

XSS

Durée Totale du Module : 7H



Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Présentation de XSS	4
Définition	4
Les différents types de failles XSS	4
Conséquences	5
Protection contre les failles XSS	5
Les différentes techniques d'injection de code malveillant	6
1-Injection de code Javascript	6
2. Failles XSS (Cross-Site Scripting) en Javascript	7
a. XSS stocké (Stored XSS)	7
b. XSS Reflétée (Reflected XSS)	8
c. DOM-based XSS	9
Sécurité contre les attaques XSS : Bonnes pratiques en JavaScript	10
Utilisation des bonnes pratiques	10
1. Validation et filtrage des données	10
2. Échapper les caractères spéciaux	10
3. Utilisation de bibliothèques de sécurisation	10
4. Utilisation de Content Security Policy (CSP)	11
5. Sensibilisation et formation	12
6. Mise à jours	12
Damn Vulnerable Web Application (DVWA)	13
Installation	13
Configuration	14
Création base de données	15
Démo : XSS Reflected	20

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Exercice XSS Reflected (Low)	21
Exercice XSS Reflected (Medium)	23
Exercice XSS Reflected (High)	25
Demo XSS Stocké	27
Exercice XSS Sotred (Low)	28
Exercice XSS Stored (Médium)	30
Démo XSS Basée sur DOM	40
Exercice XSS DOM (low)	41
Test Librairie Sanitizer	43
Exercice DOMPurify	44
Sources Utiles	45

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Présentation de XSS

Définition

Les failles XSS (Cross-Site Scripting) sont l'une des vulnérabilités de sécurité les plus courantes sur le web. Elles permettent à un attaquant d'introduire du code malveillant dans une application web, généralement par le biais d'entrées fournies par les utilisateurs. Lorsqu'un utilisateur consulte une page infectée par une faille XSS, ce code malveillant est exécuté sur son navigateur, permettant ainsi à l'attaquant d'effectuer différentes actions nuisibles, telles que la récupération des informations de l'utilisateur, la modification du contenu de la page ou encore la redirection vers des sites malveillants.

Les différents types de failles XSS

Il existe plusieurs types de failles XSS, chacun ayant ses propres caractéristiques. Voici les trois types les plus courants :

XSS stockées (Persistent) : également appelées "persistent XSS", ces failles se produisent lorsque le code malveillant est injecté dans une application web et persiste dans la base de données ou autre support de stockage. Lorsque la page contenant cette faille est affichée, le code est exécuté pour tous les utilisateurs qui la consultent.

XSS réfléchies (reflected) : dans ce cas, le code malveillant est intégré dans l'URL de la page web, puis renvoyé au navigateur de l'utilisateur lorsqu'il accède à cette URL spécifique. Cela signifie que le code n'est pas stocké et n'est exécuté qu'une seule fois, lors de la consultation de la page infectée.

XSS basées sur le DOM (DOM Based) : ce type de faille XSS exploite les manipulations du Document Object Model (DOM) dans le navigateur de l'utilisateur. Le code malveillant est injecté dans le DOM de la page, ce qui permet à l'attaquant de manipuler le contenu et le comportement de la page de manière dynamique.

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Conséquences

Les conséquences des failles XSS peuvent être très graves pour les utilisateurs et les propriétaires des applications web. Voici quelques exemples de ce que peuvent faire les attaquants grâce aux failles XSS :

Vol de données confidentielles, telles que les identifiants de connexion, les informations personnelles ou les données financières.

Manipulation du contenu de la page, ce qui peut induire les utilisateurs en erreur, diffuser de fausses informations ou proférer des attaques de phishing.

Redirection des utilisateurs vers des sites malveillants, où d'autres attaques peuvent être lancées.



<https://www.wired.com/story/british-airways-hack-details/>

Selon l'article, British Airways a révélé une faille de sécurité affectant les informations des clients pour environ 380 000 transactions effectuées entre le 21 août et le 5 septembre. Les noms, adresses, adresses e-mail et détails de cartes de paiement sensibles ont tous été compromis. Des chercheurs de la société RiskIQ ont identifié que les pirates informatiques ont injecté du code malveillant dans une composante script de la page web de British Airways liée aux informations sur les bagages, ce qui a permis de voler les données des clients lorsqu'ils soumettaient un formulaire de paiement.

Protection contre les failles XSS

Pour se protéger contre les failles XSS, il est essentiel de prendre certaines mesures de sécurité appropriées lors du développement d'une application web. Voici quelques bonnes pratiques à suivre :

Validation des entrées utilisateur : toutes les entrées fournies par les utilisateurs doivent être validées et filtrées pour supprimer ou neutraliser tout code potentiellement dangereux.

