Internship Final Report – Cyber Security (Red Teaming)

Student Details

- Name: Mostafa karam

- University: Faculty of Computer and Data Science

- Major: Computer Science / Cybersecurity

- Internship Duration: March 25, 2025 – June 25, 2025

- Company: Hack Secure

- Domain: Cybersecurity Red Teaming & Ethical Hacking

- Mentor: Mr. Nishant Prajapati

- Assistant Mentor: Mr. Aman Pandey

- Coordinator: Mr. Shivam Kapoor

Objectives

The objectives of my internship were to:

1. Strengthen practical cybersecurity skills through hands-on red team tasks.

2. Understand and apply penetration testing techniques.

3. Complete real-world exploit challenges and CTF labs.

4. Develop and deliver 3 functional Python-based security tools.

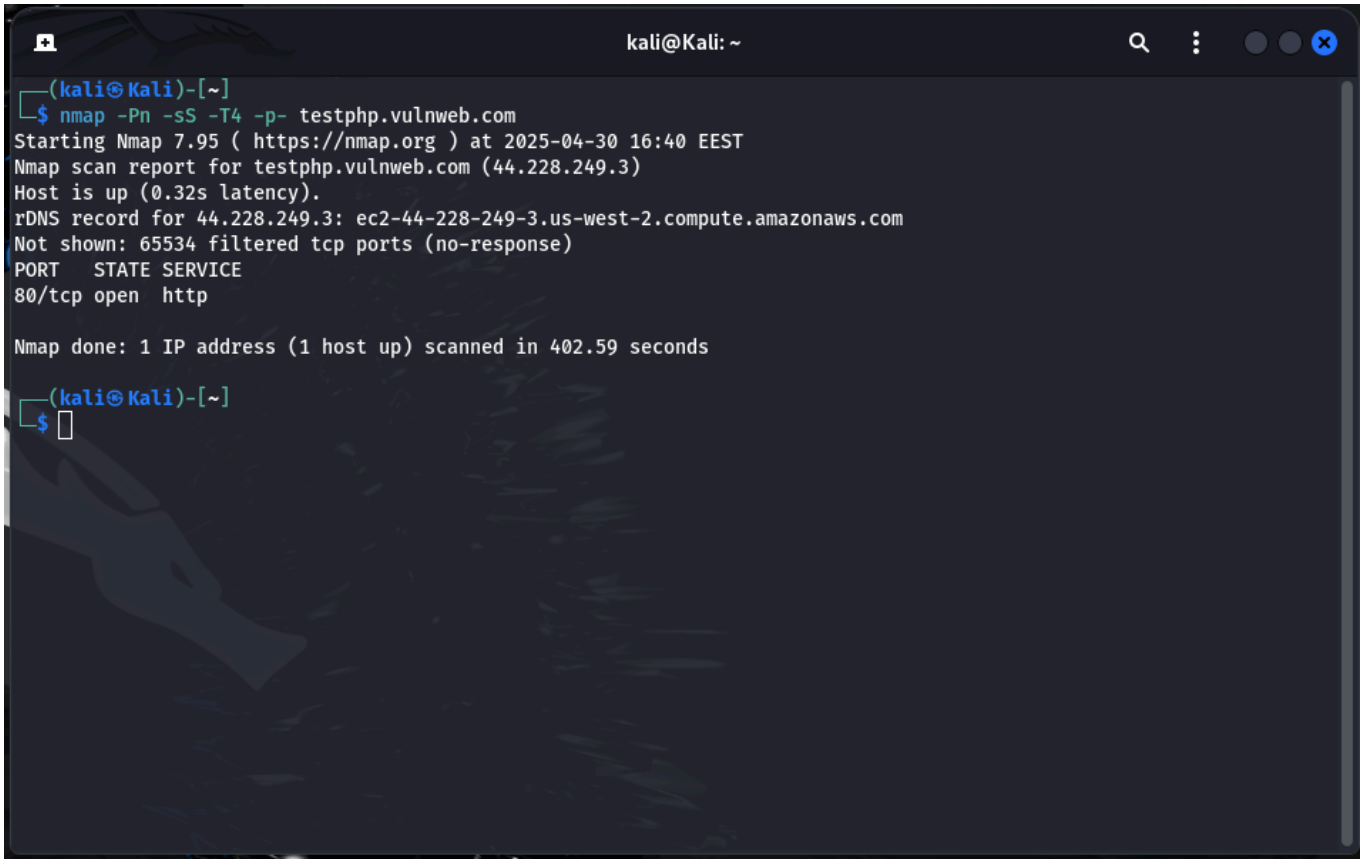
Tasks and Responsibilities:

📜 Task 1 Report: Port Scanning on `testphp.vulnweb.com`

- **Objective**:
  - `Identify open ports on the target web server http://testphp.vulnweb.com/ to understand the services exposed to the internet.`
- **Tools Used:**
  - `Nmap`
- **Command Used:**
  - `nmap -Pn -sS -T4 -p- testphp.vulnweb.com`
- **Findings (Scan Results):**
  - Open Port:
    - `80/tcp - HTTP Service`

`The scan revealed that only port 80 (HTTP) is open. All other ports are filtered, indicating no response or active firewall/security filtering.`

Screenshot:



.

📜 Task 2 Report: Directory Brute Forcing on

- **Objective:**
  - `Identify hidden or unlisted directories on the target web server, testphp.vulnweb.com, to find potential attack surfaces.`
- **Tools Used:**
  - `Gobuster`
- **Command Used:**
  - `gobuster dir -u http://testphp.vulnweb.com/ -w /usr/share/wordlists/dirb/common.txt -t 50`
- Findings (Directories Discovered):
  - `/admin` — Redirects to `/admin/`
  - `/cgi-bin/` — Forbidden (403)
  - `/crossdomain.xml` — Publicly accessible
  - `/CVS` — Redirects to `/CVS/`
  - `/CVS/Root`, `/CVS/Repository`, `/CVS/Entries` — Files found inside the CVS directory
  - `/favicon.ico` — Website's favicon
  - `/index.php` — Main page
  - `/images` — Image directory
  - `/pictures` — Pictures directory
  - `/secured` — Secure area (potentially interesting)
  - `/vendor` — Library/vendor files directory

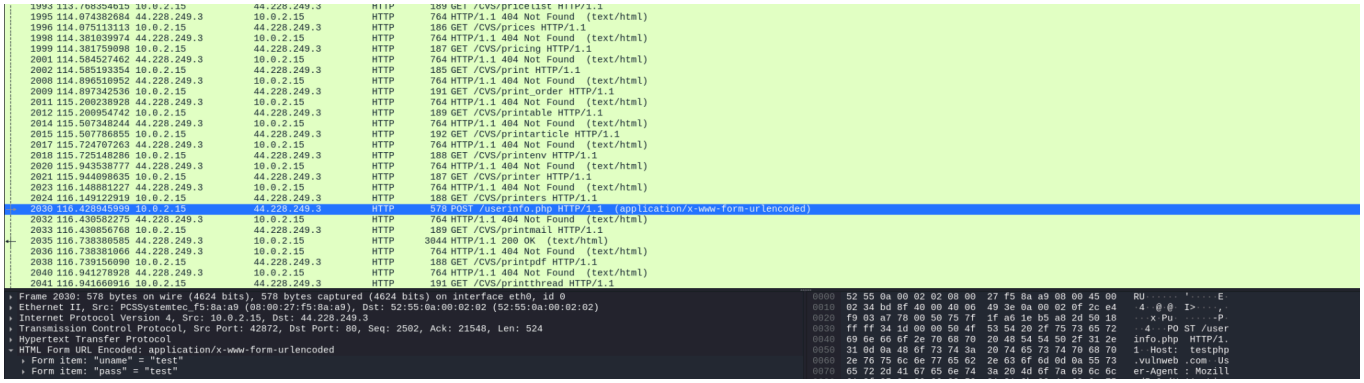
Screenshot:

```
┌──(kali㉿Kali)-[~]
└─$ 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                  (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/CVS/]
/CVS/Root             (Status: 200) [Size: 1]
/CVS/Repository       (Status: 200) [Size: 8]
/CVS/Entries          (Status: 200) [Size: 1]
/favicon.ico          (Status: 200) [Size: 894]
/index.php            (Status: 200) [Size: 4958]
/images               (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/images/]
/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
===============================================================
```

