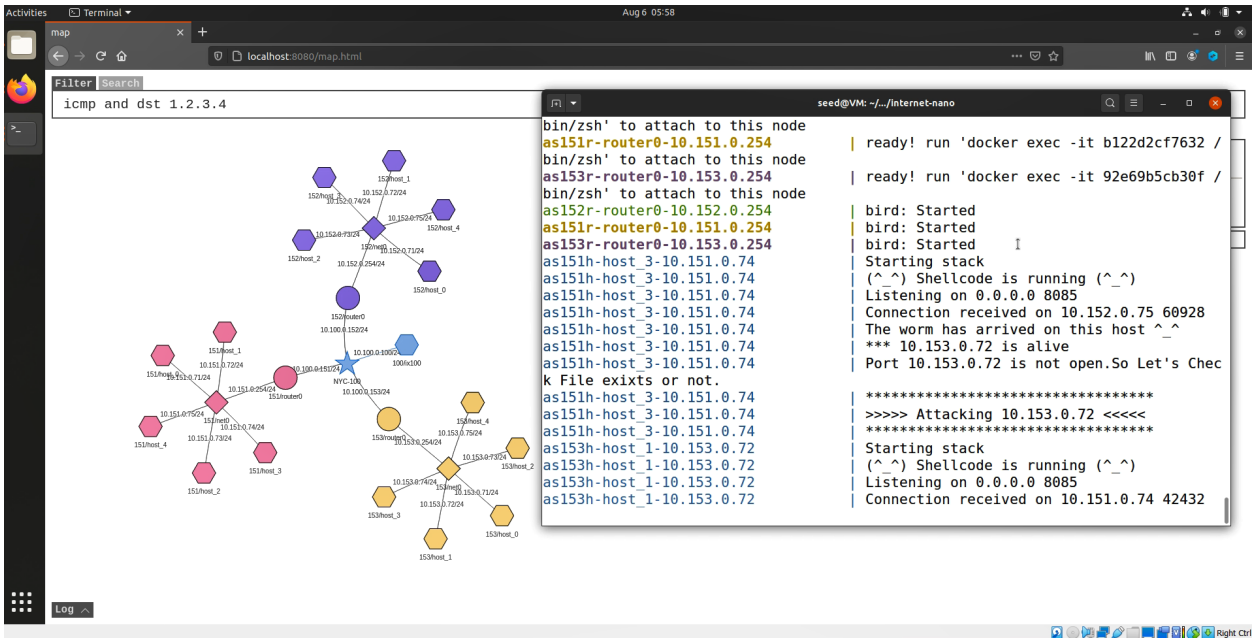


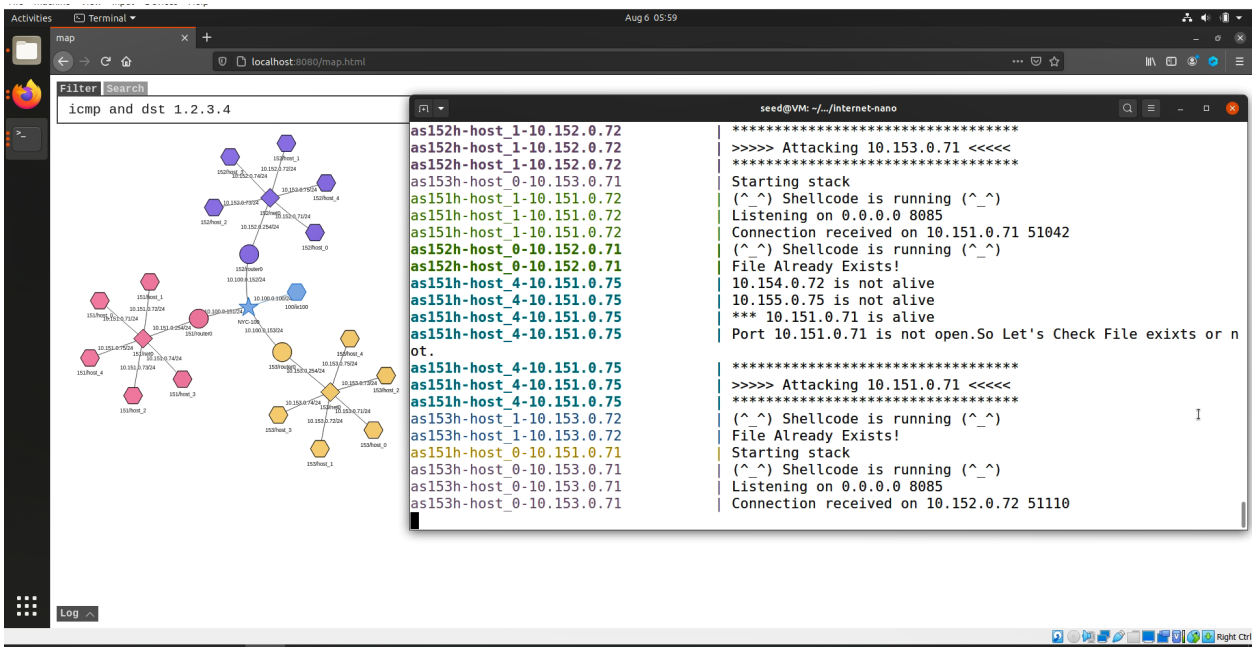
Again after entering victim machine we also checked whether there was our malicious payload (worm.py) or not .If it was present , we didn't copy it.Rather we showed an alert message (shown in Line 24).