Échappement des caractères spéciaux : avant d'afficher les données de l'utilisateur sur une page web, assurez-vous d'échapper tous les caractères spéciaux qui pourraient être interprétés comme du code exécutable.

Utilisation de fonctionnalités de sécurité fournies par les frameworks / Librairies : de nombreux frameworks web offrent des fonctionnalités de sécurité intégrées, telles que l'échappement automatique des caractères spéciaux ou la validation des entrées. Il est conseillé d'utiliser ces fonctionnalités pour éviter les erreurs de sécurité.

Formation et sensibilisation des développeurs : les développeurs doivent être conscients des risques liés aux failles XSS et être formés aux bonnes pratiques de sécurité pour minimiser les vulnérabilités.

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Les différentes techniques d'injection de code malveillant

L'injection de code malveillant est une technique couramment utilisée par les pirates informatiques afin d'exploiter les failles de sécurité dans les applications web. Elle permet d'insérer et d'exécuter du code malveillant sur le serveur ou le navigateur de l'utilisateur, compromettant ainsi la sécurité des données et pouvant causer divers dommages.

1-Injection de code Javascript

L'injection de code JavaScript est l'une des techniques les plus couramment utilisées pour exécuter du code malveillant sur un site web. Elle bénéficie de la popularité du langage JavaScript, largement utilisé dans le développement web.

Les pirates peuvent exploiter différentes failles pour réaliser une injection de code JavaScript, tels que des formulaires non sécurisés, des paramètres d'URL non filtrés ou des champs de saisie non protégés. Une fois le code injecté, il peut être exécuté directement sur le navigateur de l'utilisateur, ce qui peut avoir de graves conséquences.

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

2. Failles XSS (Cross-Site Scripting) en Javascript

Les failles XSS sont des vulnérabilités largement exploitées dans les applications web. Ces failles permettent à un attaquant d'injecter et d'exécuter du code malveillant sur les navigateurs des utilisateurs. Les failles XSS peuvent être classées en trois catégories principales :

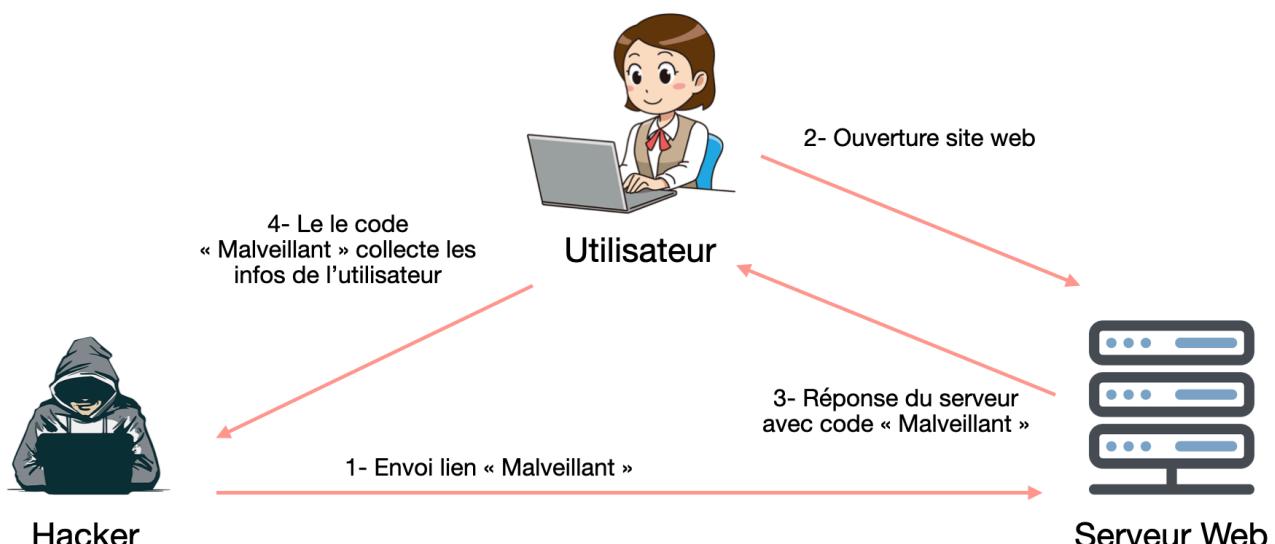
a. XSS stocké (Stored XSS)

Le XSS stocké se produit lorsque le code malveillant est stocké dans une base de données ou un système de stockage et est renvoyé aux utilisateurs chaque fois qu'ils accèdent à une page web spécifique. Cela peut conduire à une compromission généralisée des utilisateurs qui visitent cette page.

L'app reçoit des données d'une source non fiable, l'application inclut ces données non fiables dans sa réponse. En d'autres termes, le hacker injecte du code malveillant dans une appli web, ce code peut par exemple être stocké dans une BDD (Bdd, forum, formulaire, champ de commentaire). Et à chaque fois que l'utilisateur accède à l'application web, le code malveillant est exécuté.

