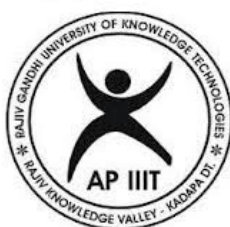


**SHADOWFOX ONE MONTH VIRTUAL
CYBERSECURITY INTERNSHIP
FINAL REPORT TILL ADVANCED LEVEL TASKS**

Submitted by

Kamsala Vandhana

kamsalavandhana07@gmail.com



Department of Computer Science

RGUKT RKV IIIT

VEMPALLI

KADAPA, ANDHRAPRADESH

June 2025 Batch

Assistant Mentor

Mr. Pranshu

ShadowFox

Mentor

Mr. Surendharan

ShadowFox



TABLE OF CONTENTS

Serial Number	Task Level	Title	Page Number
1	Beginner	Port Scanning and Fingerprinting of http://testphp.vulnweb.com/	5
2	Beginner	Directory Enumeration via Brute Force on http://testphp.vulnweb.com/	7
3	Beginner	Intercepting Login Credentials with Wireshark	10
4	Intermediate	VeraCrypt Encrypted File Decryption	13
5	Intermediate	Finding the Entry Point of VeraCrypt Executable	16
6	Advanced	Basic Pentesting Room on TryHackme	18

Introduction

With the rapid expansion of digital infrastructure, cybersecurity has become a critical concern for organizations of all sizes. Modern network environments, web applications, and data centres are frequently targeted by attackers leveraging sophisticated exploitation techniques. Despite the availability of advanced security measures, vulnerabilities persist due to outdated software, poor configurations, and insufficient security assessments. This has led to increased cyber threats, data breaches, and significant financial and reputational losses.

To address these challenges, penetration testing serves as a proactive measure to identify and exploit vulnerabilities before malicious actors can. It involves a structured approach to reconnaissance, vulnerability analysis, exploitation, and post-exploitation to simulate real-world attacks in a controlled environment.

Information about the report

This report is on the various attacks which I performed during my internship at ShadowFox. It includes *Port Scanning*, *Directory Busting*, and *intercepting network traffic via Wireshark* on the test website <http://testphp.vulnweb.com> as the **Beginner Level Tasks**.

It also includes *decrypting a hashed password* and using it to get a secret code from an encrypted VeraCrypt Disk, finding the address of the *entry point* of executable file using the *PE tool* and exploitation of Windows 10 machine using *Metasploit* by getting *reverse shell* access as the **Intermediate Level Tasks**.

This report also explains the severity, impact, steps to reproduce and mitigation steps of each attack performed.

Machines & Tools Used

1. VM Ware Workstation Pro
2. Kali Linux
3. Nmap
4. Dirbuster
5. Wireshark
6. VeraCrypt
7. PE Explorer
8. Windows 10
9. Metasploit
10. Enum4linux

I thereby assure that every attack was performed in a secure and virtual environments, abiding by the ethics of Cybersecurity.

- Swastik Gondhi

BEGINNER LEVEL - TASK 1

Find all the ports that are open on the website

<http://testphp.vulnweb.com/>

- **Attack Name**

- *Port Scanning and Fingerprinting of http://testphp.vulnweb.com/*

- **Severity**

- *CVSS Score: 5.3*
 - *Level: Medium*

- **Impact**

Port scanning and fingerprinting reveal open ports, running services, and versions, which can be further exploited if vulnerabilities exist. In this case, Nmap detected:

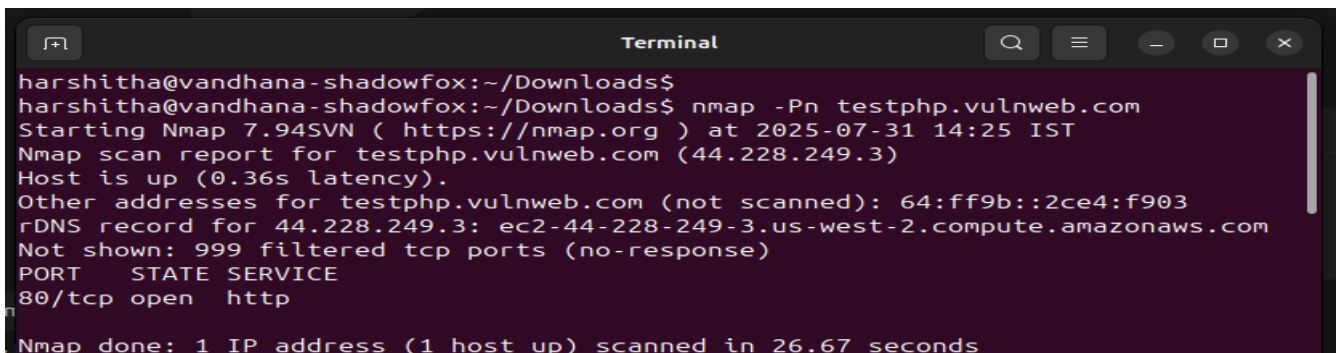
- **Port 80 (HTTP):** *Running on nginx 1.19.0.*
 - *The server is powered by **PHP 5.6.40**, which is outdated and potentially vulnerable.*

This exposure could allow attackers to probe for vulnerabilities in outdated versions of Nginx and PHP, increasing the risk of remote code execution (RCE), information leakage, and denial-of-service (DoS) attacks.

- **Steps to Reproduce**

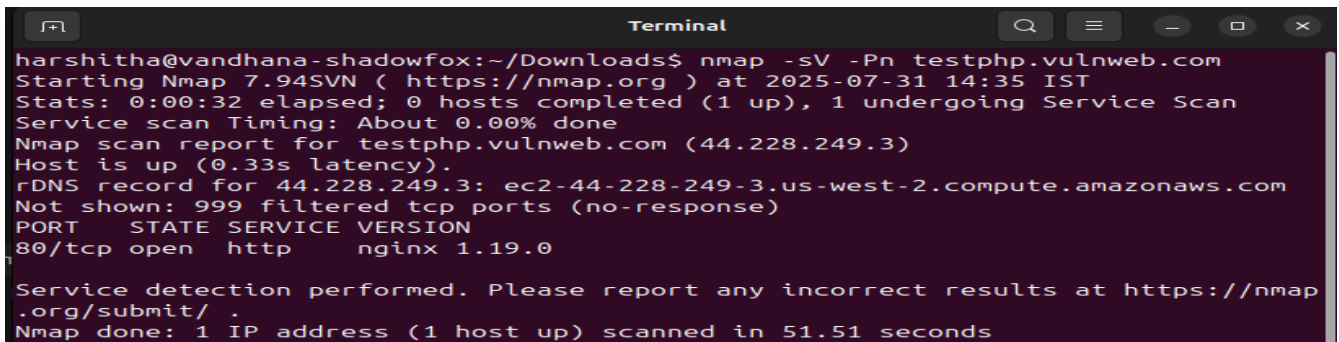
1. Website Fingerprinting:

- The scan revealed that the server is running nginx 1.19.0 on PHP 5.6.40.
- Additional information such as ActiveX, Adobe Flash, and server IP 44.228.249.3 was disclosed.



```
harshitha@vandhana-shadowfox:~/Downloads$
harshitha@vandhana-shadowfox:~/Downloads$ nmap -Pn testphp.vulnweb.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-07-31 14:25 IST
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up (0.36s latency).
Other addresses for testphp.vulnweb.com (not scanned): 64:ff9b::2ce4:f903
rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.com
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 26.67 seconds
```

2. Port Scanning with Nmap:

A terminal window titled "Terminal" showing the output of an Nmap scan. The command executed is `nmap -sV -Pn testphp.vulnweb.com`. The output includes the Nmap version (7.94SVN), the start time (2025-07-31 14:35 IST), and the scan progress (0:00:32 elapsed, 0 hosts completed, 1 up, 1 undergoing Service Scan). The scan report for `testphp.vulnweb.com (44.228.249.3)` shows the host is up with a latency of 0.33s. The rDNS record is `ec2-44-228-249-3.us-west-2.compute.amazonaws.com`. A table of open ports is shown:

PORT	STATE	SERVICE	VERSION
80/tcp	open	http	nginx 1.19.0

. The service detection performed is noted, and a link to report incorrect results is provided. The scan is done in 51.51 seconds.

