NFS Lab 4: TCP Hijacking and VPNs

I Introduction

This lab session will continue your attacks on TCP and introduce the usage of VPNs.

I.1 Prerequisites

For this lab, you should make sure to be on your own device, or at least a linux VM, with root access.

You must install the following tools:

- wireshark
- netcat
- python3
- scapy (python3 package)
- docker (cf. https://docs.docker.com/engine/install/)

II TCP Reminder

Here are two command lines that will be useful for our TCP client/server implementations, as an example we use the port 9090:

```
~$ netcat -lvn 9090
~$ netcat 127.0.0.1 9090
```

II.1 TCP Hijacking Attack

The TCP hijacking attack allows an attacker to inject packet in an established connection. When used, for instance, to attack telnet session, an attacker can send arbitrary commands.

II.1.1 Docker Setup

For this lab, we will use docker containers to create virtual machines in a local network. The file is accessible from the NFS lecture page, or directly from https://avalonswanderer.github.io/assets/zip/nfs/tp4_docker.tar.gz.

Once downloaded, you should be able to unzip the folder, and then build and start the containers.

```
~$ docker compose build             # build the containers
~$ docker compose up -d             # start the containers in the background
~$ docker compose exec client bash  # launch a bash in the client container
~$ docker compose restart           # restart all containers
~$ docker compose down              # stop the containers
```

Our containers represent three devices: a client, a server, and an attacker. Here the attacker container can be ignored if you managed to install scapy and netcat on your host/VM. If so you can imagine the following figure with the Attacker being our regular host device:

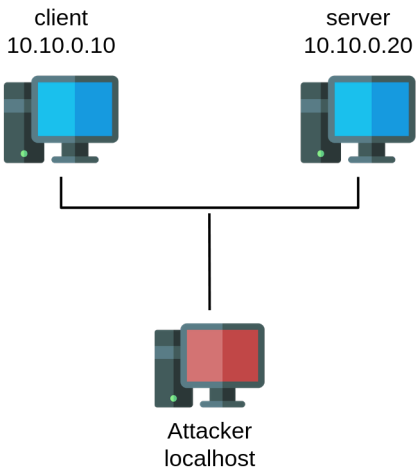

Figure 1: Network: 10.10.0.0/24

For our setup, we want to execute bash on our client (cf. docker commands above). Once on the machine, we want to open a TCP telnet connection on port 23 with the server.

```
client~$ telnet 10.10.0.20 23
```

For the telnet connection:
- login: victim
- pass: password

Remember to open wireshark on your host and to filter the packets to only see the 10.10.0.0/24 network.

II.1.2 Sending a command

Remember this code snippet from the previous lab? Update it to inject a *touch* command to create the file "*you_have_been_pwn*".

```
#!/usr/bin/python3

from scapy.all import *

ip = IP(src="XXX", dst="XXX")
tcp = TCP(sport=XXX, dport=XXX, flags="X", seq=X)
pkt = ip/tcp
send(pkt,verbose=0)
```

Tips: Do not forget to add breaklines to your command '\n'.

Checkpoint: Call your lab supervisor.

II.1.3 Going further with a reverse shell

Sending one command is cool, but what about creating a session? Update your previous code to open a connection between a netcat server running on your attacker (e.g., your host) and the server.

Using sniff filter from scapy, try to sniff the packet between the client and the server to automatically retrieve the ports and seq#, to fill your forged hijacking packet.

```
#!/usr/bin/python3
```

```
import sys
import scapy.all import *

def attack(pkt):
  old_tcp_pkt = pkt[TCP] # TCP part of the sniffed packet.
  # ...
  send(pkt,verbose=0)

filter_str = 'tcp' # for now, matching all tcp packets.
sniff(filter=filter_str, prn=attack)
```

Tips: Everything is a file in linux, even connection: */dev/tcp/XXXX/YYY* could help you. What about sending a *bash* command redirected to this?

Question 1: Make your TCP filter narrow enough, we don't want all tcp packets. Explain your filter.

Checkpoint: Call your lab supervisor.

III WireGuard

Remember IPsec? Well you're not the only one thinking it's complex. This is why other protocol exist to setup VPNs. One of them is WireGuard.

III.1 Setup the Client

Connect to the client using *docker compose exec server bash* from your host. To setup Wireguard, here are the broad steps for you to do:

- Generate a private key wireguard cli tool *wg*
- Using *ip*, create a new interface wg0 of type wireguard.
- Choose an IP in a new network for your client and add it to the interface.
  - (e.g., 20.20.20.10/24).
- With *wg*, associate wg0 with your private key.
- Activate your wg0 link (up).
- Check your config by running *wg* only.

Checkpoint: Call your lab supervisor.

III.2 Setup the Server

Connect to the docker using *docker compose exec server bash* Do the same for your server (do not forget to give it a new IP in the same network as the client!).

III.3 Connection and Inspection

One last step now!

- You need to share the public keys between them.
  - Using *wg* and the peer arguments.
  - On the client, allow only the server IP, and the otherway around.
  - On the command, be careful to specify as endpoint the real IP addresses, and the good port used by wireguard.

Create a telnet connection from your client to the wireguard IP of the server.

Checkpoint: Call your lab supervisor.

Retry your TCP session hijacking attack on the session.

Question 2: What can you observe?

Question 3: Open wireshark to sniff the communication between the client and the server. Can you identify telnet communication?

Acknowledgements

This work is inspired by previous materials from Mathieu Goessens, and the book *Internet Security: A Hands-on Approach (Computer & Internet Security) 3rd ed.* by Wenliang Du.