(Stocké / persistant car le code malveillant est stocké sur le serveur qui héberge l'application web).

XSS Stored



Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



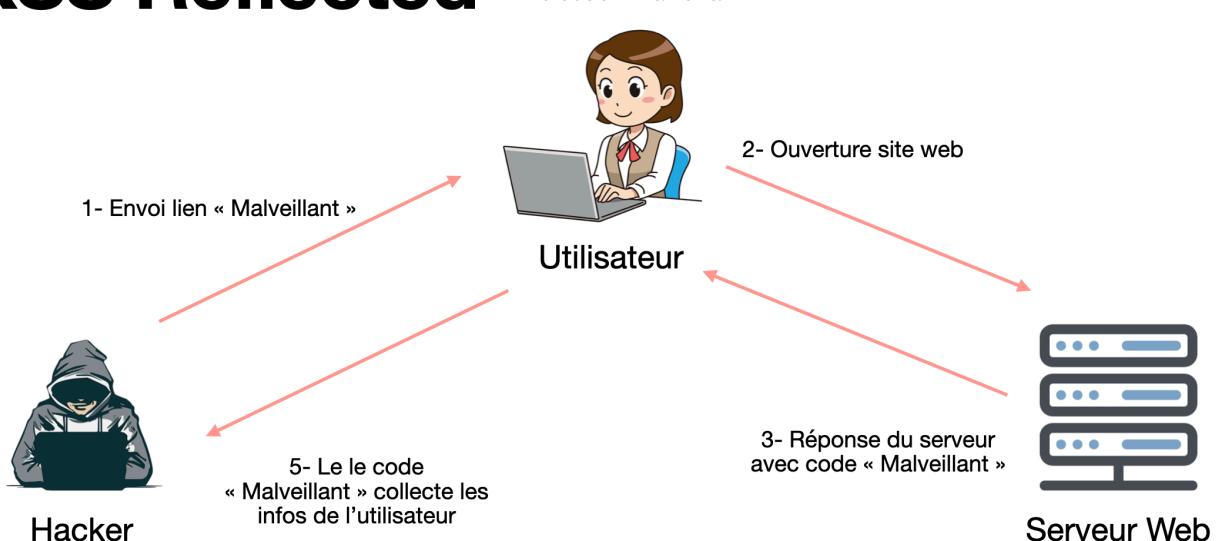
Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

b. XSS Reflétée (Reflected XSS)

Les failles XSS reflétées se produisent lorsque le code malveillant est envoyé par le biais de paramètres d'URL ou de formulaires, puis renvoyé au navigateur de l'utilisateur dans la réponse du serveur.

L'utilisateur peut être trompé pour exécuter le code malveillant en ouvrant un lien ou en soumettant un formulaire infecté.

XSS Reflected



Une application reçoit des données et traite cette requête sans vérifier le contenu, dans ce cadre on parle de XSS non persistant car le code malveillant n'est pas stocké sur le serveur mais plutôt inclus dans une URL (Cette même URL peut être distribuée dans des campagnes de phishing ou ingé. Sociale)

Auteur :
 Jean-François Pech
Relu, validé & visé par :
 Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date création :

03/03/2023

Date révision :

