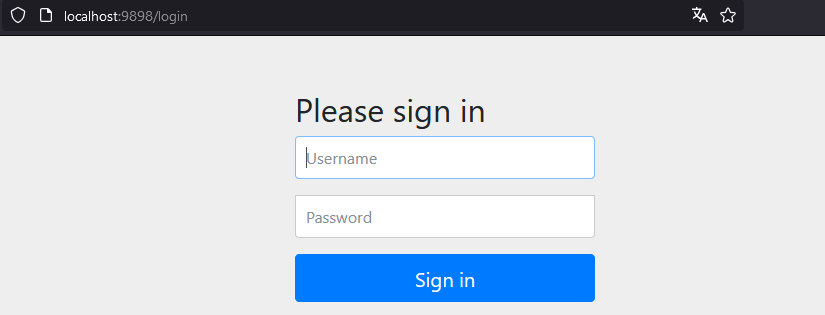
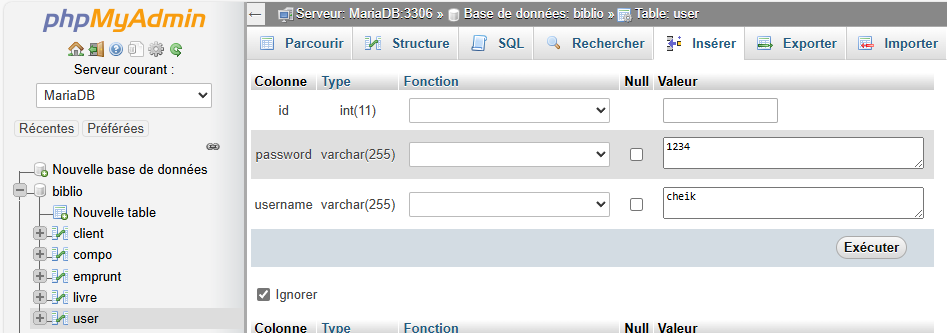
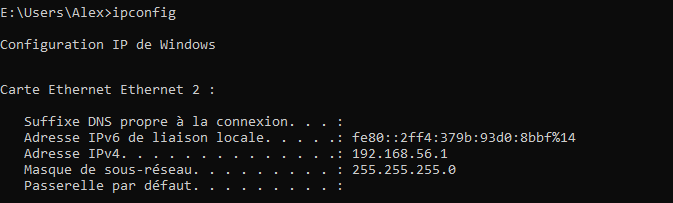
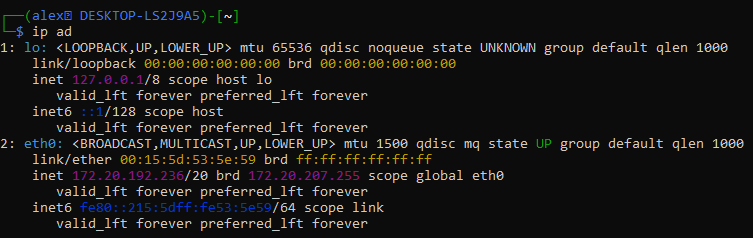
#### **Tester le lien** [**http://localhost:9898/**](http://localhost:9898/)

1. **Accéder à l'interface Web** :
   1. Ouvrez un navigateur et accédez à <http://localhost:9898/>.
   2. Si un login est demandé, ajoutez un utilisateur dans MySQL

INSERT INTO User (username, password) VALUES ('cheik', '1234');

