

STI - Projet Partie 2

Rapport

Léo Cortès

Steve Henriquet

7 janvier 2019



HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Hes·SO
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts
Western Switzerland

Contents

1	Introduction	3
2	Vulnérabilités et corrections	3
2.1	XSS	3
2.1.1	Exploitation	3
2.1.2	Correction	4
2.2	Injection SQL	4
2.2.1	Exploitation	4
2.2.2	Correction	4

Introduction

Vulnérabilités et corrections

XSS

Exploitation

Le programme est vulnérable aux attaques XSS. Les balises HTML sont correctement interprétée, ainsi que les balises de script. C'est problématique car un attaquant pourrait envoyer un message malicieux à un administrateur. Lorsque l'admin se connecte, le script pourrait récupérer ses cookies et l'attaquant pourrait les rejouer et devenir donc administrateur.

Voici le message envoyé par un attaquant quelconque :

Création utilisateur Modification utilisateur Changer de mot de passe Nouveau message Déconnexion Nom d'utilisateur : c.l

Boîte de réception

Destinataire:

Sujet:

Message:

Envoyer

Figure 1: Message malicieux

Voici le résultat dans la boîte de réception de la victime.

c.l

Test

2019-01-07 13:07:48

Répondre Supprimer

Figure 2: Boîte de réception

Lorsque la victime l'ouvre, voici le script est correctement exécuté.

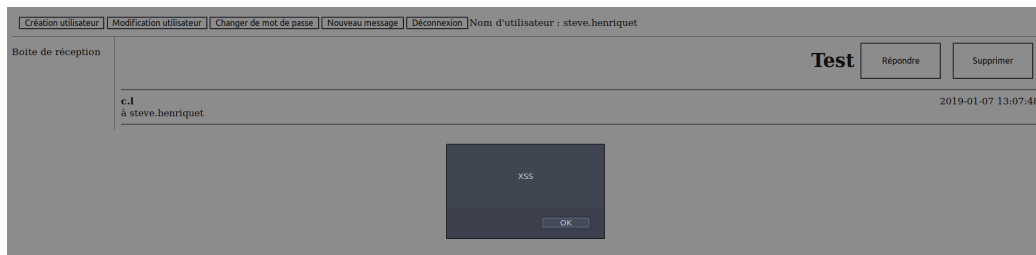


Figure 3: Exécution XSS

Correction

Les entrées ont du être "sanitisées". Nous avons utilisé le filtre *FILTER_SANITIZE_STRING* dans les divers inputs accessibles.

```
$pass = filter_var ( $_POST["pass"], filter: FILTER_SANITIZE_STRING);
$passCheck = filter_var ( $_POST["passCheck"], filter: FILTER_SANITIZE_STRING);
```

Figure 4: Exemple d'assainisation

Injection SQL

Exploitation

Correction

Tout comme pour les failles XSS, les entrées ont du être assainies.

Problèmes non résolubles

- PHP 5.6
- Gestion des cookies de session