National College of Ireland

Configuration Manual

MSc Internship
Cybersecurity

Rhea Bonnerji
18176887

School of Computing
National College of Ireland

Supervisor: Niall Heffernan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……. Rhea Bonnerji……

Student ID: ……18176887……..……

Programme: …MSc Cybersecurity…… **Year:** …2019-2020..

Module: ……MSc Internship……..……

Lecturer: ……Niall Heffernan……..……

Submission Due Date: ……17/08/2020……..……

Project Title: An approach to enhance low-interaction honeypots by enabling them to detect spoofing attacks via network analysis

Word Count: …849…… **Page Count:** ……19……..……..……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.
I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

Signature: ……Rhea Bonnerji……

Date: ……17/08/2020……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	

Date:	
Penalty Applied (if applicable):	

Configuration Manual

Rhea Bonnerji
Student ID:18176887

1 Introduction

This document acts as a manual for replicating the proposed model's setup to predict and log IP, ARP and DNS spoofing attacks to enhance the functionalities of low interaction honeypots. For this research, we created our own network in which three machines are connected (2 Kali Linux VMs and a Windows10 x86 VM) and the traffic was captured over the first Ethernet interface eth0 using tshark followed by analysis of the pcap files using different python scripts and tools. Once the experiment has been run, our set-up will be tested against our preconfigured honeynet's performance in capturing spoofed traffic to evaluate our intelligence mechanism's performance.

2 System Specification

For this project, the local host machine was running the hypervisor VMWare. It has a user-friendly interface and supports multiple OS where we have the freedom to allocate how much hardware usage we want in its configuration. We have three machines running in our VM.

Machine 1:
Operating System: Windows 10x86 (32 bit)
Memory allocated (RAM): 4GB
Network Adapter: VMnet10
Purpose: Packet generation

Machine 2:
Operating System: Kali Linux x64
Memory allocated (RAM): 4GB
Network Adapter 1: VMnet10
Network Adapter 2: NAT Network
Purpose: Hosting the honeynet consisting of the honeypots HoneyPy and SNARE, capturing network traffic using our own script and running the IP, ARP and DNS spoofing detection scripts.

Machine 3:
Operating System: Kali Linux x64
Memory allocated (RAM): 2GB
Network Adapter 1: VMnet10
Network Adapter 2: NAT Network
Purpose: Hosting the fake DNS server.

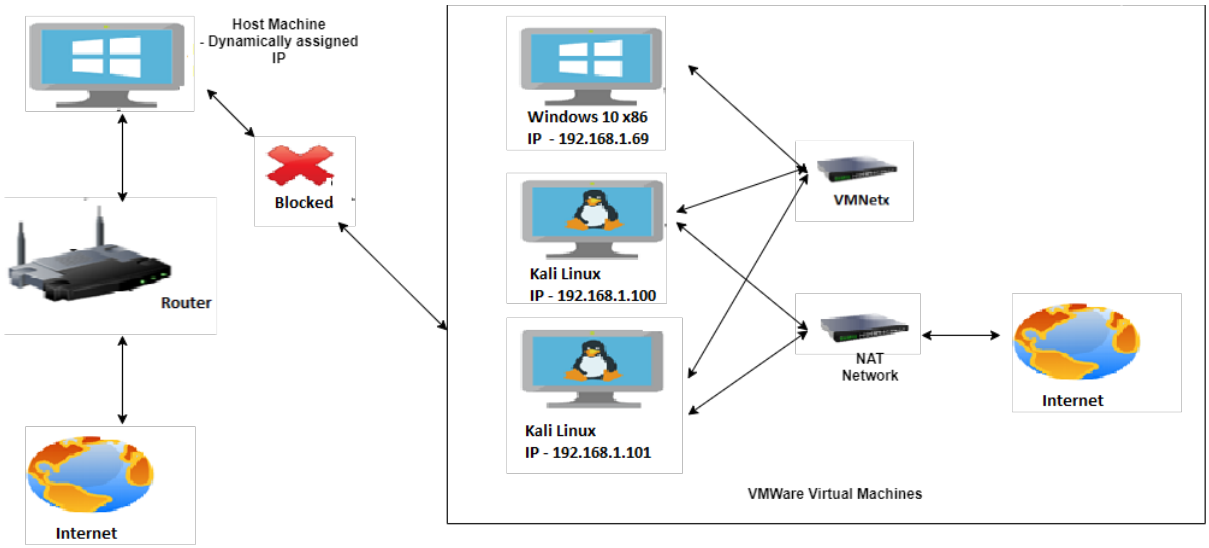


Figure 1: Network Diagram

3 Tools and Technologies

Machine 1: Windows 10x86 (32 bit)

- HyenaeFE – The packet generator

Machine 2: Kali Linux x64

- Python 2.7.17
- Python 3.8.5
- Scapy 2.4.3
- HoneyPy
- SNARE
- getmac

Machine 2: Kali Linux x64

- Ettercap

4 Implementation

4.1 Downloading HyenaeFE which is the packet generator.

HyenaeFE was installed using the their official link on SourceForge at
https://sourceforge.net/projects/hyenaefe/

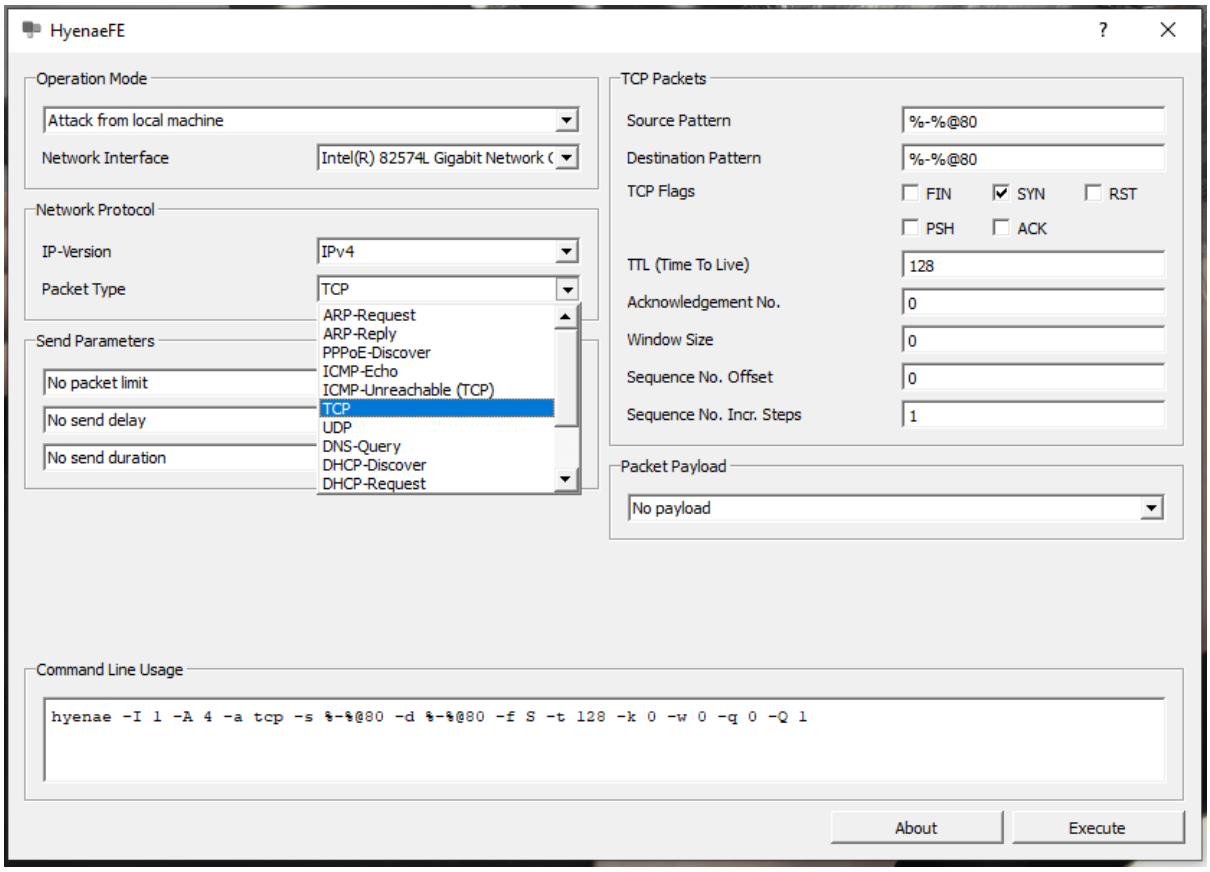


Figure 2: HyenaeFE interface displaying the multiple packet types it can generate

