27 January 2025

OFFSEC MoneyBox

Kein System ist sicher
Dreistigkeit siegt - Strebe das Unmögliche an

PantherBear

Certified GreyHat Hacker

Table of Contents

1.0 Offensive Security Exam Penetration Test Report

1.1 Introduction

The Offensive Security Exam penetration test report contains all efforts that were conducted in order to pass the Offensive Security exam. This report will be graded from a standpoint of correctness and fullness to all aspects of the exam. The purpose of this report is to ensure that the student has a full understanding of penetration testing methodologies as well as the technical knowledge to pass the qualifications for the Offensive Security Certified Professional.

1.2 Objective

The objective of this assessment is to perform an internal penetration test against the Offensive Security Exam network. The student is tasked with following a methodical approach in obtaining access to the objective goals. This test should simulate an actual penetration test and how you would start from beginning to end, including the overall report. An example page has already been created for you at the latter portions of this document that should give you ample information on what is expected to pass this course. Use the sample report as a guideline to get you through the reporting.

1.3 Requirements

The student will be required to fill out this penetration testing report fully and to include the following sections:

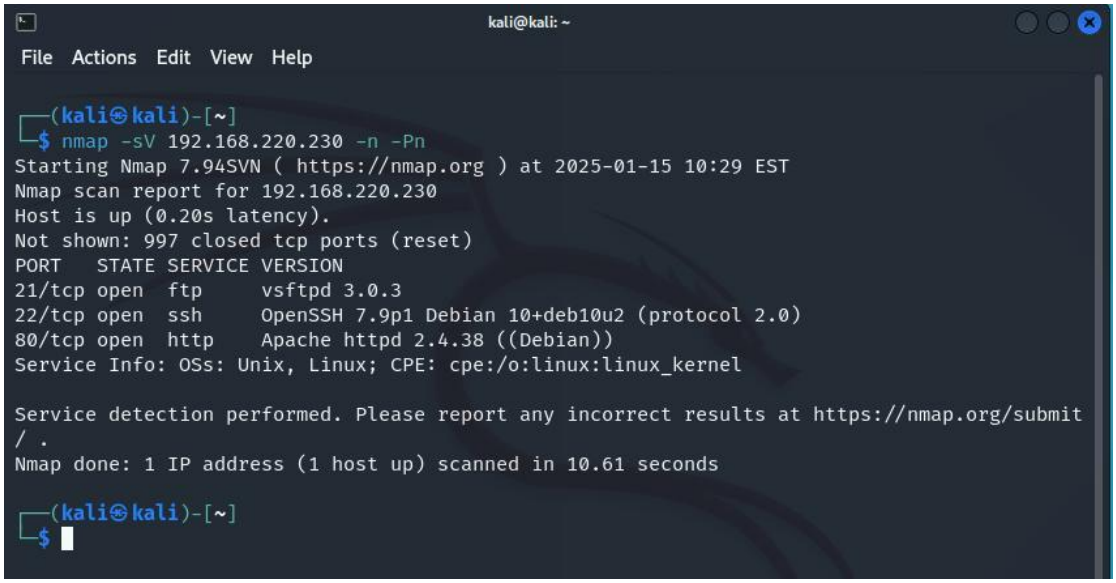
- Overall High-Level Summary and Recommendations (non-technical)
- Methodology walkthrough and detailed outline of steps taken
- Each finding with included screenshots, walkthrough, sample code, and proof.txt if applicable.
- Any additional items that were not included

2.0 System IP: 192.168.220.230

2.1 Service Enumeration

Port no.	Service	Version
21	ftp	vsftpd 3.0.3
22 80	ssh http	OpenSSH 7.9p1 Apache httpd 2.4.38

Nmap Scan Results:



