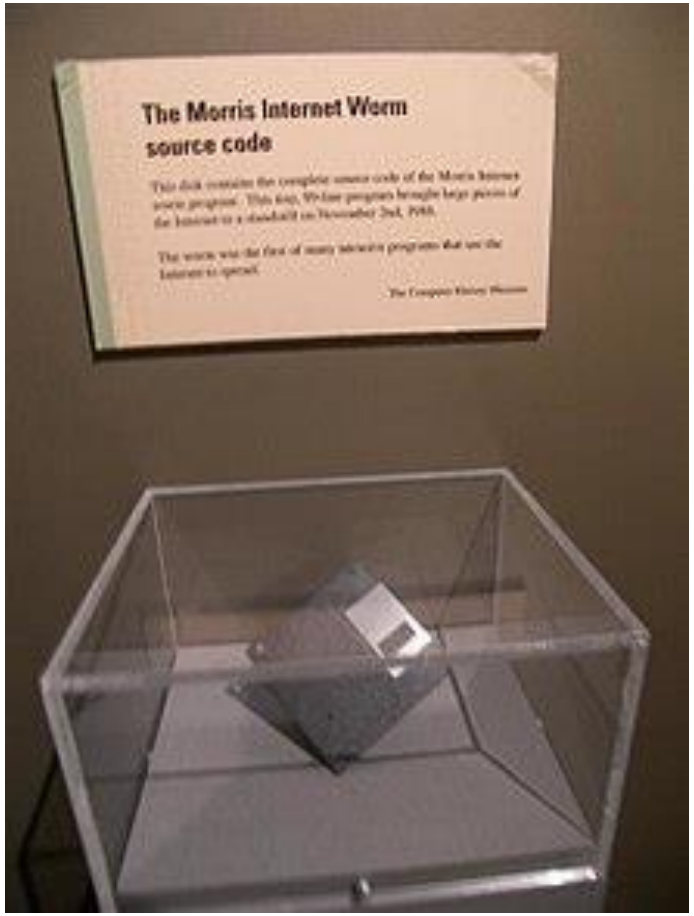
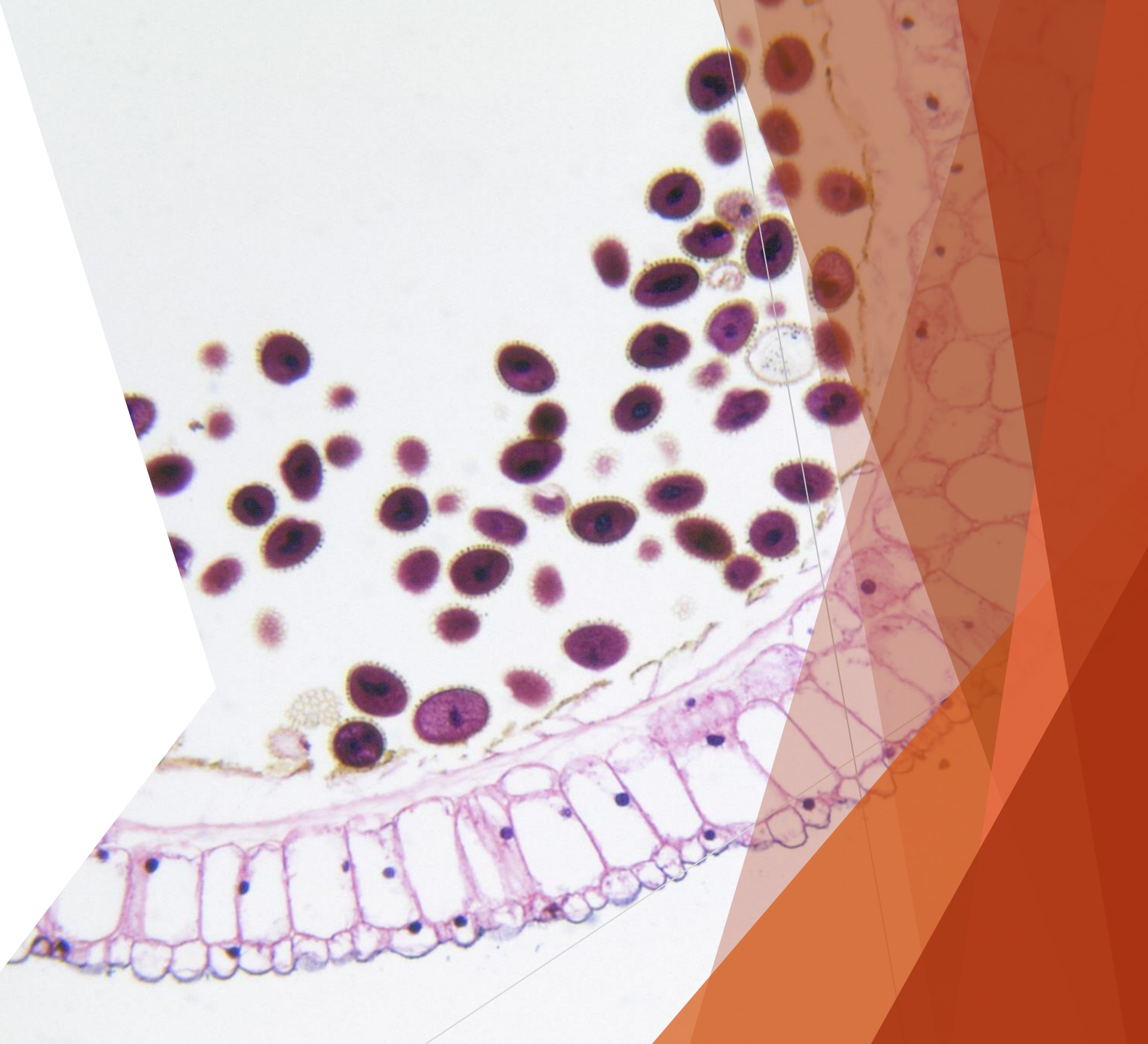


Morris Worm Attack

Overview

- What is Morris Worm?
- How to demonstrate the attack?
- Morris Worm lab
  - Setup VM https://github.com/seed-labs/seed-labs/blob/master/manuals/vm/seedvm-manual.md
  - Credits https://seedsecuritylabs.org/Labs_20.04/Networking/Morris_Worm/



http://www.loundy.com/CASES/US_v_Morris2.html

What is Morris Worm?

A malware was developed by Robert Tappan MORRIS,

- was a first-year graduate student in Cornell University's computer science Ph.D. program
- did undergraduate work at Harvard
- a tenured professor at MIT in 2006
- had a Unix account the Cornell

Morris' goal

- a program can self-spread across a national network of computers after being inserted at one computer location connected to the network
- not destructive
- demonstrate the inadequacies of current security measures on computer networks by exploiting the security defects that he had discovered

Attacking Method

through a "hole" or "bug" (an error) in sendmail

- a computer program that transfers and receives electronic mail on a computer

through a bug in the "finger deamon" program

- a program that permits a person to obtain limited information about the users of another computer

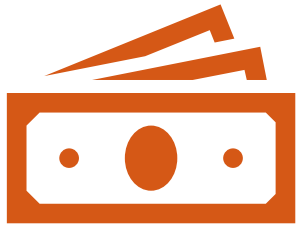
through the "trusted hosts" feature

- which permits a user with certain privileges on one computer to have equivalent privileges on another computer without using a password; and

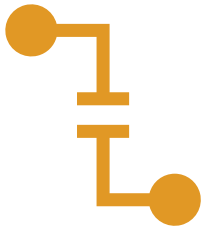
through a program of password guessing,

- whereby various combinations of letters are tried out in rapid sequence in the hope that one will be an authorized user's password, which is entered to permit whatever level of activity that user is authorized to perform.