4.2 Checking if the packet generator works by sending TCP packets and capturing them using Wireshark in the Kali VM.

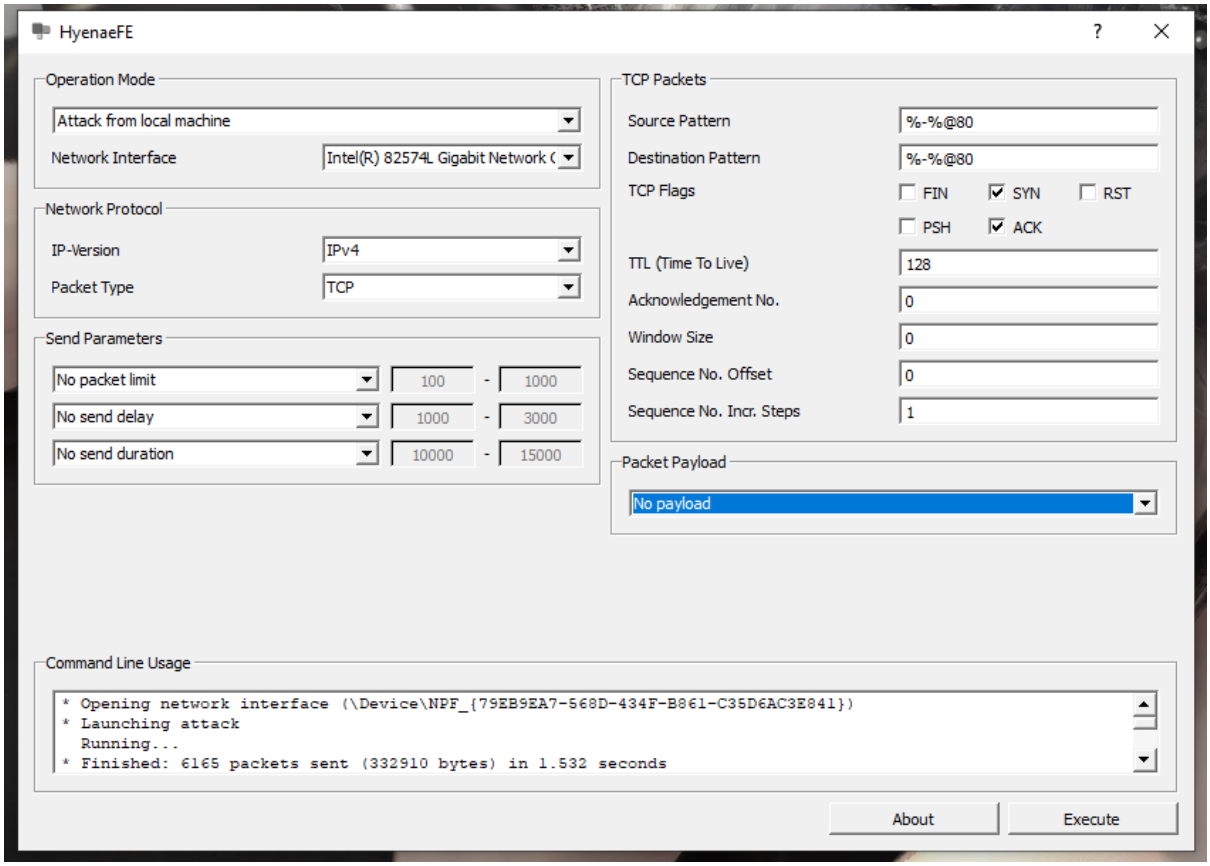


Figure 3: Sending random TCP packets on port 80

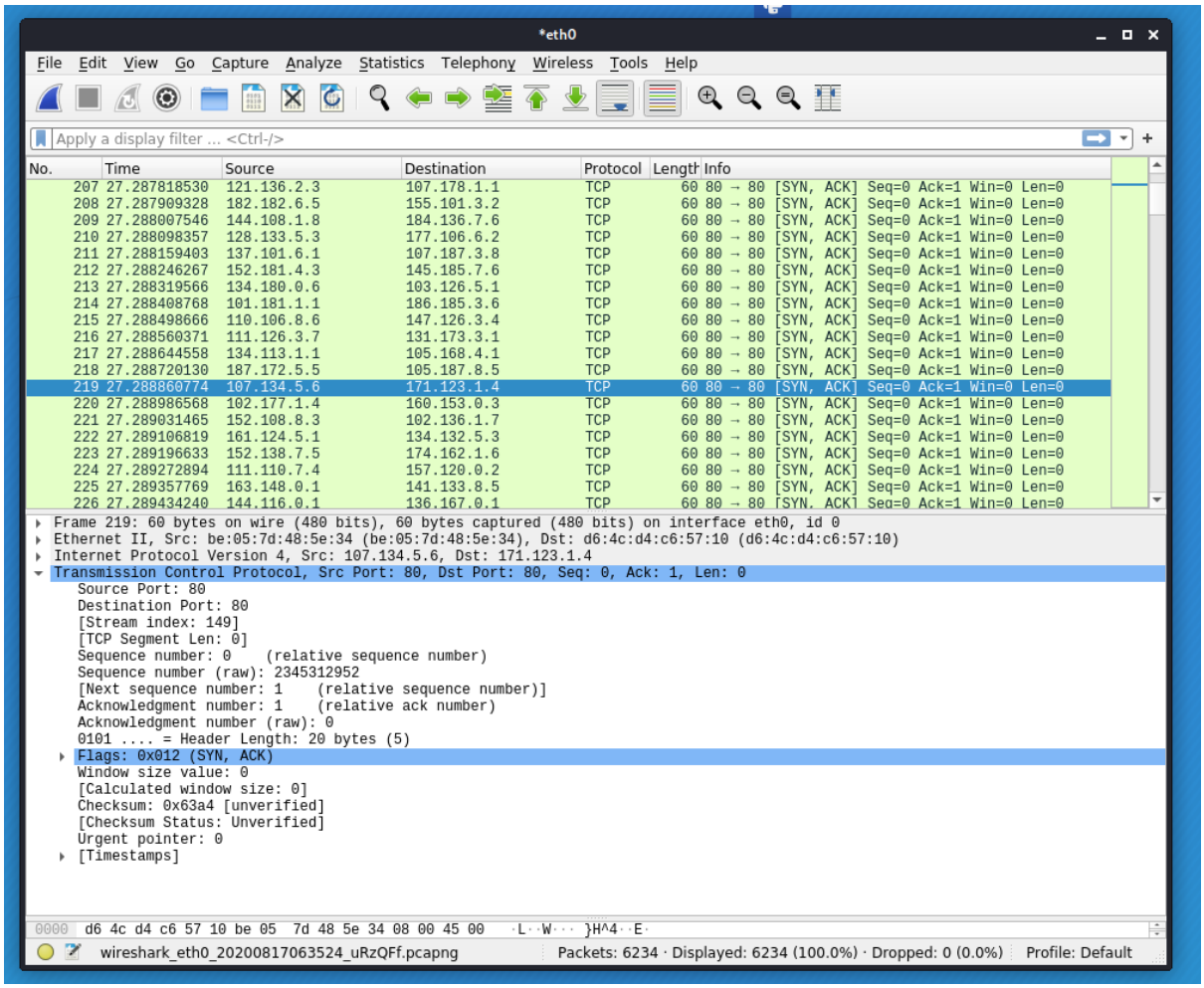


Figure 4: Wireshark is capturing all the packets sent by HyenaeFE

4.3 Installing and Configuring HoneyPy

HoneyPy was installed using the their official link available on github at https://github.com/foospidy/HoneyPy.

The services that are enabled on the HoneyPy honeypot and which port and plugin to use for each service is determined in the services.cfg file. We have not used the default service profile but used the services.linux.profile (available in the HoneyPy/etc/profiles) for the services.cfg file by copying its contents into it because of the extensive list of services present there.