- o Discovered open **Port 80 (HTTP)** running on nginx 1.19.0.

• Mitigations Steps

1. Update Software Versions:

- o Upgrade nginx to the latest stable version.
- o Upgrade PHP to a more secure version (preferably 8.x) to patch known vulnerabilities.

2. Restrict Information Disclosure:

- o Hide version information in server headers (use `server_tokens off;` in nginx configuration).
- o Remove ActiveX and outdated Adobe Flash if not necessary.

3. Firewall Configurations:

- o Apply firewall rules to limit port exposure to only necessary ones.

4. Run Regular Vulnerability Scans:

- o Perform regular scans to identify outdated services and vulnerabilities.

BEGINNER LEVEL – TASK 2

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

- **Attack Name**
 - *Directory Enumeration via Brute Force on <http://testphp.vulnweb.com/>*
- **Severity**
 - *CVSS Score: 7.5*
 - *Level: High*
- **Impact**

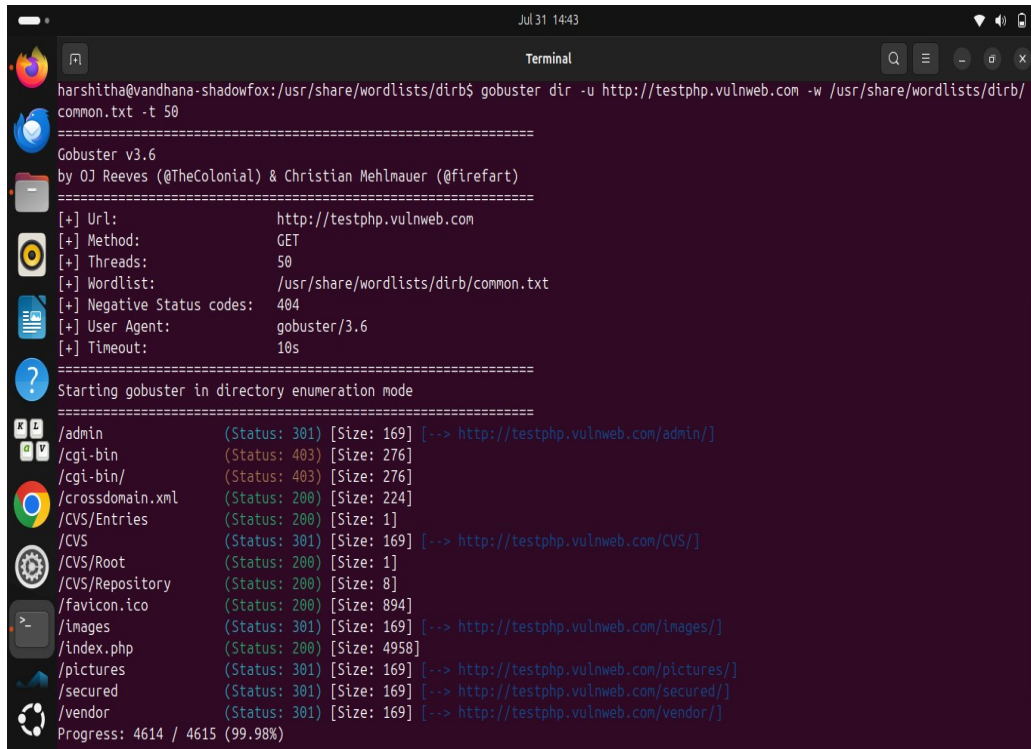
The directory brute force attack exposed sensitive directories that may contain configuration files, admin panels, and source code repositories. This increases the attack surface, allowing potential access to:

- `/admin/` - Possible admin panel (403 Forbidden, but visible)
- `/cgi-bin/` - Common directory for executing scripts (403 Forbidden, but visible)
- `/CVS/` - Version control repository exposing Entries, Repository, and Root
- `/crossdomain.xml` - Cross-domain policy file, potentially exposing configurations
- `/images/` and `/pictures/` - Image directories that may leak sensitive content
- `/check.php` - A script that is directly accessible (Status 200, Size: 4958)

Exposed CVS directories may allow attackers to access historical changes, configurations, and even source code, increasing the risk of source code disclosure and configuration weaknesses.

- **Steps to Reproduce**

1. Directory Brute Forcing using Dirb:



```
harshitha@vandhana-shadowfox: /usr/share/wordlists/dirb$ gobuster dir -u http://testphp.vulnweb.com -w /usr/share/wordlists/dirb/common.txt -t 50

=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://testphp.vulnweb.com
[+] Method: GET
[+] Threads: 50
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s
=====
Starting gobuster in directory enumeration mode
=====
/admin (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/admin/]
/cgi-bin (Status: 403) [Size: 276]
/cgi-bin/ (Status: 403) [Size: 276]
/crossdomain.xml (Status: 200) [Size: 224]
/CVS/Entries (Status: 200) [Size: 1]
/CVS (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/CVS/]
/CVS/Root (Status: 200) [Size: 1]
/CVS/Repository (Status: 200) [Size: 8]
/favicon.ico (Status: 200) [Size: 894]
/images (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/images/]
/index.php (Status: 200) [Size: 4958]
/pictures (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/pictures/]
/secured (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/secured/]
/vendor (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/vendor/]
Progress: 4614 / 4615 (99.98%)
```

2. Analysis of Results:

Accessible Directories:

- /admin
- /CVS
- /images
- /pictures

- **Mitigation Steps:**

1. **Restrict Directory Access:**

- o Apply proper .htaccess rules to deny directory listing and access to sensitive directories like /admin/, /cgi-bin/, and /CVS/.

2. **Disable Unused Services:**

- o If cgi-bin is not in use, disable it from the web server configuration.