Morris Worm vs. Ransomware



non-destructive vs. asking ransom fee



The techniques are still the same

exploit vulnerabilities

self-duplication

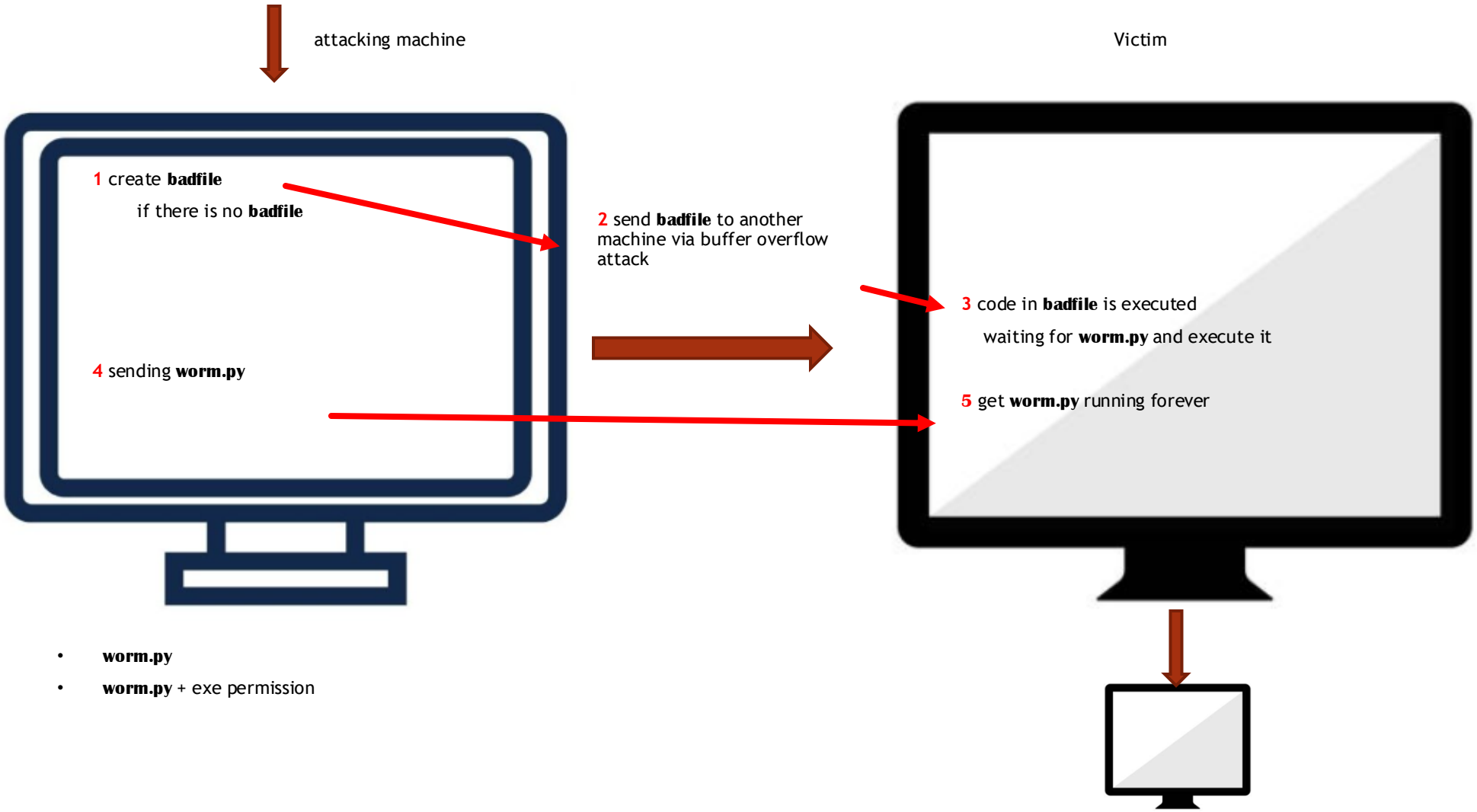
self-spreading

How to demonstrate the attack?

A virtual lab design by Wenliang Du at Syracuse university in 2021

Running on an Internet emulator

A simplified version writing in Python



- **worm.py**
- **worm.py** + exe permission

Characters of Du's **worm.py**

Exploit buffer overflow vulnerability	• with a crafted badfile containing shellcode and commands
Use shellcode	• to gain the control of victim's machines
Use commands	• Receive worm.py at victim's machines in Bash Shell sent by attacking machines
Ensures only one copy of worm.py is running	• keeps spreading from attack machine to victim machines and being executed • tests the exists of badfile to identify whether a victim has been infected • if true, it assume other worm.py is running and kills itself

Morris Worm lab

- Please follow https://seedsecuritylabs.org/Labs_20.04/Files/Morris_Worm/Morris_Worm.pdf

Download Ubuntu Image

Ubuntu 20.04 VM

If you prefer to create a SEED VM on your local computers, there are two ways to do that: (1) use a pre-built SEED VM; (2) create a SEED VM from scratch.

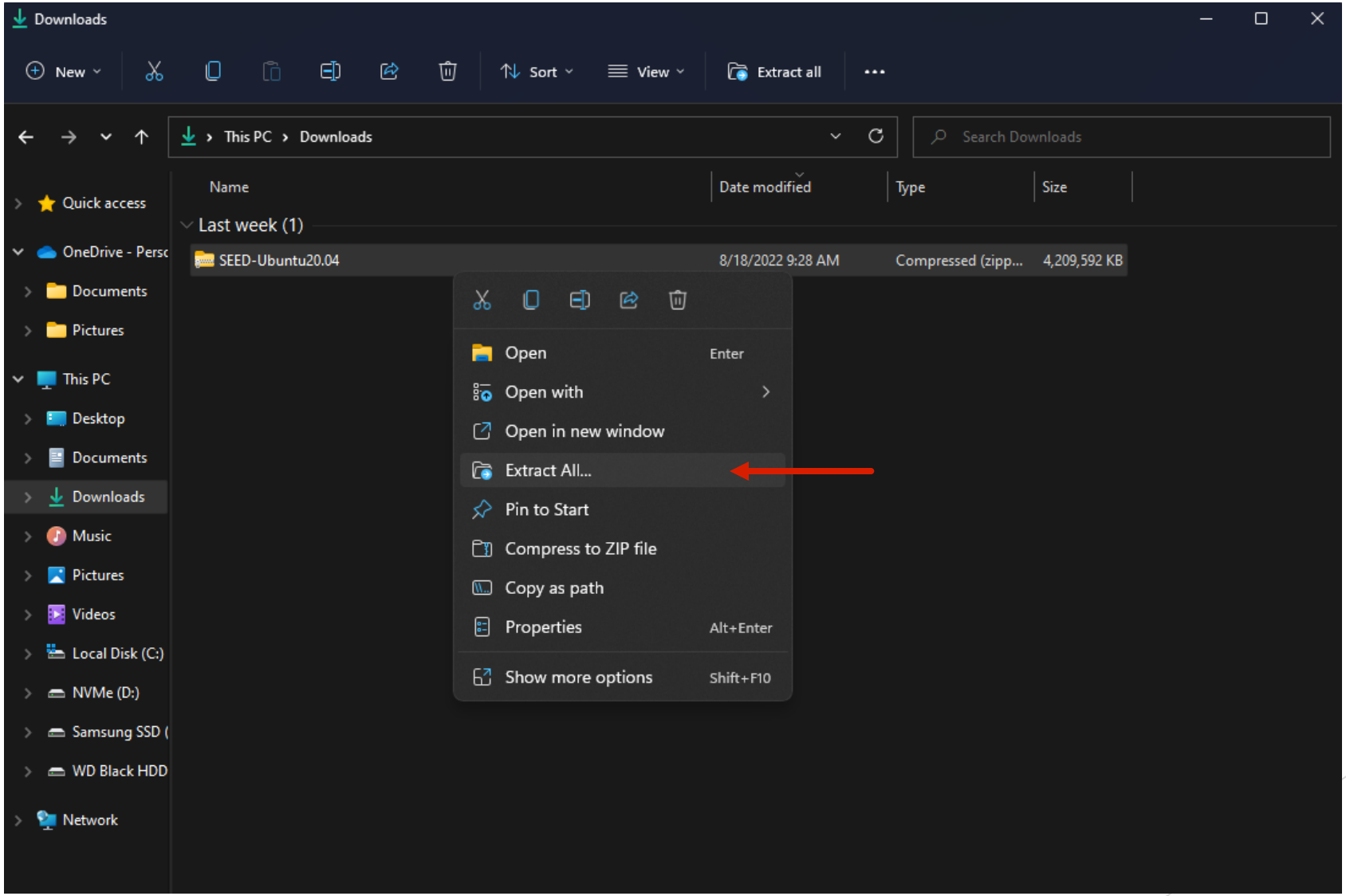
Approach 1: Use a pre-built SEED VM. We provide a pre-built SEED Ubuntu 20.04 VirtualBox image (`SEED-Ubuntu20.04.zip`, size: 4.0 GB), which can be downloaded from the following links.

- Google Drive
- DigitalOcean
- MD5 value: f3d2227c92219265679400064a0a1287
- VM Manual: follow this manual to install the VM on your computer

Approach 2: Build a SEED VM from scratch. The procedure to build the SEED VM used in Approach 1 is fully documented, and the code is open source. If you want to build your own SEED Ubuntu VM from scratch, you can use the following manual.