10/03/2023

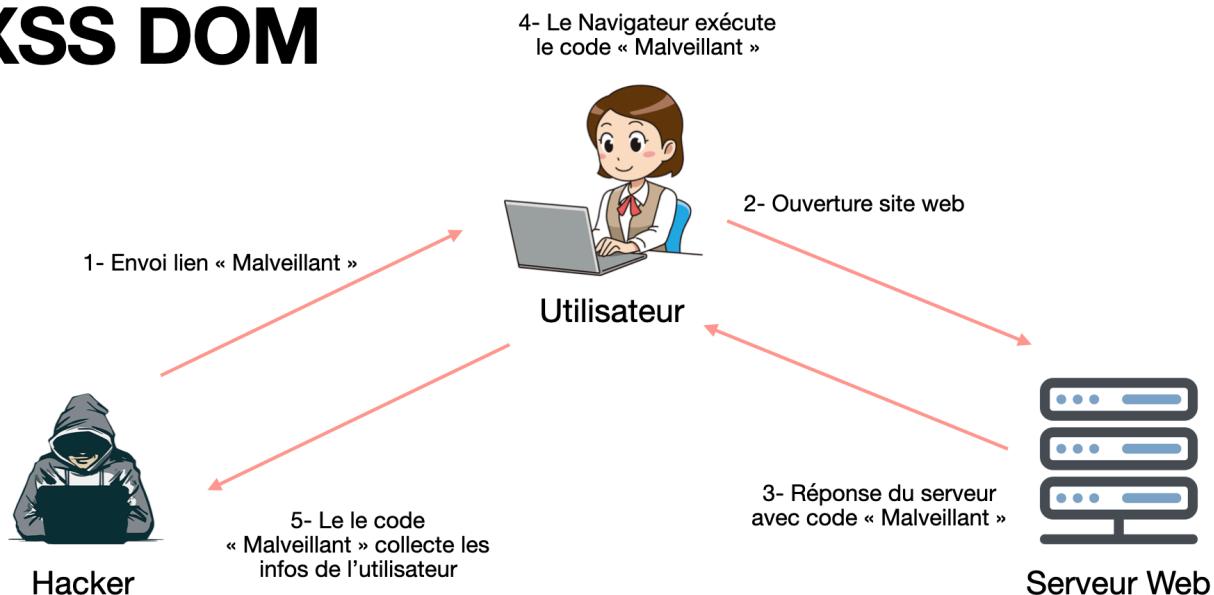


Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

c. DOM-based XSS

Le DOM-based XSS se produit lorsque le code malveillant est injecté directement dans le DOM (Document Object Model) de la page web. Lorsque le navigateur interprète le code, il exécute le code malveillant, pouvant entraîner des conséquences néfastes.

XSS DOM



XSS Dom Based se produit quand une app utilise du JS côté client qui traite des données provenant d'une source non fiable

Les données traitées vont permettre de modifier le DOM

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Sécurité contre les attaques XSS : Bonnes pratiques en JavaScript

Les attaques XSS (Cross-Site Scripting) sont l'une des principales vulnérabilités rencontrées dans les applications Web. Ces attaques permettent à des attaquants d'injecter du code malveillant dans les pages Web et de compromettre la sécurité des utilisateurs. Il est donc essentiel de connaître et d'appliquer les bonnes pratiques pour se protéger contre les attaques XSS.

Utilisation des bonnes pratiques

1. Validation et filtrage des données

L'une des meilleures façons de se protéger contre les attaques XSS est de valider et de filtrer toutes les données entrantes et sortantes. Cela permet de s'assurer que seules les données sécurisées et autorisées sont affichées sur les pages Web. Utilisez des techniques de validation des données, telles que les filtres de caractères, l'échappement des caractères spéciaux, et les fonctions de validation intégrées aux frameworks ou bibliothèques JavaScript.

2. Échapper les caractères spéciaux

Une autre bonne pratique consiste à échapper les caractères spéciaux présents dans les données dynamiques affichées sur les pages Web. Utilisez des fonctions d'échappement intégrées pour convertir les caractères spéciaux en entités HTML ou en codes de caractères. Cela garantit que les caractères ne sont pas interprétés comme des balises HTML ou des scripts.

`htmlspecialchars()` une fonction native de **PHP** qui va encoder les caractères (exemple converti le '<' en « < »)

3. Utilisation de bibliothèques de sécurisation

Les bibliothèques de sécurisation, telles que **DOMPurify** ou **Bleach**, peuvent être utilisées pour nettoyer et filtrer automatiquement les données avant de les afficher sur les pages Web. Ces bibliothèques détectent les balises HTML et les scripts malveillants et les suppriment ou les désactivent, protégeant ainsi contre les attaques XSS.

Auteur :

Jean-François Pech

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date création :

03/03/2023

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

4. Utilisation de Content Security Policy (CSP)

La mise en place d'une politique de sécurité du contenu (CSP) est également une pratique recommandée pour se protéger contre les attaques XSS. Cette politique permet de définir des règles strictes sur les types de contenu autorisés à être chargés et exécutés sur une page Web. Elle bloque tout contenu provenant de sources non autorisées, réduisant ainsi les risques d'attaques XSS.

Permet de définir des domaines autorisés pour l'exécution d'un script.

Exemple d'une CSP (c'est une entête HTTP)

Liste blanche

```
Content-Security-Policy
script-src 'self' *.pinterest.com *.pinimg.com *.google.com connect.facebook.net
*.google-analytics.com *.accountkit.com *.facebook.com *.googletagmanager.com
*.branch.io *.yozio.com cdn.ampproject.org *.cdn.ampproject.org radar.cedexis.com
static.zdassets.com ekr.zdassets.com *.zopim.com *.zopim.org *.zopim.io;
https://analytics.tiktok.com https://cdn.cookielaw.org/scripttemplates/
https://js.datadome.co https://www.dwin1.com/ https://cdn.polyfill.io
https://widget.simplybook.pro/ https://*.app.smart-tribune.com https://connect.facebook.net/
https://cdn.pbb1.co/ https://*.takingbackjuly.com/ https://embed.typeform.com https://*.inside-graph.com/
https://geolocation.onetrust.com/cookieconsentpub/v1/geo/location
https://googleads.g.doubleclick.net/ https://intgepi.bglobale.com https://maps.googleapis.com/
```

