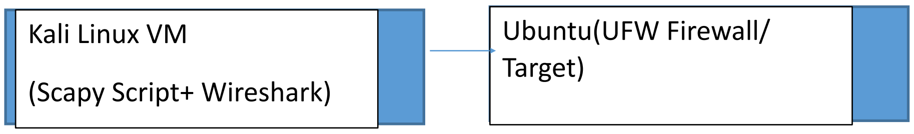
Part B: Prg2

Use a packet crafting tool such as Scapy (Python) to create custom TCP, UDP, or ICMP packets. Send these packets to a target system and observe how a firewall responds. Use Wireshark to capture and analyze the traffic, including flags, headers, and dropped packets.

i) Kali Linux-> Ubuntu(Target-ubuntu vm is used to set up UFW firewall settings)

Note: *UFW stands for Uncomplicated Firewall.It is the default firewall management tool used in Ubuntu and other Debian-based Linux systems.*



Step 1: Network Setup:

Check the IP address of source VM Kali Linux and Target VM Ubuntu by using ifconfig command in terminal.

Step 2: Configure Ubuntu as firewall target

a. **Enable UFW**

```
sudo   ufw   enable
```

b. **Add Rules**

```
sudo ufw deny 80/tcp      # block web traffic
sudo ufw allow 22/tcp     # allow SSH
```

For icmp packets to block we can add following block rules:

```
sudo  iptables  -I  INPUT 1   -p  icmp  --icmp-type  echo-request  -j  DROP
```

Step 4: Scapy Script in Kali Linux

```
  vi   packet_lab.py


#!/usr/bin/env python3

from scapy.all import *

import time


TARGET_IP = "192.168.56.103"   # Ubuntu VM

DEST_PORT = 80

DEST_PORT2 = 22

UDP_PORT = 53

PACKET_DELAY = 1



def banner():

    print("\n===========================================")

    print("      SCAPY PACKET-CRAFTING LAB SCRIPT")

    print("===========================================\n")



def send_icmp():

    print("[*] Sending ICMP Echo Request...")

    packet = IP(dst=TARGET_IP) / ICMP()

    send(packet, verbose=0)

    print("[+] ICMP packet sent.\n")
```

```
def send_tcp_syn(destination_port):
    print(f"[*] Sending TCP SYN to port {destination_port}...")
    packet = IP(dst=TARGET_IP) / TCP(dport=destination_port, flags="S")
    send(packet, verbose=0)
    print("[+] TCP SYN sent.\n")


def send_tcp_null(destination_port):
    print("[*] Sending TCP NULL packet...")
    packet = IP(dst=TARGET_IP) / TCP(dport=destination_port, flags=0)
    send(packet, verbose=0)
    print("[+] NULL packet sent.\n")


def send_tcp_fin(destination_port):
    print("[*] Sending TCP FIN packet...")
    packet = IP(dst=TARGET_IP) / TCP(dport=destination_port, flags="F")
    send(packet, verbose=0)
    print("[+] FIN sent.\n")


def send_udp():
    print(f"[*] Sending UDP to port {UDP_PORT}...")
    packet = IP(dst=TARGET_IP) / UDP(dport=UDP_PORT) / Raw(load="TestUDP")
    send(packet, verbose=0)
    print("[+] UDP sent.\n")


def send_custom_payload():
    print("[*] Sending TCP packet with custom payload...")
            packet = IP(dst=TARGET_IP) / TCP(dport=DEST_PORT, flags="PA") /
Raw(load="HelloFromScapy")
    send(packet, verbose=0)
    print("[+] Payload packet sent.\n")


def main():
    banner()
    time.sleep(1)

    # Send packets
    send_icmp()
    time.sleep(PACKET_DELAY)

    send_tcp_syn(DEST_PORT2)
    time.sleep(PACKET_DELAY)

    send_tcp_null(DEST_PORT2)
    time.sleep(PACKET_DELAY)
    send_tcp_fin(DEST_PORT2)
    time.sleep(PACKET_DELAY)

    send_udp()
    time.sleep(PACKET_DELAY)

    send_custom_payload()

    print("\n[+] Experiment Completed.")


if __name__ == "__main__":
    main()
```

Note: Type the python script and save it in vi editor

Step 5: Start Wireshark in Kali Linux

Start wireshark and chose interface etho

Step 6: Run the Scapy Script

```
sudo  python3  packet_lab.py
```

Step 7: Observe the output:

Now output can be checked in 2 ways:

1st way:open Wireshark once packets are captured and check with the below filters:

- ➔ icmp
- ➔ tcp.flags.syn==1

  we will see TCP [SYN] Kali-> Unix(port 80).. then no RST, no Syn/Ack.

- ➔ tcp.port==22

  In wireshark we will see

  TCP[SYN] kali -> Ubuntu

  Tcp[Syn, ACk] Ubuntu-> kali

  TCP [ACK] Kali-> Ubuntu

2nd way: in kali linux type the below command and check the response

a) ping <Ubuntu-ip>

b) nc –v <Ubuntu-ip> 80

c) nc –v <Ubuntu-ip> 22

And also to check ssh connection is working i.e port 22 is working.

Type the below command in kalilinux terminal

```
ssh ubuntu@<ubuntu-ip>
```

And also in Ubuntu VM check the logs by using below commands

```
sudo ufw logging on
tail –f /var/log/ufw.log
```

Note:

Once the experiment completes remove the rule in ubuntu.

```
sudo  iptables –D  INPUT  -p  icmp  --icmp-type  echo-request  -j  DROP
```