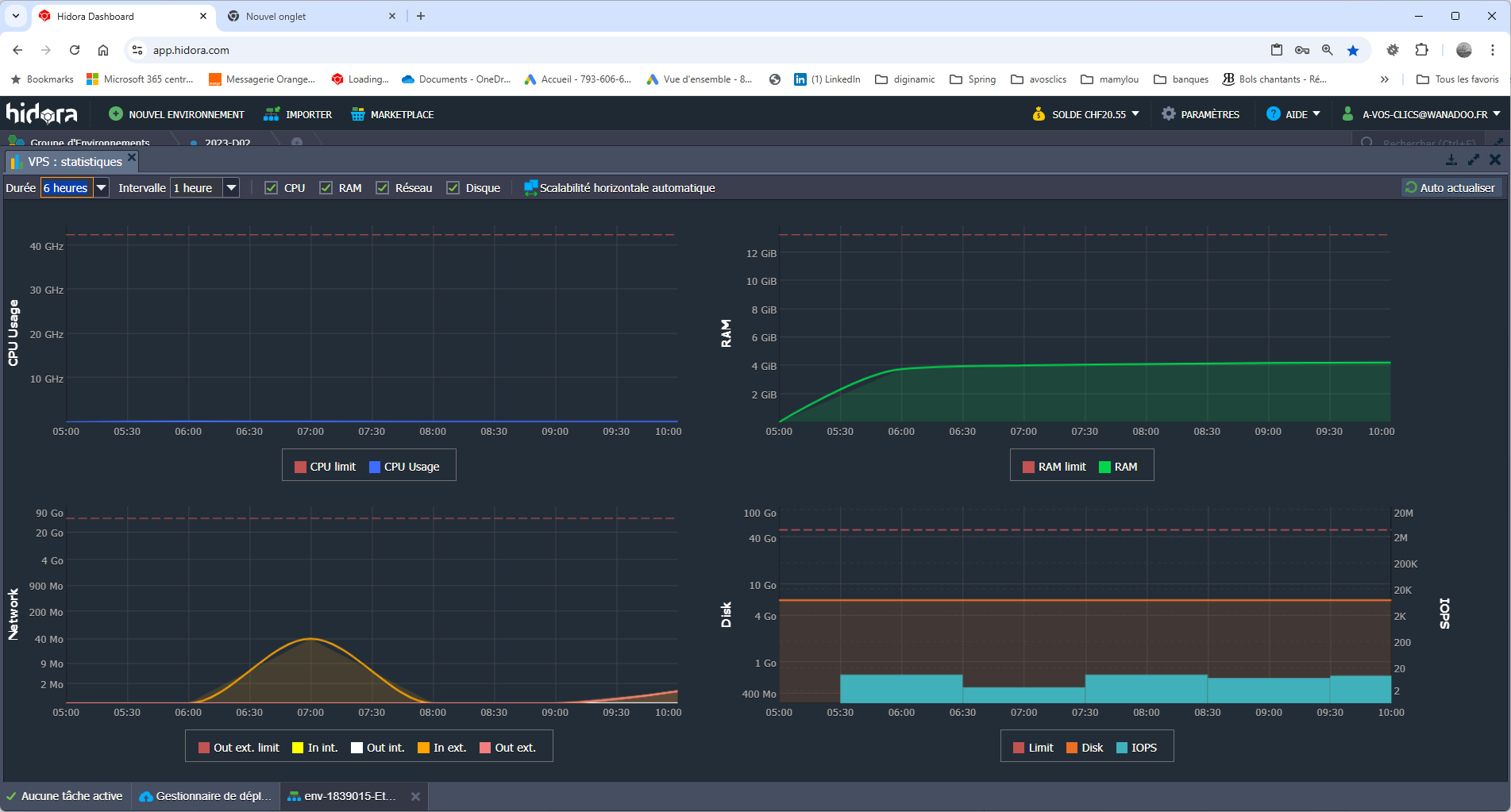
**Adresse IP de la machine cible**

Dans un invite de commande, taper ipconfig sous windowsIp ad sous linuxl’adresse ip de la machine sur laquelle est hébergée l’application spring boot est : 192.168.56.1

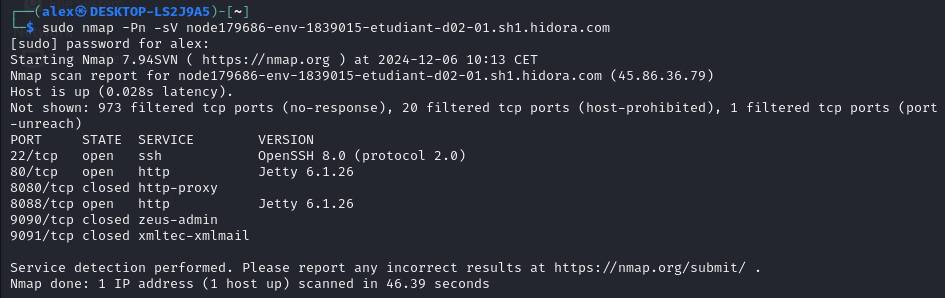
Le pentest sera réalisé depuis un environnement Kali Linux.

**Réalisation d'un Pentest sur un Serveur Laravel et VM Big Data avec Kali Linux**

Scanner la VM et l’Application Laravel avec Nmap :

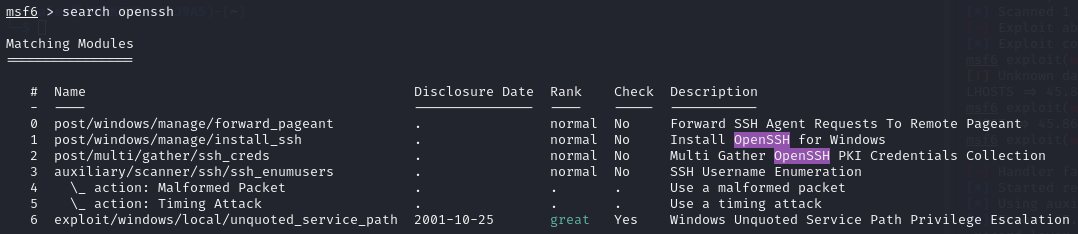
**Scan des Ports de la VM Big Data (node179686-env-1839015-etudiant-d02-01.sh1.hidora.com) :**

