NATIONAL INSTITUTE OF BUSINESS MANAGEMENT

School of Computing

BSc (Hons) Ethical Hacking and Network Security

Higher National Diploma in Network Engineering (HNDNE)

Batch – 23.2F

Network Programming Design

Coursework - Firewall

Name	Navod Zoysa , Isira Lakshan
Index #	COHNDNE232F-045 , COHNDNE232F-046
Submission:	10/19/2024

Ethical Declaration of Original Work

I declare that the work presented in this coursework is entirely my own. I confirm that:

1. The work presented in this coursework is conducted by me, and any contributions from other individuals are appropriately acknowledged.
2. Any external sources of information and ideas used in this work are cited and referenced accurately. I have provided proper credit to the original authors through citations in the text and a comprehensive list of references.
3. The data and findings presented in this work are genuine and have not been manipulated or fabricated. Any assistance received in the collection and analysis of data is acknowledged appropriately.
4. I have not submitted this work, or any part of it, for any other academic qualification.
5. I understand the ethical principles governing academic work, including honesty, integrity, and accountability. I have adhered to these principles throughout the process.

I am aware of the consequences of academic misconduct and understand that any violation of ethical standards may result in disciplinary action.

Navod

……

[Navod De Zoysa Leader]

[19/10/2024]

Contents

1. Installation Process

Update System Packages

- # yum update -y

Install Python

- Ensure Python is installed. CentOS 7 and later usually come with Python 2.x or 3.x. To check, run

  python3 –version

- If Python 3 is not installed, you can install it

  # yum install python3 -y

Install Required Packages

- You will need several Python packages, including scapy, tkinter, and any other dependencies for your script. Install them using pip

  # yum install python3-pip -y

  # yum install scapy

Install Firewall Utilities

- Since the script uses iptables, ensure that the firewall utilities are installed and accessible

  # yum install iptables-services -y

2. Create a Script and Give Permissions

#touch /root/firewall.py

#chmod +x /root/firewall.