📜Task 3 Report: Capturing Login Credentials Using Wireshark

- Objective:
  - Intercept and analyze network traffic during a login attempt on `http://testphp.vulnweb.com/` to check if credentials are transmitted securely.
- Tools Used:
  - Wireshark
- Steps Followed:
  1. Opened Wireshark and started capturing traffic on the active network interface.
  2. I visited `http://testphp.vulnweb.com/` and attempted to log in by submitting a test username and password.
  3. Applied HTTP filters in Wireshark and identified a POST request directed to `/userinfo.php`.
  4. Analyzed the packet payload, revealing the username and password in plain text.
- Findings:
  - Form Fields Captured:
    - Username (`uname`) = `test`
    - Password (`pass`) = `test`
- Data was transmitted unencrypted due to the usage of HTTP rather than HTTPS.

Screenshot:



📜 Task 4 Report: SQL Injection

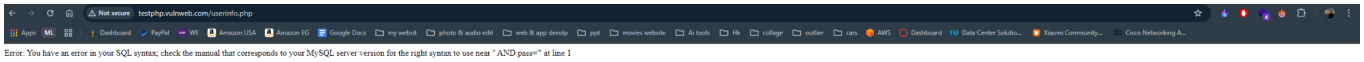
- Objective:
  - Test whether the login or search functionalities on the website are vulnerable to SQL Injection, and attempt to extract information from the backend database.
- Tools Used:
  - Manual SQLi Testing via Browser Input Field
- Steps Followed:
  1. Navigated to the login form on `http://testphp.vulnweb.com/`.
  2. Entered the following SQL payload in the username field: `' OR 1=1 –`
  3. Left the password field blank or with any value.
     Upon form submission, the following error message was displayed:

```
Error: You have an error in your SQL syntax; check the manual that corresponds to
your MySQL server version for the right syntax to use near '' AND pass='''' at line 1
```

Analysis:

- The error message confirms the input is directly injected into the SQL query without proper sanitization.
- This indicates the website is vulnerable to SQL Injection.
- An attacker could potentially exploit this vulnerability to:
  - Bypass login
  - Enumerate users or databases
  - Dump sensitive data

Screenshot:

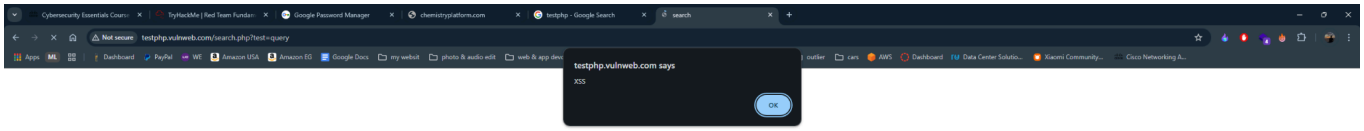


📜 Task 5 Report: Cross-Site Scripting (XSS)