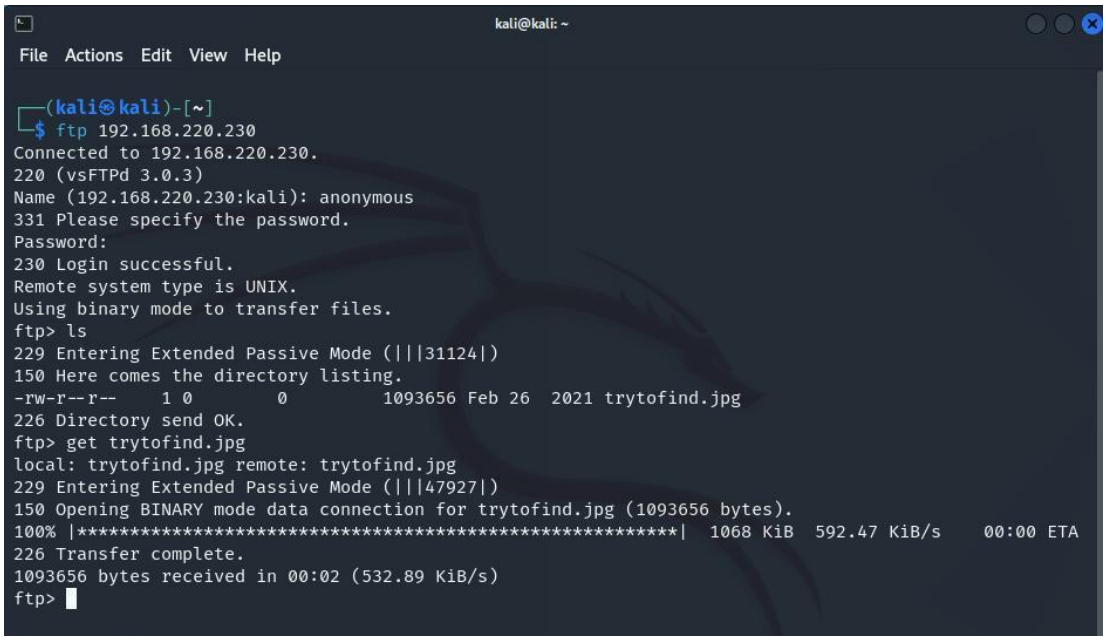
Command: **nmap -sV 192.168.220.230 -n -Pn**

The command lists the service versions running on open ports, and the network scan is accelerated by disabling host discovery and DNS resolution.

2.2 System IP: FootHold on FTP

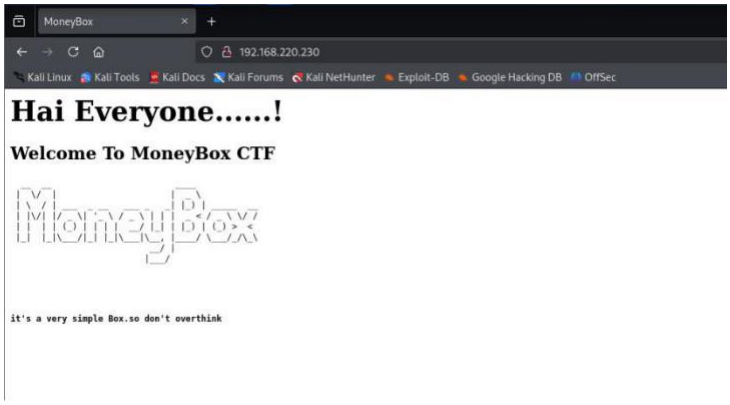
Command: **nmap --script ftp-anon -p 21 192.168.220.230**

The Nmap command uses the **ftp-anon** script to scan the target machine on port 21 to check if the FTP server permits anonymous access. The scan result shows FTP code 230, indicating that the server does allow anonymous access. Additionally, a file named **trytofind.jpg** is found on the server.