3. Code Explanation

- **Enter the following Python code into the script:**

```
import os
import logging
import time
import tkinter as tk
from tkinter import messagebox, ttk
import subprocess
import threading
from scapy.all import sniff  # Ensure scapy is installed

# Set up logging for firewall actions
logging.basicConfig(filename='firewall.log', level=logging.INFO,
format='%(asctime)s - %(message)s')
firewall_logger = logging.getLogger()

# Set up logging for traffic
traffic_logger = logging.getLogger('TrafficLogger')
traffic_handler = logging.FileHandler('traffic.log')
traffic_handler.setLevel(logging.INFO)
formatter = logging.Formatter('%(asctime)s - %(message)s')
traffic_handler.setFormatter(formatter)
traffic_logger.addHandler(traffic_handler)

# Set up logging for malicious activity
malicious_activity_logger = logging.getLogger('MaliciousActivityLogger')
malicious_handler = logging.FileHandler('malicious_activity.log')  # New log file
for malicious activity
malicious_handler.setLevel(logging.INFO)
malicious_handler.setFormatter(formatter)
malicious_activity_logger.addHandler(malicious_handler)

# Directory to monitor for malicious files
MONITOR_DIR = '/home/centos/malicious'

class FirewallGUI:
    def __init__(self, master):
        self.master = master
        master.title("Advanced Firewall & Packet Capture Tool")
        master.geometry("800x600")
        master.configure(bg="#eaeaea")

        ############## Create Main Menu
        self.main_menu()
```

```
    def main_menu(self):
        self.menu_frame = tk.Frame(self.master, bg="#eaeaea")
        self.menu_frame.pack(fill="both", expand=True)

        self.title_label = tk.Label(self.menu_frame, text="Main Menu",
font=("Helvetica", 20), bg="#eaeaea")
        self.title_label.pack(pady=20)

        # Buttons for navigation
        self.firewall_button = ttk.Button(self.menu_frame, text="Firewall
Management", command=self.create_firewall_frame)
        self.monitoring_button = ttk.Button(self.menu_frame, text="Monitor
Malicious Files", command=self.create_monitoring_frame)
        self.packet_capture_button = ttk.Button(self.menu_frame, text="Packet
Capture", command=self.create_packet_capture_frame)

        self.firewall_button.pack(pady=10)
        self.monitoring_button.pack(pady=10)
        self.packet_capture_button.pack(pady=10)

    def create_firewall_frame(self):
        self.menu_frame.pack_forget()  # Hide main menu
        self.firewall_frame = tk.Frame(self.master, bg="#f0f0f0", padx=20, pady=20)
        self.firewall_frame.pack(fill="both", expand=True)

        tk.Label(self.firewall_frame, text="Source IP:", bg="#f0f0f0").grid(row=0,
column=0, sticky='e', padx=5, pady=5)
        tk.Label(self.firewall_frame, text="Destination IP:",
bg="#f0f0f0").grid(row=1, column=0, sticky='e', padx=5, pady=5)
        tk.Label(self.firewall_frame, text="Port:", bg="#f0f0f0").grid(row=2,
column=0, sticky='e', padx=5, pady=5)
        tk.Label(self.firewall_frame, text="Protocol:", bg="#f0f0f0").grid(row=3,
column=0, sticky='e', padx=5, pady=5)
        tk.Label(self.firewall_frame, text="Action:", bg="#f0f0f0").grid(row=4,
column=0, sticky='e', padx=5, pady=5)

        self.source_entry = tk.Entry(self.firewall_frame, width=30)
        self.dest_entry = tk.Entry(self.firewall_frame, width=30)
        self.port_entry = tk.Entry(self.firewall_frame, width=30)
        self.protocol_entry = tk.Entry(self.firewall_frame, width=30)

        self.source_entry.grid(row=0, column=1, padx=5, pady=5)
        self.dest_entry.grid(row=1, column=1, padx=5, pady=5)
        self.port_entry.grid(row=2, column=1, padx=5, pady=5)
        self.protocol_entry.grid(row=3, column=1, padx=5, pady=5)

        ############## Action selection
```

```
        self.action_var = tk.StringVar(value="ACCEPT")  # Default to ACCEPT
        self.action_menu = ttk.OptionMenu(self.firewall_frame, self.action_var,
"ACCEPT", "ACCEPT", "DROP")
        self.action_menu.grid(row=4, column=1, padx=5, pady=5)

        ## Buttons
        self.add_button = ttk.Button(self.firewall_frame, text="Add Rule",
command=self.add_rule)
        self.remove_button = ttk.Button(self.firewall_frame, text="Remove Rule",
command=self.remove_rule)
        self.view_button = ttk.Button(self.firewall_frame, text="View Rules",
command=self.view_rules)

        self.add_button.grid(row=5, column=0, pady=10, padx=5)
        self.remove_button.grid(row=5, column=1, pady=10, padx=5)
        self.view_button.grid(row=5, column=2, pady=10)

        ##Status Label
        self.status_label = tk.Label(self.firewall_frame, text="", bg="#f0f0f0")
        self.status_label.grid(row=6, column=0, columnspan=3)

        ##Back to Main Menu Button
        self.back_button = ttk.Button(self.firewall_frame, text="Back to Main
Menu", command=self.back_to_main_menu)
        self.back_button.grid(row=7, column=0, columnspan=3, pady=10)

    def create_monitoring_frame(self):
        self.menu_frame.pack_forget()  # Hide main menu
        self.monitoring_frame = tk.Frame(self.master, bg="#f0f0f0", padx=20,
pady=20)
        self.monitoring_frame.pack(fill="both", expand=True)

        self.start_monitoring_button = ttk.Button(self.monitoring_frame, text="Start
Monitoring", command=self.start_malicious_file_monitoring)
        self.start_monitoring_button.pack(pady=20)

        self.status_label = tk.Label(self.monitoring_frame, text="Status: Not