3. **Secure Version Control Paths:**

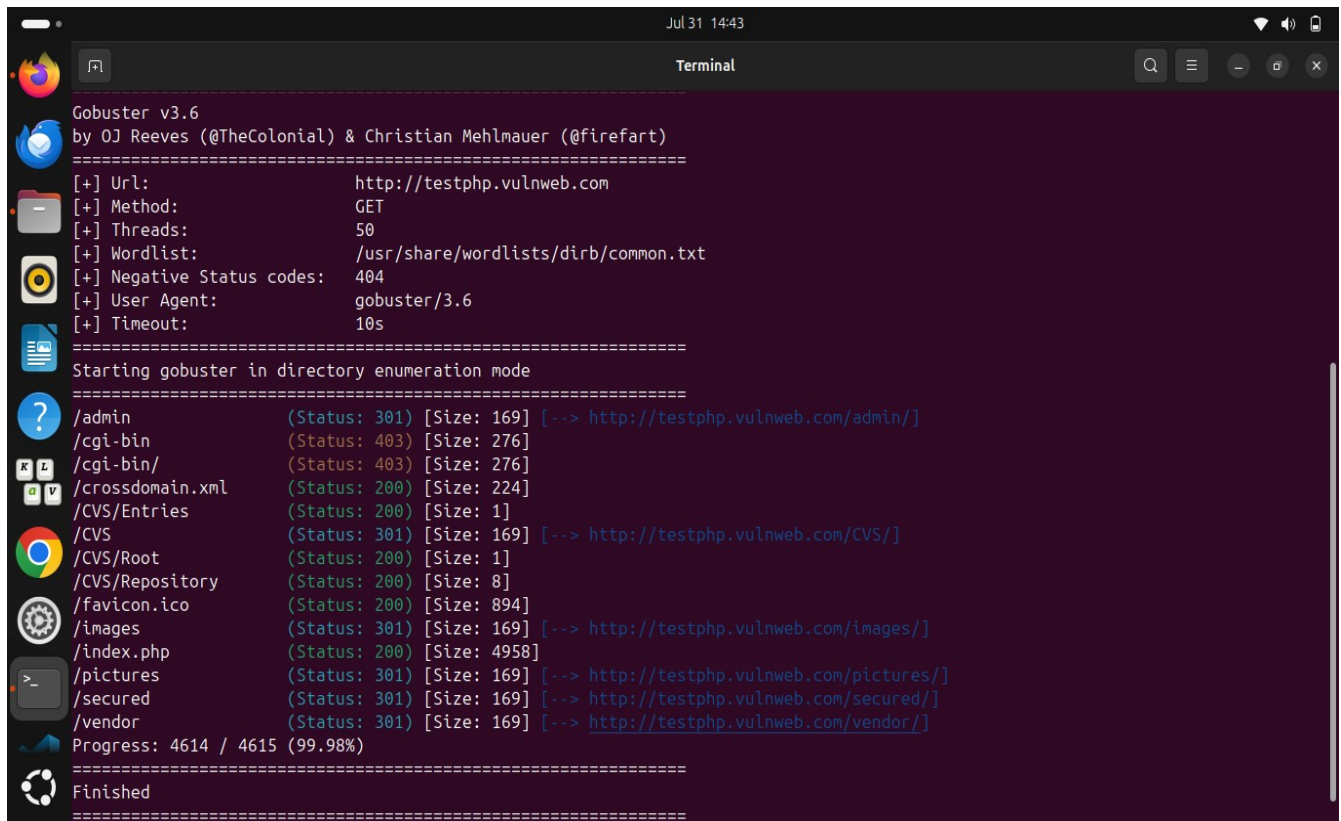
- o Ensure version control paths (/CVS/) are not publicly accessible.

4. **Validate Cross-Domain Policies:**

- o Ensure `crossdomain.xml` is securely configured to allow only trusted domains.

5. Regular Security Audits:

- o Perform regular scans and audits to identify exposed paths and sensitive directories.



```
Jul 31 14:43
Terminal

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://testphp.vulnweb.com
[+] Method:       GET
[+] Threads:      50
[+] Wordlist:      /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent:    gobuster/3.6
[+] Timeout:      10s
=====
Starting gobuster in directory enumeration mode
=====
/admin          (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/admin/]
/cgi-bin        (Status: 403) [Size: 276]
/cgi-bin/       (Status: 403) [Size: 276]
/crossdomain.xml (Status: 200) [Size: 224]
/CSV/Entries    (Status: 200) [Size: 1]
/CSV            (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/CSV/]
/CSV/Root       (Status: 200) [Size: 1]
/CSV/Repository (Status: 200) [Size: 8]
/favicon.ico    (Status: 200) [Size: 894]
/images         (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/images/]
/index.php      (Status: 200) [Size: 4958]
/pictures       (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/pictures/]
/secured        (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/secured/]
/vendor         (Status: 301) [Size: 169] [-> http://testphp.vulnweb.com/vendor/]
Progress: 4614 / 4615 (99.98%)
=====
Finished
=====
```


BEGINNER LEVEL – TASK 3

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.

- **Attack Name**

- *Intercepting Login Credentials with Wireshark*

- **Severity**

- *CVSS Score: 7.5*

- *Level: High*

- **Impact**

The attack allows interception of plain-text credentials (username and password) transmitted over an unencrypted HTTP connection. An attacker positioned within the same network (Man-in-the-Middle) can easily capture sensitive information, leading to unauthorized access and potential data breaches.

- **Steps to Reproduce**

1. **Navigate to the Target Website:**

- o Open a browser and go to `http://testphp.vulnweb.com/`.

2. **Initiate a Login Attempt:**

- o Fill in the username and password fields with sample credentials and submit the form.

3. **Launch Wireshark:**

- o Open Wireshark and start capturing traffic on the active network interface.

4. **Filter the Traffic:**

- o Use the display filter `http` to isolate HTTP traffic.