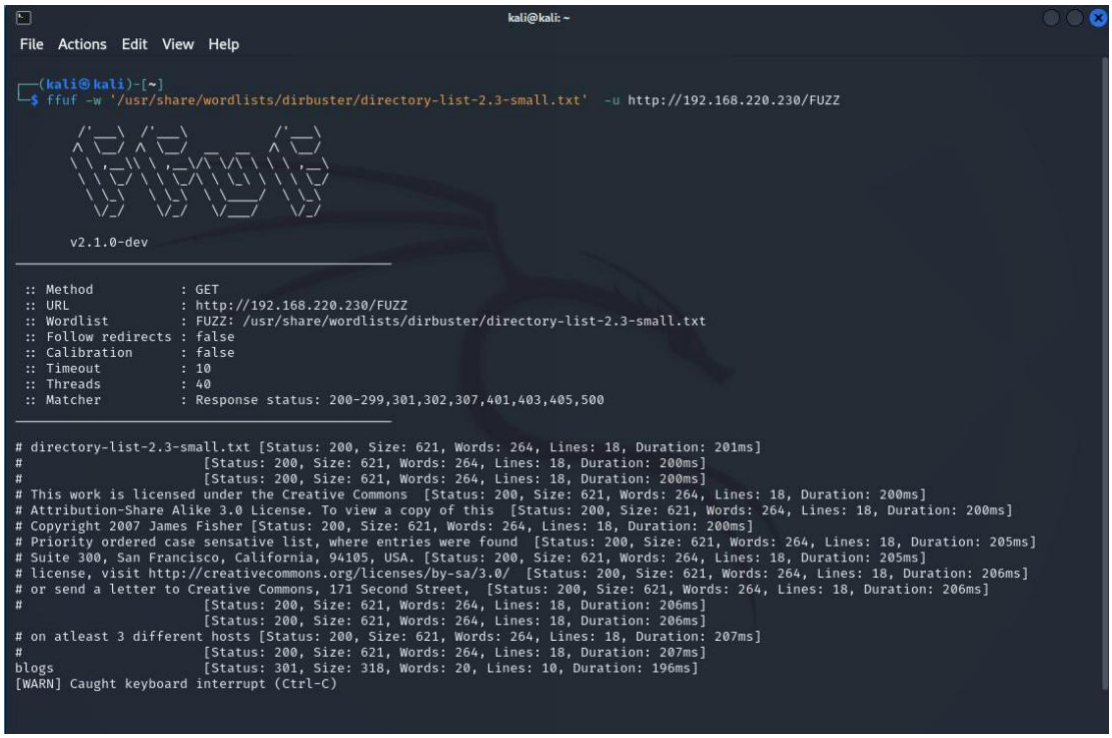


Using FTP Anonymous login and exfiltrating the file **trytofind.jpg**

2.3 Website Enumeration

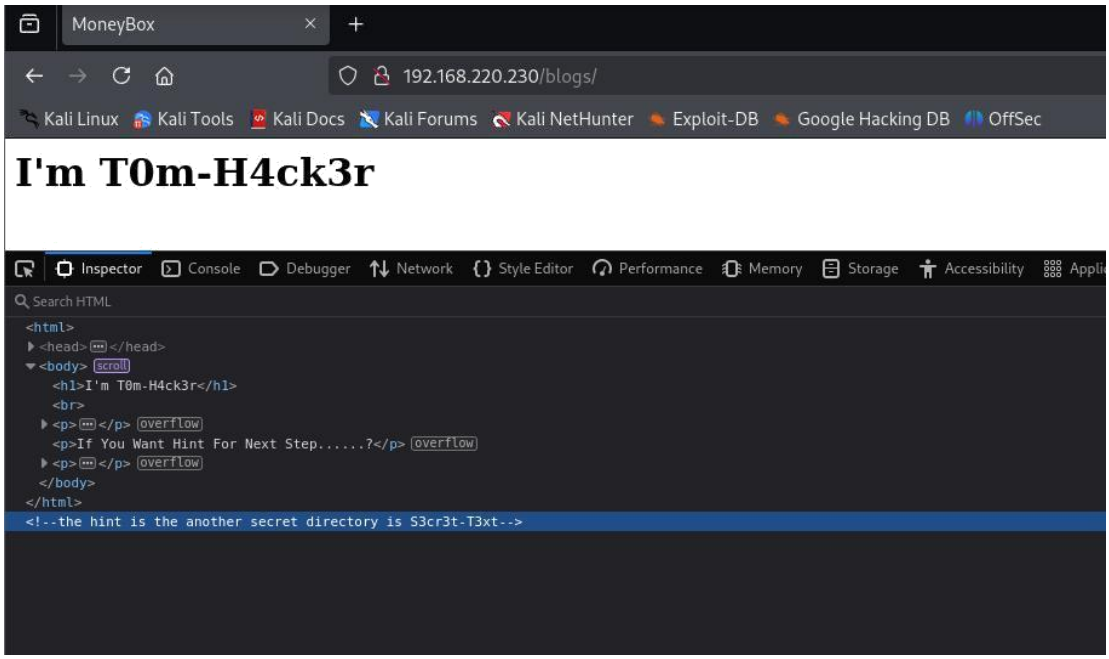


There are no new leads from this point even when source code is inspected.