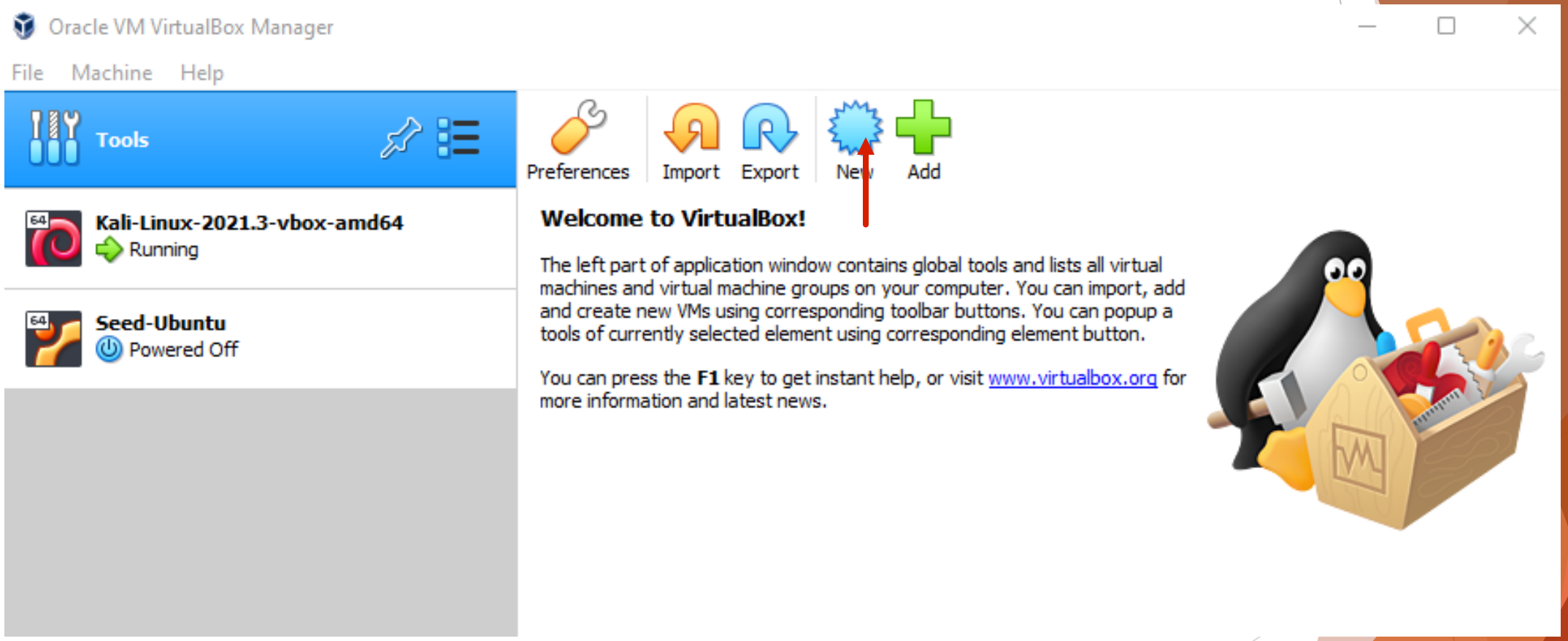
- How to build a SEED VM from scratch

Extract zip file



SAJIN SHIVDAS

Open Oracle VM VirtualBox Manager



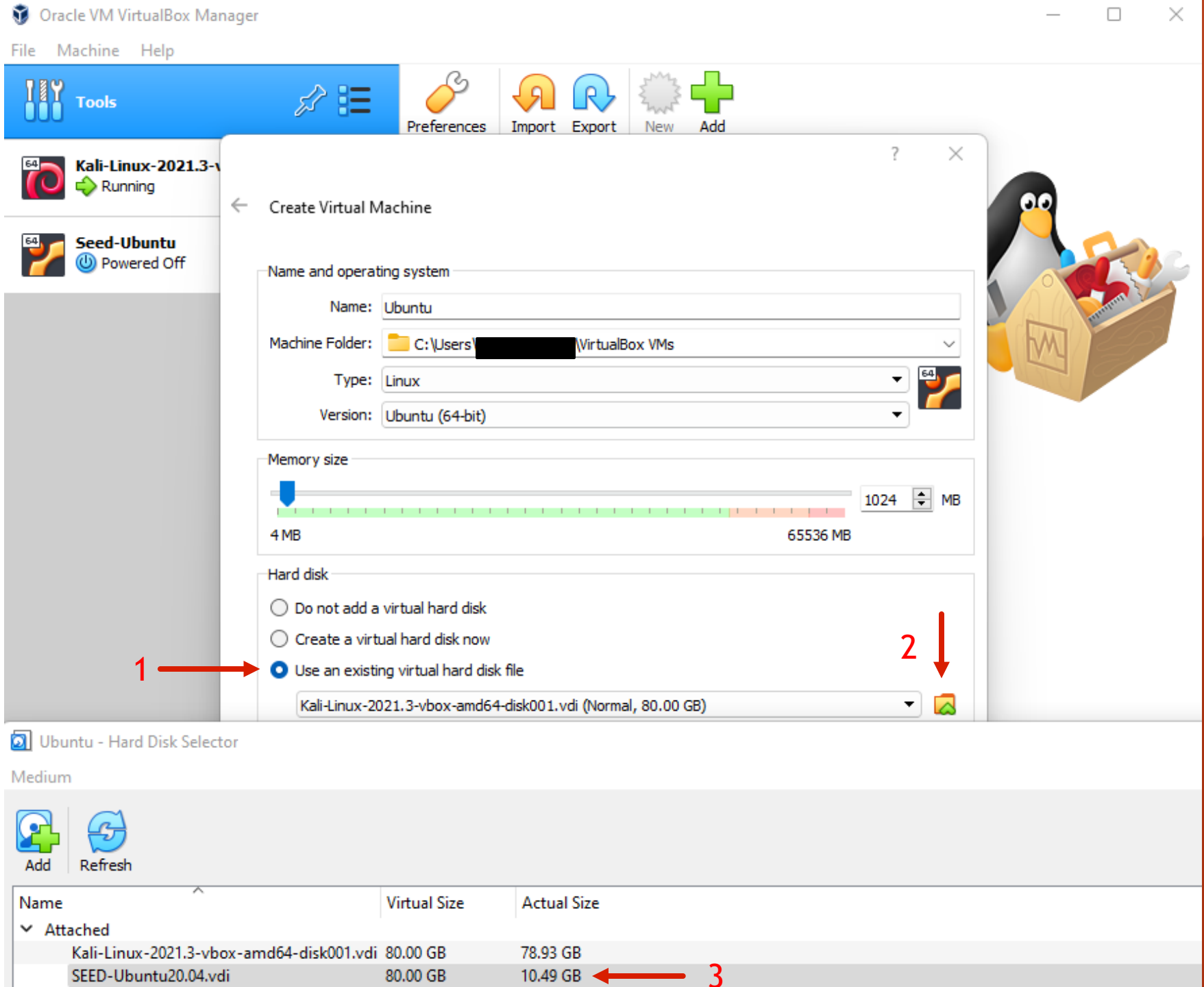
SAJIN SHIVDAS

1: Select *Use an existing virtual hard disk file.*

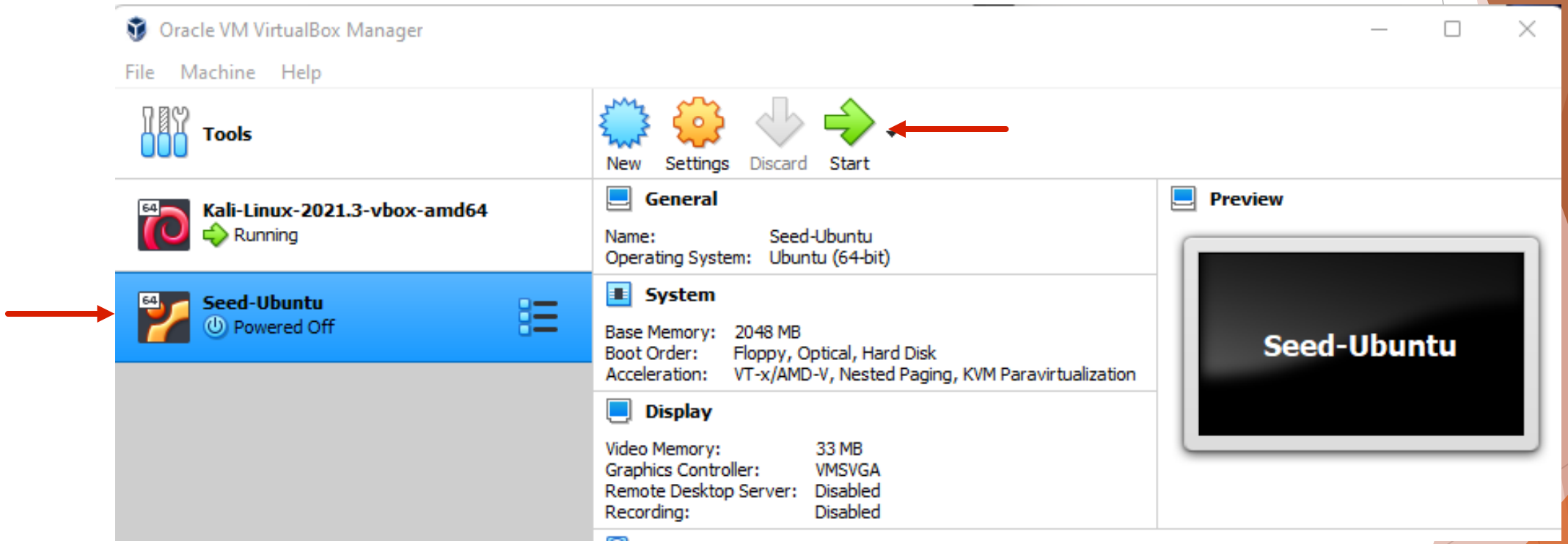
2: Select *folder icon to browse files.*