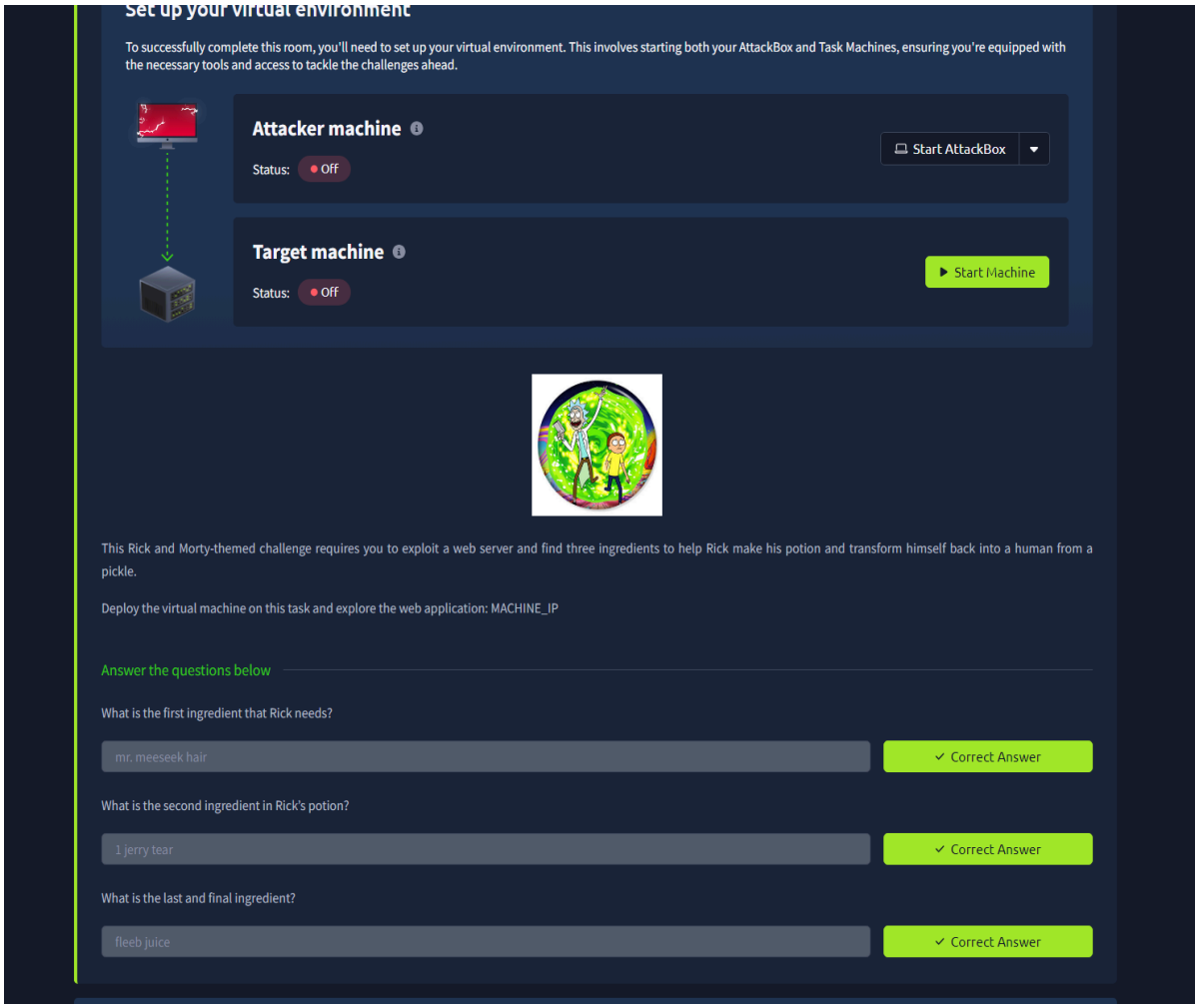
- Objective:

    - Test for Cross-Site Scripting (XSS) vulnerabilities by injecting malicious scripts into input fields to evaluate input sanitization on the website.
- Tools Used:
    - Web Browser (Manual Testing)
- Steps Followed:
  1. Navigated to the search page at: `http://testphp.vulnweb.com/search.php?test=query`
  2. Injected the following JavaScript payload directly into the URL: `<script>alert('XSS')</script>`
  3. Reloaded the page and observed the result.
- Findings:
    - The browser immediately rendered an alert box with the message `XSS`.
    - This confirms that the input was reflected and executed as code on the page, demonstrating a Reflected XSS vulnerability.