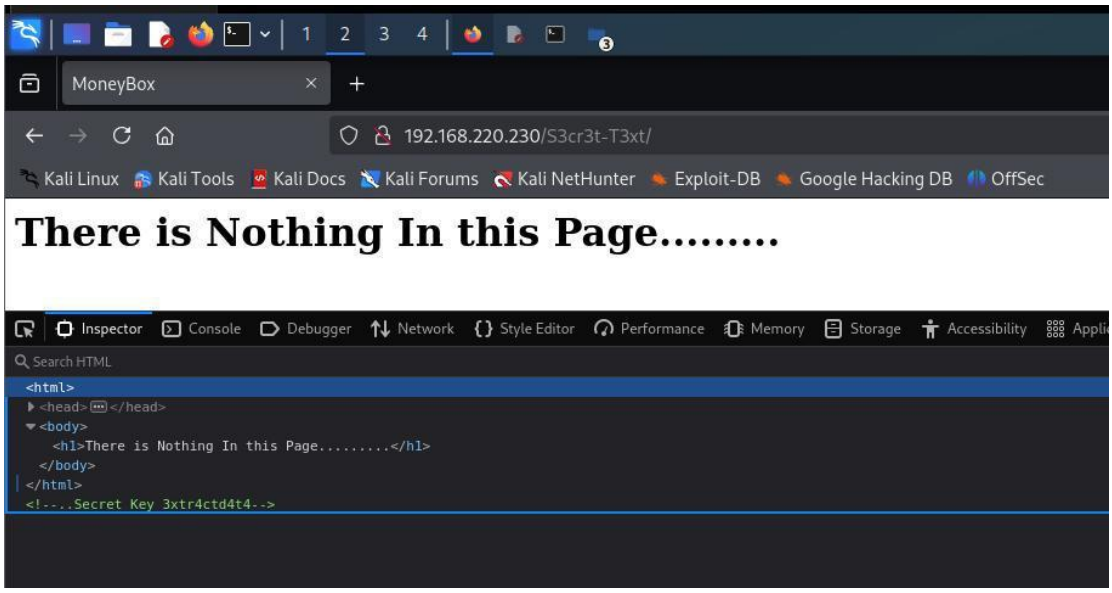


Command: **ffuf -w '/usr/share/wordlists/dirbuster/directory-list-2.3-small.txt' -u http://192.168.220.230/FUZZ**

I used FFUF to fuzz potential subdirectories on the website and discovered a sub-directory named **blogs**, which returned an HTTP status code **301 - Moved Pernamently**.



While inspecting the source code of **http://192.168.220.230/blogs**, there was a comment found that points to another directory called **S3cr3t-T3xt**