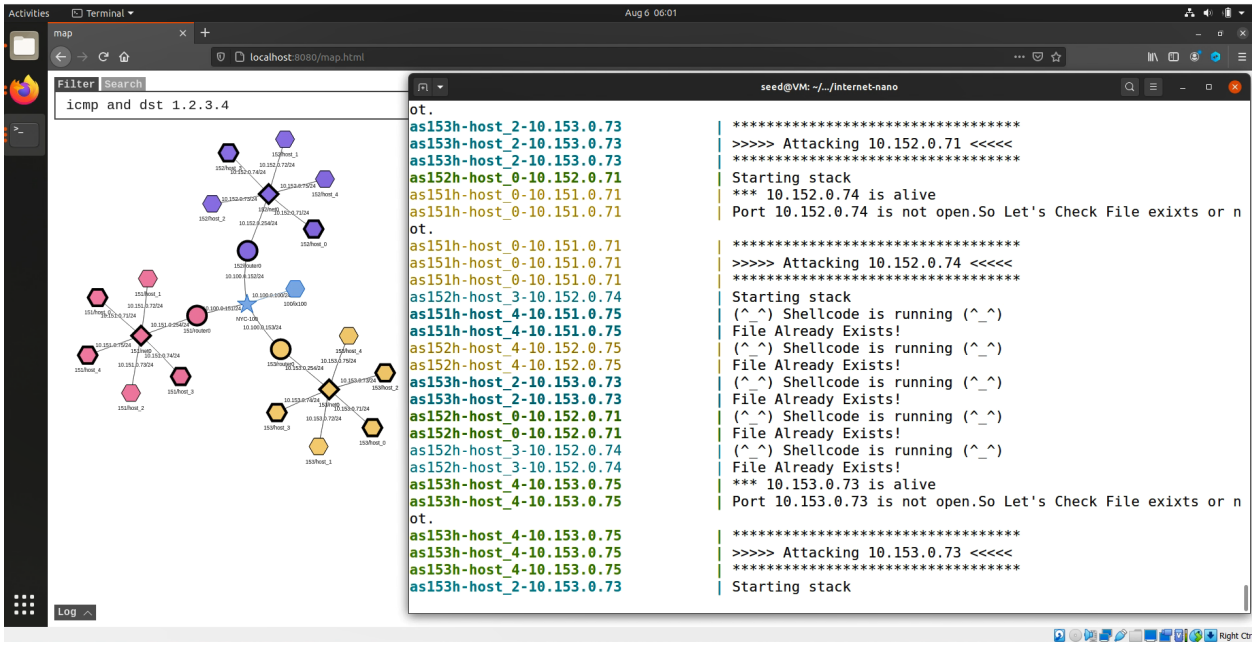
If not, then we let it copied , permitted for execution and then finally executed it for further propagation of infection.