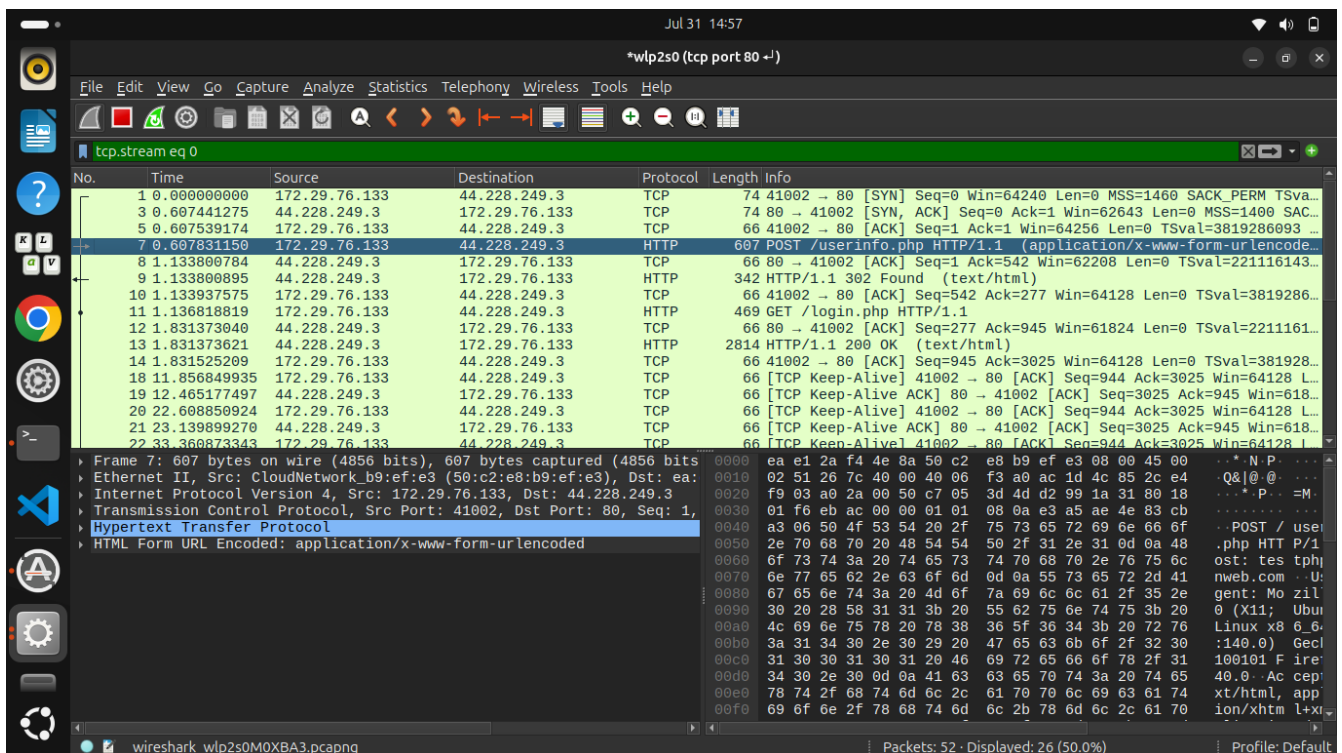
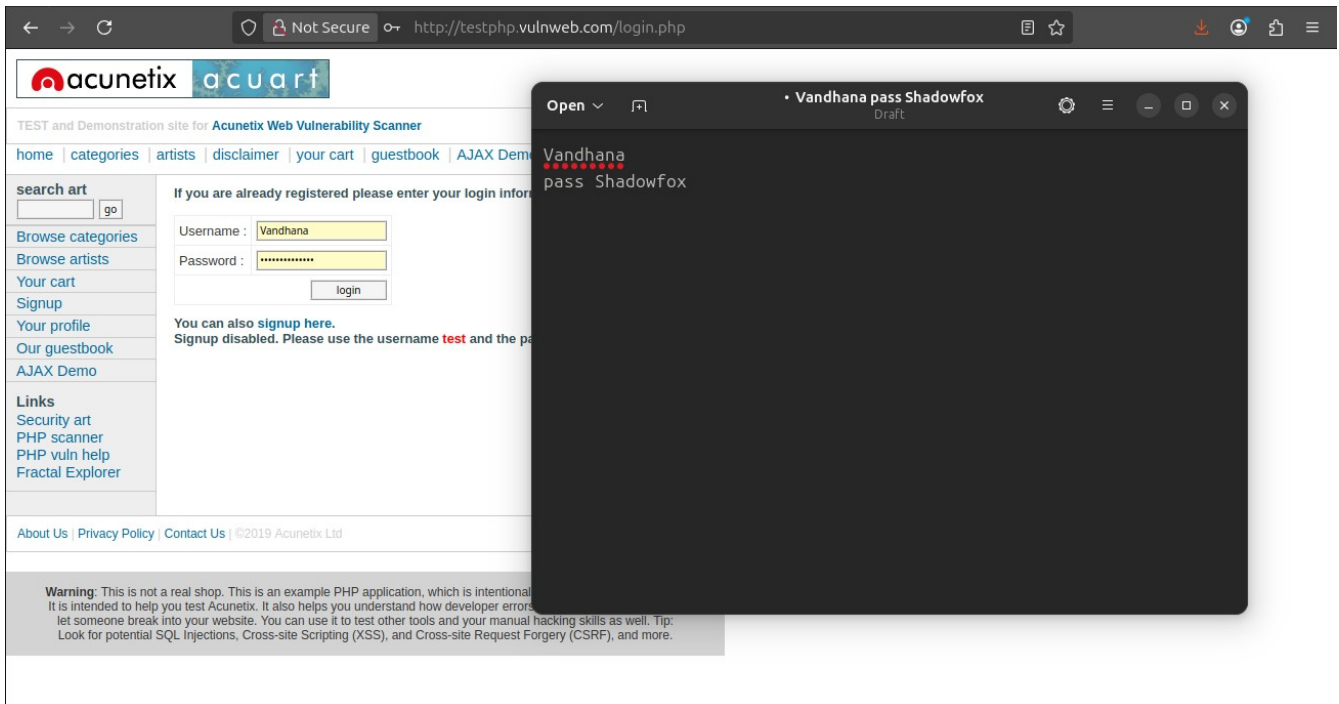
5. **Locate the Login Request:**

- o Search for a POST request to `/login.php` or similar endpoint.

- o Inspect the packet to find the username and password parameters in plain text.

6. Capture the Credentials:

- o Right-click the packet → Follow → HTTP Stream.
- o The credentials should be visible in the stream.



• Mitigation Steps

1. Enforce HTTPS:

- o Use SSL/TLS to encrypt HTTP traffic and prevent credential interception.

2. Use Secure Cookies:

- o Mark cookies as Secure and HttpOnly to avoid exposure in unencrypted channels.

3. Implement Strong Authentication Mechanisms:

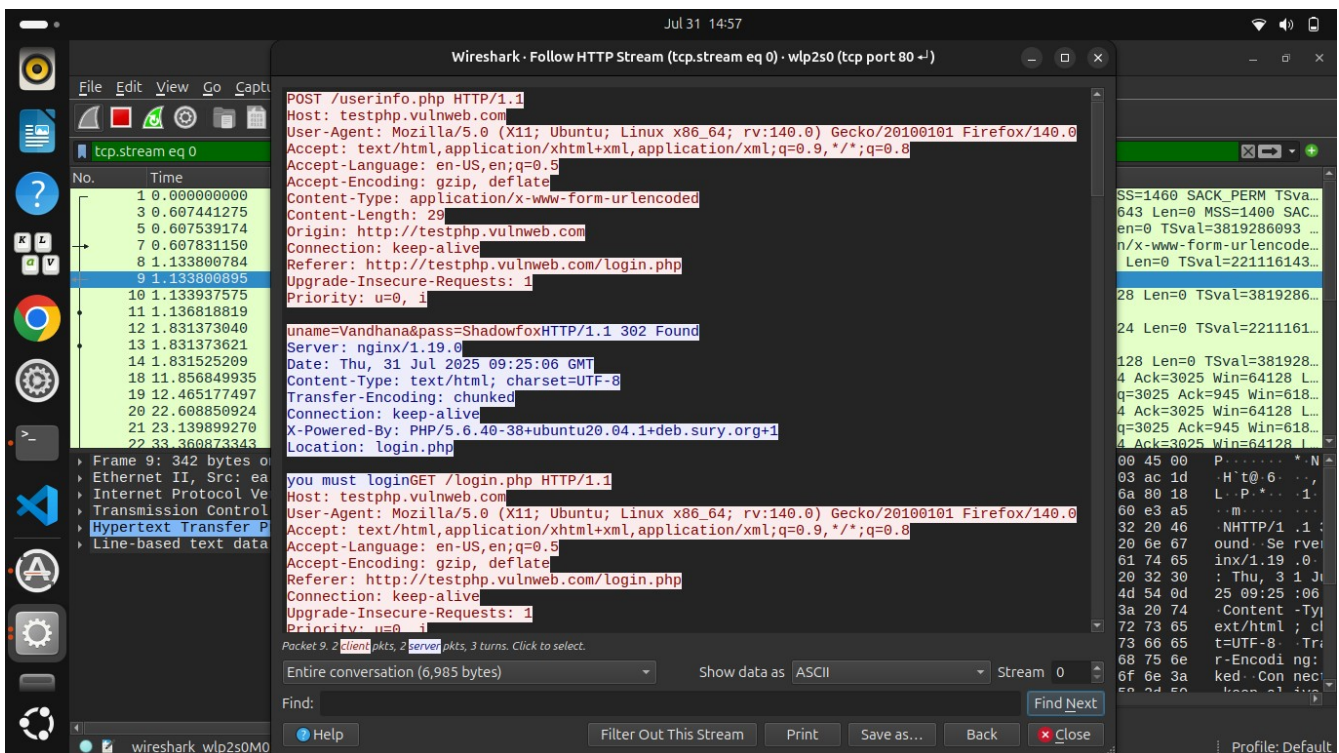
- o Utilize multi-factor authentication (MFA) to add a second layer of protection.