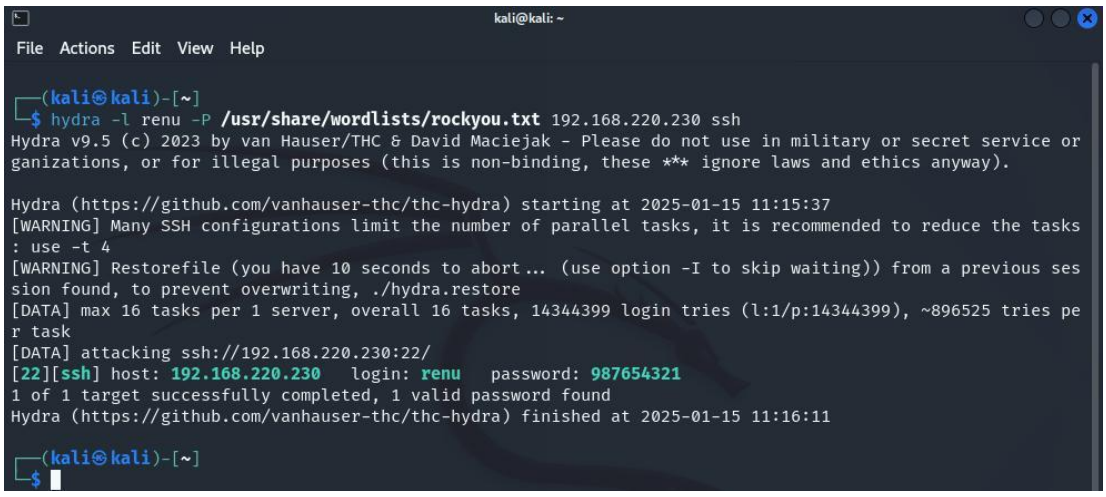


While inspecting the source code of **http://192.168.220.230/S3cr3t-T3xt**, there was a comment that mentioned a Secret Key: **3xtr4cted4t4**

2.4 Probing Around

Command: **steghide extract -sf trytofind.jpg**

The file **trytofind.jpg** was about 11MB in size. By using the command above, the hidden data within the JPEG file via steganography could be extracted. This process required the Secret Key: **3xtr4cted4t4**

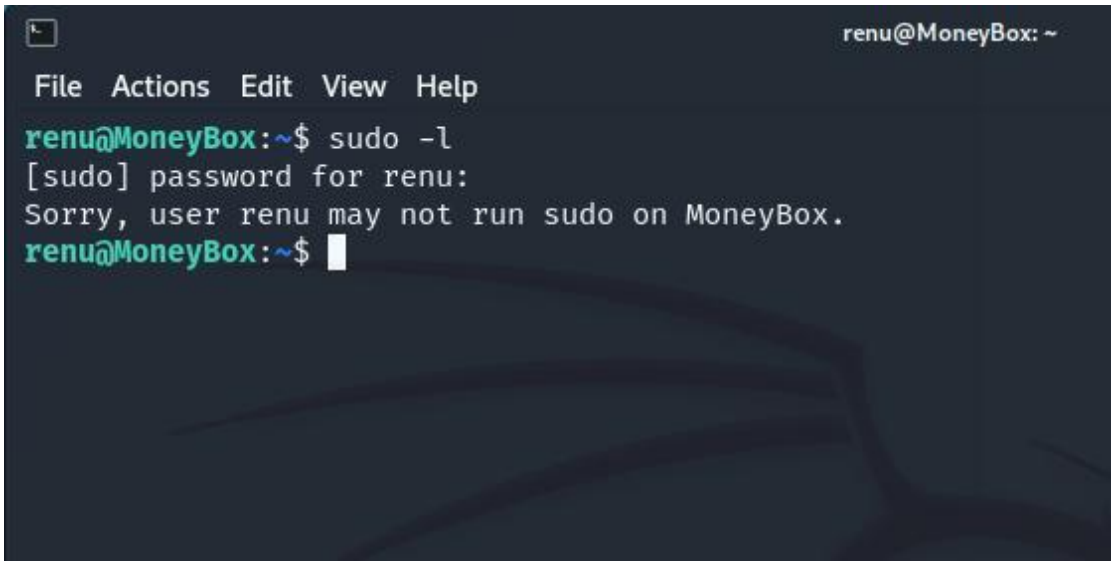


Command: **hydra -l renu -P /usr/share/wordlists/rockyou.txt 192.168.220.230 ssh**

Cracking user **renu** SSH login credentials against rockyou.txt using Dictionary attack. The user password is: **987654321**