As no modifications have been made on this file, screenshots have not been provided but the file can be found using the path HoneyPy/etc/profiles.

Next, we redirect the lower ports to higher ports because the lower ports generally have system services running on them. This has been achieved by writing a script called **ipfix.sh** in the **ipt-kit folder**.

1. ipfix.sh

/home/malware/Desktop/HoneyPy/ipt-kit/ipfix.sh - Mousepad

File Edit Search View Document Help

```
#!/bin/bash
 cd /home/malware/Desktop/HoneyPy/ipt-kit/
 sudo ./ipt_set_tcp 7 10007
 sudo ./ipt_set_tcp 19 10019
 sudo ./ipt_set_tcp 20 10020
 sudo ./ipt_set_tcp 21 10021
 sudo ./ipt_set_tcp 22 10022
 sudo ./ipt_set_tcp 23 10023
 sudo ./ipt_set_tcp 25 10025
 sudo ./ipt_set_tcp 42 10042
 sudo ./ipt_set_tcp 43 10043
 sudo ./ipt_set_tcp 49 10049
 sudo ./ipt_set_tcp 53 10053
 sudo ./ipt_set_tcp 67 10067
 sudo ./ipt_set_tcp 68 10068
 sudo ./ipt_set_tcp 69 10069
 sudo ./ipt_set_tcp 70 10070
 sudo ./ipt_set_tcp 79 10079
 sudo ./ipt_set_tcp 80 10080
 sudo ./ipt_set_tcp 110 10110
 sudo ./ipt_set_tcp 113 10113
 sudo ./ipt_set_tcp 119 10119
 sudo ./ipt_set_tcp 123 10123
 sudo ./ipt_set_tcp 135 10135
 sudo ./ipt_set_tcp 137 10137
 sudo ./ipt_set_tcp 138 10138
 sudo ./ipt_set_tcp 139 10139
 sudo ./ipt_set_tcp 143 10143
 sudo ./ipt_set_tcp 161 10161
 sudo ./ipt_set_tcp 162 10162
 sudo ./ipt_set_tcp 177 10177
 sudo ./ipt_set_tcp 179 10179
 sudo ./ipt_set_tcp 201 10201
 sudo ./ipt_set_tcp 264 10264
 sudo ./ipt_set_tcp 318 10318
 sudo ./ipt_set_tcp 381 10381
 sudo ./ipt_set_tcp 382 10382
 sudo ./ipt_set_tcp 383 10383
 sudo ./ipt_set_tcp 389 10389
 sudo ./ipt_set_tcp 411 10411
 sudo ./ipt_set_tcp 412 10412
 sudo ./ipt_set_tcp 443 10443
 sudo ./ipt_set_tcp 445 10445
 sudo ./ipt_set_tcp 464 10464
 sudo ./ipt_set_tcp 465 10465
 sudo ./ipt_set_tcp 497 10497
```

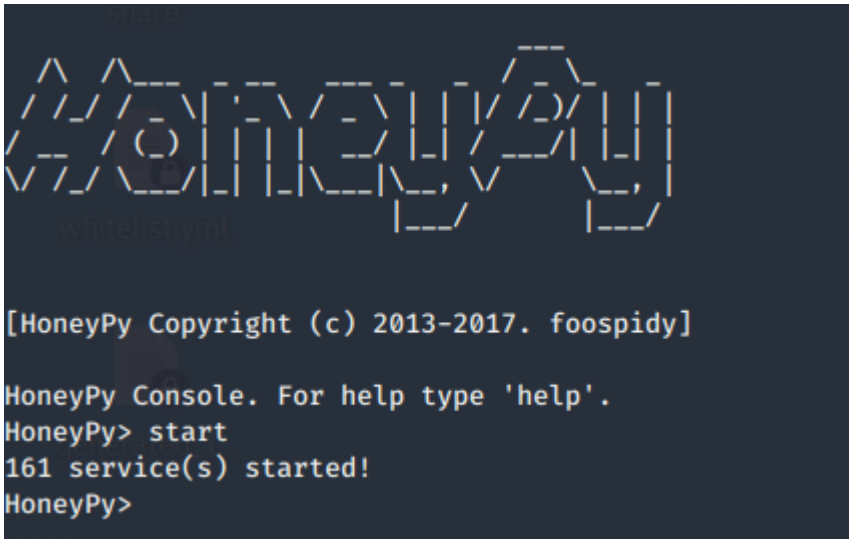
```
sudo ./ipt_set_tcp 497 10497
sudo ./ipt_set_tcp 500 10500
sudo ./ipt_set_tcp 512 10512
sudo ./ipt_set_tcp 513 10513
sudo ./ipt_set_tcp 514 10514
sudo ./ipt_set_tcp 515 10515
sudo ./ipt_set_tcp 520 10520
sudo ./ipt_set_tcp 521 10521
sudo ./ipt_set_tcp 540 10540
sudo ./ipt_set_tcp 546 10546
sudo ./ipt_set_tcp 554 10554
sudo ./ipt_set_tcp 547 10547
sudo ./ipt_set_tcp 560 10560
sudo ./ipt_set_tcp 563 10563
sudo ./ipt_set_tcp 587 10587
sudo ./ipt_set_tcp 591 10591
sudo ./ipt_set_tcp 593 10593
sudo ./ipt_set_tcp 631 10631
sudo ./ipt_set_tcp 636 10636
sudo ./ipt_set_tcp 639 10639
sudo ./ipt_set_tcp 646 10646
sudo ./ipt_set_tcp 691 10691
sudo ./ipt_set_tcp 860 10860
sudo ./ipt_set_tcp 873 10873
sudo ./ipt_set_tcp 902 10902
sudo ./ipt_set_tcp 989 10989
sudo ./ipt_set_tcp 990 10990
sudo ./ipt_set_tcp 993 10993
sudo ./ipt_set_tcp 995 10995
cd ..
sudo ./Honey.py
```

4.4 Running HoneyPy

Commands to start HoneyPy

```
malware@kali:~$ cd Desktop
malware@kali:~/Desktop$ cd HoneyPy
malware@kali:~/Desktop/HoneyPy$ cd ipt-kit
malware@kali:~/Desktop/HoneyPy/ipt-kit$ sudo ./ipfix.sh
[sudo] password for malware:
```

HoneyPy is running



4.5 Looking at HoneyPy's logging abilities

First, we will try to send some TCP packets to see what HoneyPy logs. So, for this, we will set the destination IP on the packet generator to be the Kali VM containing HoneyPy. We find that the honeypot doesn't capture and log any of this, thus not generating any log file. So, we try to SSH and FTP into the machine and we also send an HTTP request to see if it logs any of this. Upon doing this, we find that log files have been generated logging the SSH, FTP and HTTP request attempts. Therefore, we can conclude that HoneyPy fails to capture any of the spoofing attack attempts.