Monitoring", bg="#f0f0f0")
        self.status_label.pack()

        ##Back to Main Menu Button
        self.back_button = ttk.Button(self.monitoring_frame, text="Back to Main
Menu", command=self.back_to_main_menu)
        self.back_button.pack(pady=10)

    def create_packet_capture_frame(self):
        self.menu_frame.pack_forget()  # Hide main menu
```

```
        self.packet_capture_frame = tk.Frame(self.master, bg="#f0f0f0", padx=20,
pady=20)
        self.packet_capture_frame.pack(fill="both", expand=True)

        self.packet_listbox = tk.Listbox(self.packet_capture_frame, width=100,
height=20)
        self.packet_listbox.pack(pady=10)

        self.start_capture_button = ttk.Button(self.packet_capture_frame,
text="Start Packet Capture", command=self.start_packet_capture)
        self.start_capture_button.pack(pady=20)

        ## Back to Main Menu Button
        self.back_button = ttk.Button(self.packet_capture_frame, text="Back to
Main Menu", command=self.back_to_main_menu)
        self.back_button.pack(pady=10)

    def run_command(self, command):
        try:
            output = subprocess.check_output(command, stderr=subprocess.STDOUT,
shell=True)
            firewall_logger.info(f"Executed command: {command}")
            return output.decode()
        except subprocess.CalledProcessError as e:
            error_msg = e.output.decode()
            firewall_logger.error(f"Error executing command: {command} -
{error_msg}")
            messagebox.showerror("Error", error_msg)
            return None

    def add_rule(self):
        source_ip = self.source_entry.get()
        dest_ip = self.dest_entry.get()
        port = self.port_entry.get()
        protocol = self.protocol_entry.get()
        action = self.action_var.get()  # Get the selected action (ACCEPT or DROP)

        cmd = f"sudo iptables -A INPUT -p {protocol}"
        if source_ip:
            cmd += f" -s {source_ip}"
        if dest_ip:
            cmd += f" -d {dest_ip}"
        if port:
            cmd += f" --dport {port}"
        cmd += f" -j {action}"  # Use the selected action

        firewall_logger.info(f"Adding rule: {cmd}")
```

```
        self.run_command(cmd)
        self.status_label.config(text=f"Added rule for {source_ip} -> {dest_ip} on port
{port} with action {action}")

    def remove_rule(self):
        source_ip = self.source_entry.get()
        dest_ip = self.dest_entry.get()
        port = self.port_entry.get()
        protocol = self.protocol_entry.get()
        action = self.action_var.get()  # Get the selected action

        cmd = f"sudo iptables -D INPUT -p {protocol}"
        if source_ip:
            cmd += f" -s {source_ip}"
        if dest_ip:
            cmd += f" -d {dest_ip}"
        if port:
            cmd += f" --dport {port}"
        cmd += f" -j {action}"  # Use the selected action

        firewall_logger.info(f"Removing rule: {cmd}")
        self.run_command(cmd)
        self.status_label.config(text=f"Removed rule for {source_ip} -> {dest_ip} on
port {port} with action {action}")

    def view_rules(self):
        cmd = "sudo iptables -L -n -v"
        output = self.run_command(cmd)
        if output:
            messagebox.showinfo("Current Firewall Rules", output)

    def start_malicious_file_monitoring(self):
        self.status_label.config(text="Status: Monitoring...")
        while True:
            for filename in os.listdir(MONITOR_DIR):
                if filename.endswith(".malicious"):
                    traffic_logger.info(f"Malicious file detected: {filename}")
                    messagebox.showwarning("Malicious File Detected", f"Malicious file
detected: {filename}")
                    malicious_activity_logger.info(f"Malicious activity detected:
{filename}")  # Log to malicious_activity.log
            time.sleep(10)  # Check every 10 seconds

    def start_packet_capture(self):
        self.packet_listbox.delete(0, tk.END)  # Clear previous packets
        traffic_logger.info("Started packet capture")
```

```
        capture_thread = threading.Thread(target=lambda:
sniff(prn=self.capture_packet, store=False), daemon=True)
        capture_thread.start()

    def capture_packet(self, packet):
        if packet.haslayer('IP'):  # Ensure packet has an IP layer
            ip_layer = packet['IP']
            packet_info = f"{ip_layer.src} -> {ip_layer.dst} (Protocol: {ip_layer.proto})"
            traffic_logger.info(packet_info)

            ######### Check for malicious activity (for example, port scanning)
            if packet.haslayer('TCP') and packet['TCP'].dport not in [22, 80, 443]:  #
Adjust based on your requirements
                malicious_activity_logger.info(f"Potential malicious activity detected:
{packet_info}")

            self.packet_listbox.insert(tk.END, packet_info)
        else:
            packet_info = f"Non-IP packet detected: {packet.summary()}"
            traffic_logger.info(packet_info)
            self.packet_listbox.insert(tk.END, packet_info)

    def back_to_main_menu(self):
        self.firewall_frame.pack_forget() if hasattr(self, 'firewall_frame') else None
        self.monitoring_frame.pack_forget() if hasattr(self, 'monitoring_frame') else
None
        self.packet_capture_frame.pack_forget() if hasattr(self,
'packet_capture_frame') else None
        self.main_menu()

if __name__ == "__main__":
    root = tk.Tk()
    app = FirewallGUI(root)
    root.mainloop()
```