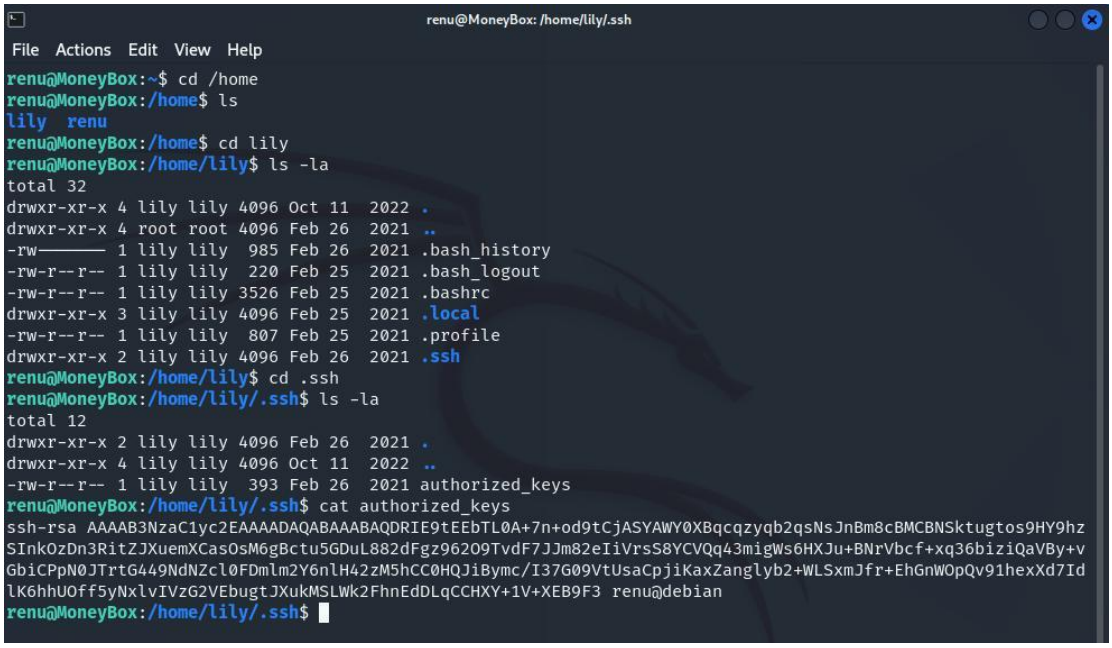
2.5 Gaining Access: local.txt

SSH into the user **renu** using the password **987654321** and dumping the contents of **local.txt** in the current directory

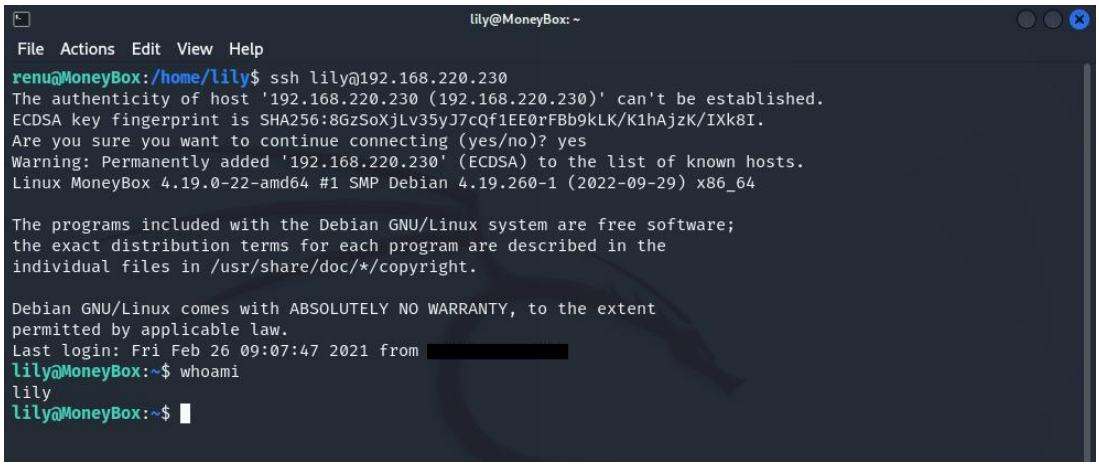


The user **renu** is unable to run anything on **MoneyBox** due to the restriction in privileges