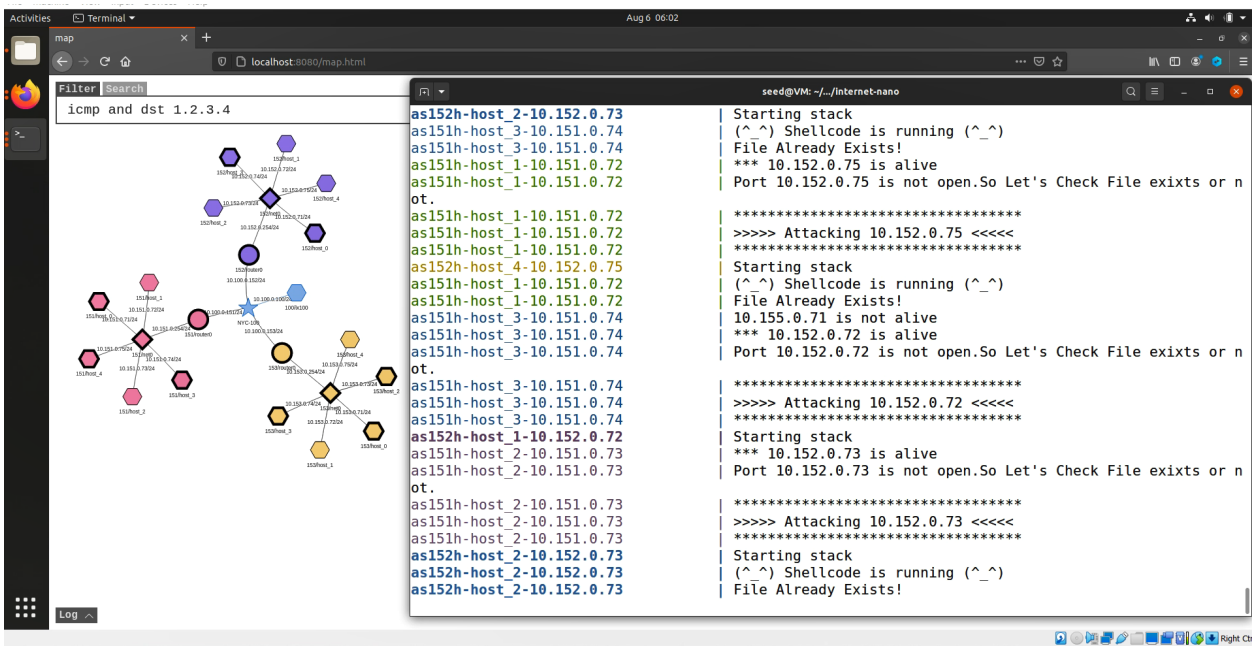
Lastly we started our attack from the first of our machine and then we started observing.

We put some snapshots of propagation of infection from the next page.


Activities
Terminal
Aug 6 05:58
map
localhost:8080/map.html
Filter
Search
icmp and dst 1.2.3.4
seed@VM: ~/.../internet-nano
bin/zsh' to attach to this node
as151r-router0-10.151.0.254 | ready! run 'docker exec -it b122d2cf7632 /
bin/zsh' to attach to this node
as153r-router0-10.153.0.254 | ready! run 'docker exec -it 92e69b5cb30f /
bin/zsh' to attach to this node
as152r-router0-10.152.0.254 | bird: Started
as151r-router0-10.151.0.254 | bird: Started
as153r-router0-10.153.0.254 | bird: Started
as151h-host_3-10.151.0.74 | Starting stack
as151h-host_3-10.151.0.74 | (^_^) Shellcode is running (^_^)
as151h-host_3-10.151.0.74 | Listening on 0.0.0.0 8085
as151h-host_3-10.151.0.74 | Connection received on 10.152.0.75 60928
as151h-host_3-10.151.0.74 | The worm has arrived on this host ^_^
as151h-host_3-10.151.0.74 | *** 10.153.0.72 is alive
as151h-host_3-10.151.0.74 | Port 10.153.0.72 is not open.So Let's Chec
k File exixts or not.
as151h-host_3-10.151.0.74 | **********
as151h-host_3-10.151.0.74 | >>>>> Attacking 10.153.0.72 <<<<<
as151h-host_3-10.151.0.74 | **********
as153h-host_1-10.153.0.72 | Starting stack
as153h-host_1-10.153.0.72 | (^_^) Shellcode is running (^_^)
as153h-host_1-10.153.0.72 | Listening on 0.0.0.0 8085
as153h-host_1-10.153.0.72 | Connection received on 10.151.0.74 42432
Log
Right Ctrl


Activities
Terminal
Aug 6 05:59
map
localhost:8080/map.html
Filter
Search
icmp and dst 1.2.3.4
seed@VM: ~/.../internet-nano
as152h-host_1-10.152.0.72 | **********
as152h-host_1-10.152.0.72 | >>>>> Attacking 10.153.0.71 <<<<<
as152h-host_1-10.152.0.72 | **********
as153h-host_0-10.153.0.71 | Starting stack
as151h-host_1-10.151.0.72 | (^_^) Shellcode is running (^_^)
as151h-host_1-10.151.0.72 | Listening on 0.0.0.0 8085
as151h-host_1-10.151.0.72 | Connection received on 10.151.0.71 51042
as152h-host_0-10.152.0.71 | (^_^) Shellcode is running (^_^)
as152h-host_0-10.152.0.71 | File Already Exists!
as151h-host_4-10.151.0.75 | 10.154.0.72 is not alive
as151h-host_4-10.151.0.75 | 10.155.0.75 is not alive
as151h-host_4-10.151.0.75 | *** 10.151.0.71 is alive
as151h-host_4-10.151.0.75 | Port 10.151.0.71 is not open.So Let's Check File exixts or n
ot.
as151h-host_4-10.151.0.75 | **********
as151h-host_4-10.151.0.75 | >>>>> Attacking 10.151.0.71 <<<<<
as151h-host_4-10.151.0.75 | **********
as153h-host_1-10.153.0.72 | (^_^) Shellcode is running (^_^)
as153h-host_1-10.153.0.72 | File Already Exists!
as151h-host_0-10.151.0.71 | Starting stack
as153h-host_0-10.153.0.71 | (^_^) Shellcode is running (^_^)
as153h-host_0-10.153.0.71 | Listening on 0.0.0.0 8085
as153h-host_0-10.153.0.71 | Connection received on 10.152.0.72 51110
Log
Right Ctrl


