Challenge pour les hackers !

Elément	Description
Type de travail	Individuel
Objectif pédagogique	Protéger une application **PHP** contre les attaques par injection SQL en utilisant le connecteur **PDO**
Durée estimée	30 min
Fichiers sources	Le code de l'application **PHP** est fourni par l'enseignant et est disponible dans **Teams** dans le canal 151
A produire	Répondre aux questions directement dans ce document
Moyens d'aide	Internet

1 Introduction

Cet exercice a pour but de vous faire découvrir les attaques par injection **SQL**. Mais surtout de vous montrer, comment, en tant que développeur, vous devez protéger vos applications contre ce genre d'attaque.

Avant toute de chose, vous devez commencer par installer l'application **PHP**. Le code source de cette application est disponible sous **Teams** dans le **canal 151**.

2 Installation de l'application PHP

Pour installer l'application **PHP** sur votre environnement, vous devez :

- Récupérer le code de l'application présent sur **Teams**.
- Exécuter le fichier **dump.sql** présent à la racine du répertoire fourni.

Comme vous en avez maintenant l'habitude, l'outil **uWamp** est recommandé pour héberger l'application **PHP**.

Une fois le serveur web et la base de données démarrés, vous devriez avoir dans votre navigateur quelque chose qui ressemble à cela :

Ensuite, vous pouvez vérifier que vous pouvez vous connecter grâce à l'utilisateur **toto.**

Pour cela, vous devez saisir :

- Dans le champ « Nom de l'utilisateur » : **toto**
- Dans le champ « Mot de passe » : **toto**

Voilà le résultat obtenu :

A noter que l'application :

- Indique que l'utilisateur est bien connecté
- Dit bonjour à l'utilisateur connecté
- Affiche également la requête **SQL** effectuée à partir des informations saisies par l'utilisateur

3 Votre mission :

Votre mission consiste dans un 1er temps à exploiter la faille puis ensuite à corriger le code pour que l'application ne soit plus vulnérable à l'attaque par injection **SQL**.

3.1 Exploiter la faille

Cette application a plusieurs problèmes liés à la sécurité (vulnérabilités du code, mauvaises pratiques, etc).

Vous devez exploiter une faille (une vulnérabilité) de l'application.
En effet, vous devez réaliser une **attaque par injection SQL** qui va vous **permettre de vous authentifier** (= connecter) à l'application.

Attention :
Vous n'avez pas le droit de modifier le code de l'application.

La seule chose que vous avez le droit de faire est de saisir des informations dans les champs **Nom de l'utilisateur** et/ou **Mot de passe**.

Mais évidemment, sans utiliser les informations de l'un des deux utilisateurs présents en base de données.

En effet, le hacker ne dispose d'aucune information sur les données.

Merci d'indiquer ci-dessous les valeurs saisies dans les différents champs permettant l'attaque par injection **SQL**.

3.2 Corriger le code de l'application

Maintenant que vous avez compris comment réaliser une attaque par injection **SQL**, votre mission en tant que développeur est de corriger le code afin que l'application ne soit plus vulnérable à ce type d'attaque.

Expliquez ci-dessous comment allez-vous faire ?

Maintenant que vous avez corrigé le code de l'application, vous devez vérifier que le code est maintenant robuste et fiable en tentant à nouveau l'attaque réalisée à la section 3.1.

Quel résultat constatez-vous ?

4 Conclusion

Grâce à cette activité, vous savez maintenant comment protéger vos applications **PHP** contre les attaques par injection **SQL**.

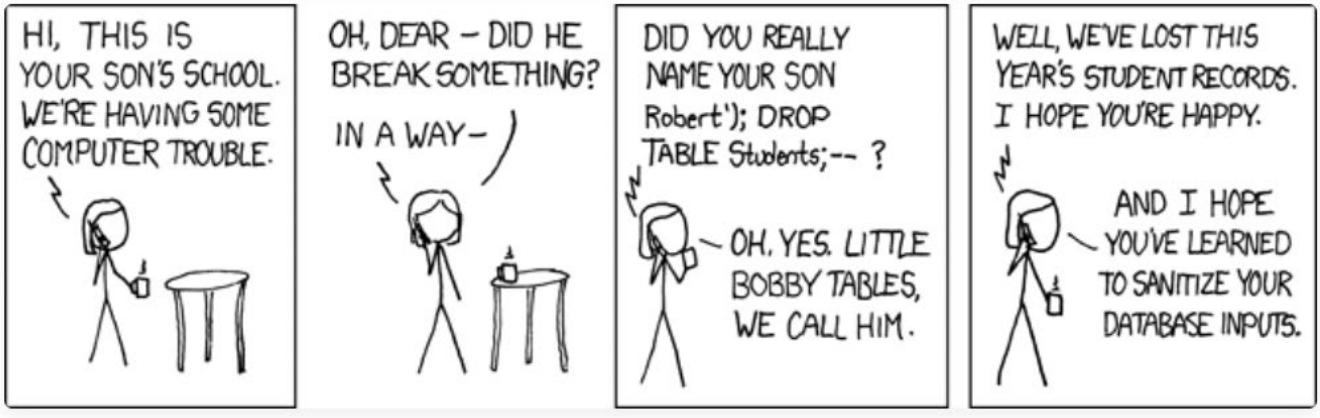
En effet, pour chaque requête **SQL** qui nécessite une concaténation avec des valeurs provenant de l'utilisateur (ou de l'extérieur de l'application), vous devez utiliser la méthode **prepare()** de **PDO** et **binder** les variables à concaténer.

5 Et maintenant ?

Maintenant, vous allez pouvoir reprendre le code de votre application des « surnoms des enseignants » et utiliser la méthode **prepare()** là où cela est nécessaire.

Mais avant, je vous propose un quizz pour vérifier que vous avez acquis toutes les connaissances requises : https://forms.office.com/r/if6pEDReH7

Avant de nous quitter …



– Un dessin animé sur l'injection SQL (source d'image : XKCD)