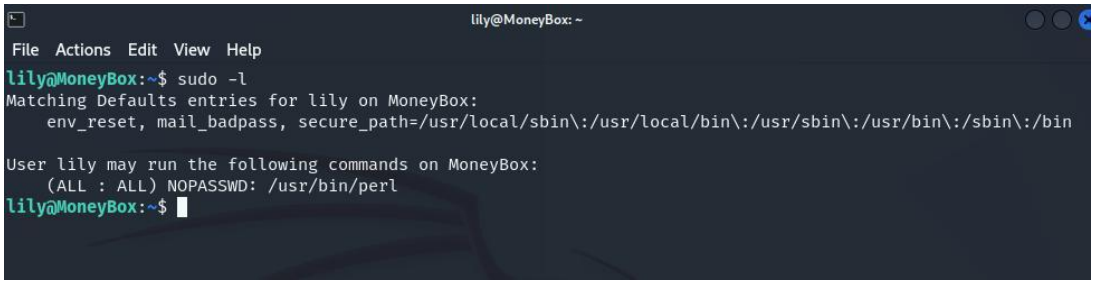
2.6 Horizontal Privilege Escalation



While digging through the /home directory, I found another user, **lily**. Upon navigating into the **lily** user directory, two items stood out: the .bash_history file and the .ssh directory, both of which have a high likelihood of containing sensitive information. After checking the .bash_history file for user **lily**, nothing of interest was found. However, within the **authorized_keys** file under the .ssh directory, there is a public SSH key (**ssh-rsa**) belonging to user **renu**. This means that user **renu** can log in to the **lily** account via SSH using SSH key authentication, without the need for a password prompt.

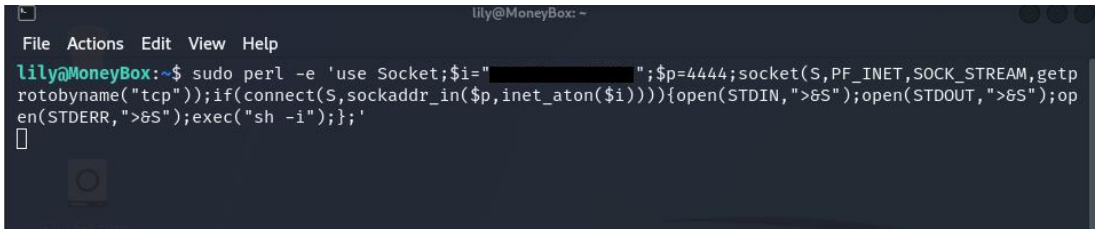


2.7 Privilege Escalation: root.txt



Command: **sudo -l**

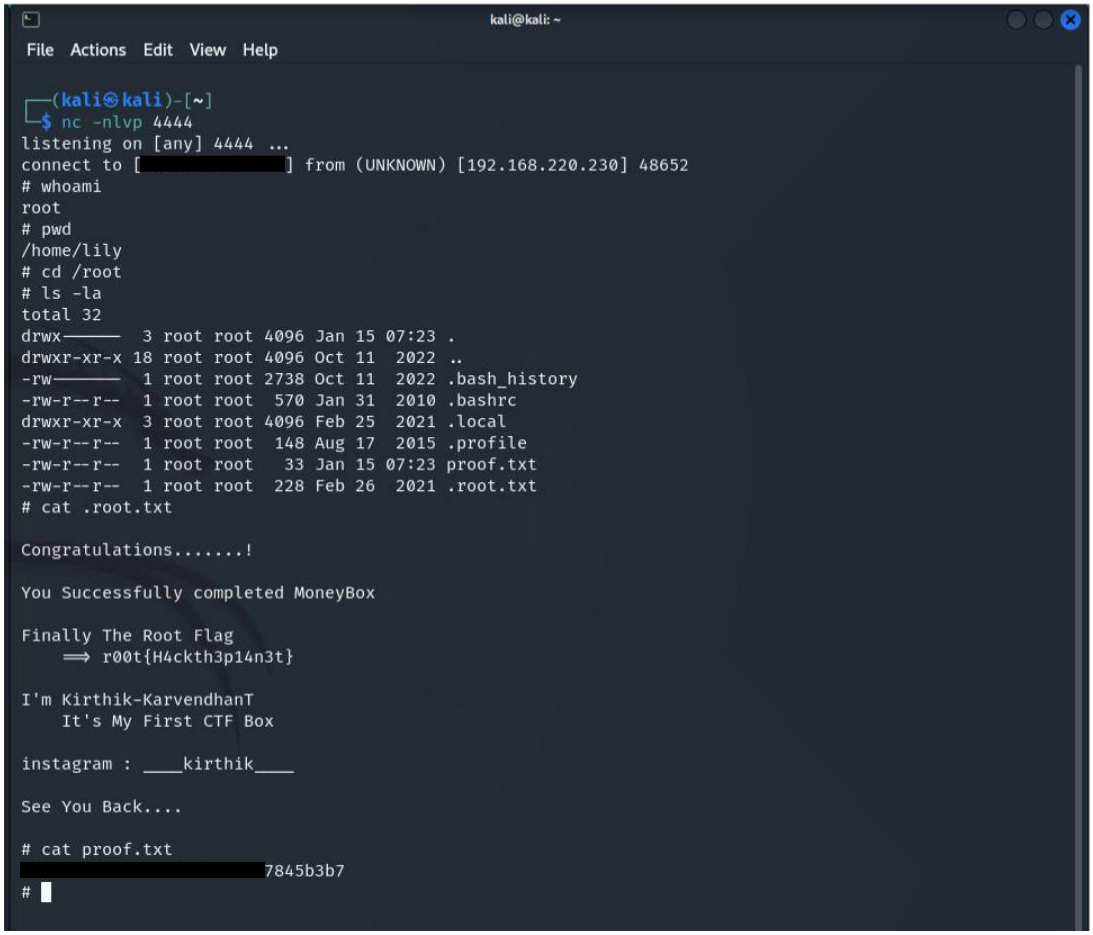
The command is used to list the allowed commands that the current user can run with sudo privileges. User **lily** can run the perl binary from **/usr/bin/perl** as root (Administrator privilege) without password required.



