Lab 02 Report

Name: Madene Meriem Group: 01

October 16, 2025

Objective

Lab Title: Authentication in Practice
Main Goal:

Environment Setup

Virtual Machines:

VM Name	IP Address
Kali Linux	192.168.153.131
Metasploitable2	192.168.153.128

Tools Used:

- gpg encryption
- scp to transfer files securely
- nmap
- john the ripper password cracker
- hydra

Exercise 1: GPG Encryption and Decryption

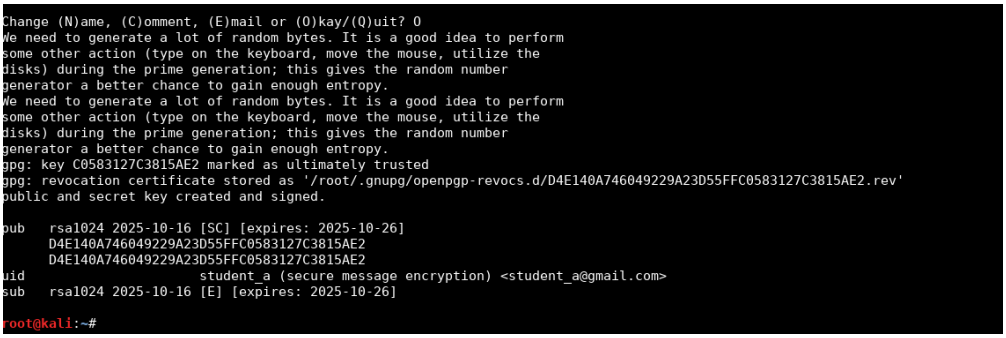
Description:
Encrypting a message and transfer it from metasploit machine to kali using GPG encryption and decryption .

Step 1: Generate GPG Keypair on Kali (Student A)

Commands Executed:

```
$ gpg --full-generate-key
```

Screenshot:



Analysis :

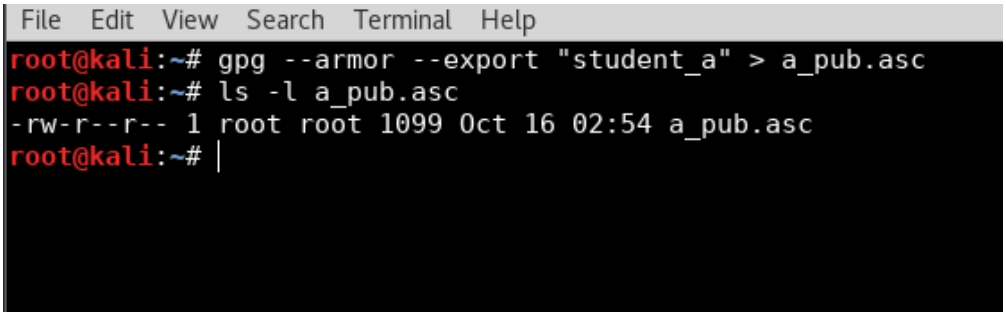
Generating a public key (RSA 1024bits expires in 10days) in order to use it in the encryption and the decryption process .

Step 2: Export Public Key

Commands Executed:

```
$ gpg --armor --export "student_a" > a_pub.asc
$ ls -l a_pub.asc
```

Screenshot:



Analysis :

Exporting the public key of the "student_a" to create a shareable file that will be used in future decryption by the reciever .

Step 3: Transfer Public Key to Metasploitable2

Commands Executed:

```
$ scp a_pub.asc msfadmin@192.168.153.129:/home/msfadmin
```

Screenshot:

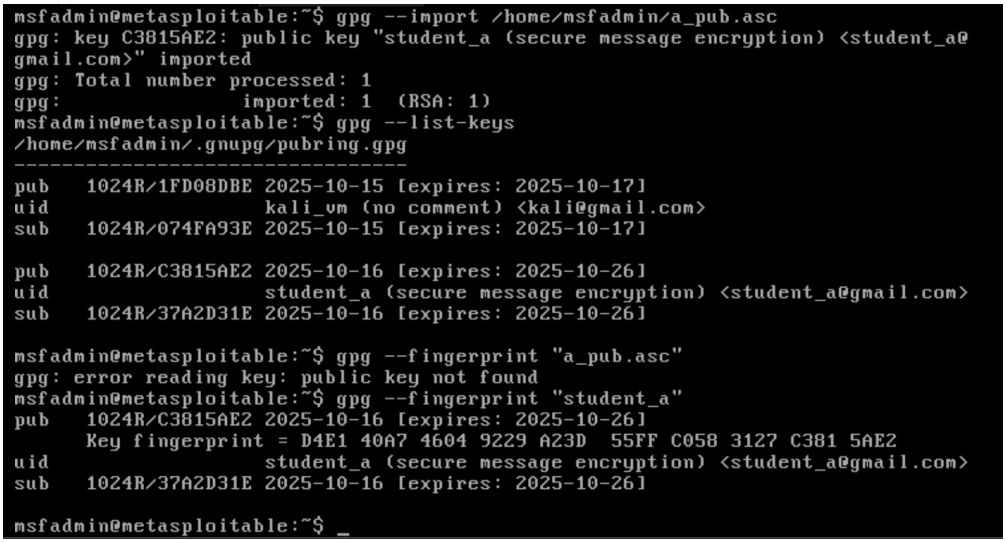
Analysis :
scp command help in copying the key file securely to the metasploit machine in order to use it in decrypting the message .

Step 4: Import and Verify Public Key on Metasploitable2

Commands Executed:

```
$ gpg --import /home/msfadmin/a_pub.asc
$ gpg --list-keys
$ gpg --fingerprint "a_pub.asc"
```

Screenshot:



Analysis :
Importing the key from the specified location to use it later Listing the keys to confirm the successful import GPG creates a shorter or summary hashed version of the key called fingerprint , its like an ID of the key's owner used to verify that the key is not changed by a mim or any other unauthorised entities .

DELIVERABLE - Public Key Fingerprint:

D4e1 40a7 4604 9229 a23d 55ff c058 3127 c381 5ae2

DELIVERABLE - Verification Method:

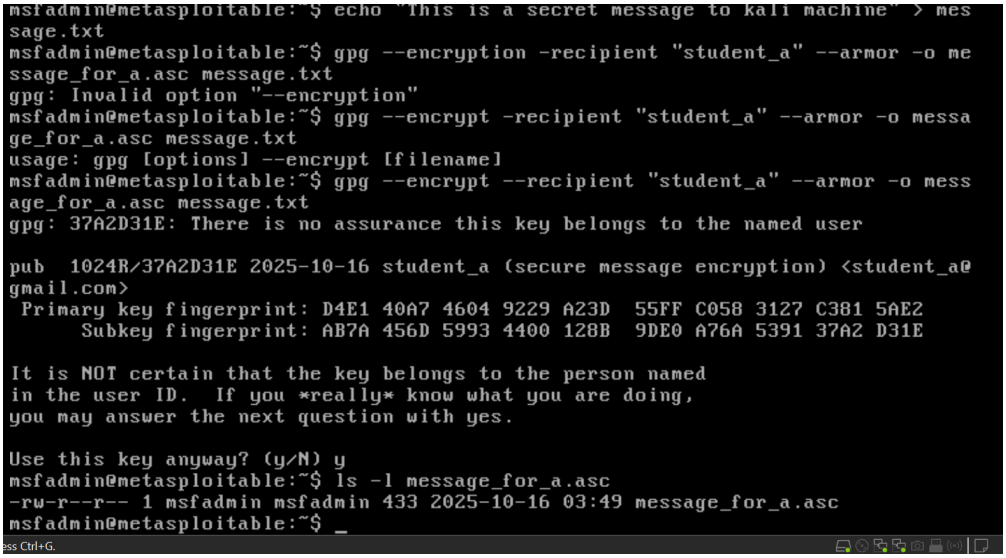
I verified the fingerprint by comparing it in both machines