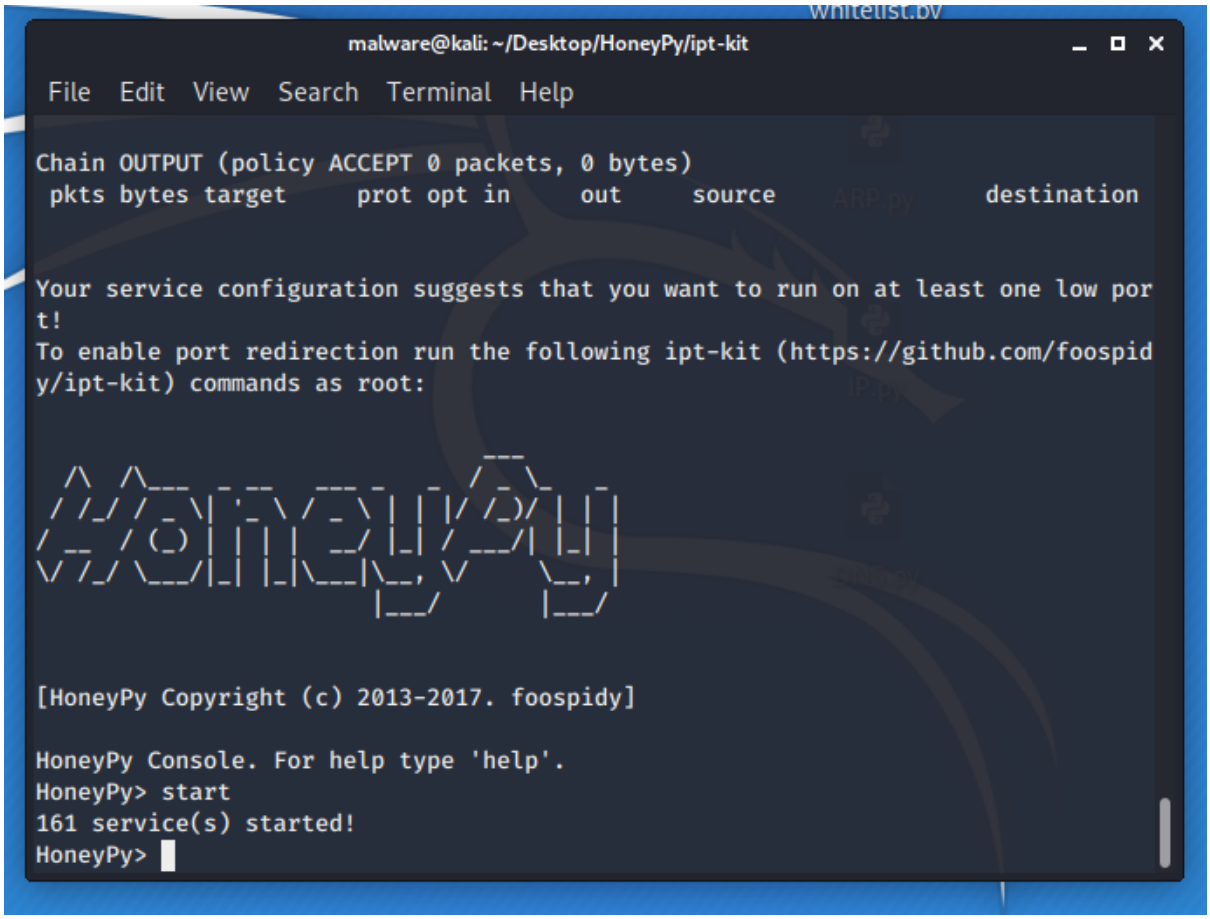


Figure 5: HoneyPy has been started.

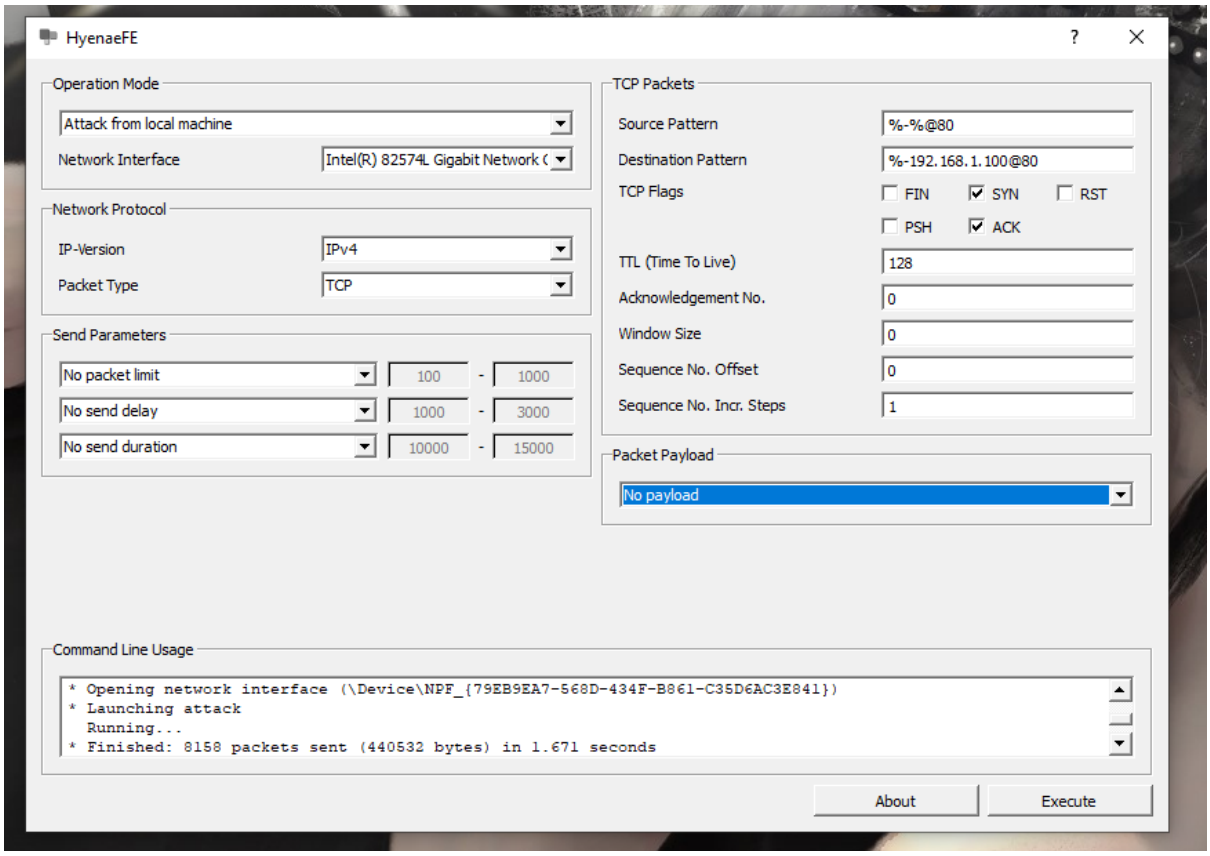


Figure 6: Sending TCP packets by fixing the destination IP to the Kali VM containing HoneyPy.

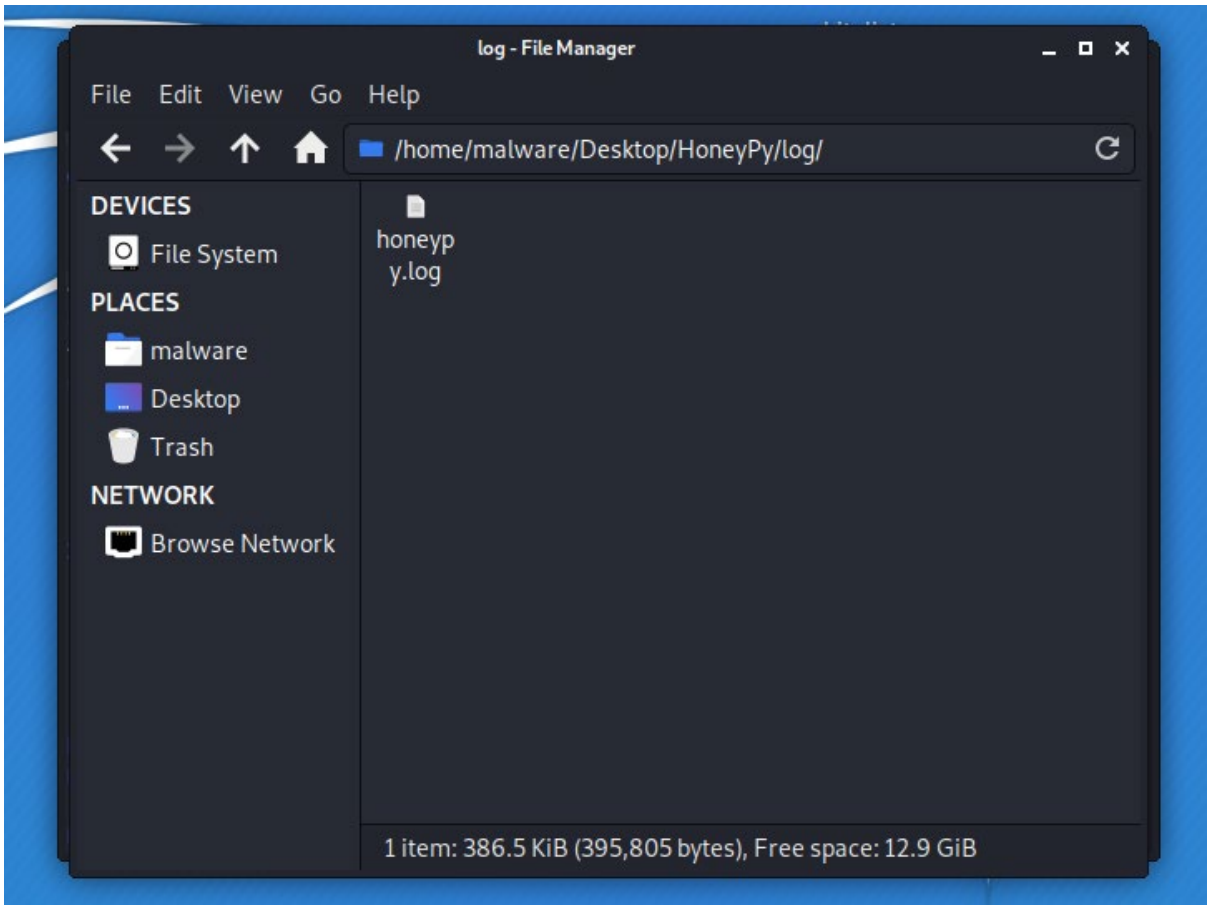


Figure 7: No logs have been generated for the sent TCP packets.