4.Verify and Testing

Firewall rules Testing

- Create a client machine
  IP -192.168.188.129

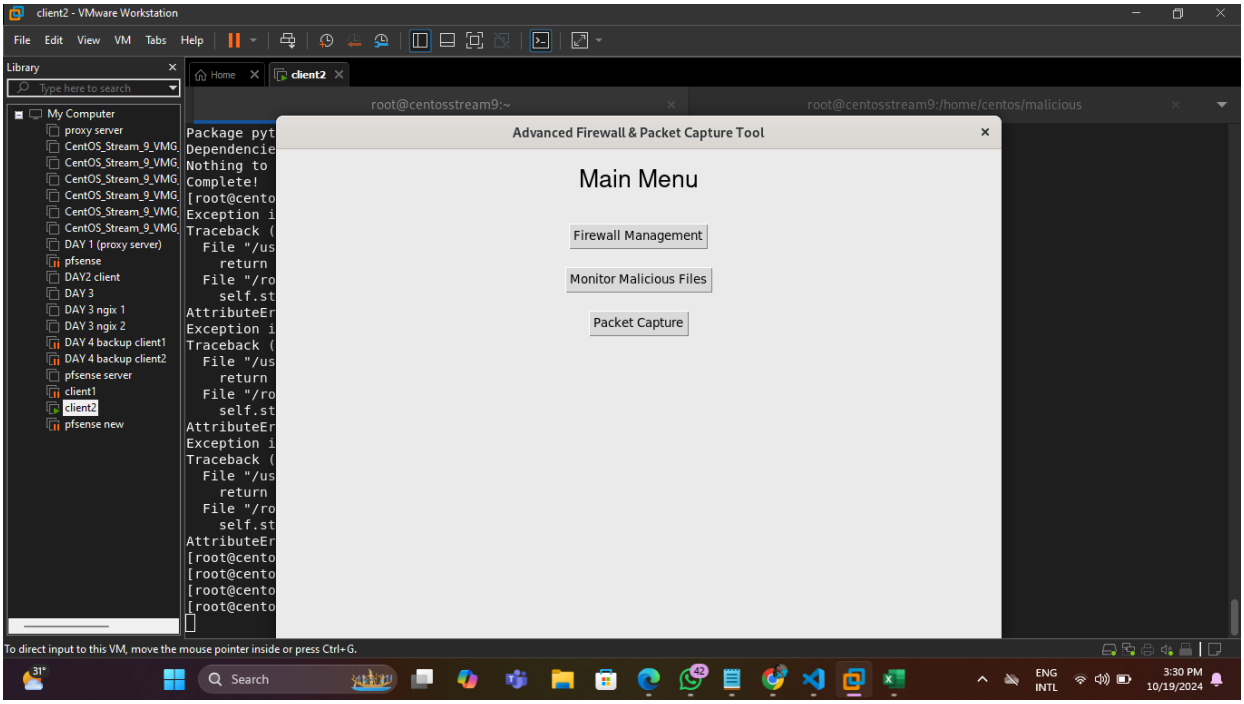
- Add rules to firewall
  Source- 192.168.188.129(client IP)
  Destination -192.168.188.130 (firewall machine IP)
  Port- any
  Protocol-ICMP
- Ping from client
  PING 192.168.188.130
- Then Can see packet loss occur