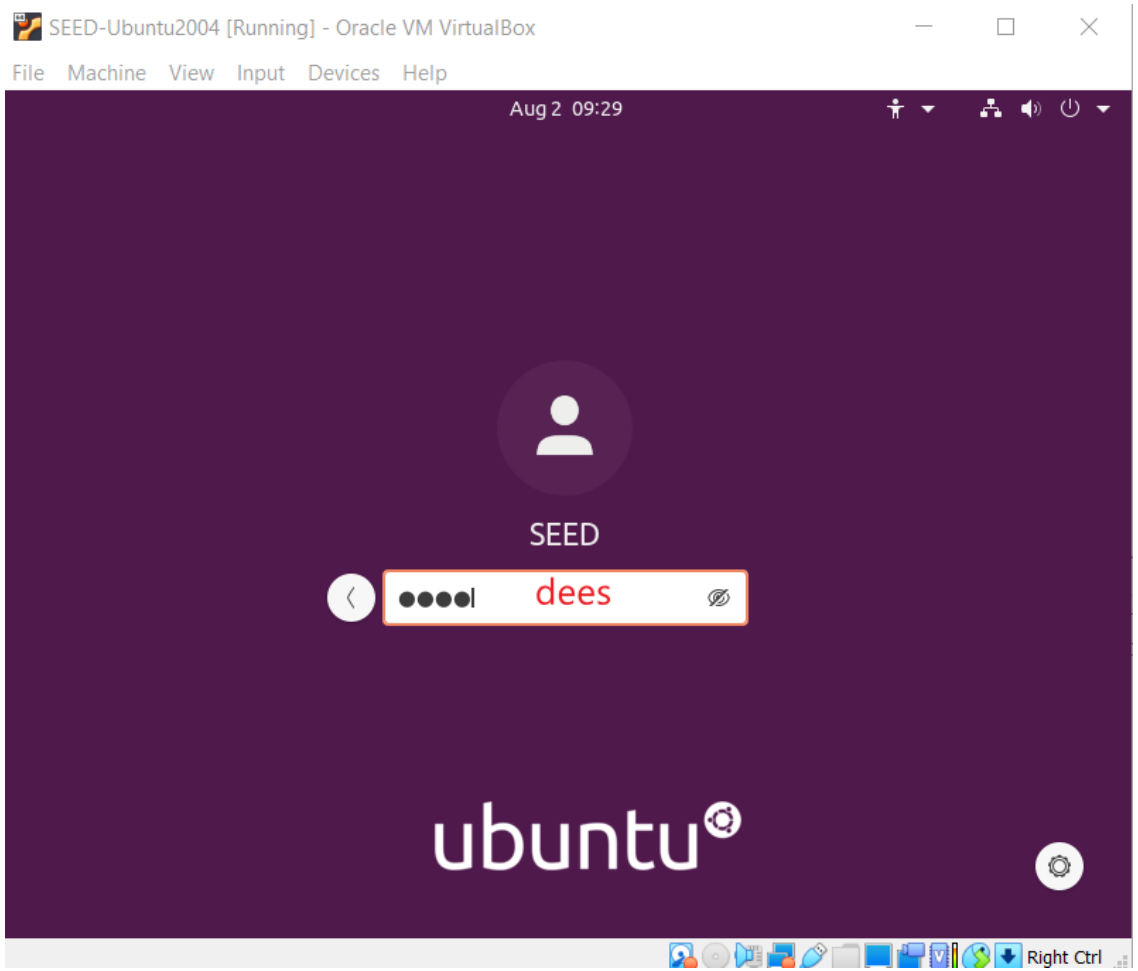
3: Point to *SEED-Ubuntu20.04.vdi*

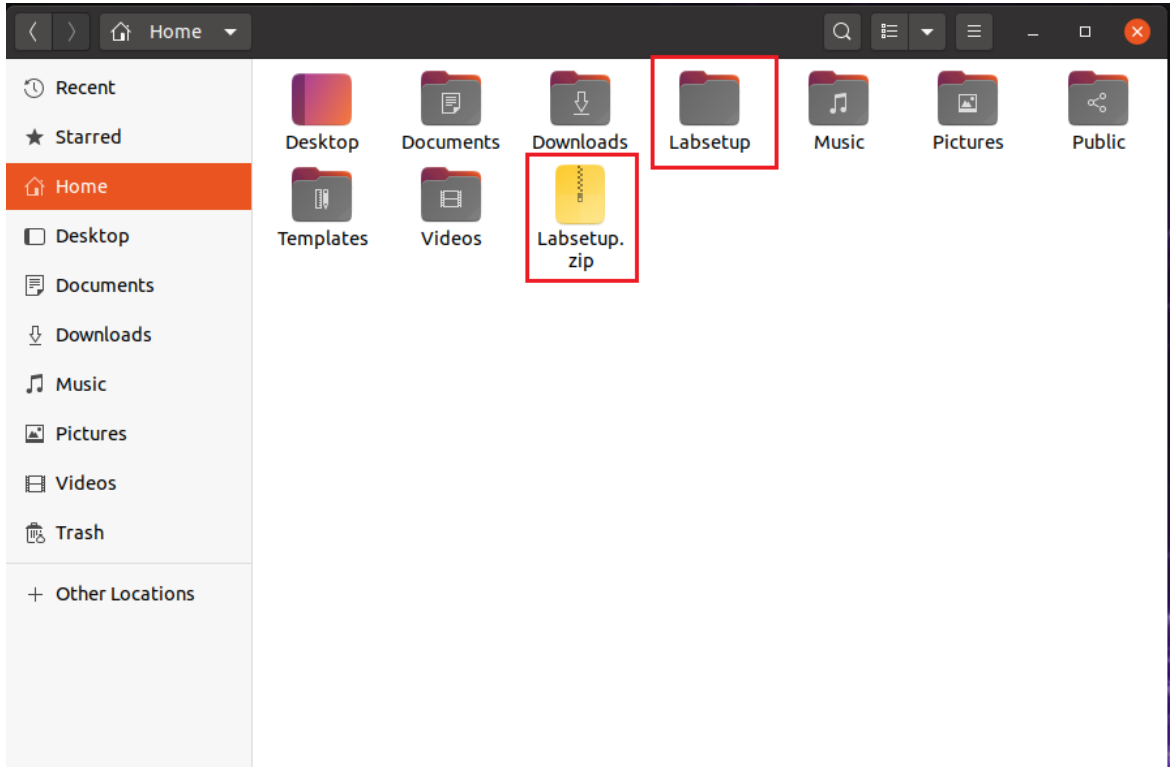
4: Click *Finish*



Click start to launch

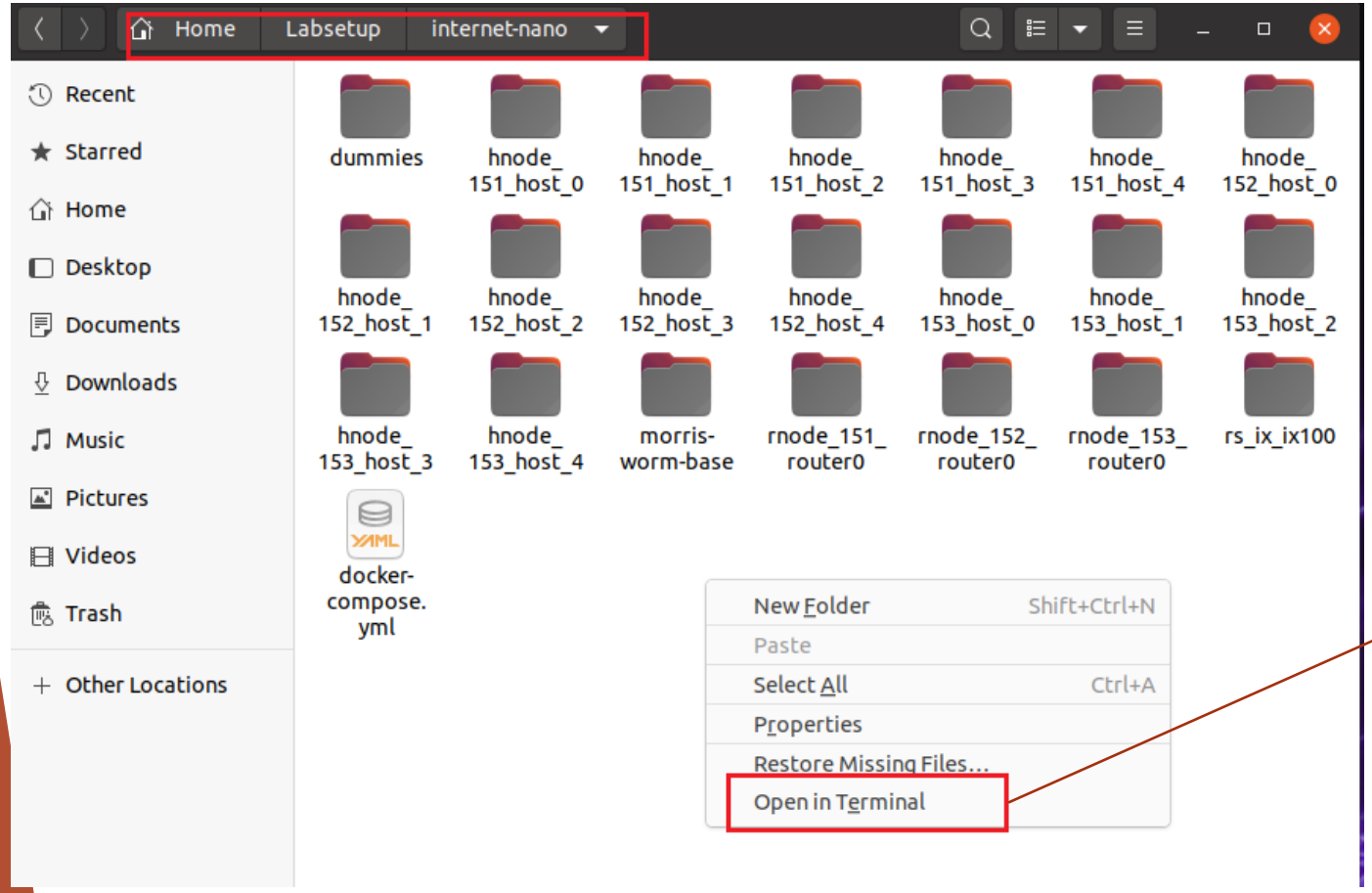


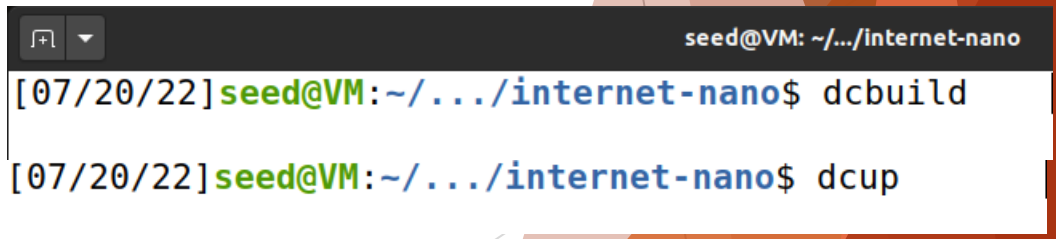

SEED-Ubuntu2004 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 2 09:29
SEED
dees
ubuntu
Right Ctrl