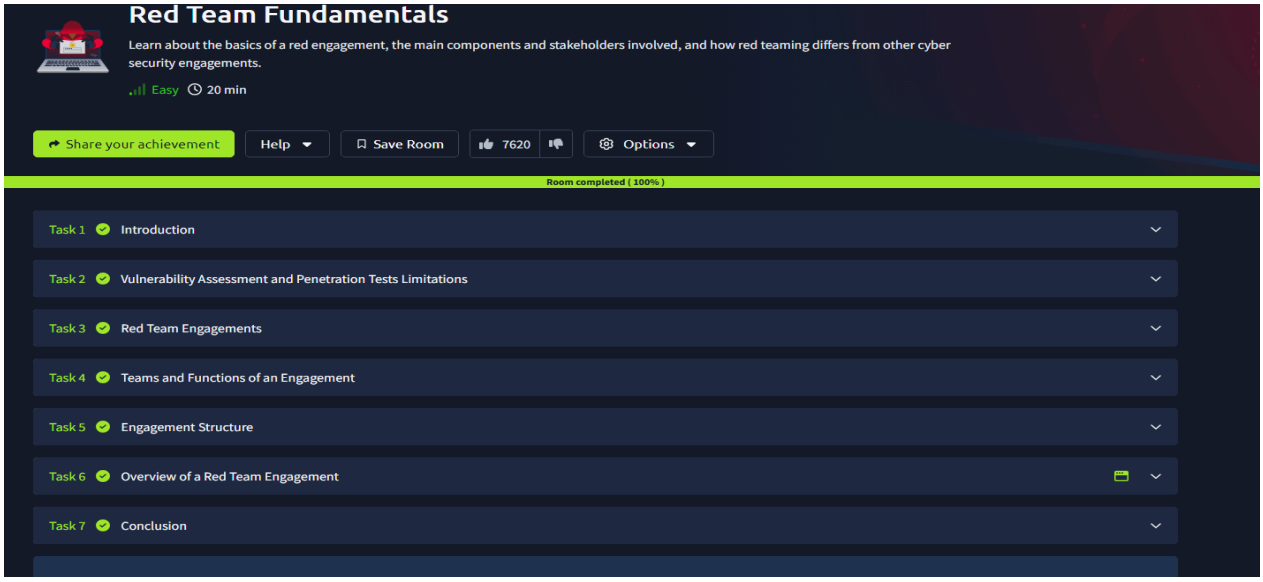
Screenshot:



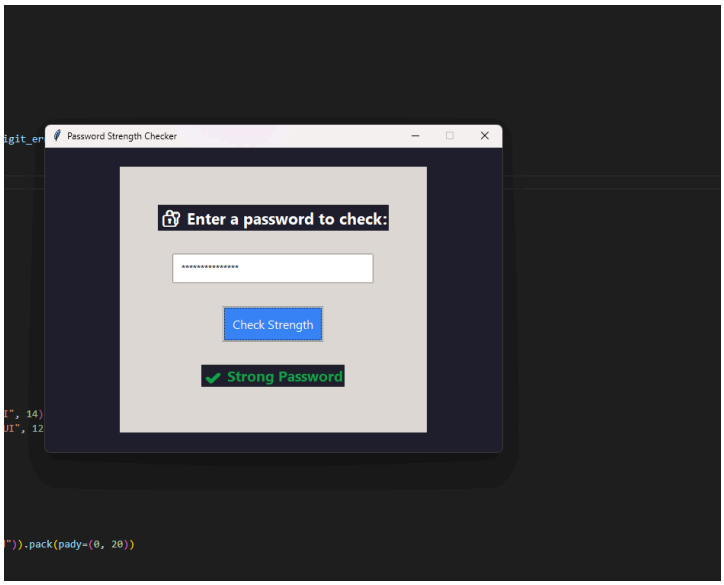
📜CTF Task – TryHackMe: PickleRick



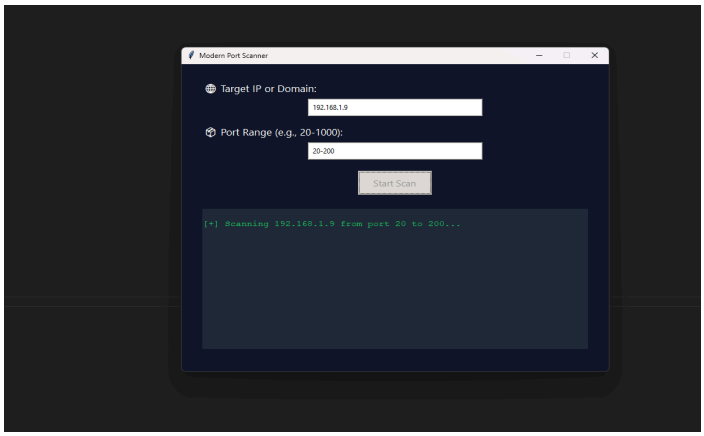
📜CTF Task – TryHackMe: Red Team Fundamentals



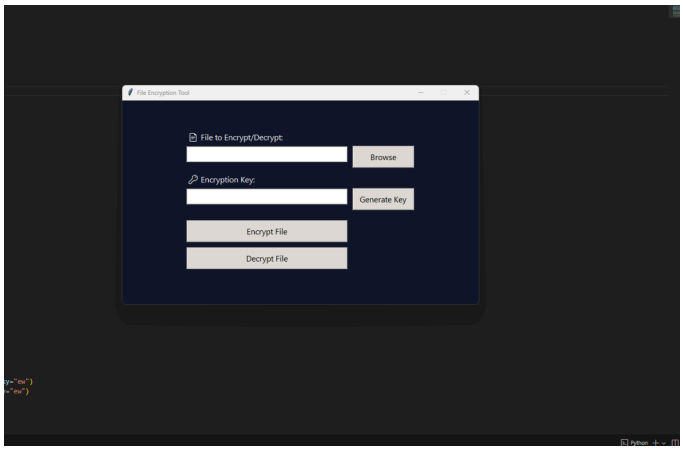
📜Python Project 1: Password Strength Checker



📜Python Project 2: Basic Port Scanner



📜Python Project 3: File Encryption/Decryption Too



📜Learning Outcomes

- Enhanced practical understanding of web-based vulnerabilities.

- Gained experience using security tools (Nmap, Gobuster, Wireshark, Metasploit, etc.)

- Understood red team methodology and adversarial simulation.

- Built modern Python GUI tools using `tkinter`, threading, and libraries like `socket` and `cryptography`.

📜Challenges & Solutions

- XSS & SQLi Debugging: Initially struggled to detect injection points, solved by analyzing source and error responses.

- GUI Design: Creating a modern look with `tkinter` required styling with `ttk` and layout improvements.

- Decryption Errors: Fixed key mismatch bugs by adding input validation and error messages.

📜Conclusion

This internship helped me bridge academic knowledge with professional-level cybersecurity experience. I gained a deeper interest in ethical hacking and hands-on red teaming, which I aim to pursue further in my career.

📜Acknowledgments

I sincerely thank Hack Secure and my mentors, Mr. Aman Pandey and Mr. Prabhat Raj for guiding me throughout the internship. Their support helped me grow technically and professionally.

Appendix

- GitHub Repo: [https://github.com/mostafa-karam/HackSecure-Internship]