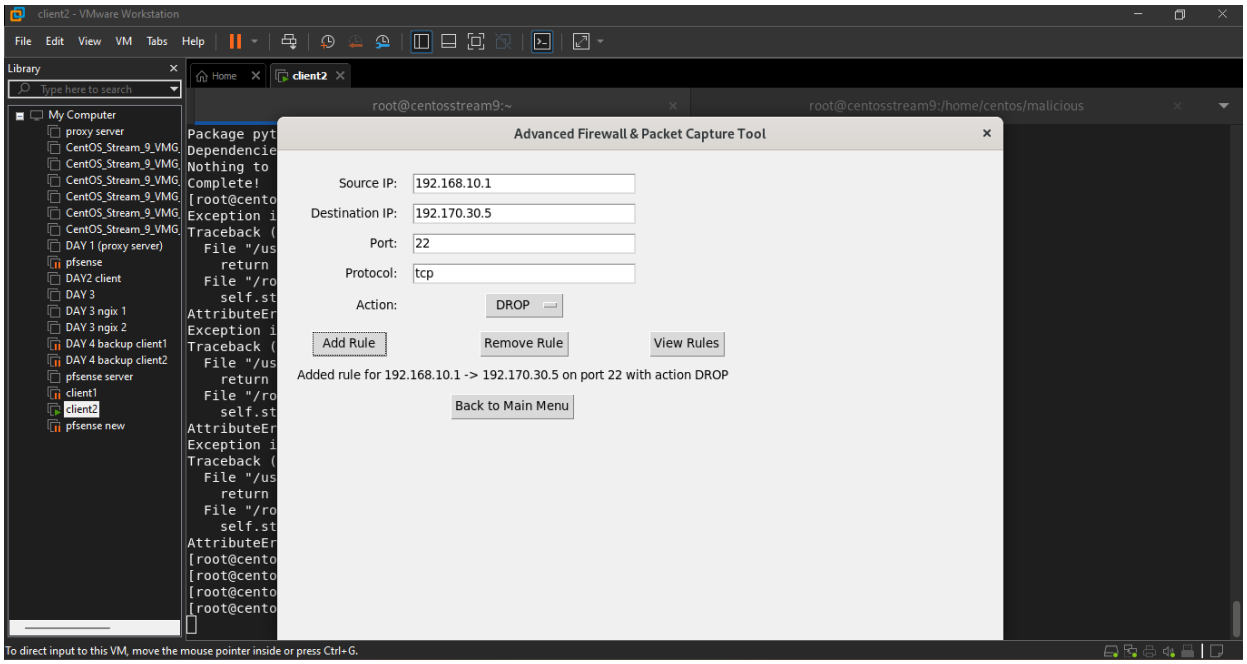
Packet capturing Can be check by Click the packet capturing Icon

