Cours de sécurité accéléré no. 2 - Injections SQL

Cours de sécurité accéléré no. 2 - Injections SQL

Cet article a été écrit par Damien Metzger, et publié sur le blog de PrestaShop le 8 août 2011.

Une injection SQL consiste pour un pirate à insérer dans une requête SQL des données qui ne sont pas celles attendues par le script. Sans contrôle adéquat, il est très facile de modifier le comportement d'une requête, c'est pourquoi les tentatives d'attaques de ce type sont très courantes et qu'il doit être dans votre seconde nature de vous en protéger.

Un exemple valant mieux qu'un long discours, voici ce qui peut se passer sur une page d'identification.

Votre requête :

```
SELECT id FROM users WHERE email = ‘damien@prestashop.com' AND password = ‘$plop'
```

Exploit

Le pirate renseigne comme mot de passe `‘ OR ‘1' = ‘1`
La requête finale sera donc :

```
SELECT id FROM users WHERE email = ‘damien@prestashop.com' AND password = ‘‘ OR ‘1' = ‘1'
```

Elle renverra donc toujours l'identifiant de l'utilisateur, sans avoir à connaître son mot de passe. Comment s'en protéger alors ? Rien de plus simple, tant qu'on y pense à tous les coups !

Il y a 2 types d'injections courantes :

1. Celles qui exploitent une absence de protection des quotes et double quotes
2. Celles qui exploitent l'absence de contrôle des types de données

Pour la première, vous devez donc "échapper" les quotes simples et doubles avec une fonction adéquate :

- La fonction standard de PHP est addslashes().
- Si vous avez une instance de connexion à MySQL ouverte, vous pouvez privilégier la fonction mysql_real_escape_string(), qui couvre également quelques caractères supplémentaires.
- Dans PrestaShop, il faut utiliser pSQL(), qui est encore plus strict avec tout ce qui ressemble à du HTML et gère les magic quotes (les magic quotes sont une option de configuration sur certains serveurs qui automatise l'échappement des quotes ; si l'idée part d'un bon sentiment, sa présence ou son absence provoque des différences de comportements entre les serveurs gênants, PrestaShop en déconseille donc l'utilisation).

Pour la seconde, qui concerne principalement les données de type numérique, il suffit de faire un cast. Il est beaucoup plus difficile d'exploiter une requête lorsqu'on est limité aux entiers et aux flottants !

Enfin, la plupart des outils proposant une couche d'abstraction pour l'accès à la base de données intègrent différents mécanismes de protection. En particulier, le binding de paramètre est un bon moyen de sécuriser ses requêtes.

Pour mettre à l'épreuve vos scripts, il existe des outils automatisant les tentatives d'injections. Sans avoir la finesse d'un pirate et sans non plus offrir de garantie de sécurité, c'est un bon moyen de vérifier s'il n'y a pas eu quelques oublis.

Pour l'anecdote, il y avait des élections l'année dernière en Suède. Il faut savoir que la loi autorise les votants à écrire leurs bulletins à la main. Les résultats étant très serrés, le gouvernement a choisi de publier – anonymement – la liste des réponses des bulletins papiers. Parmi les nombreux noms de candidats, un malin avait écrit sur son bulletin de vote « pwn DROP TABLE VALJ » : sachant que les bulletins seraient numérisés et insérés en base de données, il n'avait pour ambition – sérieuse ou non – que de complètement supprimer la table contenant les résultats ! Comme quoi tout est possible pour parvenir à ses fins.