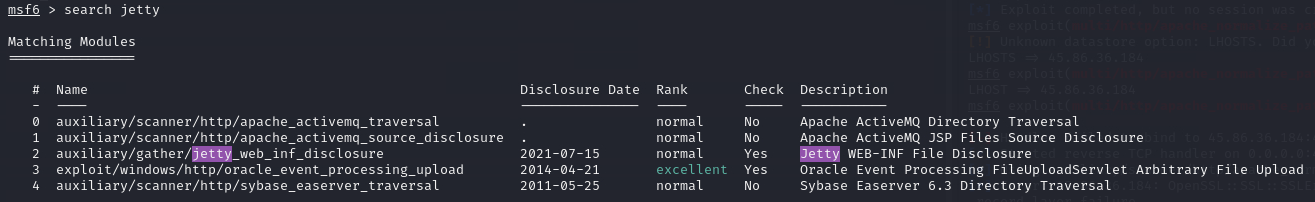
sudo nmap -Pn -sV node179686-env-1839015-etudiant-d02-01.sh1.hidora.com

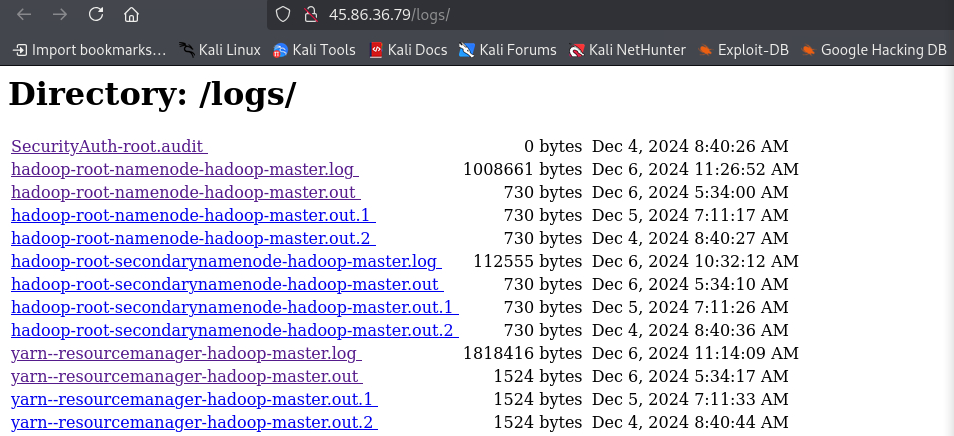
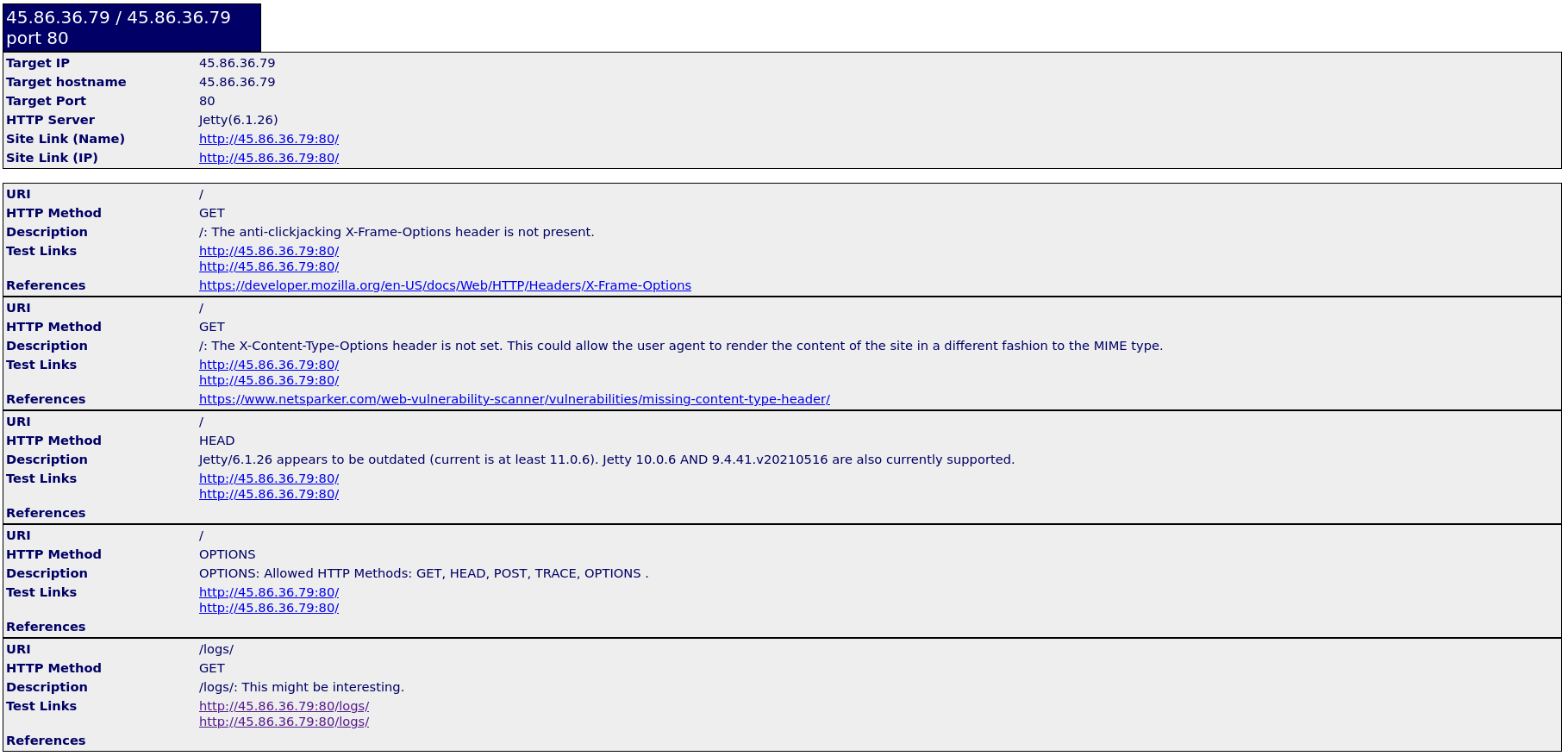