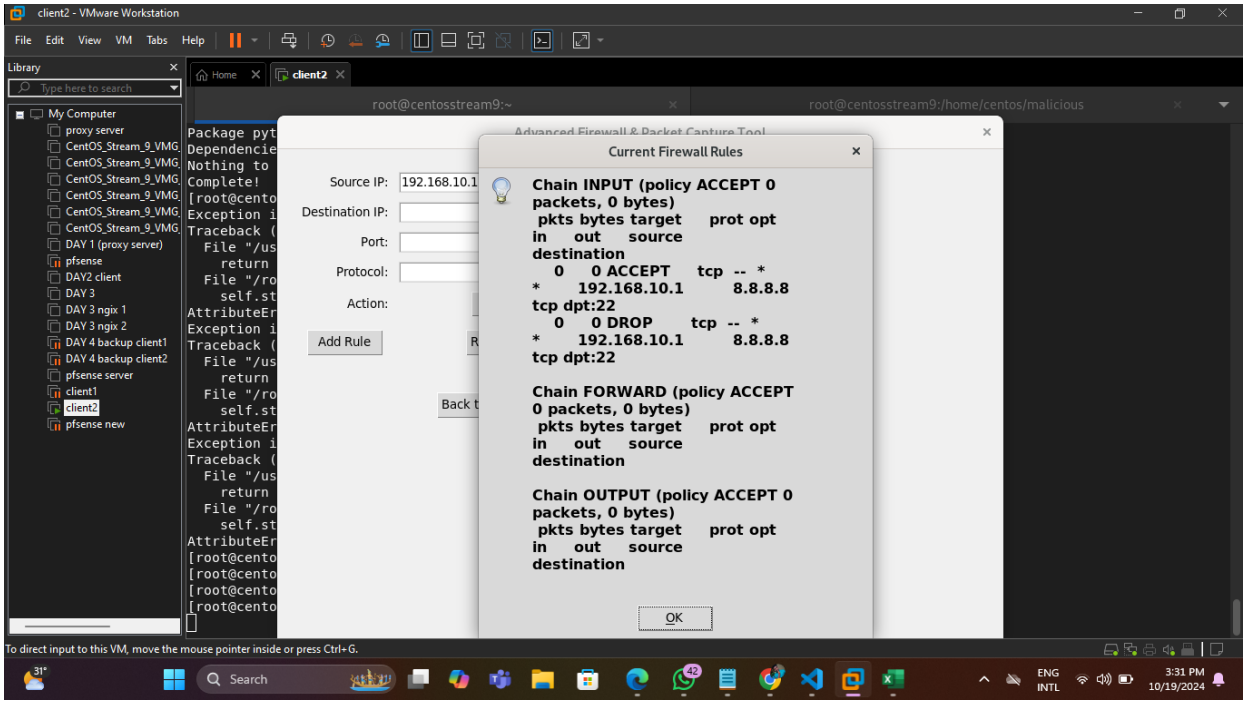
Testing the logs

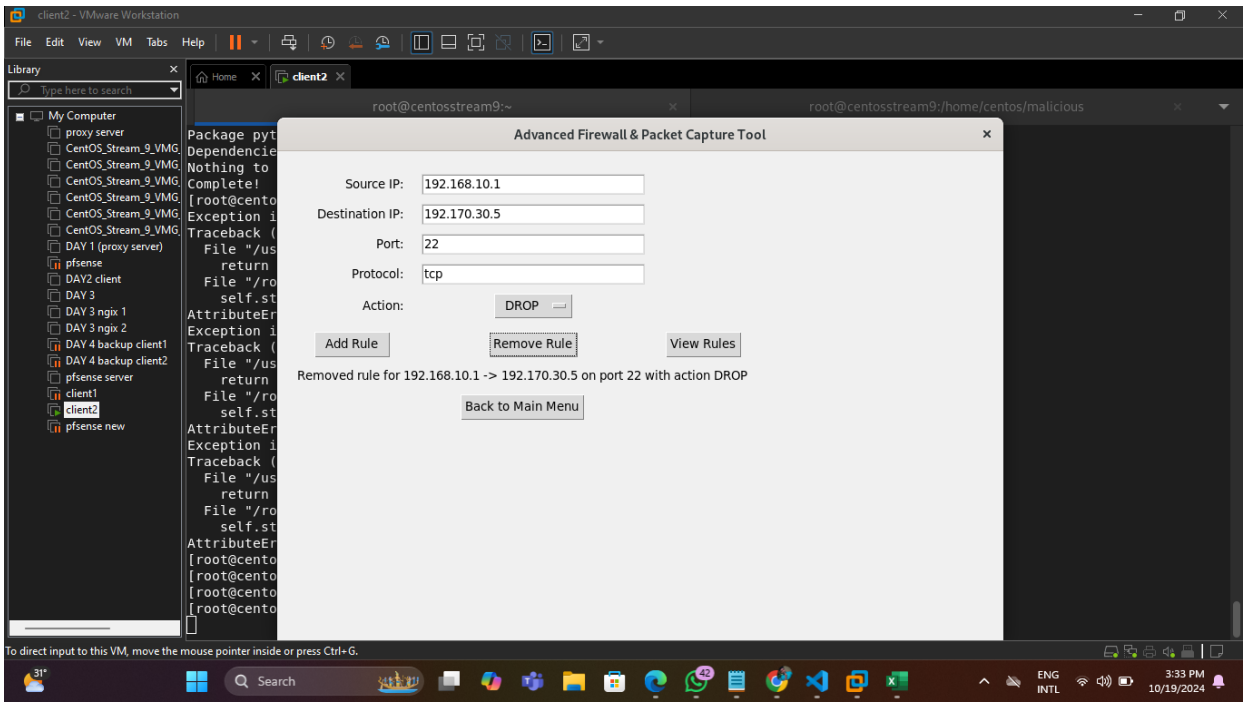
#cat /root/firewall.log
#cat /root/traffic.log
#cat /root/malicious_activity.log