Dans l'exemple d'une liste blanche de src de script, en premier on a un self qui correspond au nom de domaine sur lequel se trouve votre application et ensuite une liste de nom de domaine dont ont autorisé les scripts

Ci dessus c'est un extrait mais sur certaine appli ces CSP peuvent être très voire trop conséquentes

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

5. Sensibilisation et formation

Enfin, la sensibilisation et la formation des développeurs et des utilisateurs sont essentielles pour se protéger contre les attaques XSS. En enseignant les bonnes pratiques de sécurité, telle que l'importance de la validation des données et de l'échappement des caractères spéciaux, il est possible de réduire les risques d'attaques XSS au sein des applications Web.

⚠ aux liens suspects (QRCode ,mails, forums, réseaux sociaux, sms etc...)

6. Mise à jours

La mise à jour des serveurs web (technologies, os (derniers patch de sécurité)

Mettre à jour les navigateurs web (mise à jour auto ?)

AntiVirus ? (Norton :) ?)

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Damn Vulnerable Web Application (DVWA)

L'application Damn Vulnerable Web App (V) est une application Web PHP/MySQL qui est « Damn » vulnérable. Ses principaux objectifs sont d'aider les professionnels de la sécurité à tester leurs compétences et leurs outils dans un environnement légal, d'aider les développeurs Web à mieux comprendre les processus de sécurisation des applications Web et d'aider les enseignants/étudiants à enseigner/apprendre la sécurité des applications Web dans un cadre scolaire.

Installation

Pré-requis :

De quoi simuler un serveur web (Apache, MySQL) donc une application comme Wamp, Mamp ou Xamp

Récupérer le projet depuis le repository Git :

<https://github.com/digininja/DVWA>

Renommer le dossier en « DVWA »

(Extraire l'archive et placer le dossier du projet dans votre répertoire « www » ou « htdocs »

(Ouvrir le dossier du projet dans votre éditeur de code)

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Configuration

Dans le dossier config le fichier **config.inc.php.dist** renommer le en « **config.inc.php** »

De base dans ce fichier de config, on a un système de tableau pour gérer différents paramètres de l'application

Nous allons ré-utiliser ces informations pour configurer (côté phpMyAdmin) la base de données, un utilisateur et son mot de passe (à voir si besoin de modifier le port utilisé par la base de données (8889)).

```
$_DVWA = array();  
$_DVWA[ 'db_server' ] = getenv('DB_SERVER') ?: '127.0.0.1';  
$_DVWA[ 'db_database' ] = 'dvwa';  
$_DVWA[ 'db_user' ] = 'dvwa';  
$_DVWA[ 'db_password' ] = 'p@ssw0rd';  
$_DVWA[ 'db_port' ] = '3306';
```