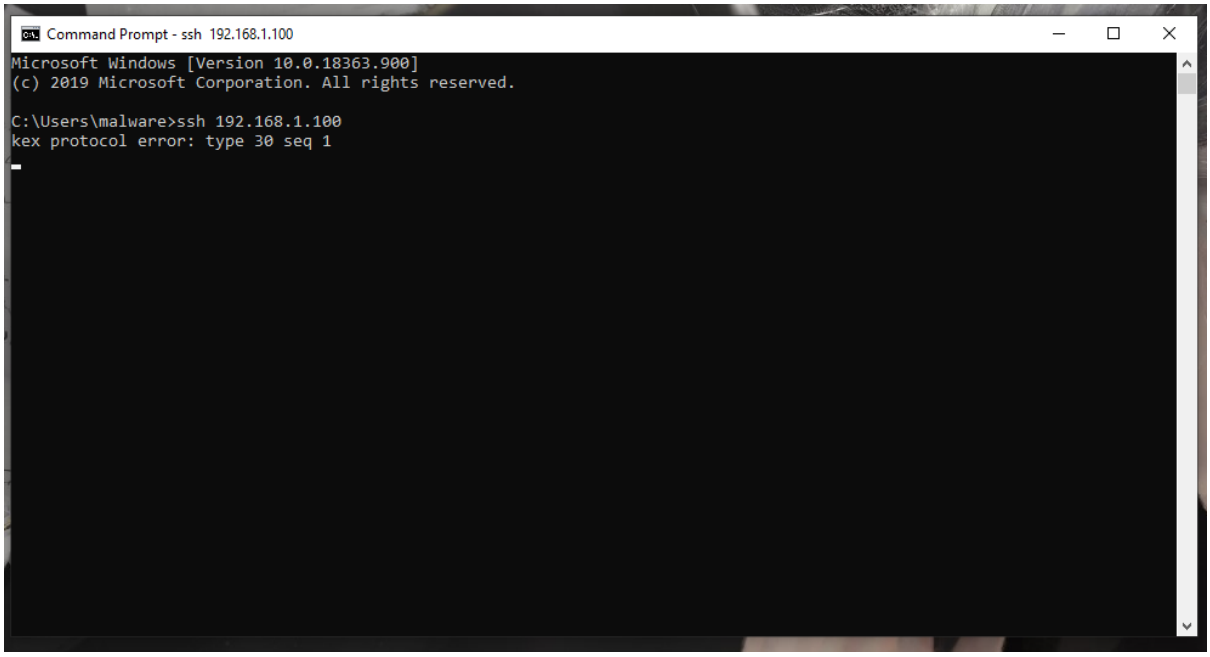


Figure 8: SSH into the Kali VM containing HoneyPy.

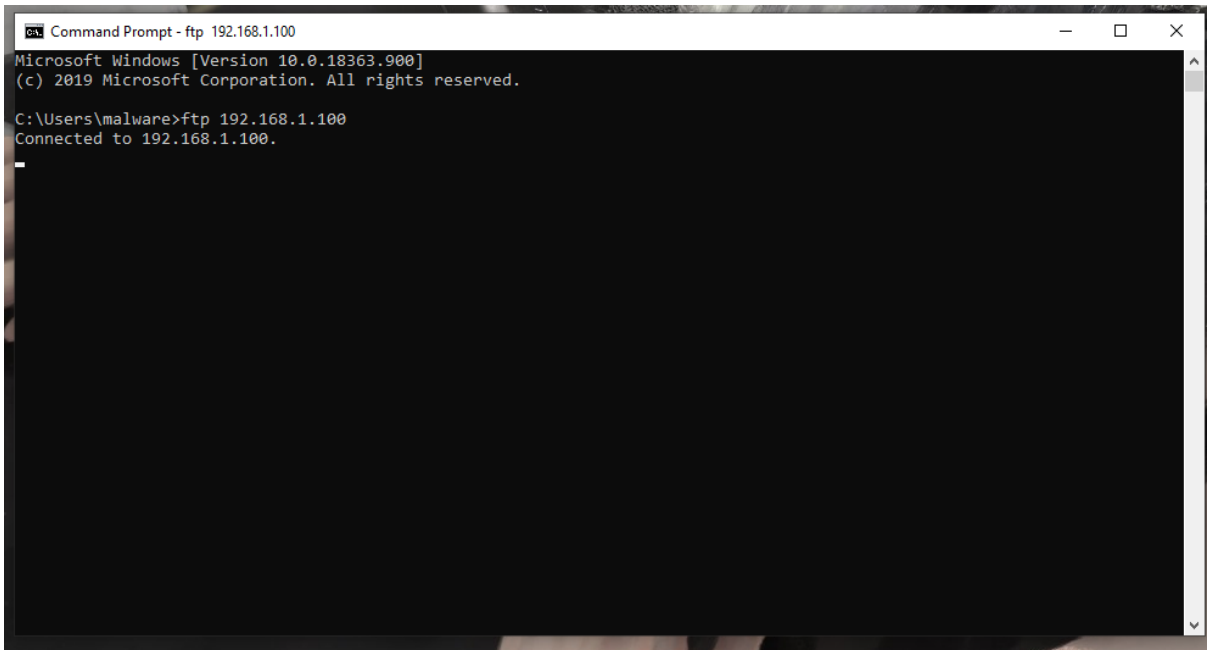


Figure 9: FTP into the Kali VM containing HoneyPy.

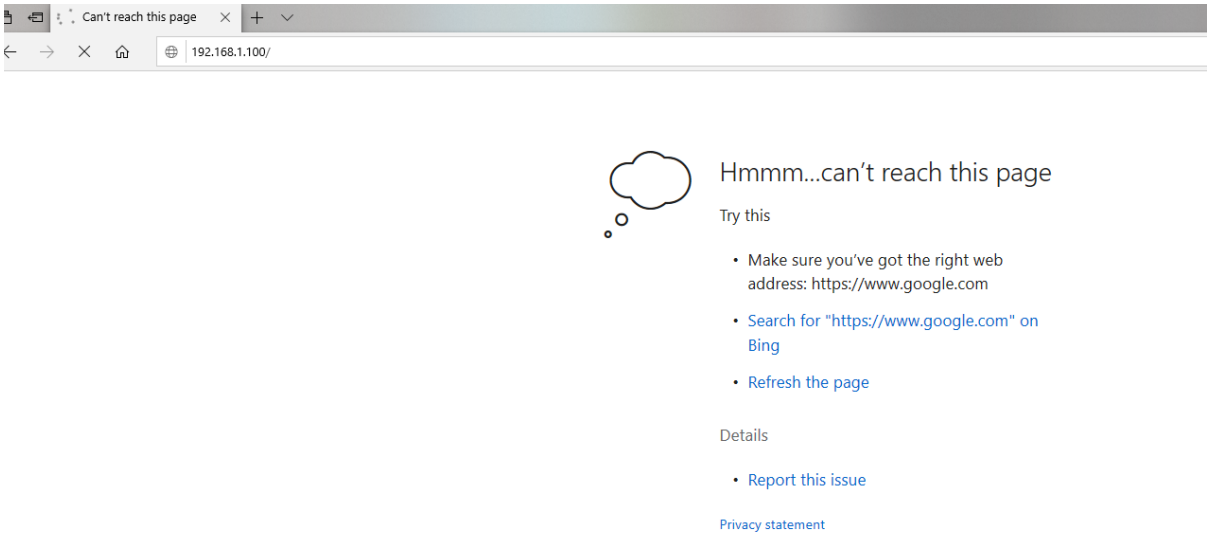


Figure 10: Sending a HTTP request to the Kali VM containing HoneyPy.