Step 5: Create and Encrypt Message on Metasploitable2

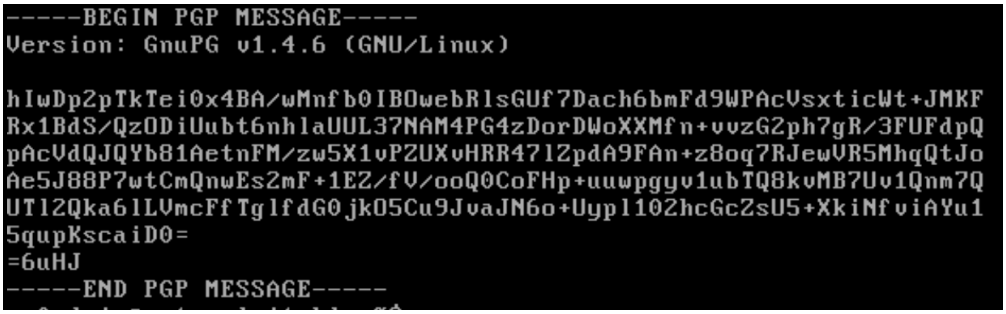
Commands Executed:

```
$ echo "This is a secret message for kali machine" > message.txt
$ gpg --encrypt --recipient "student_a" --armor -o message_for_a.asc message.txt
$ ls -l message_for_a.asc
```

Screenshot:



DELIVERABLE - Encrypted File Content:

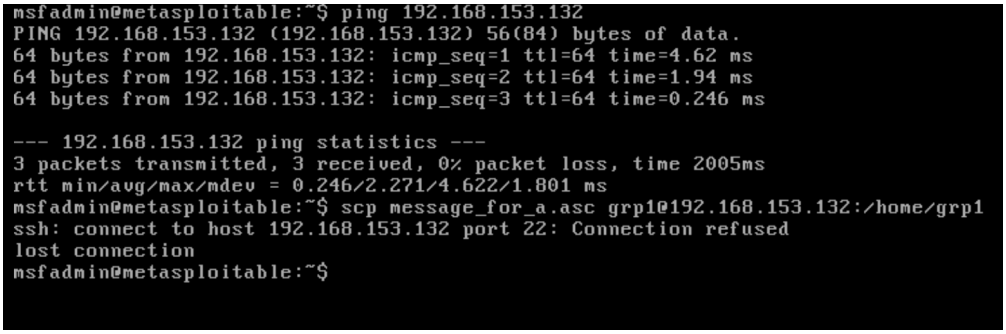


Step 6: Transfer Encrypted File to Kali

Commands Executed:

```
$ scp message_for_a.asc grp1@192.168.153.132:/home/grp1
```

Screenshot:



Analysis :
Successful secure message transfer from metasploit to kali machine with scp command .

Step 7: Decrypt Message on Kali

Commands Executed:

```
$ gpg --decrypt /home/grp1/message_for_a.asc > message.txt && cat message.txt
```

Screenshot:

Analysis:
Successful decryption process using the gpg keys in kali machine . Kali machine asked for the passphrase entered in the first step because the private key is protected .