Download lab files

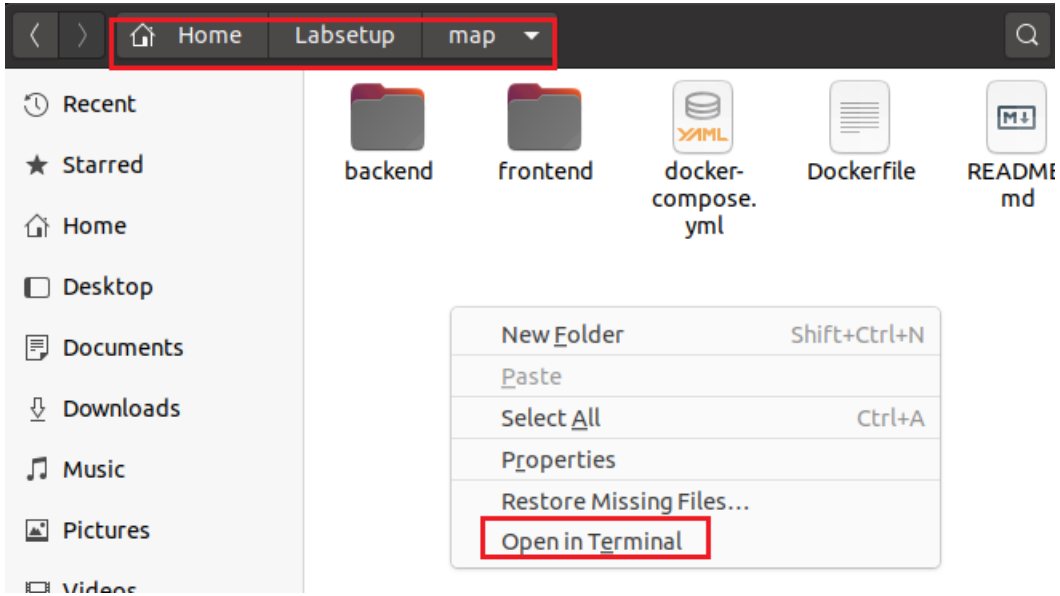
Start nano internet

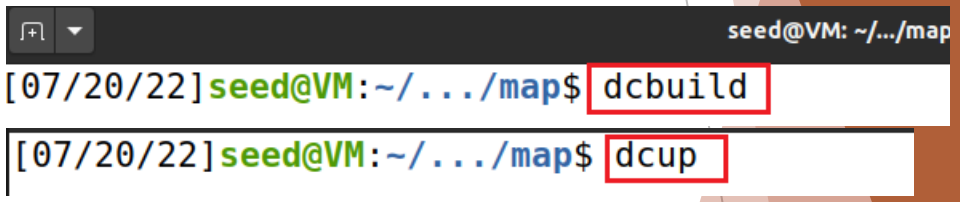




SAJIN SHIVDAS

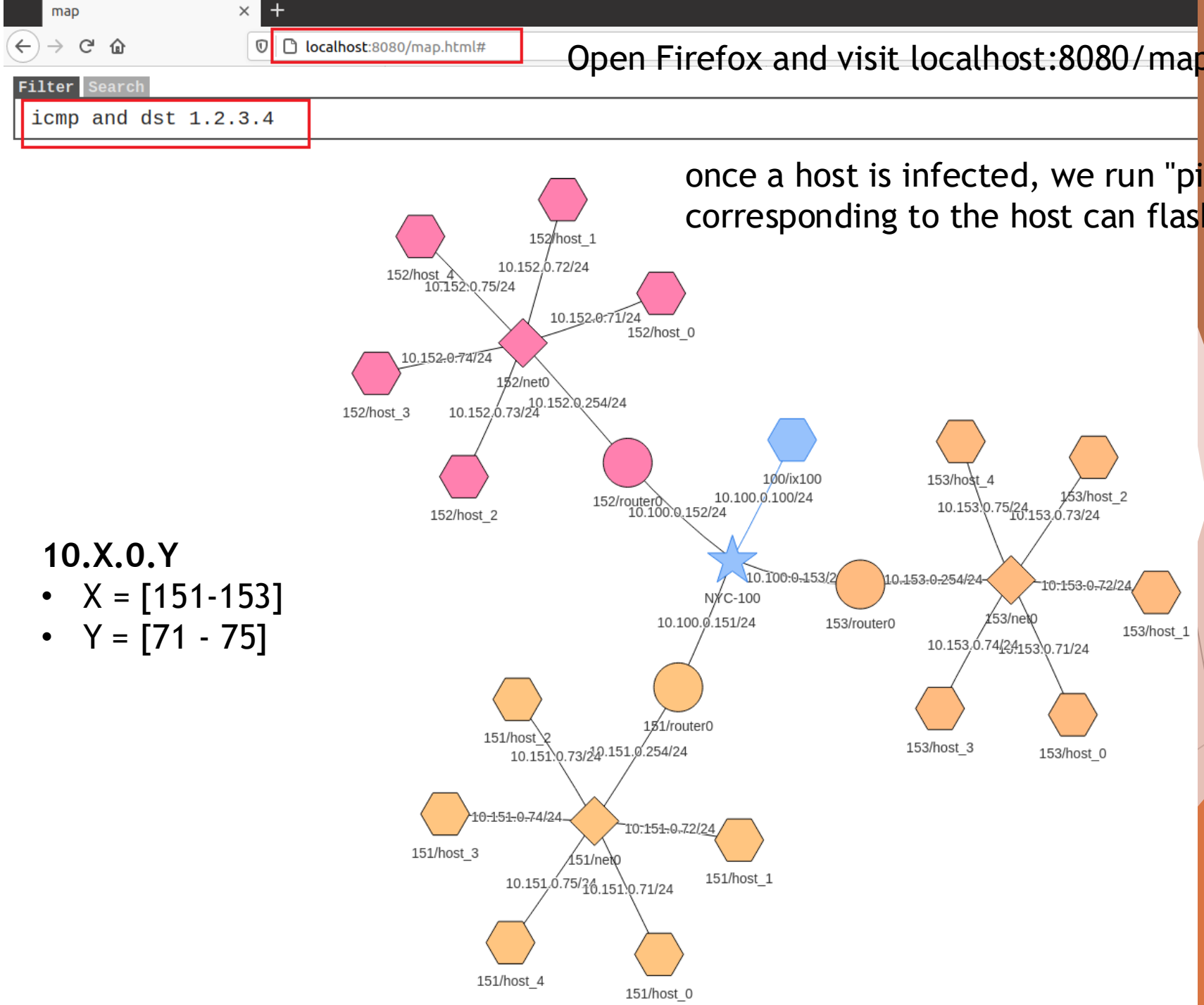
Start internet map for visualization





SAJIN SHIVDAS

Open Firefox and visit localhost:8080/map.html#



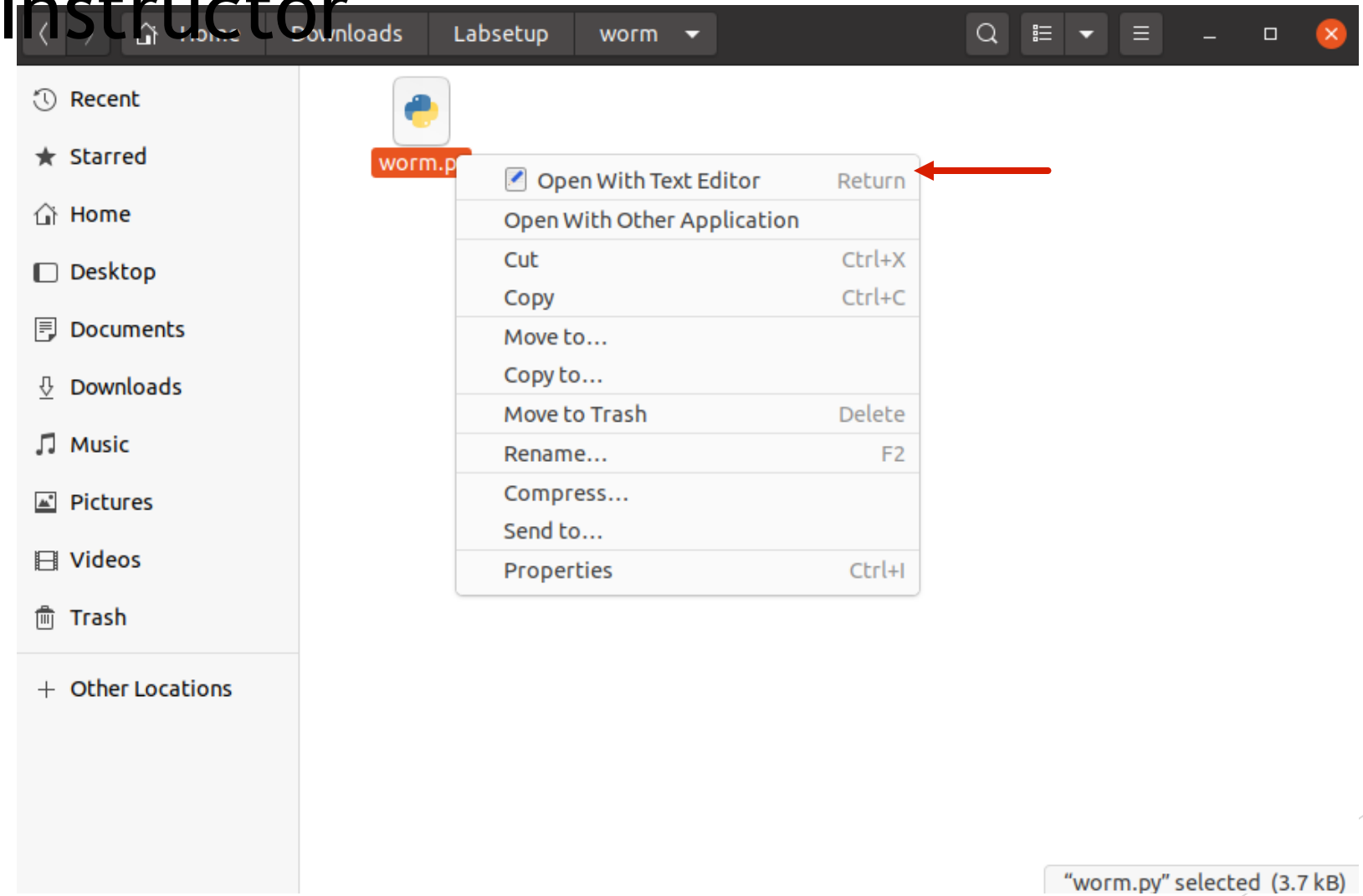
once a host is infected, we run "ping 1.2.3.4", so the r
corresponding to the host can flash on the map.

10.X.0.Y

- X = [151-153]
- Y = [71 - 75]

SAJIN SHIVDAS

Open **worm.py** and replace original code from Instructor



SAJIN SHIVDAS

Review attacking code

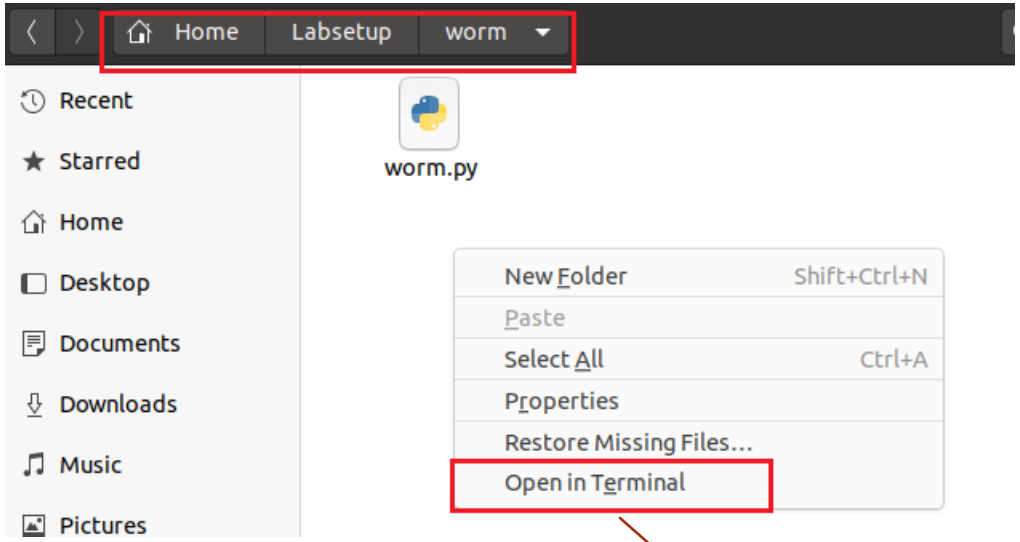
```
# Find the next victim (return an IP address).
# Check to make sure that the target is alive.
def getNextTarget():
    while True:
        a = randint(151, 153)
        b = randint(71, 75)
```