4. Network Segmentation:

- o Isolate critical applications from public networks to reduce exposure.

5. Regular Monitoring:

- o Continuously monitor network traffic for signs of interception or anomalies.



INTERMEDIATE LEVEL – TASK 1

A file is encrypted using VeraCrypt (A disk encryption tool). The password to access the file is encrypted in a hash format and provided to you in the drive with the name encoded.txt. Decode the password and enter in the vera crypt to unlock the file and find the secret code in it.

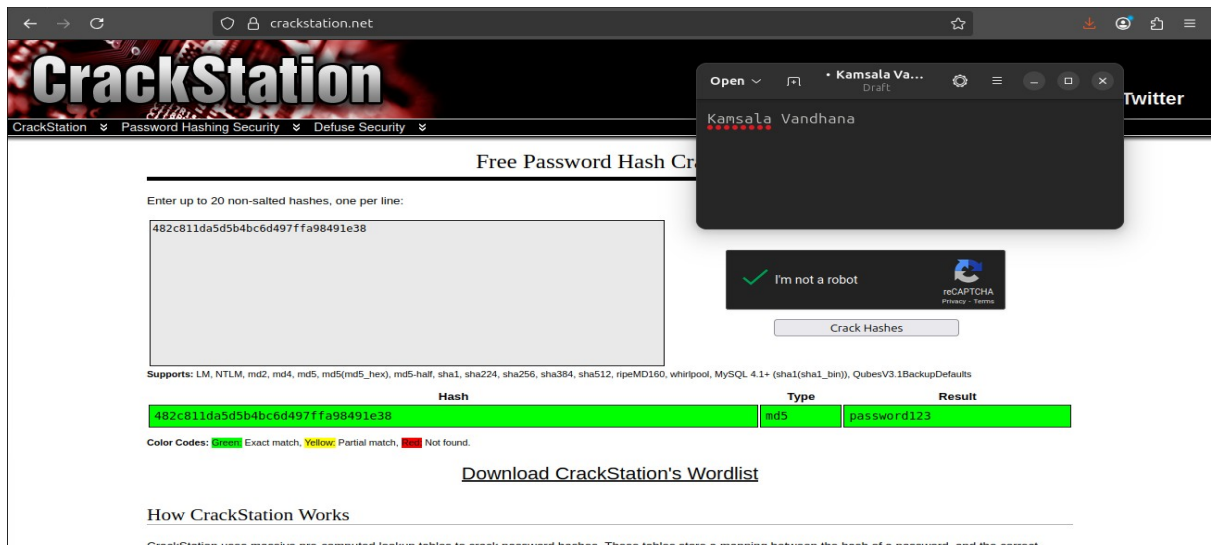
- **Attack Name**
 - *VeraCrypt Encrypted File Decryption*
- **Severity**
 - *CVSS Score: 6.8*
 - *Level: Medium*
- **Impact**

The attack demonstrates the ability to decrypt a password-protected file container using VeraCrypt if the hash of the password is known and can be cracked. This exposes sensitive information if password hashes are not properly secured.

- **Steps to Reproduce**

1. **Locate the Encoded Password File:**
 - o Access the drive and locate encoded.txt which contains the hashed password.
2. **Decode the Hash:**
 - o Use the online tool md5hashing.net to decode the hash value.
 - o The password was successfully decoded as password123.

The screenshot shows the CrackStation website in a browser. The page title is 'Free Password Hash Cracker'. It features a large text input field for entering hashes, with the example '482c811da5d5b4bc6d497ffa98491e38' pasted in. To the right of the input field is a reCAPTCHA widget with the text 'I'm not a robot' and a 'Crack Hashes' button. Below the input field, there is a list of supported hash types: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, rpeMD160, whirlpool, MySQL 4.1+ (sha1 sha1_bin), QubesV3.1BackupDefaults. At the bottom, there is a link to 'Download CrackStation's Wordlist' and a section titled 'How CrackStation Works' which explains the use of pre-computed lookup tables.



3. Download and Install VeraCrypt:

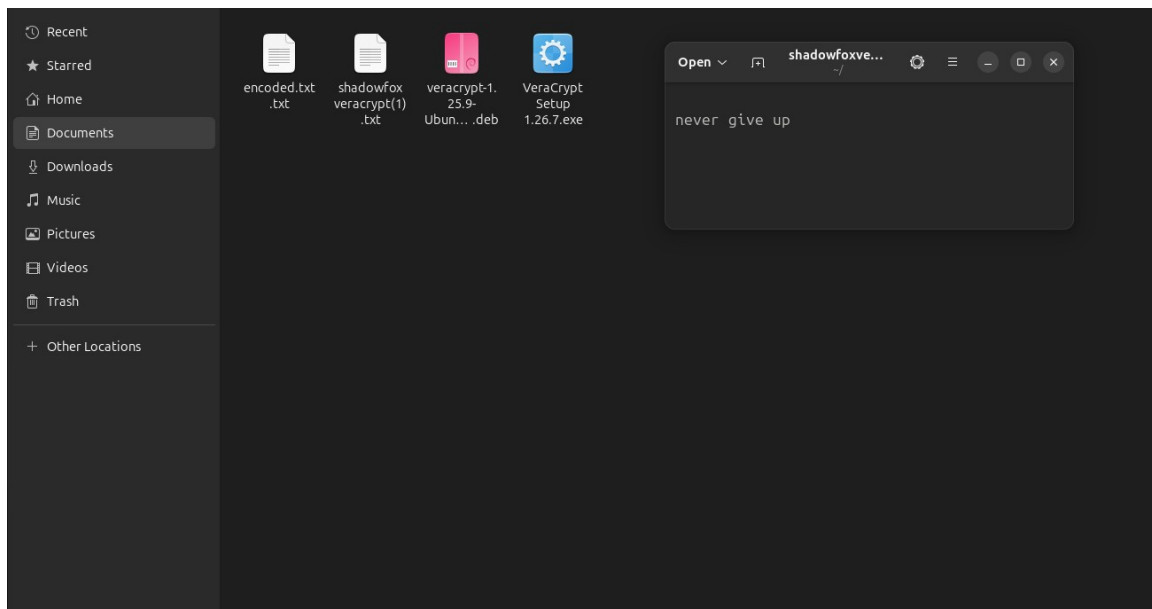
- o Install VeraCrypt and mount the encrypted container.

4. Enter the Decoded Password:

Input password is password123, entered the password

5. Access the Secret Code:

Upon successful decryption, open the file and retrieve the secret code:
never give up



- **Mitigation Steps**

1. **Use Strong Passwords:**

- o Avoid using simple, easily crackable passwords. Implement password complexity policies.

2. **Hash with Salt:**

- o Always hash passwords with a unique salt value to prevent hash-based attacks.

3. **Limit Hash Exposure:**

- o Store hashed passwords securely and avoid exposing them unnecessarily.

4. **Multi-Factor Authentication:**

- o Implement MFA to add an additional layer of security to encrypted files.

5. **Monitor Access Logs:**

- o Regularly review access logs to detect unauthorized access attempts.