Activities
Terminal
Aug 6 06:01
map
localhost:8080/map.html
Filter
Search
icmp and dst 1.2.3.4
Log
seed@VM: ~/.../internet-nano
as153h-host_2-10.153.0.73 | **********
as153h-host_2-10.153.0.73 | >>>>> Attacking 10.152.0.71 <<<<<
as153h-host_2-10.153.0.73 | **********
as152h-host_0-10.152.0.71 | Starting stack
as151h-host_0-10.151.0.71 | *** 10.152.0.74 is alive
as151h-host_0-10.151.0.71 | Port 10.152.0.74 is not open.So Let's Check File exixts or n
ot.
as151h-host_0-10.151.0.71 | **********
as151h-host_0-10.151.0.71 | >>>>> Attacking 10.152.0.74 <<<<<
as151h-host_0-10.151.0.71 | **********
as152h-host_3-10.152.0.74 | Starting stack
as151h-host_4-10.151.0.75 | (^_^) Shellcode is running (^_^)
as151h-host_4-10.151.0.75 | File Already Exists!
as152h-host_4-10.152.0.75 | (^_^) Shellcode is running (^_^)
as152h-host_4-10.152.0.75 | File Already Exists!
as153h-host_2-10.153.0.73 | (^_^) Shellcode is running (^_^)
as153h-host_2-10.153.0.73 | File Already Exists!
as152h-host_0-10.152.0.71 | (^_^) Shellcode is running (^_^)
as152h-host_0-10.152.0.71 | File Already Exists!
as152h-host_3-10.152.0.74 | (^_^) Shellcode is running (^_^)
as152h-host_3-10.152.0.74 | File Already Exists!
as153h-host_4-10.153.0.75 | *** 10.153.0.73 is alive
as153h-host_4-10.153.0.75 | Port 10.153.0.73 is not open.So Let's Check File exixts or n
ot.
as153h-host_4-10.153.0.75 | **********
as153h-host_4-10.153.0.75 | >>>>> Attacking 10.153.0.73 <<<<<
as153h-host_4-10.153.0.75 | **********
as153h-host_2-10.153.0.73 | Starting stack


Activities
Terminal
Aug 6 06:02
map
localhost:8080/map.html
Filter
Search
icmp and dst 1.2.3.4
Log
seed@VM: ~/.../internet-nano
as152h-host_2-10.152.0.73 | Starting stack
as151h-host_3-10.151.0.74 | (^_^) Shellcode is running (^_^)
as151h-host_3-10.151.0.74 | File Already Exists!
as151h-host_1-10.151.0.72 | *** 10.152.0.75 is alive
as151h-host_1-10.151.0.72 | Port 10.152.0.75 is not open.So Let's Check File exixts or n
ot.
as151h-host_1-10.151.0.72 | **********
as151h-host_1-10.151.0.72 | >>>>> Attacking 10.152.0.75 <<<<<
as151h-host_1-10.151.0.72 | **********
as152h-host_4-10.152.0.75 | Starting stack
as151h-host_1-10.151.0.72 | (^_^) Shellcode is running (^_^)
as151h-host_1-10.151.0.72 | File Already Exists!
as151h-host_3-10.151.0.74 | 10.155.0.71 is not alive
as151h-host_3-10.151.0.74 | *** 10.152.0.72 is alive
as151h-host_3-10.151.0.74 | Port 10.152.0.72 is not open.So Let's Check File exixts or n
ot.
as151h-host_3-10.151.0.74 | **********
as151h-host_3-10.151.0.74 | >>>>> Attacking 10.152.0.72 <<<<<
as151h-host_3-10.151.0.74 | **********
as152h-host_1-10.152.0.72 | Starting stack
as151h-host_2-10.151.0.73 | *** 10.152.0.73 is alive
as151h-host_2-10.151.0.73 | Port 10.152.0.73 is not open.So Let's Check File exixts or n
ot.
as151h-host_2-10.151.0.73 | **********
as151h-host_2-10.151.0.73 | >>>>> Attacking 10.152.0.73 <<<<<
as151h-host_2-10.151.0.73 | **********
as152h-host_2-10.152.0.73 | Starting stack
as152h-host_2-10.152.0.73 | (^_^) Shellcode is running (^_^)
as152h-host_2-10.152.0.73 | File Already Exists!