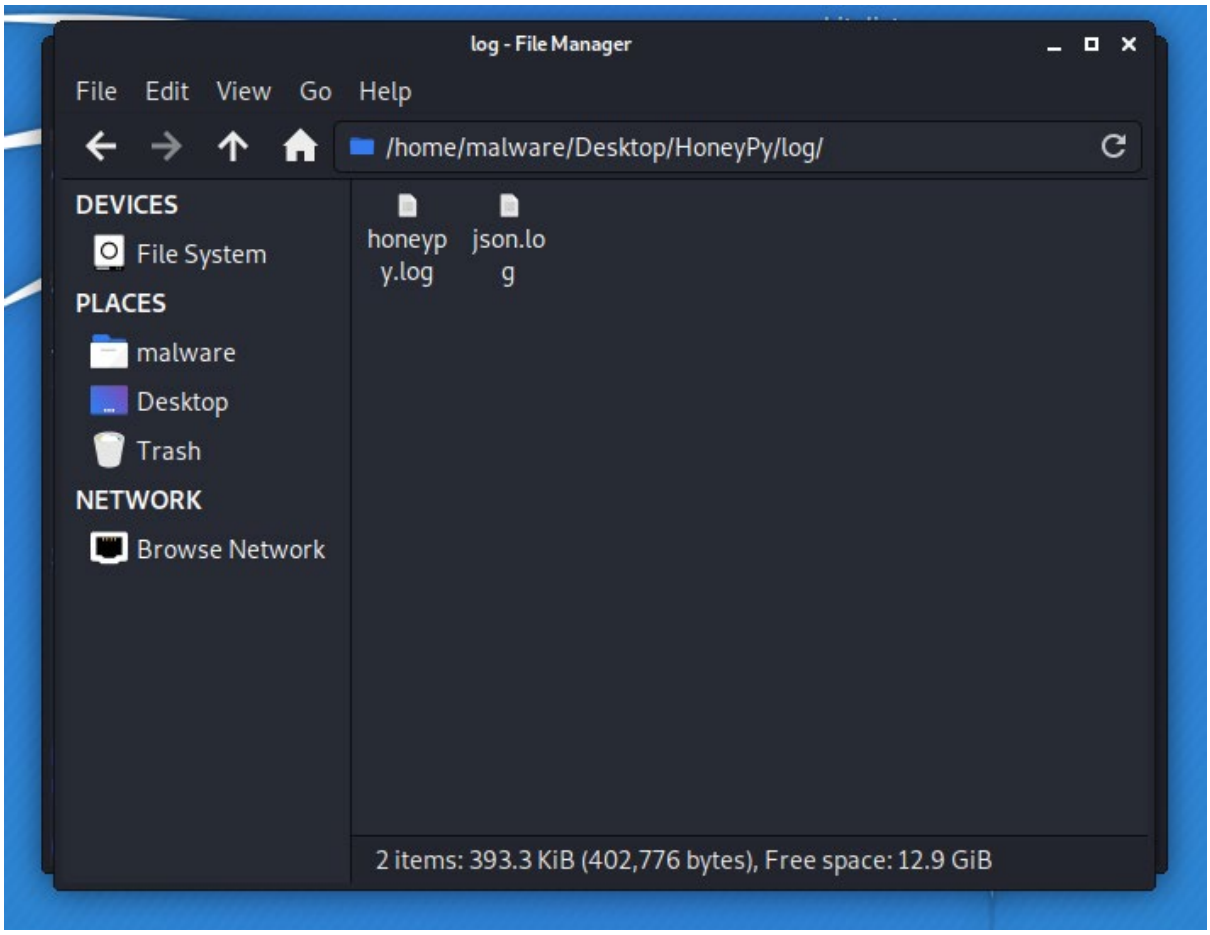


Figure 11: Logs have been generated.

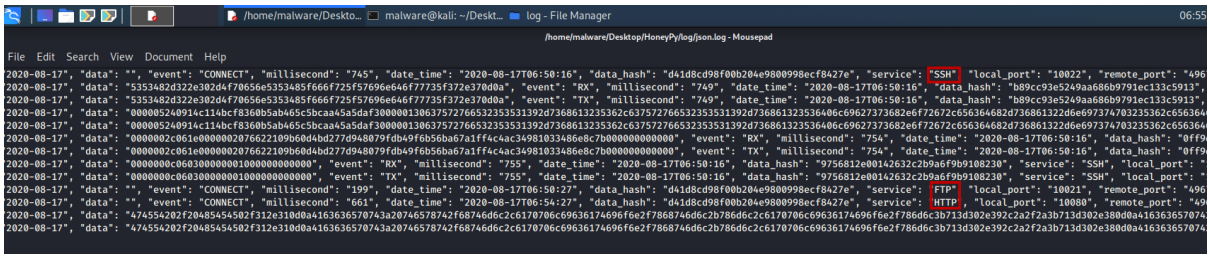


Figure 12: Checking the generated log.

4.6 Installing SNARE

SNARE was installed using the their official link available on github at https://github.com/mushorg/snare.

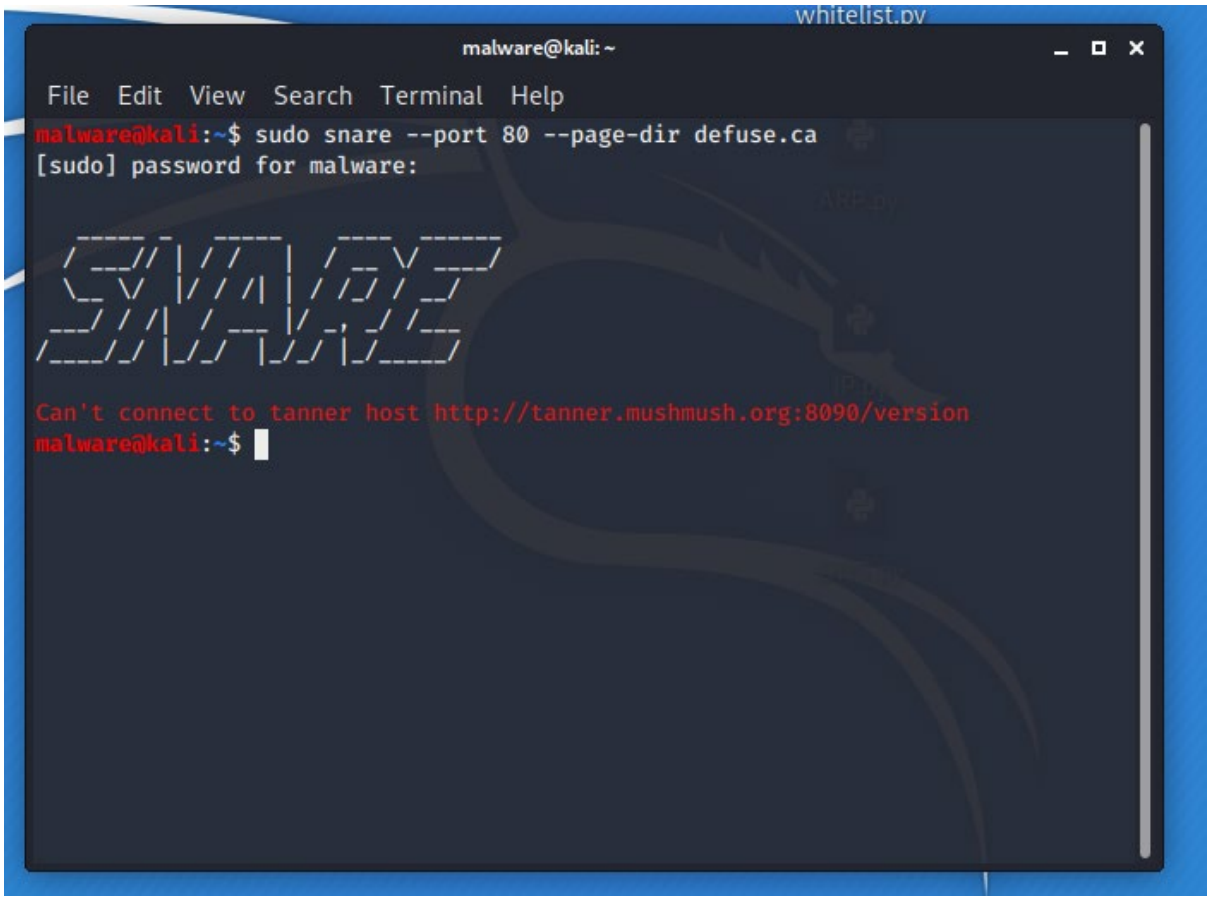


Figure 13: Running SNARE where defuse.ca is the website we are cloning for use with SNARE.