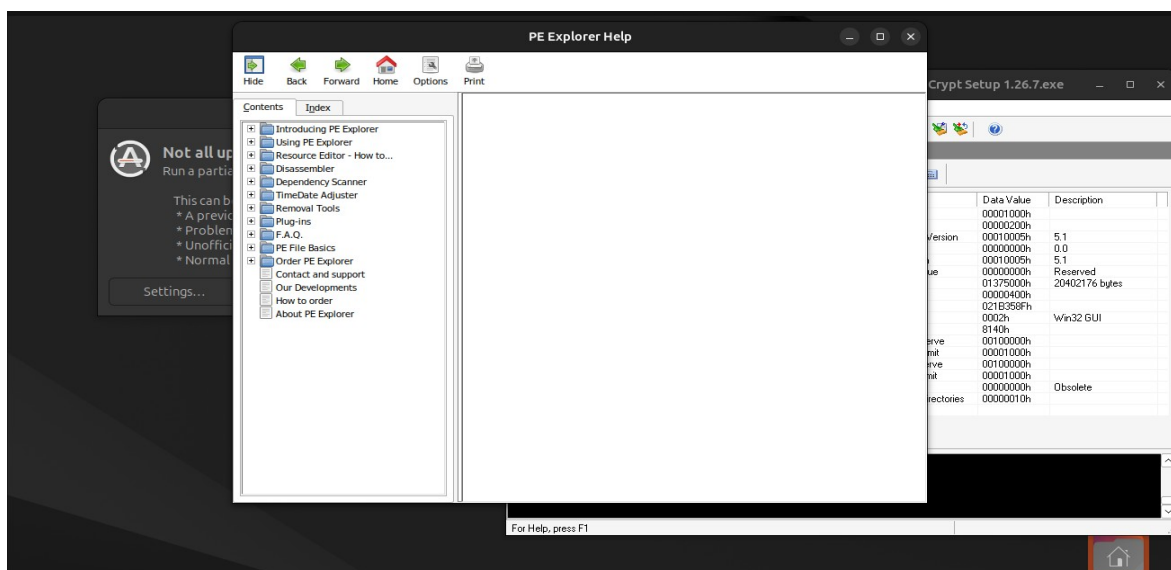
INTERMEDIATE LEVEL – TASK 2

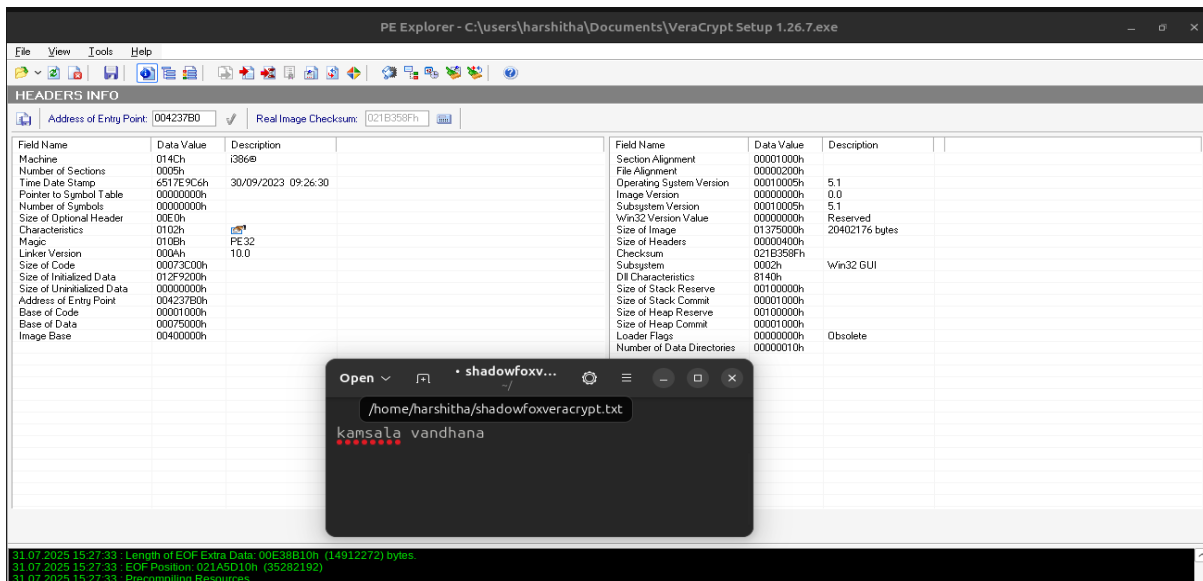
An executable file of VeraCrypt will be provided to you. Find the address of the entry point of the executable using PE explorer tool and provide the value as the answer as a screenshot

- **Attack Name**
 - *Finding the Entry Point of VeraCrypt Executable*
- **Severity**
 - *CVSS Score: 6.8*
 - *Level: Medium*
- **Impact**

Identifying the entry point of an executable is crucial for reverse engineering and vulnerability analysis. Gaining this information helps in understanding the program's control flow and potential attack vectors for exploitation.

- **Steps to reproduce**
 1. **Obtain VeraCrypt Executable:**
 - Download or access the VeraCrypt executable file.
 2. **Open PE Explorer:**
 - Launch the PE Explorer tool.
 3. **Upload the VeraCrypt Executable:**
 - Load the executable into PE Explorer.





4. Locate the Entry Point:

- o Navigate to the headers section and identify the **Address of Entry Point (AEP)**.

5. Record the Address:

- o Note the value displayed as the entry point address.

• Mitigation Steps

1. Binary Obfuscation:

- o Use obfuscation techniques to make it harder to identify entry points.

2. Packers and Encryptors:

- o Apply binary packers to complicate reverse engineering efforts.

3. Runtime Checks:

- o Implement runtime verification to detect tampering or unauthorized access.

4. Regular Updates:

- o Keep VeraCrypt and PE Explorer tools updated to prevent exploitation through known vulnerabilities.

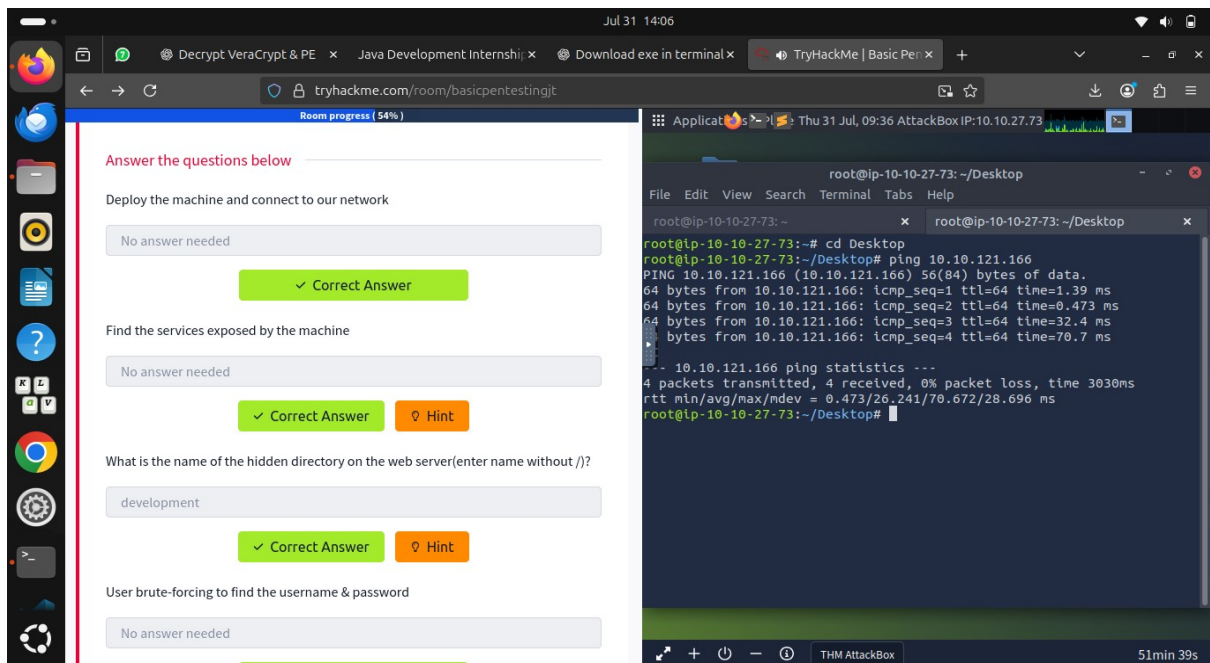
5. Limit Executable Access:

- o Restrict access to executable files and monitor any modifications.