Make sure generate valid IPs for attacking automatically

```
# Check whether the current host is already infected with the worm
def isInfectedAlready():
    exists = os.path.exists('badfile')
    if exists:
        subprocess.Popen(["ping -q -i2 1.2.3.4"], shell=True)
        return True
    else:
        return False
```

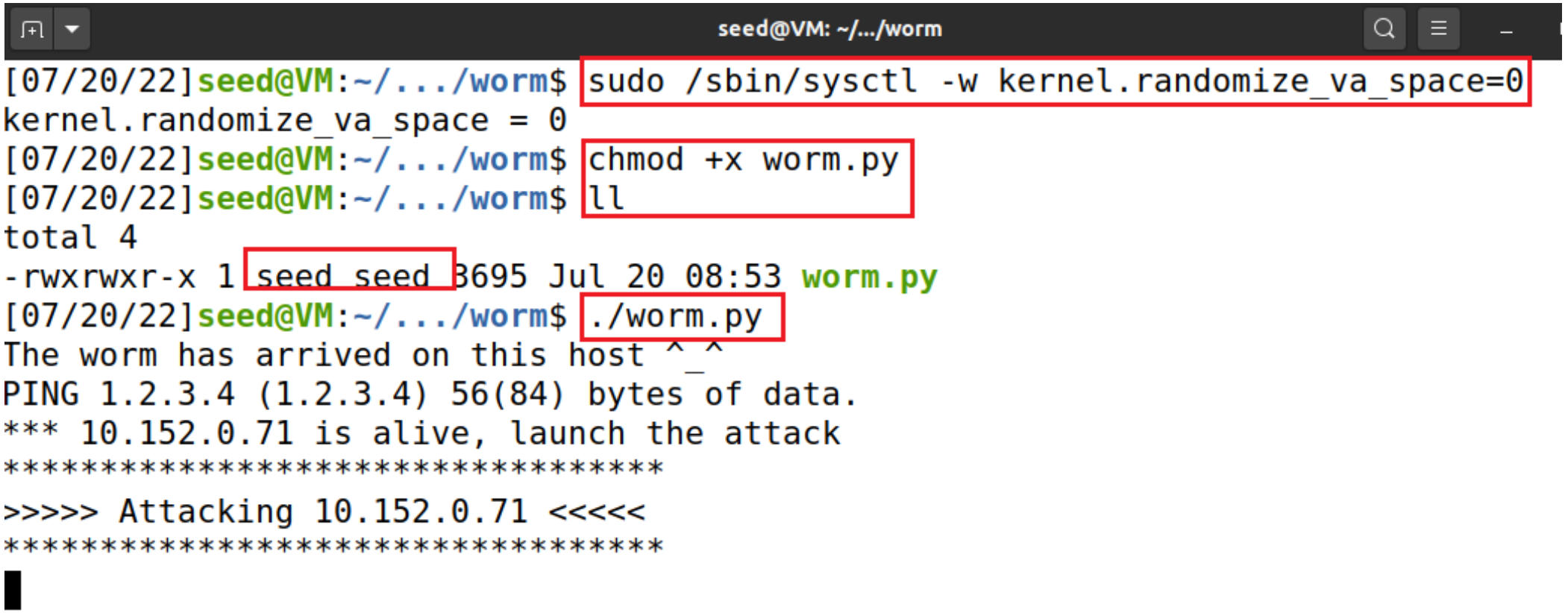
Visualizing attacked hosts even the **badfile** exists