We see that the honeypot SNARE is not working because the website hosting the tanner service is down. So, we have raised an issue on their official Github page.

But, this shouldn't be a problem because HoneyPy also works on port 80 like we have already seen above.

4.7 startCapture.py

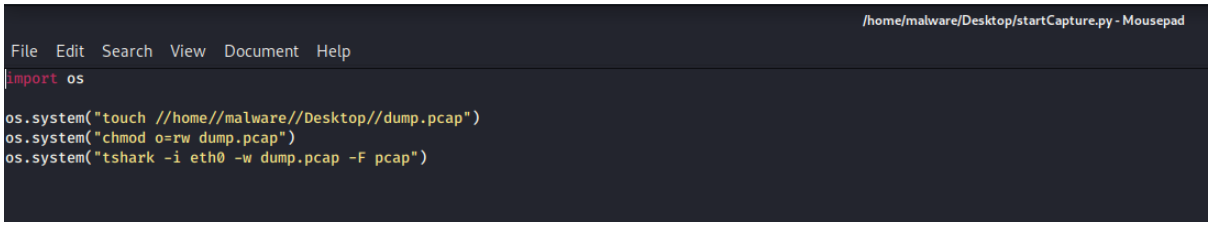


Figure 14: startCapty.py source code.

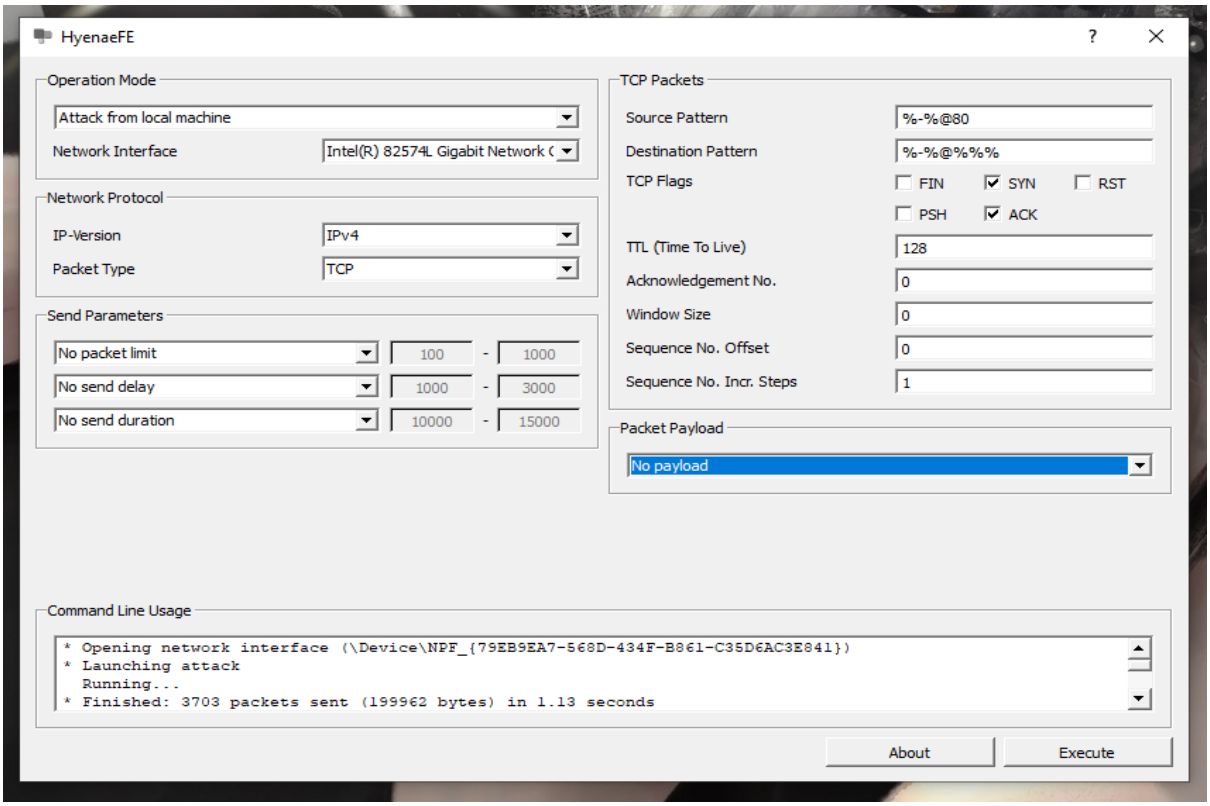


Figure 13: Sending random TCP packets to see if startCapture.py captures the traffic