**Analyse des résultats** : Notez les ports critiques, comme 22 (SSH), 80 (HTTP), 3306 (MySQL)

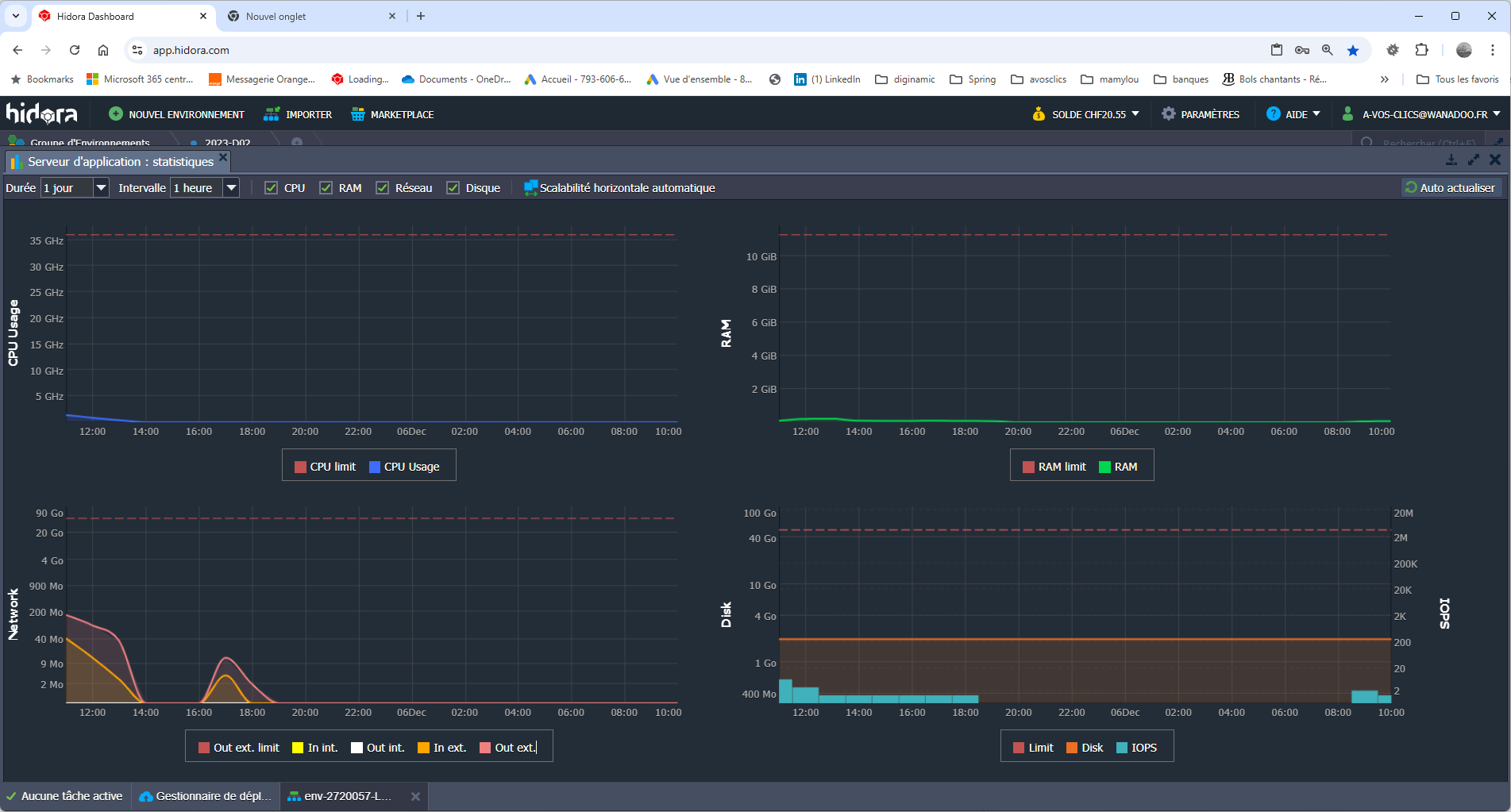
Port 22/tcp : OpenSSH 8.0 (protocol 2.0)