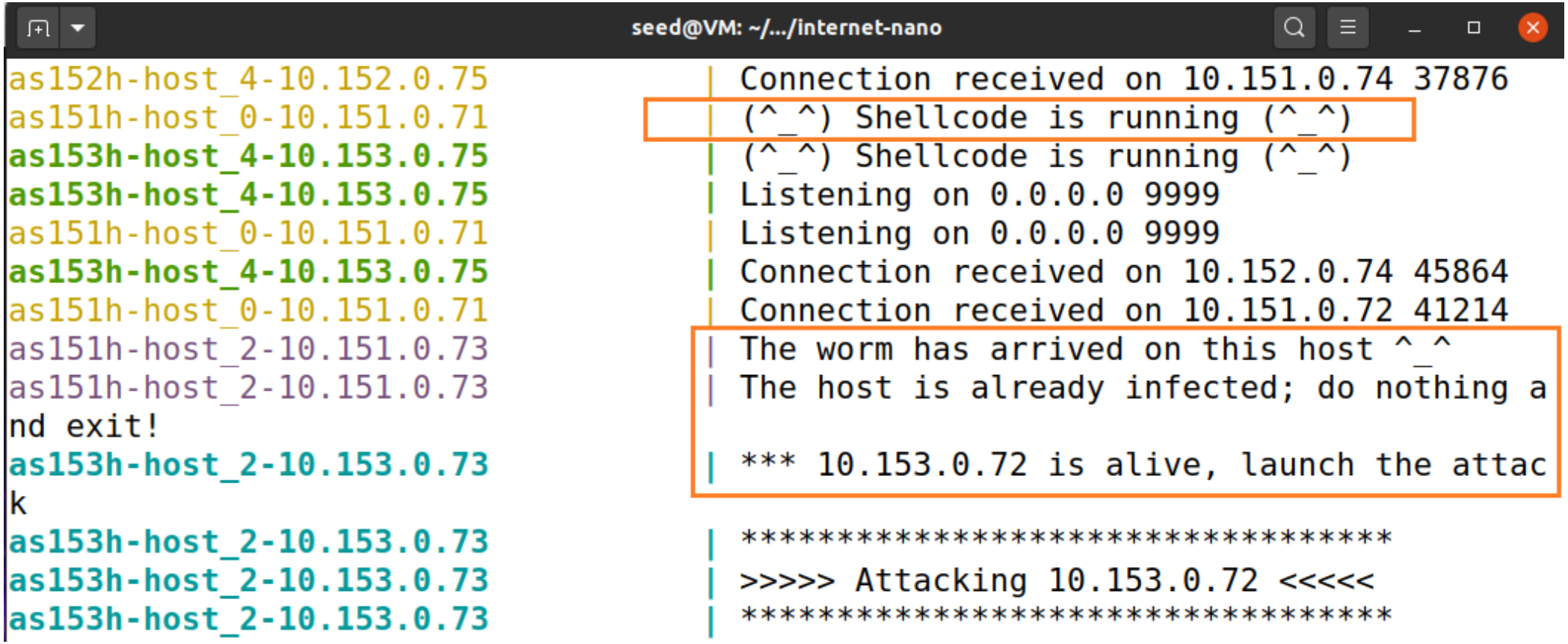
Start attack

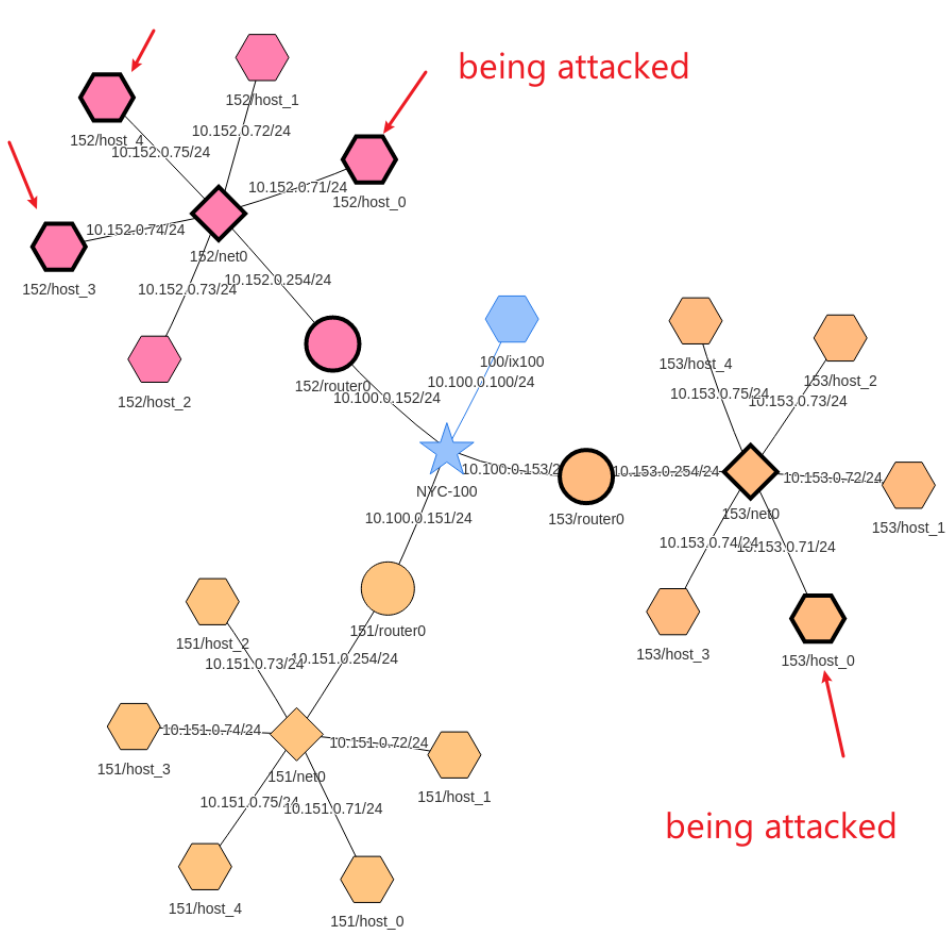


disable address randomization

SAJIN SHIVDAS


seed@VM: ~/.../worm
[07/20/22]seed@VM:~/.../worm$ sudo /sbin/sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
[07/20/22]seed@VM:~/.../worm$ chmod +x worm.py
[07/20/22]seed@VM:~/.../worm$ ll
total 4
-rwxrwxr-x 1 seed seed 3695 Jul 20 08:53 worm.py
[07/20/22]seed@VM:~/.../worm$ ./worm.py
The worm has arrived on this host ^_^
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
*** 10.152.0.71 is alive, launch the attack
****
>>>>> Attacking 10.152.0.71 <<<<<
****


seed@VM: ~/.../internet-nano
as152h-host_4-10.152.0.75 | Connection received on 10.151.0.74 37876
as151h-host_0-10.151.0.71 | (^_^) Shellcode is running (^_^)
as153h-host_4-10.153.0.75 | (^_^) Shellcode is running (^_^)
as153h-host_4-10.153.0.75 | Listening on 0.0.0.0 9999
as151h-host_0-10.151.0.71 | Listening on 0.0.0.0 9999
as153h-host_4-10.153.0.75 | Connection received on 10.152.0.74 45864
as151h-host_0-10.151.0.71 | Connection received on 10.151.0.72 41214
as151h-host_2-10.151.0.73 | The worm has arrived on this host ^_^
as151h-host_2-10.151.0.73 | The host is already infected; do nothing a
nd exit!
as153h-host_2-10.153.0.73 | *** 10.153.0.72 is alive, launch the attac
k
as153h-host_2-10.153.0.73 | **********