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

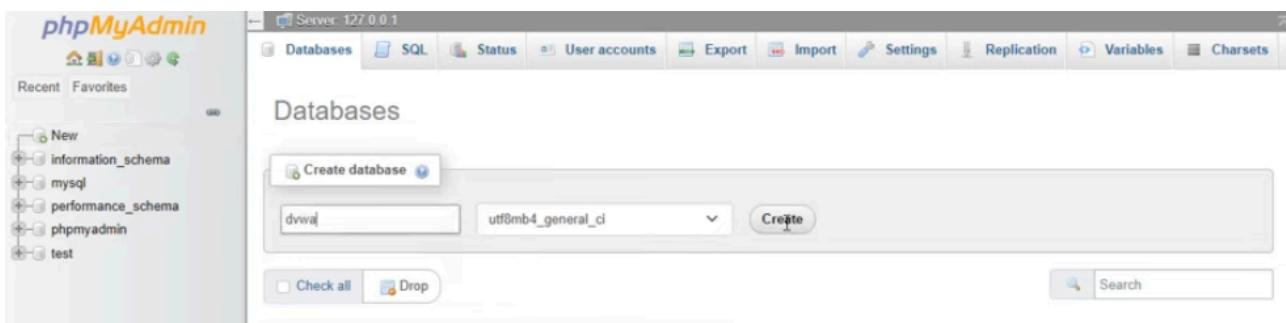
10/03/2023



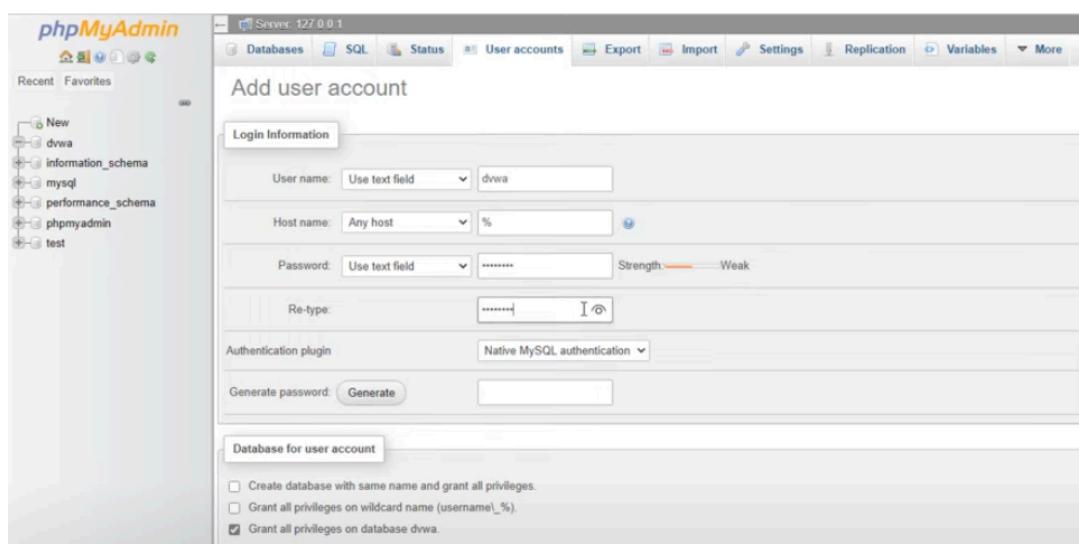
Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

Création base de données

Dans PhpMyAdmin créez une nouvelle base de données « dvwa »

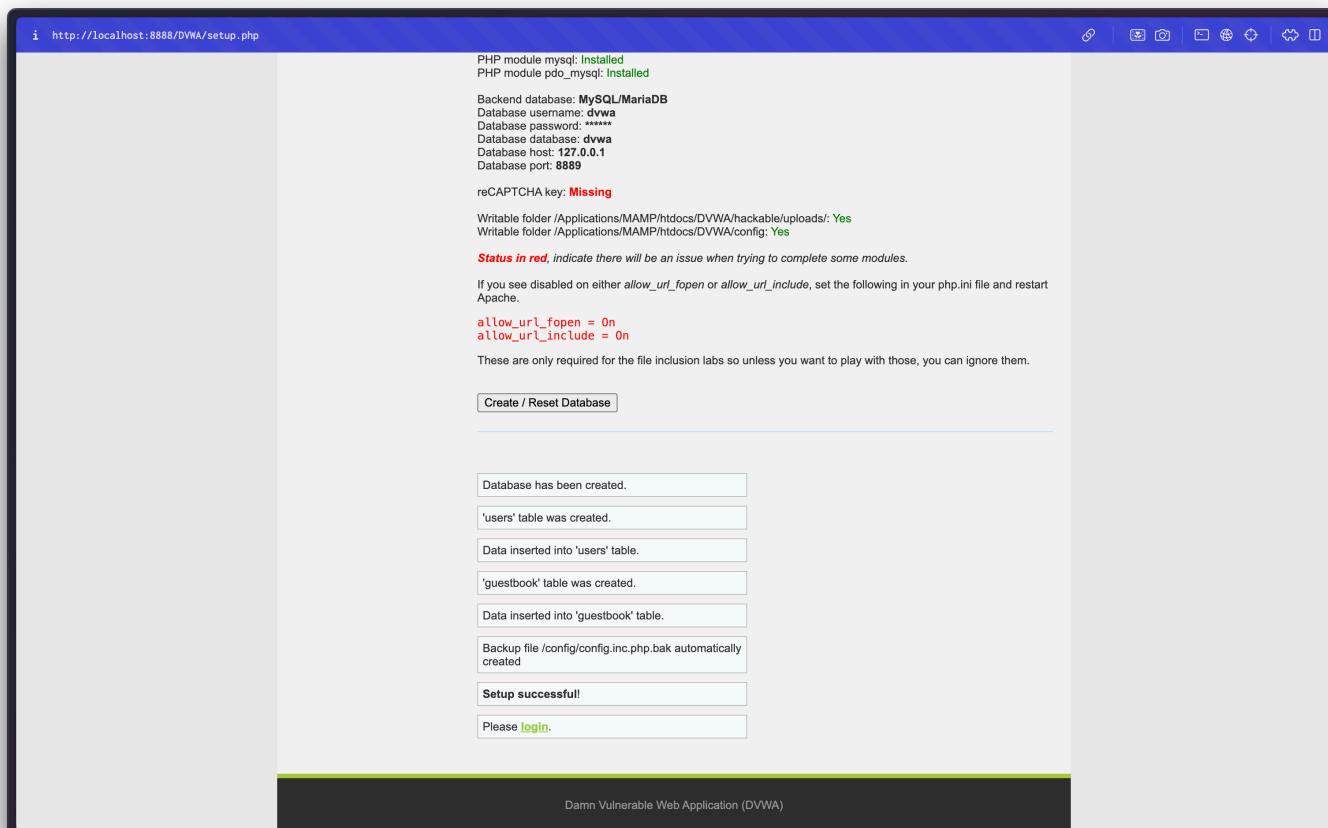


Dans votre base de données « dvwa », dans l'onglet privilèges, ajouter un utilisateur et son mot de passe : « p@ssw0rd »