ADVANCED LEVEL TASK

TryHackMe's Basic Pentesting Room

- **Attack Name**
 - *Penetration Test on the Basic Pentesting Room*
- **Severity**
 - *Level: High*
- **Impact**
 1. Unauthorized remote shell access to the target machine.
 2. Disclosure of sensitive user information via SMB shares.
 3. Exposure of login portal and internal files.
 4. Full root access gained.
- **Steps to reproduce**
 1. Checking whether target is alive or not



2. Scanning for open ports & services

The screenshot shows a web browser window with the URL `tryhackme.com/room/basicpentestingjt`. The page displays a progress bar at 63% and a list of tasks. The first task, "Deploy the machine and connect to our network", is completed. The second task, "Find the services exposed by the machine", is also completed. The third task, "What is the name of the hidden directory on the web server (enter name without /)?", is completed with the answer "development". The fourth task, "User brute-forcing to find the username & password", is completed. The fifth task, "What is the username?", is currently in progress. On the right, a terminal window shows the output of an nmap scan on 10.10.121.166, identifying open ports 22, 80, 139, 445, 8080 and services ssh, http, netbios-ssn, microsoft-ds, and ajp13.

Room progress (63%)

Deploy the machine and connect to our network

No answer needed

✓ Correct Answer

Find the services exposed by the machine

No answer needed

✓ Correct Answer Hint

What is the name of the hidden directory on the web server (enter name without /)?

development

✓ Correct Answer Hint

User brute-forcing to find the username & password

No answer needed

✓ Correct Answer

What is the username?

```
root@ip-10-10-27-73: ~
File Edit View Search Terminal Tabs Help
root@ip-10-10-27-73: ~
root@ip-10-10-27-73: ~# nmap 10.10.121.166
Starting Nmap 7.80 ( https://nmap.org ) at 2025-07-31 09:37 BST
Nmap scan report for ip-10-10-121-166.eu-west-1.compute.internal (10.10.121.166)
Host is up (0.040s latency).
Not shown: 994 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
8080/tcp   open  ajp13
Nmap done: 1 IP address (1 host up) scanned in 1.82 seconds
root@ip-10-10-27-73: ~#
```

3. Directory fuzzing to find hidden directories

The screenshot shows the same TryHackMe room interface as before. The progress bar is still at 63%. The tasks are the same, but the terminal window now shows the output of a gobuster scan on 10.10.121.166, which has found a hidden directory named "/development".

Room progress (63%)

Deploy the machine and connect to our network

No answer needed

✓ Correct Answer

Find the services exposed by the machine

No answer needed

✓ Correct Answer Hint

What is the name of the hidden directory on the web server (enter name without /)?

development

✓ Correct Answer Hint

User brute-forcing to find the username & password

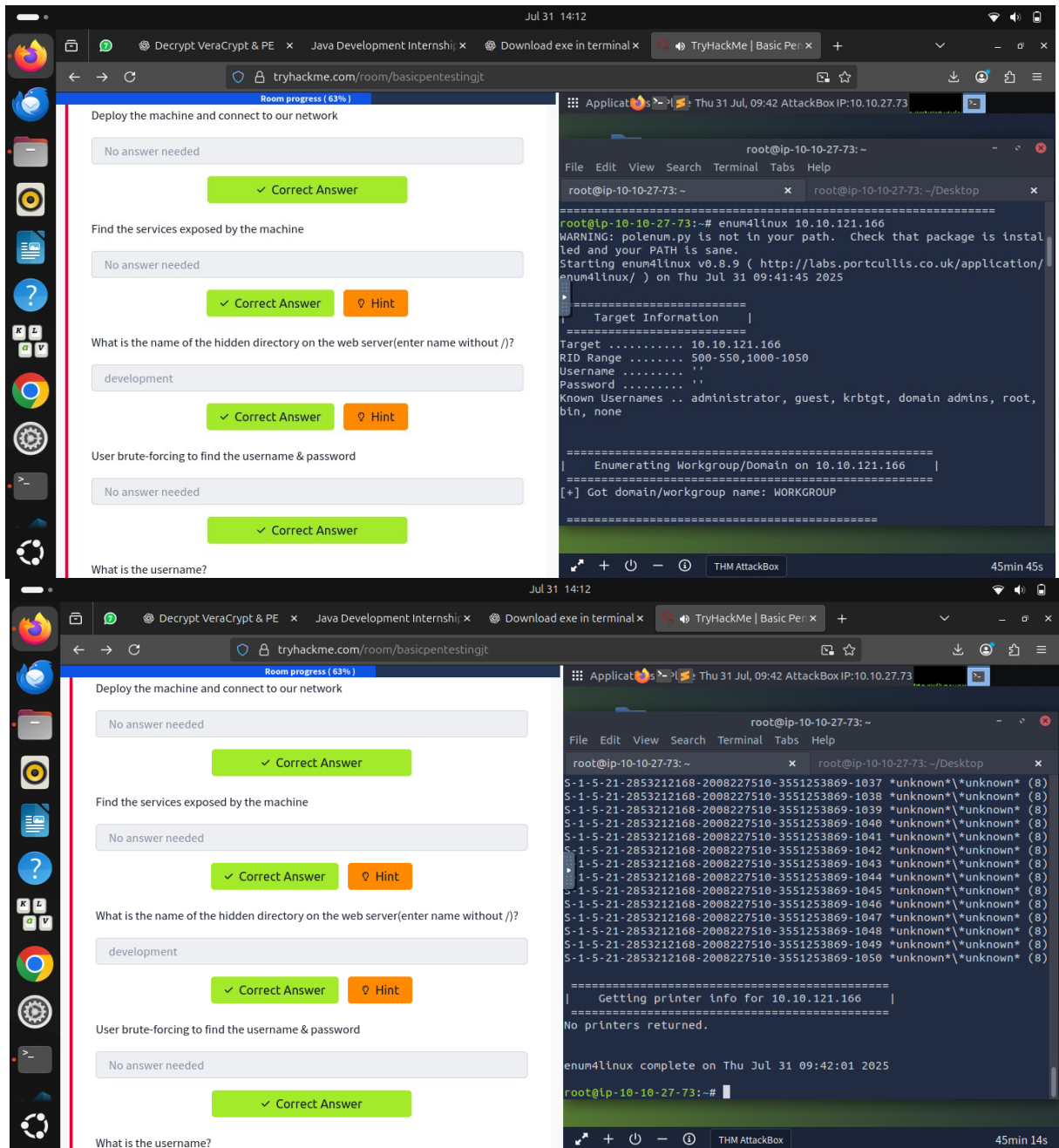
No answer needed

✓ Correct Answer

What is the username?

```
root@ip-10-10-27-73: ~
File Edit View Search Terminal Tabs Help
root@ip-10-10-27-73: ~
root@ip-10-10-27-73: ~# gobuster dir -u http://10.10.121.166 -w /usr/share/wordlists/dirbuster/directory-list-1.0.txt
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehrlauer (@firefart)
=====
[+] Url: http://10.10.121.166
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-1.0.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s
=====
Starting gobuster in directory enumeration mode
=====
/development (Status: 301) [Size: 320]
Progress: 141708 / 141709 (100.00%)
Finished
root@ip-10-10-27-73: ~#
```