as153h-host_2-10.153.0.73 | >>>>> Attacking 10.153.0.72 <<<<<
as153h-host_2-10.153.0.73 | **********

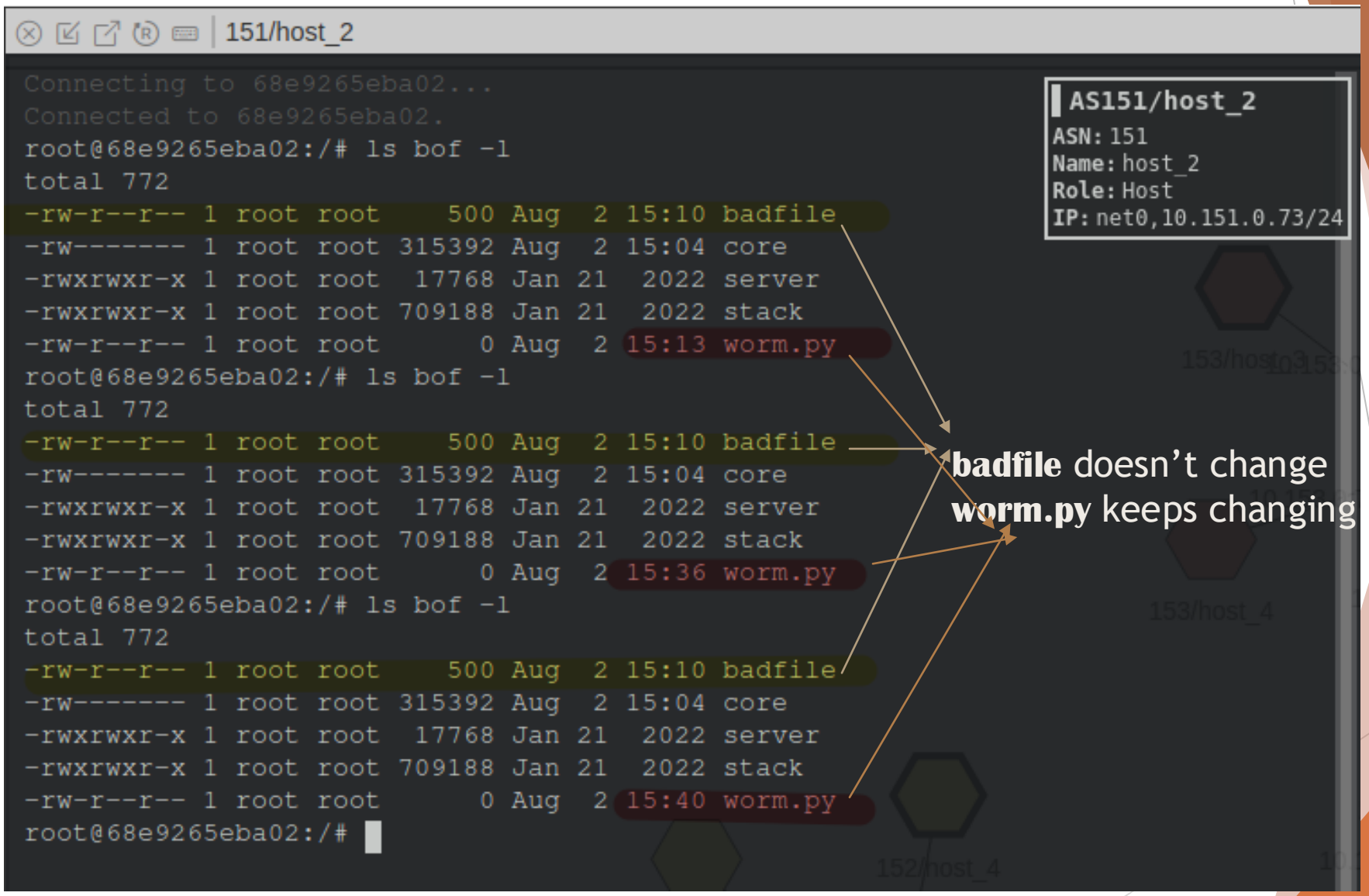

being attacked
152/host_1
152/host_4
10.152.0.72/24
10.152.0.75/24
10.152.0.71/24
152/host_0
10.152.0.74/24
152/net0
152/host_3
10.152.0.73/24
10.152.0.254/24
152/host_2
152/router0
100/ix100
10.100.0.100/24
10.100.0.152/24
NYC-100
10.100.0.151/24
153/router0
153/host_4
153/host_2
10.153.0.75/24
10.153.0.73/24
10.153.0.254/24
10.153.0.72/24
153/host_1
153/net0
10.153.0.74/24
10.153.0.71/24
153/host_3
153/host_0
151/router0
151/host_2
10.151.0.73/24
10.151.0.254/24
10.151.0.74/24
10.151.0.72/24
151/host_3
151/net0
151/host_1
10.151.0.75/24
10.151.0.71/24
151/host_4
151/host_0
being attacked

```
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

net0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.151.0.74  netmask 255.255.255.0  broadcast 10.151.0.255
        ether 02:42:0a:97:00:4a  txqueuelen 1000  (Ethernet)
        RX packets 80  bytes 9269 (9.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@76b886ec9f0a:/# ls  bof -l
total 716
-rwxrwxr-x 1 root root  17768 Jan 21  2022 server
-rwxrwxr-x 1 root root 709188 Jan 21  2022 stack
root@76b886ec9f0a:/#
```

AS151/host_3
ASN: 151
Name: host_3
Role: Host
IP: net0,10.151.0.74/24

153/host_3

153/host_4



SAJIN SHIVDAS