Port 80/tcp : Jetty 6.1.26

Utilisation de nikto :

nikto -h http://45.86.36.79 -o nikto\_report.html -Format html

**Scan des Vulnérabilités sur l’App Laravel (**[**http://45.86.36.184/public/**](http://45.86.36.184/public/)**)**



**Analyse des résultats** : Identifiez les vulnérabilités connues (failles CVE) et exploitez-les si possible (ex : failles d'injection SQL, failles XSS).

Pas de résultats avec les injections SQL :

**Réaliser un Crawl de l’App Laravel avec Httrack**

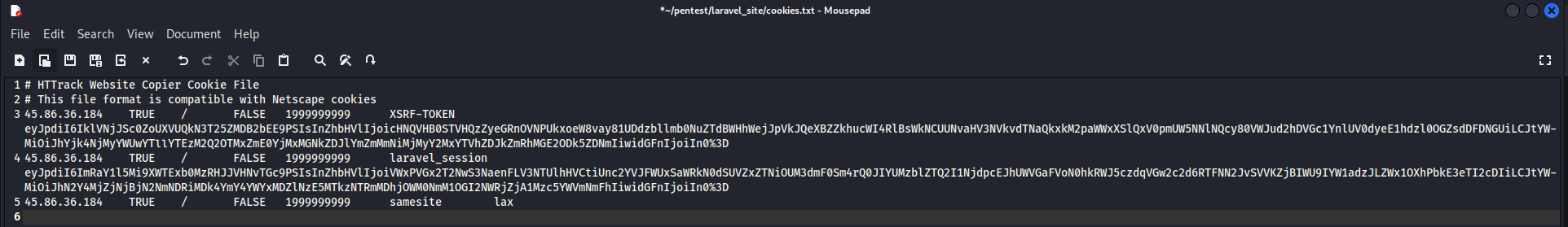
**Objectif : Télécharger la structure du site pour une analyse hors ligne**

Installation de Httrack:

sudo apt install httrack -y

Cloner le site Laravel : httrack http://45.86.36.184/public/ -O ~/pentest/laravel\_site -v**Analyse des résultats** :

Vérifiez les fichiers sensibles exposés : .env, config.php, fichiers de logs

Un fichier trouvé contenant deux clés :