Une fois configuré on peut se rendre sur notre serveur en local dans le dossier de l'application et se rendre sur la page setup.php

En bas de cette page on peut cliquer sur le bouton « Create / Reset DataBase », si on a bien tout configuré précédemment on est redirigé vers la page de login.php



The screenshot shows a web browser displaying the DVWA setup.php page. The URL in the address bar is `http://localhost:8888/DVWA/setup.php`. The page contains configuration details and a success message.

Configuration Details:

- PHP module mysql: **Installed**
- PHP module pdo_mysql: **Installed**
- Backend database: **MySQL/MariaDB**
- Database username: **dwvwa**
- Database password: *********
- Database database: **dwvwa**
- Database host: **127.0.0.1**
- Database port: **8888**

reCAPTCHA key: **Missing**

Writable folder: /Applications/MAMP/htdocs/DVWA/hackable/uploads/: **Yes**
Writable folder: /Applications/MAMP/htdocs/DVWA/config: **Yes**

Status in red, indicate there will be an issue when trying to complete some modules.

If you see disabled on either `allow_url_fopen` or `allow_url_include`, set the following in your `php.ini` file and restart Apache.

allow_url_fopen = On
allow_url_include = On

These are only required for the file inclusion labs so unless you want to play with those, you can ignore them.

Create / Reset Database

Success Message:

- Database has been created.
- 'users' table was created.
- Data inserted into 'users' table.
- 'guestbook' table was created.
- Data inserted into 'guestbook' table.
- Backup file /config/config.inc.php.bak automatically created
- Setup successful!**
- Please [login](#).

Damn Vulnerable Web Application (DVWA)

Auteur :

Jean-François Pech

Relu, validé & visé par :

Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date création :

03/03/2023

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

On peut se connecter en tant que « admin » avec « password »



A screenshot of a web browser displaying the DVWA (Damn Vulnerable Web Application) login page. The URL in the address bar is `http://localhost:8888/DVWA/login.php`. The page features a large DVWA logo at the top center. Below it is a form with two input fields: 'Username' containing 'admin' and 'Password' containing 'password'. A 'Login' button is positioned below the password field. At the bottom of the page, a small link reads 'Damn Vulnerable Web Application (DVWA)'.

Auteur :

Jean-François Pech

Date création :

03/03/2023

Relu, validé & visé par :

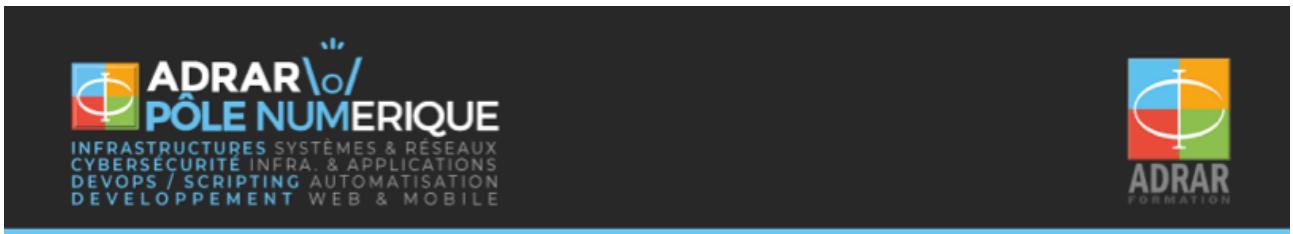
Jérôme CHRETIENNE
 Sophie POULAKOS
 Mathieu PARIS

Date révision :

10/03/2023



Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.