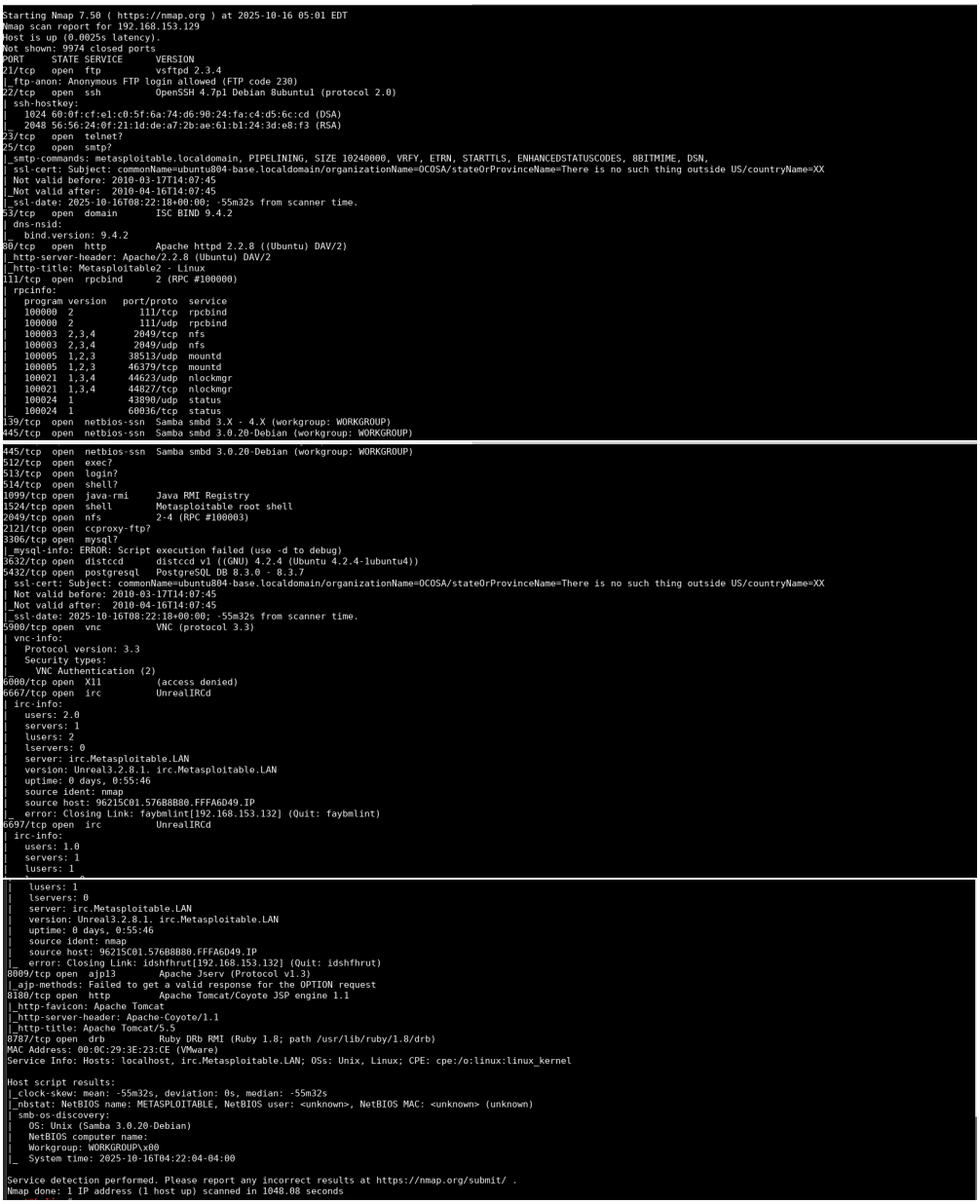
Exercise 2: Reconnaissance & Discovery

Description:
Scanning metasploit machine to look for any authentication related services

Commands Executed:

```
$ nmap -sC -sV -p 1-10000 -o metasploit_scan 192.168.153.129
```

DELIVERABLE - Screenshot of Scan Results:



Authentication Services Identified:

Service	Port
FTP	21
ssh	22

Analysis:

Nmap results show multiple open ports , one of them are related to authentication (FTP & SSH) , these security breaches can lead to multiples security issues like exploiting weakness , brute-forcing credentials , gaining shell access ..etc

Exercise 3: Simple Hashing & Cracking

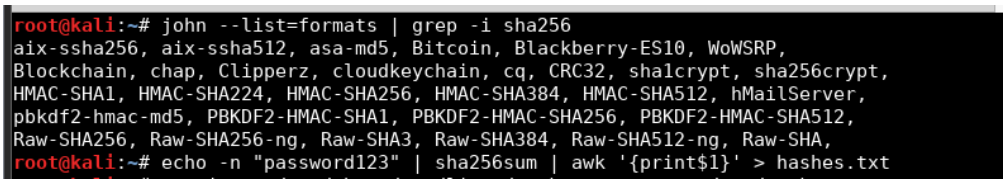
Description:
Password cracking using a word list with John the ripper (password cracking tool)

Task 1: Compute SHA-256 Hash

Commands Executed:

```
$ john --list=formats | grep -i sha256
$ echo -n "password123" | sha256sum | awk '{print$1}' > hashes.txt
```

Screenshot:



Key Findings:

- Chosen password: password123
- SHA-256 hash: ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881f383d4473e94f (in base64)

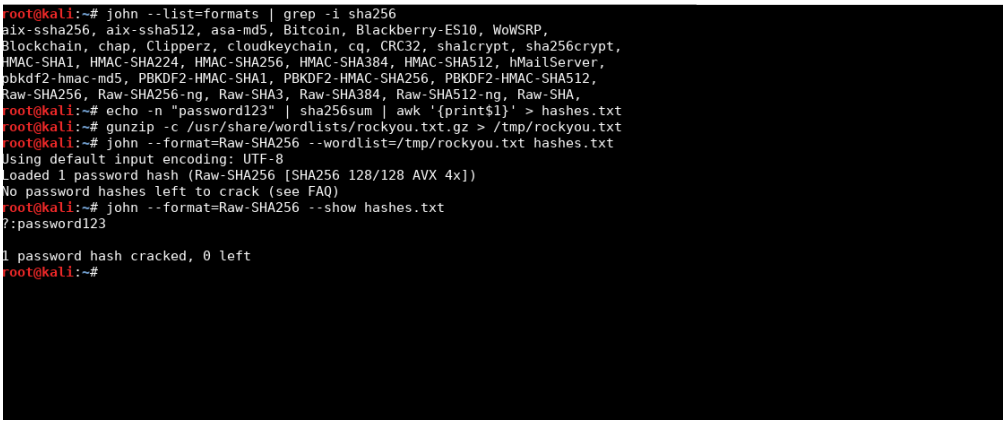
Analysis:
Hashing the password using Raw SHA256 , this method is fast but easy to crack and vulnerable especially to wordlist attacks if the password is common .