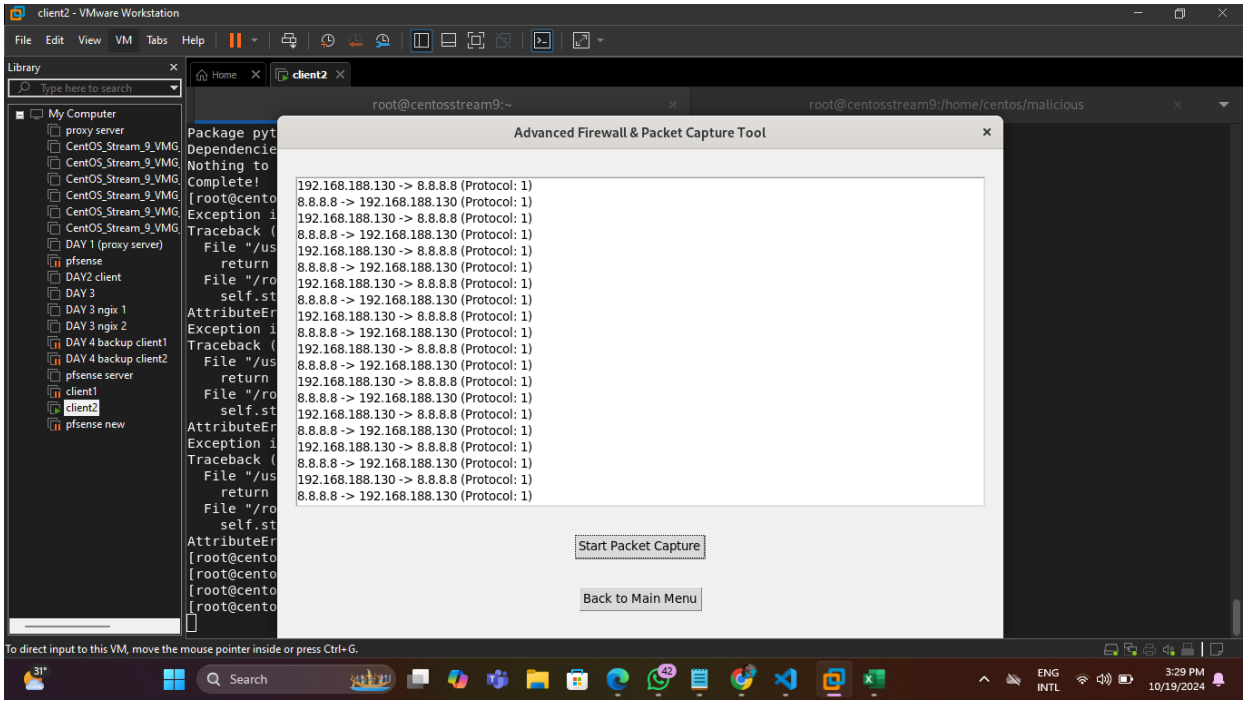
5.Appendices






client2 - VMware Workstation
Current Firewall Rules
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
0 0 ACCEPT tcp -- * * 192.168.10.1 8.8.8.8 tcp dpt:22
0 0 DROP tcp -- * * 192.168.10.1 8.8.8.8 tcp dpt:22
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
OK
Source IP:
Destination IP:
Port:
Protocol:
Action:
Add Rule


client2 - VMware Workstation
Advanced Firewall & Packet Capture Tool
Source IP: 192.168.10.1
Destination IP: 192.170.30.5
Port: 22
Protocol: tcp
Action: DROP
Add Rule
Remove Rule
View Rules
Removed rule for 192.168.10.1 -> 192.170.30.5 on port 22 with action DROP
Back to Main Menu


client2 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to search
My Computer
proxy server
CentOS_Stream_9_VMG
CentOS_Stream_9_VMG
CentOS_Stream_9_VMG
CentOS_Stream_9_VMG
CentOS_Stream_9_VMG
CentOS_Stream_9_VMG
DAY 1 (proxy server)
pfsense
DAY2 client
DAY 3
DAY 3 ngix 1
DAY 3 ngix 2
DAY 4 backup client1
DAY 4 backup client2
pfsense server
client1
client2
pfsense new
Home
client2
root@centosstream9:~
root@centosstream9:/home/centos/malicious
Advanced Firewall & Packet Capture Tool
192.168.188.130 -> 8.8.8.8 (Protocol: 1)
8.8.8.8 -> 192.168.188.130 (Protocol: 1)
Start Packet Capture
Back to Main Menu
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
Search
ENG
INTL
3:29 PM
10/19/2024

6.Video Demonstration

Link-https://youtu.be/-s5Yv0SBjjs