4. Username & password Bruteforcing using enum4linux



5. Logging in to user using SSH service

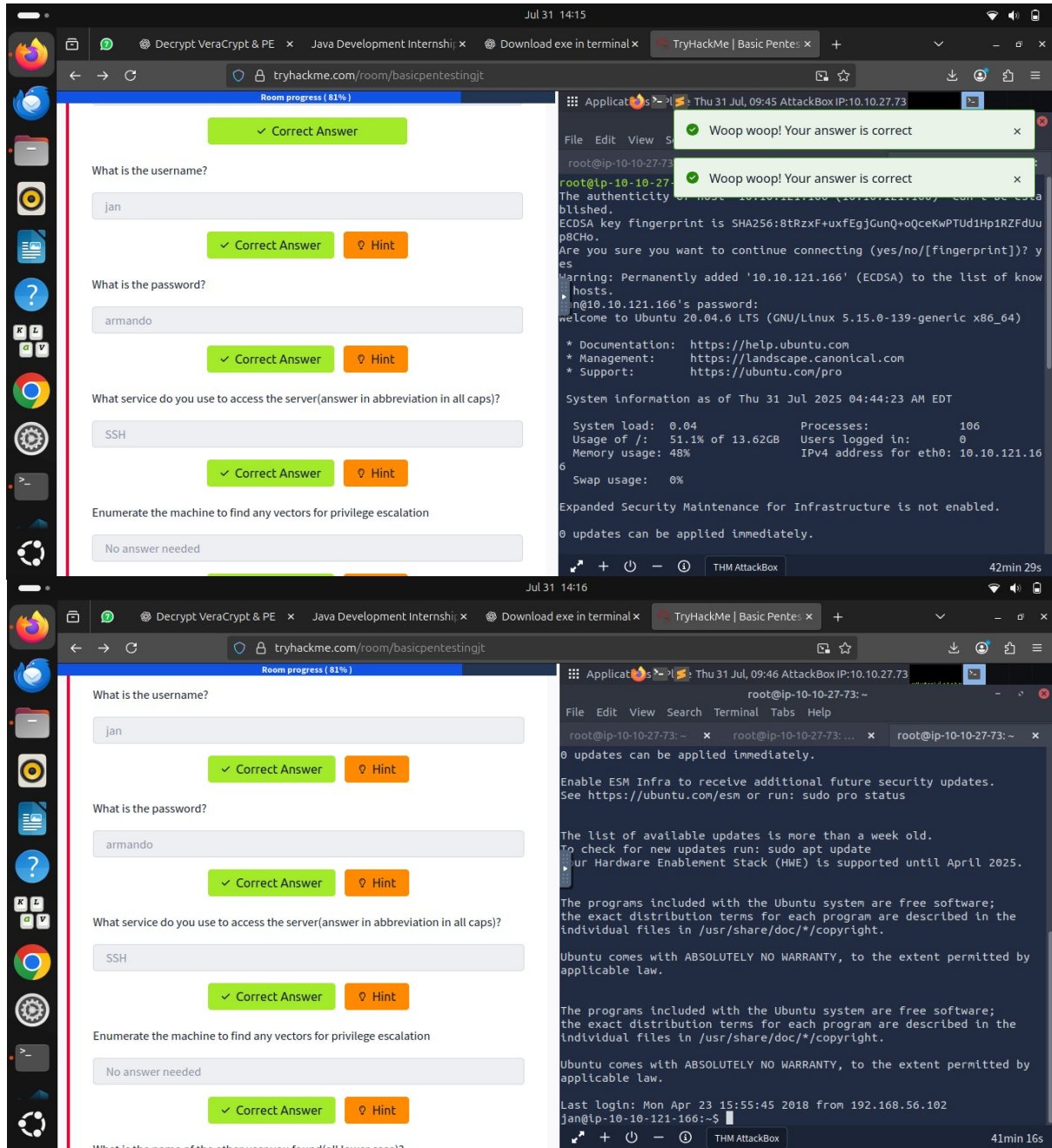
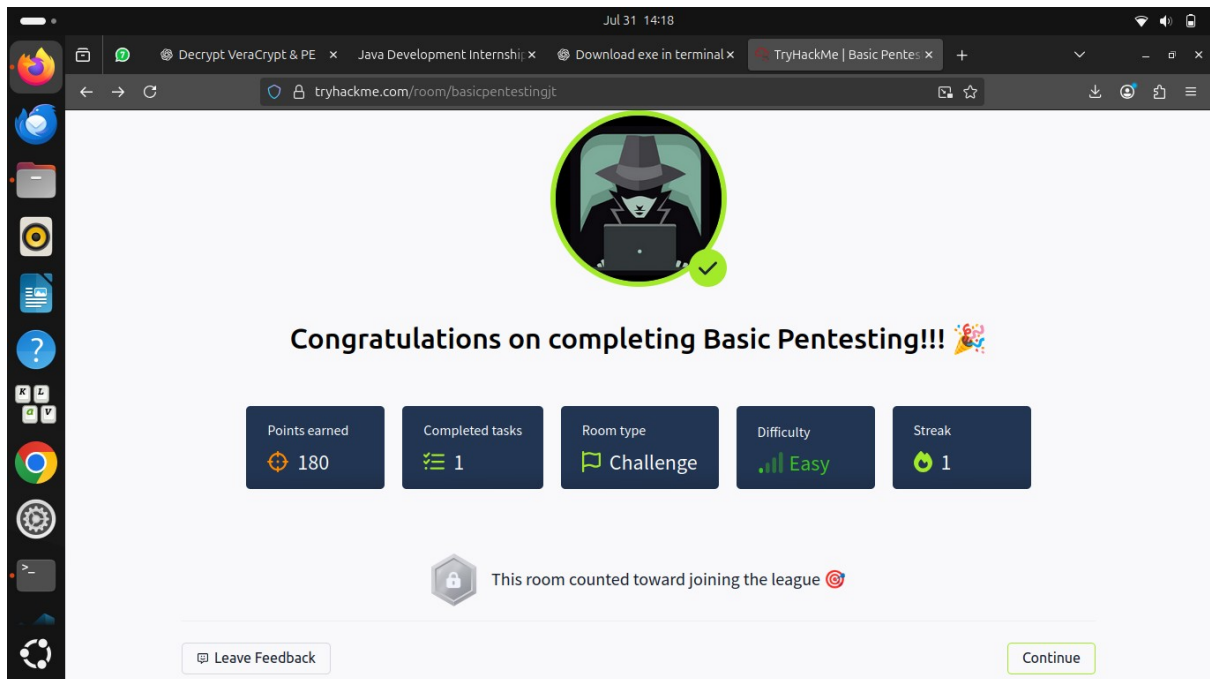


Figure 29 - SSH Login



- **Mitigation Steps**

1. Implement account lockout policy for repeated failed login attempts.
2. Disable SSH for default or weak accounts.
3. Enforce strong password policies and use key-based authentication.
4. Implement account lockout policy for repeated failed login attempts.
5. Disable SSH for default or weak accounts.
6. Enforce strong password policies and use key-based authentication.

END OF THE REPORT