Task 2: Crack the Hash

Commands Executed:

```
$ gunzip -c /usr/share/wordlists/rockyou.txt.gz > /tmp/rockyou.txt
$ john --format=Raw-SHA256 --wordlist=/tmp/rockyou.txt hashes.txt
$ john --format=Raw-SHA256 --show hashes.txt
```

Screenshot:



Cracking Result:

?:password123

1 password hash cracked, 0 left

DELIVERABLE - Question 1: Name two solutions to fix the flaw:

1. Using a strong hashing algorithme to ensure the security of the password

2. The use of strong password and non common ones that contains characters (small and caps), numbers, special characters .. etc so that the password cannot be cracked easily .

DELIVERABLE - Question 2: Implementation of chosen solution:

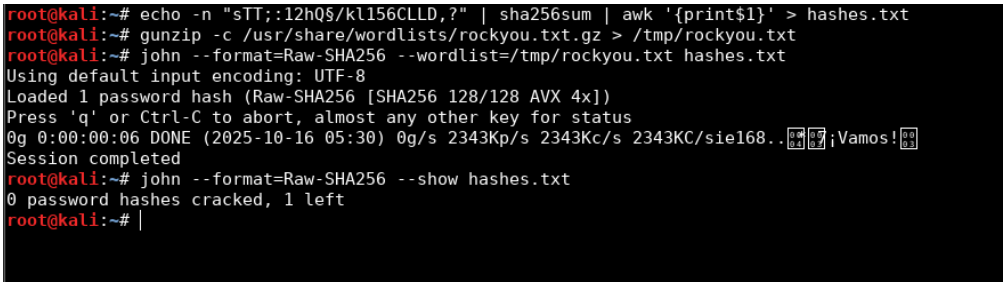
(Describe which solution you chose and how you implemented it)

Using a stronger password like : sTT;:12hQ§/kl156CLLD,?

Commands for Implementation:

```
$ echo -n "sTT;:12hQ§/kl156CLLD,?" | sha256sum | awk '{print$1}' > hashes.txt
```

Screenshot of Implementation:



Analysis:

The use of strong password took a longer time to hash than the weak one . Also john the ripper failed to crack the password as it is not in the word list

Exercise 4: Simulated Online Attack & Defensive iptables Rule

Description:

Simulate a small brute-force using hydra tools and block the attacker's IP with iptables.

Step 1: Create Wordlist

Commands Executed:

```
$ cat > passlist.txt # followed by the list
```

Screenshot:

```
root@kali:~# cat passlist.txt
msfadmin
msf
msf4dmln
password
password123
123456
12345678
1234
12345
123456789
admin
admin123
ftp
ftpuser
anonymous
guest
guest123
toor
root
root123
default
changeme
letmein
welcome
qwerty
qwerty123
pass
passw0rd
passwd
login
user
user123
test
test123
123qwe
abc123
iloveyou
```

Step 2: Launch Brute-Force Attack with Hydra

Commands Executed:

```
$ hydra -l msfadmin -P passlist.txt ftp:/192.168.153.129 -t 4
```

Screenshot:

```
root@kali:~# hydra -l msfadmin -P  passlist.txt ftp://192.168.153.129 -t 4
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2025-10-16 05:54:07
[DATA] max 4 tasks per 1 server, overall 64 tasks, 47 login tries (l:1/p:47), ~0 tries per task
[DATA] attacking service ftp on port 21
[21][ftp] host: 192.168.153.129   login: msfadmin   password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2025-10-16 05:54:14
root@kali:~#
```

Attack Result:

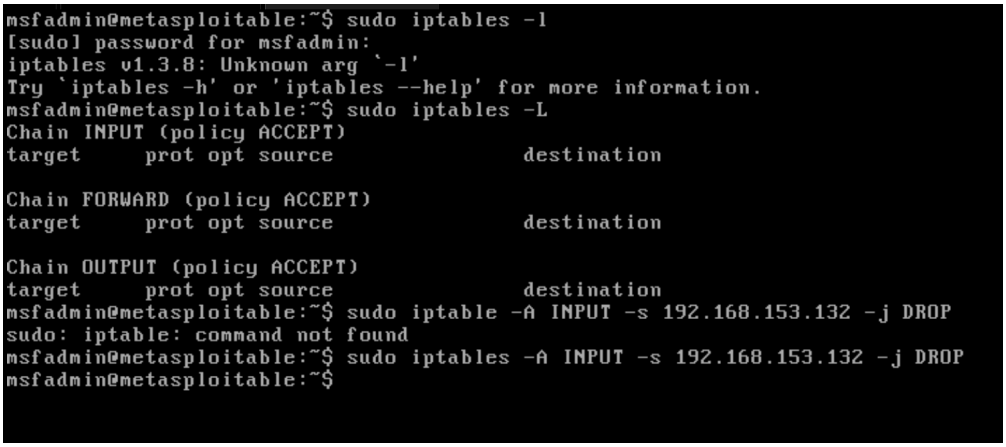
Successful login to the FTP service using the hydra tool

Step 3: Block Kali IP with iptables on Metasploitable2

Commands Executed:

```
$ sudo iptables -A INPUT -s 192.168.153.132 -j DROP
```

Screenshot:



Blocking Rule Applied:

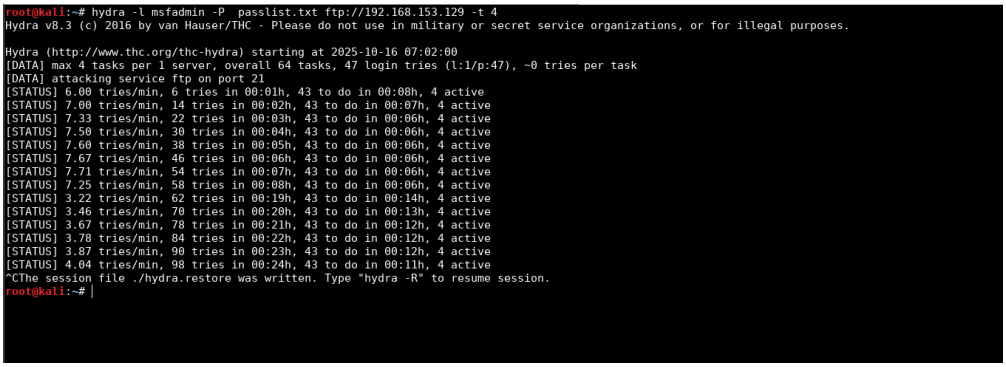
Applied rule is DROP which blocks all the incoming packets from the specified ip address (kali machine's address)

Step 4: Re-run Hydra After Blocking

Commands Executed:

```
$ hydra -l msfadmin -P passlist.txt ftp:/192.168.153.129 -t 4
```

Screenshot:



Result After Blocking:

Failed login attempts because of the blocked ip address , meaning that you can't login to FTP service from that ip again.

Analysis:

Dropping the ip address with iptables in metasploits successfully blocked that ip from logging in remotely to the FTP service . But we still can log in from a different ip , so it is better practice to use a more secure service like FTPS .

Conclusion

Summary:

In this lab we performed many security basics and concepts :

- GPG encryption and how does it work and transfer messages between two users in a secure way
- Discovery and scanning metasploit machine to check any open ports especially authentication ports like FTP and SSH
- Simple password cracking and hashing using password cracking tools like John the ripper and SHA256 hashing algorithm
- Simulate a remote brute force attack on a target machine using hydra tool with a passlist over ftp serivce which was successful .

Security Implications:

- Weak passwords can be cracked easily in short time due to the lack of entropy
- Unencrypted authentication services leak credentials to unauthorized entities
- The lack of firewall rules can lead to unauthorized access and many other security threats

Lessons Learned:

- Always use strong passwords and non guessable ones , and use strong hashing algorithms
- Use encrypted services to prevent credentials leaks
- Use strong and secure rules in the Firewall to keep you system secure .