**Extraction de Données Supposées Supprimées avec Foremost**

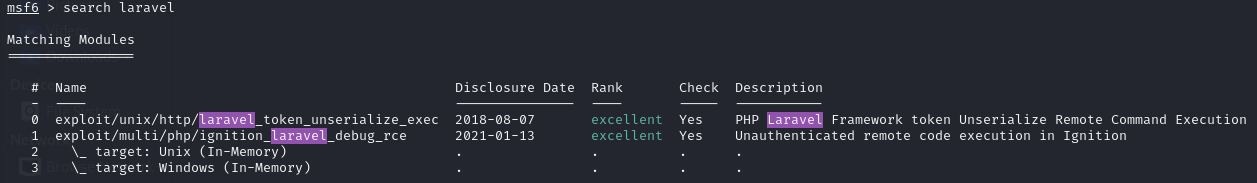
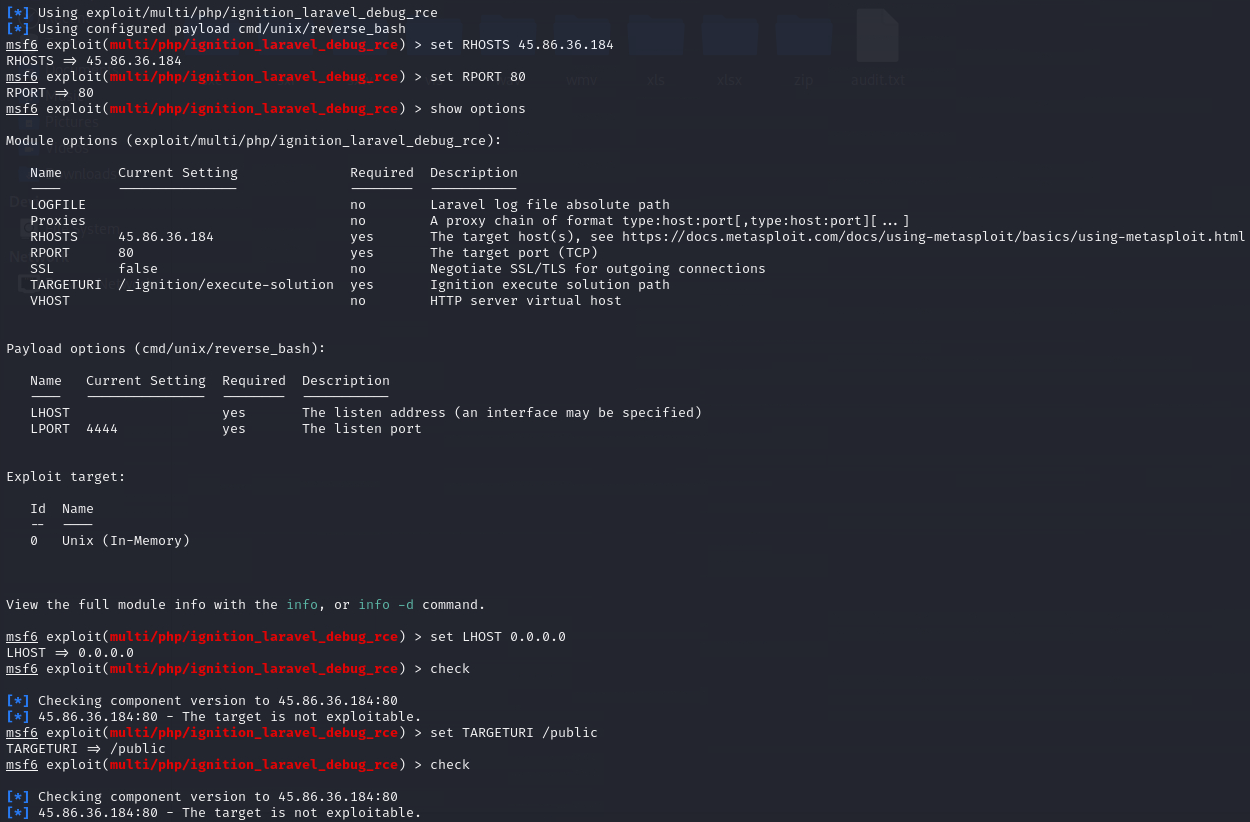
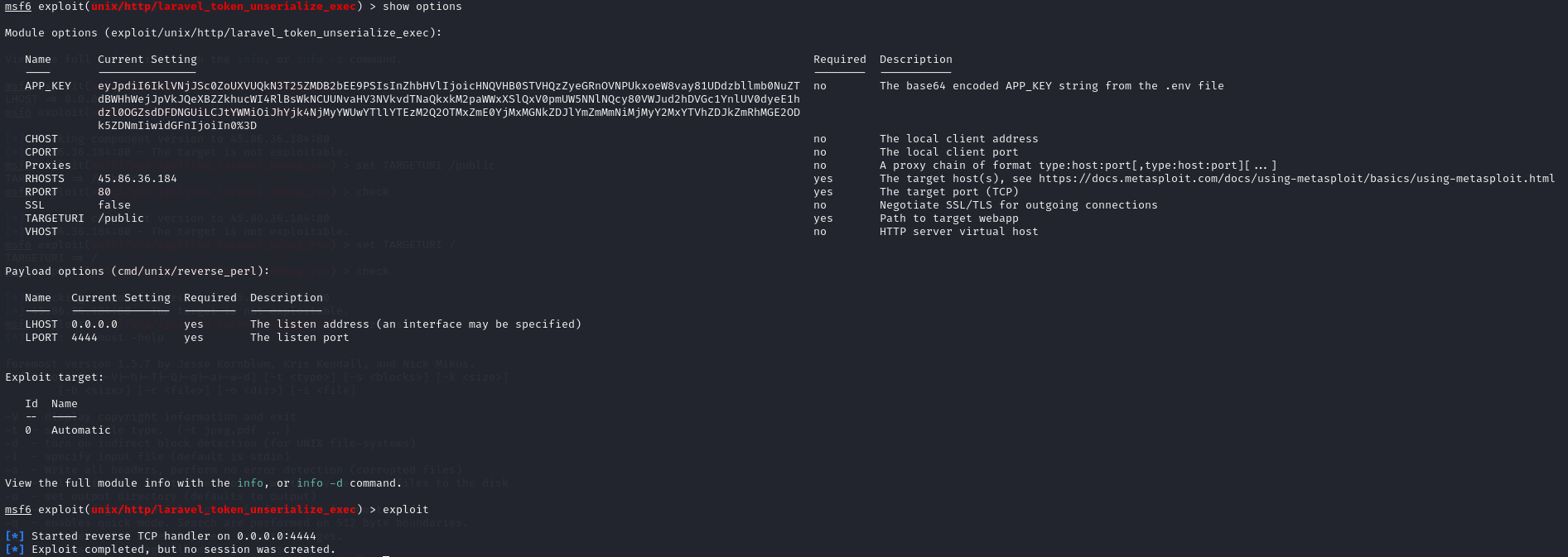
Objectif : Récupérer des fichiers supprimés dans la VM ou dans les réponses du serveur Laravel:

**Installation de Foremost:**

sudo apt install foremost –y

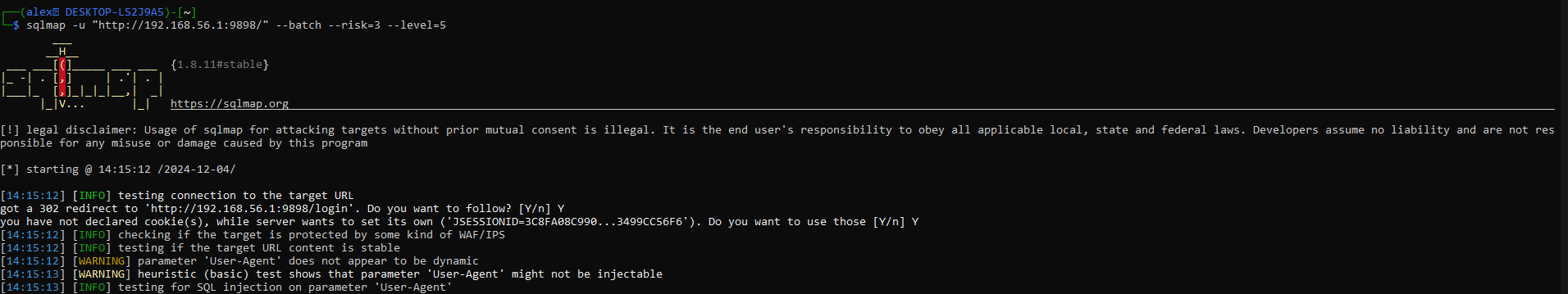
**Télécharger une copie des logs ou des données accessibles**

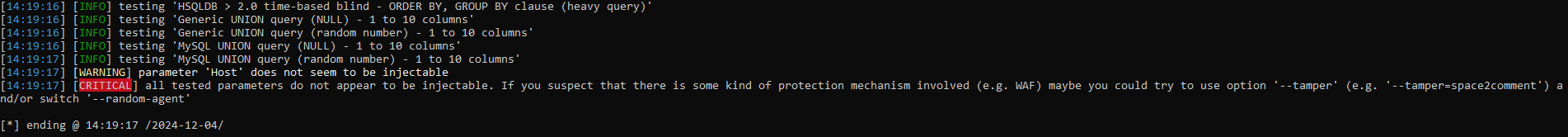
**Analyse des résultats** : Vérifiez le répertoire ~/pentest/foremost\_output pour des fichiers extraits tels que des documents sensibles, des bases de données, ou des images.

**Tester les Failles Communes dans l’App Laravel**tentative d’exploitation de la faille la plus récenteTentative avec la seconde faille avec les clés du fichier cookies.txt

**Test d'intrusion avec sqlmap:**

sqlmap -u "http://192.168.56.1:9898/" --batch --risk=3 --level=5

Sqlmap teste différentes méthodes d’injection SQL



#### **Tester le mapping** [**http://localhost:9898/api/clients**](http://localhost:9898/api/clients)

* **Avec Postman** :
  + Méthode : GET
  + URL : <http://localhost:9898/api/clients>
* **Avec sqlmap**

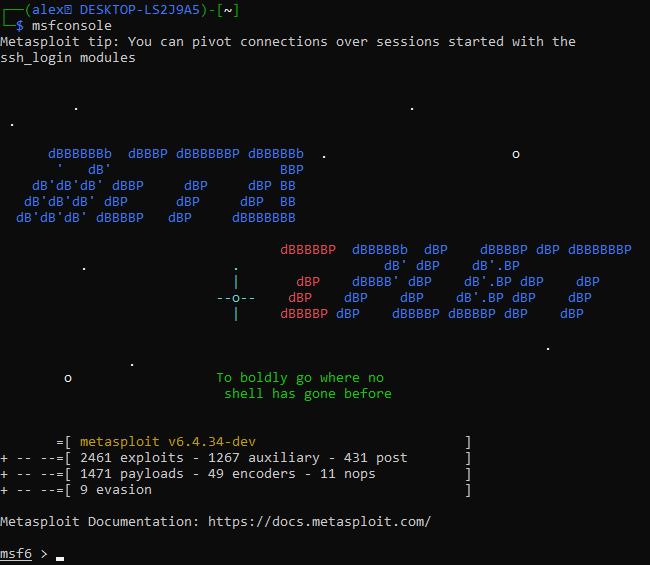
sqlmap -u "http://192.168.56.1:9898/api/clients" --batch