Et on arrive sur l'interface de base de l'application DVWA(su la gauche vous avec plusieurs catégories, nous allons surtout nous intéresser à celles basées sur XSS)

The screenshot shows the DVWA homepage. The URL in the address bar is <http://localhost:8888/DVWA/index.php>. The page title is 'DVWA'. On the left, there is a sidebar menu with the following items:

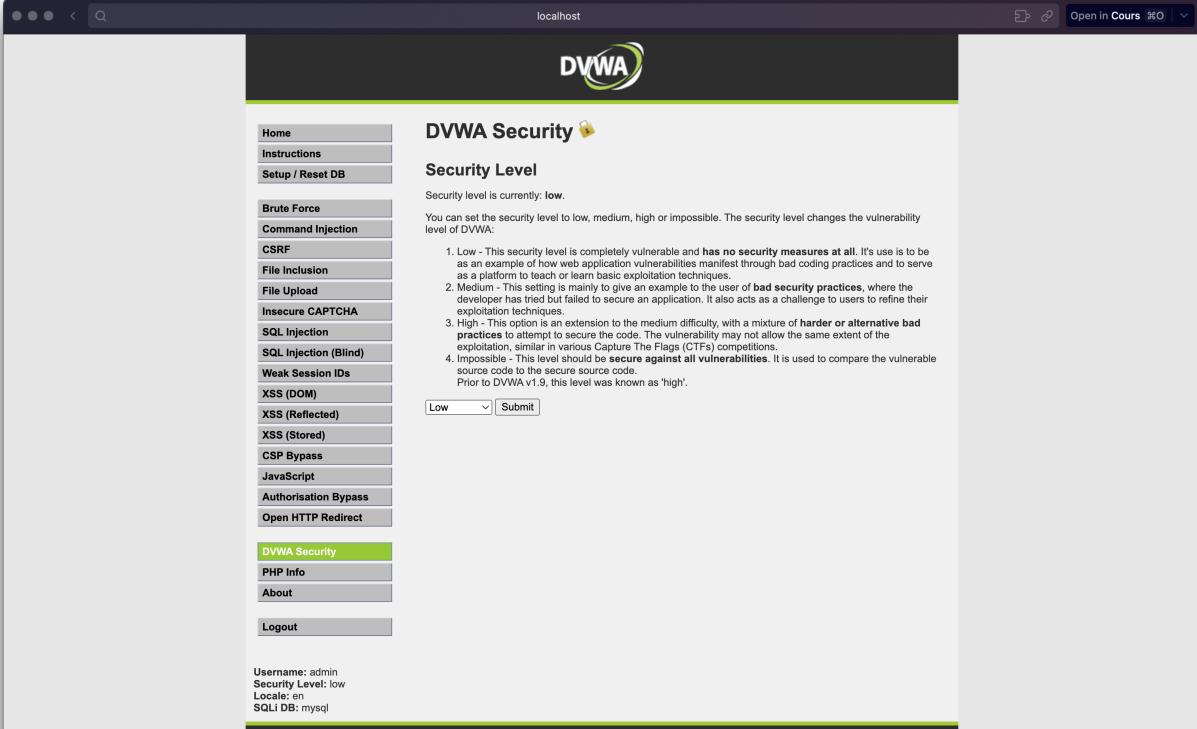
- Home
- Instructions**
- Setup / Reset DB
- Brute Force
- Command Injection
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection
- SQL Injection (Blind)
- Weak Session IDs
- XSS (DOM)
- XSS (Reflected)
- XSS (Stored)
- CSP Bypass
- JavaScript
- Authorisation Bypass
- Open HTTP Redirect
- DVWA Security
- PHP Info
- About
- Logout

The main content area starts with 'Welcome to Damn Vulnerable Web Application!'. It explains that DVWA is a PHP/MySQL web application designed to help security professionals test their skills and tools in a legal environment. It aims to teach web developers about securing web applications and aid students and teachers. The goal is to practice common web vulnerabilities at various difficulty levels. The page also includes sections for 'General Instructions', 'WARNING!', and 'More Training Resources', along with a note about the application being vulnerable and instructions for running it on a virtual machine. At the bottom, it says 'You have logged in as 'admin''.

Auteur :	Date création :
Jean-François Pech	03/03/2023
Relu, validé & visé par :	Date révision :
<input checked="" type="checkbox"/> Jérôme CHRETIENNE <input checked="" type="checkbox"/> Sophie POULAKOS <input checked="" type="checkbox"/> Mathieu PARIS	10/03/2023

Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.

De plus, sachez que l'application va proposer 3 niveau de sécurité différent que l'on peut régler dans cette page : security.php



The screenshot shows the DVWA Security Level interface. On the left, there's a sidebar with various exploit options like Brute Force, Command Injection, and SQL Injection. The main area is titled "DVWA Security" with a gear icon. It says "Security Level" and "Security level is currently: low." Below that is a detailed description of the security levels:

- Low - This setting is completely vulnerable. It has no security measures at all. Its use is to be as an example of how an application's vulnerabilities can exist through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
- Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
- High - This option is an extension on medium difficulty, with a mixture of **harder or alternative bad practices** to challenge users to learn more. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
- Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code. Prior to DVWA v1.9, this level was known as 'high'.

A dropdown menu shows "Low" is selected, with a "Submit" button next to it. At the bottom, it shows the user information: "Username: admin", "Security Level: low", "Locale: en", and "SQLI DB: mysql".