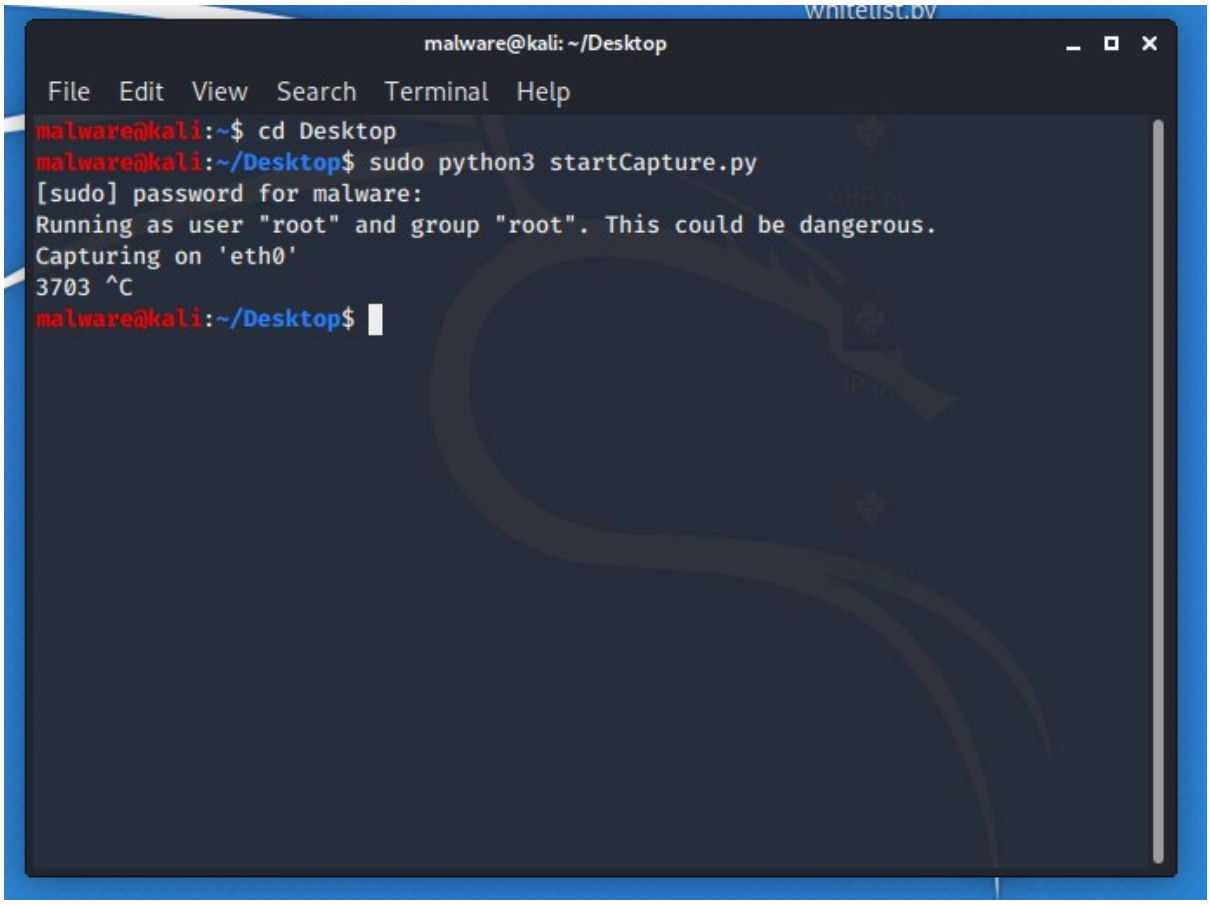


Figure 14: Packet capturing on eth0

Figure 15: Dump.pcap file created

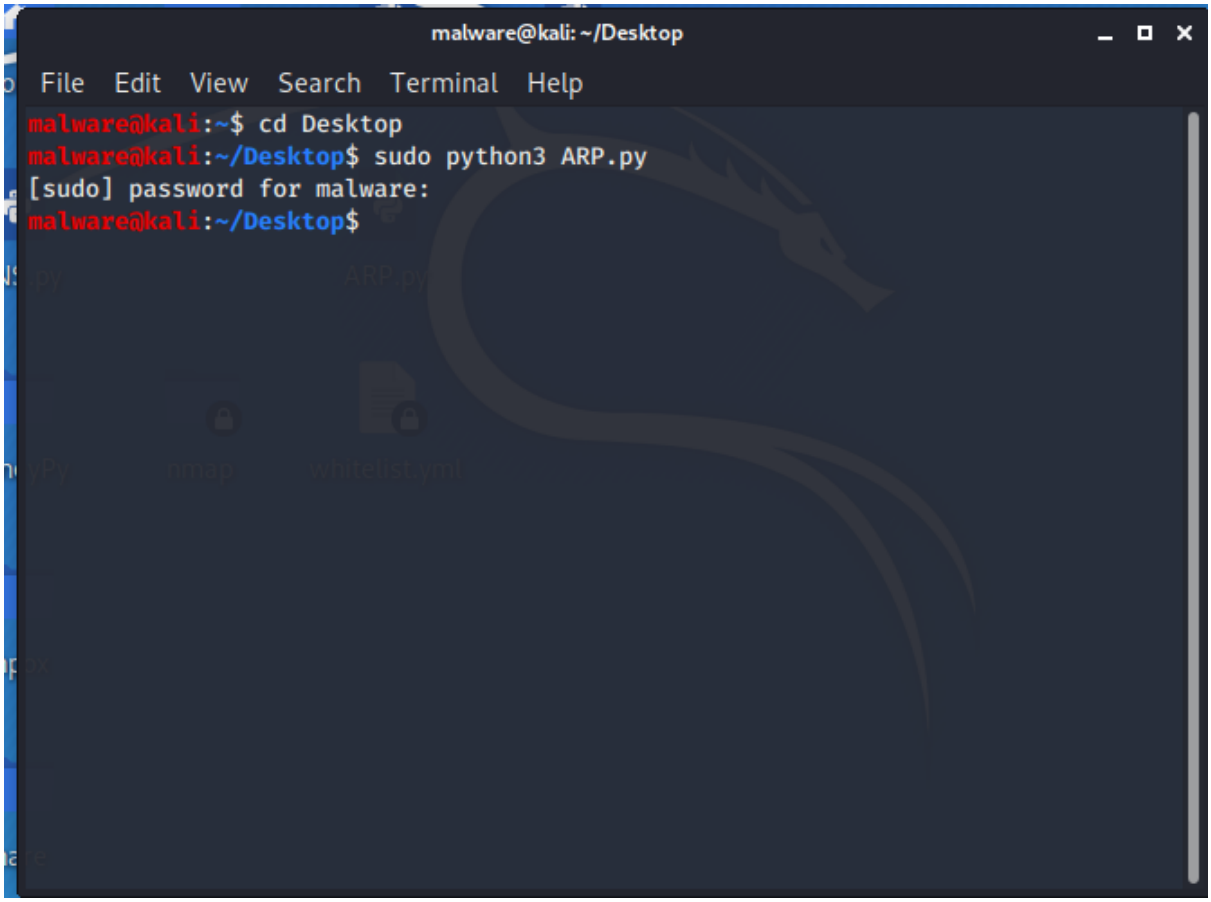


Figure 16: Running ARP.py

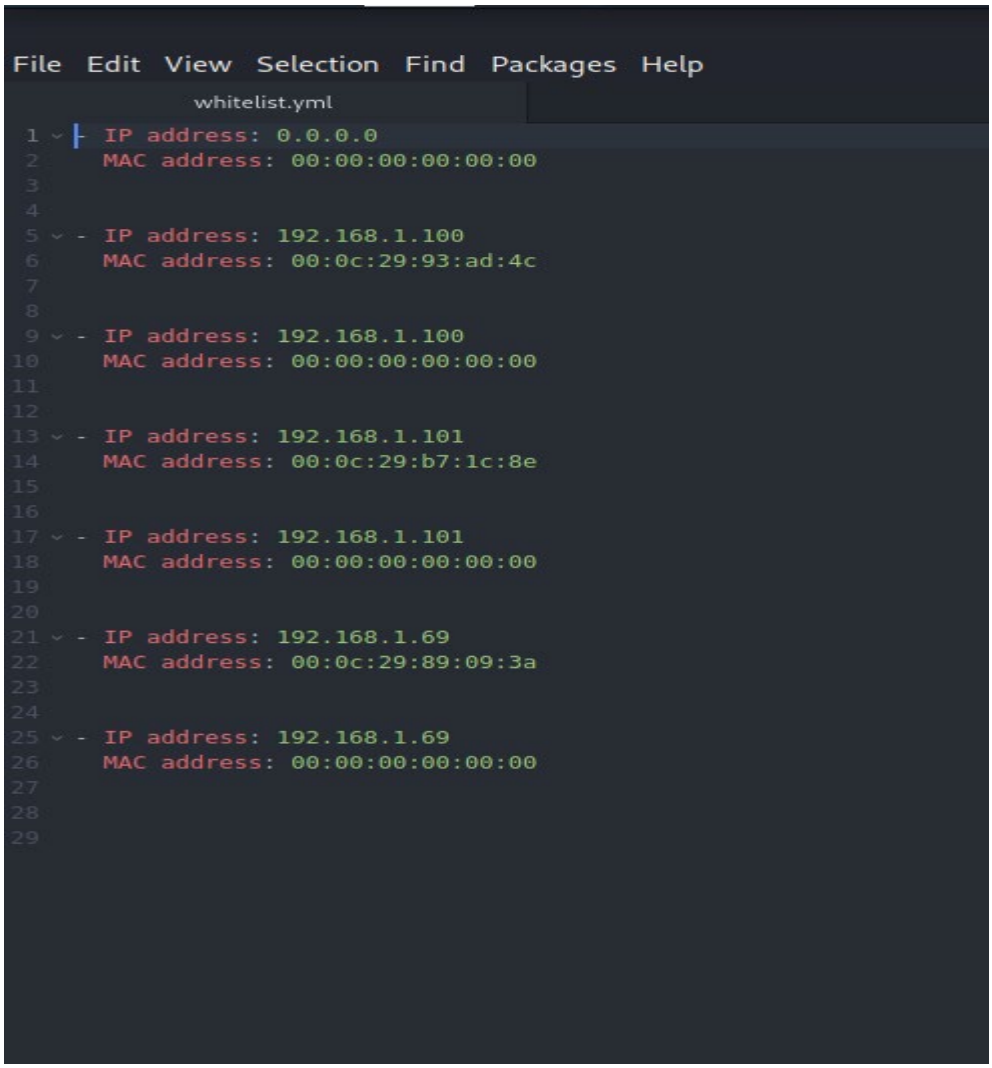


Figure 17: Output of whitelist.py as whitelist.yml

All the scripts work on the dump.pcap file and all the python scripts are executed using the commands shown in Figure 16. Please refer to the video as the generated logs are really long. For the rest of the scripts, please refer to the ICT solutions.