### **Mettre en place le pentest :**

#### **Metasploit**

1. **Lancer Metasploit**

Msfconsole



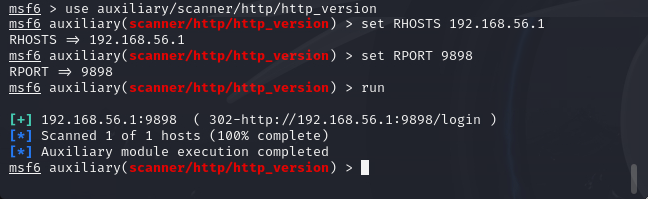
**Scanner des failles** :

use auxiliary/scanner/http/http\_version

set RHOSTS 192.168.56.1

set RPORT 9898

run



**Utilisation avec Metasploit**

Lancer Metasploit :

msfconsole

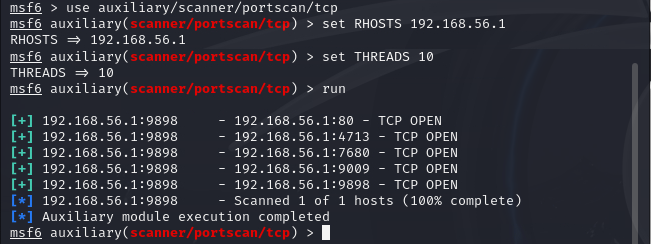
Utiliser un module de scan :

use auxiliary/scanner/portscan/tcp

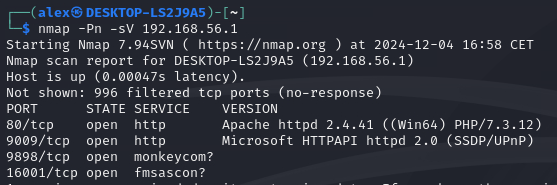
set RHOSTS 185.34.102.254

set THREADS 10

run



La commande nmap avec le parametre –sV affiche les informations du service utilisant chaque port (s’il est connu).



**Tester l'IP avec SQLMap**

**sqlmap -u "http://185.34.102.254" --batch --risk=3 --level=5**

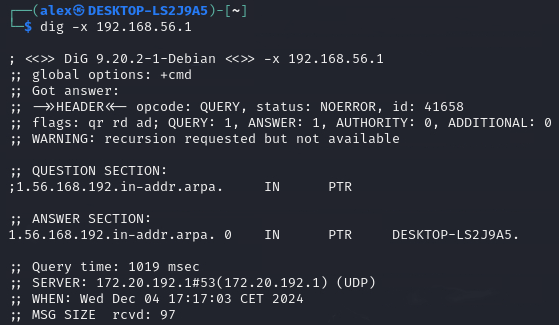
Analyser l’IP avec Whois et Dig

whois 192.168.56.1



Dig (DNS) :

dig -x 192.168.56.1



### **Précautions :**

### Basées sur les tests et failles identifiées, voici des recommandations pour améliorer la sécurité :

#### **Sécurisation du projet Spring Boot**

**Renforcer l'authentification** :

Utiliser des mots de passe forts (8+ caractères, incluant majuscules, chiffres, caractères spéciaux).

Implémenter une limitation des tentatives de connexion pour éviter les attaques par force brute.

**Protéger les Endpoints** :

Implémenter un contrôle d'accès basé sur les rôles (RBAC) pour restreindre /api/clients et /client/add.

Valider toutes les entrées utilisateur avec des outils comme Hibernate Validator.

**Prévenir les injections SQL** :

Utiliser des requêtes préparées (Prepared Statements).

Analyser les logs pour détecter des tentatives d’injections SQL.

**Mettre en œuvre des protections CSRF** (Cross-Site Request Forgery) :

Ajouter un token CSRF pour les requêtes POST.

Supprimer les sauvegardes ou fichiers non nécessaires sur le serveur.

**Mise à jour des composants** :

Assurez-vous que Laravel, Apache, et OpenSSH sont à jour.

Vérifiez régulièrement les CVE et appliquez les correctifs disponibles.

**Configurer le serveur Apache** :

Désactiver le listing de répertoire avec cette directive dans httpd.conf

#### **Sécurisation de l'application Laravel**

**Sécuriser les fichiers sensibles** :

Empêcher l'accès public aux fichiers .env et .git en configurant .htaccess

Supprimer les sauvegardes ou fichiers non nécessaires sur le serveur.

**Mise à jour des composants** :

Assurez-vous que Laravel, Apache, et OpenSSH sont à jour.

Vérifiez régulièrement les CVE et appliquez les correctifs disponibles.

**Configurer le serveur Apache** :

Désactiver le listing de répertoire avec cette directive dans httpd.conf

Restreindre l'accès à certaines ressources par IP.

**Activer les logs Laravel** :

Configurez un système de monitoring des logs pour détecter des comportements suspects.

#### **Bonnes pratiques générales**

**Isoler les environnements** :

Placez les bases de données dans un réseau interne inaccessible depuis l'extérieur.

Utilisez des conteneurs (Docker) pour isoler les applications.

**Scanner régulièrement les vulnérabilités** :

Planifiez des audits réguliers avec des outils comme Nessus ou OpenVAS.

**Former les développeurs** :

Organisez des formations sur les concepts de sécurité comme l'OWASP Top 10.