The above is a Perl reverse shell that was generated from the website **https://www.revshells.com**. This reverse shell is part of a broader exploitation technique where I abused the permissions of user **lily** to execute the Perl binary as root on the target machine without requiring a password. I was able to run the reverse shell through Perl command as root, which effectively connects the target machine back to my local system which is captured by my NetCat Listener on port **4444**. This allowed me to establish a stable and interactive session, ultimately granting me access to the target system with root privileges. The result is a fully functional root shell on the compromised machine.

Alternatively, the below command can also be used in **lily**s SSH session to escalate current shell to root.

Command: sudo perl -e 'system("/bin/bash")'



I navigated to the **/root** directory and successfully accessed the contents needed for submission.

2.8 Vulnerability Fix and Severity

Vulnerability 1: Anonymous FTP login

FTP server has poor security configurations in supporting FTP Anonymomus login

Fix and Mitigations

- **Restricting access to specific directories that are intended for anonymous usage**
- **Using firewalls and network configurations to restrict who can access the anonymous FTP service**
- **Adopt Data Classification when uploading documents in terms of sensitivity of information**

Vulnerability 2: Weak SSH credentials

Renu's password cracked and gained initial foothold using SSH

Fix and Mitigations

- **Use a easy-to-remember but stronger password**

Vulnerability 3: Horizontal Privilege Escalation to Lily's account

Renu's SSH-RSA public key is found in Lily's .ssh/.authorized_keys directory and gained access to Lily's account via SSH

Fix and Mitigations

- **Set proper permissions on .ssh and .authorized_keys**
  chmod 700 ~/.ssh (disable access to .ssh directory for all besides the owner)
  chmod 600 ~/.ssh/authorized_keys (Grants read and write permissions to the user only, disallowing other users from accessing it)

- **Limit SSH key usage within .authorized_keys file**
  Restrict access to specific commands

  command="/path/to/command" ssh-rsa AAAAB3... key

- **Set SSH Key Expiration**
- **Restrict SSH Access to specific Users and Groups**

  Configure the SSH server (/etc/ssh/sshd_config) to restrict which users or groups can use SSH keys for access.

Vulnerability 4: Vertical Privilege Escalation to root

The misconfiguration of sudo permissions in user Lily's account allowed the abuse of the Perl binary to escalate privilege levels.

Fix and Mitigations

- **Audit Sudoers File for Dangerous**
- **Entries Limit Sudo Access**

  Ensure users are only granted **specific, necessary privileges**. Limit what each user can do via sudo and explicitly specify which commands they can run.

- **Fix Privilege Escalation Vectors**

  Changing ownership or permissions of the binary to prevent users from executing it unnecessarily.

- **Restriction of commands executed by sudo**

  The sudoers file can be edited to restrict **command arguments** and **environment variables** that can be passed to commands when they are executed with sudo.

  eg1. Restrict a user to running a specific command with **no arguments parsed**

  eg2. Prevent users from using environment variables that might cause privilege escalation

- **Restrict sudo to only trusted users**