Les injections SQL classiques

[sommaire]

1. **Objectifs**

- Etre capable d'exploiter l'injections SQL comme : bypass de formulaire, récupération de données etc... ;

- **Présentations**

- L'**injection SQL** est une vulnérabilité de sécurité Web qui permet à un attaquant d'interférer avec les requêtes adressées par une application à sa base de données.
- Cela permet généralement à un attaquant de consulter des données qu'il n'est normalement pas en mesure de récupérer. Cela peut inclure des données appartenant à d'autres utilisateurs, ou toute autre donnée à laquelle l'application elle-même est capable d'accéder.
- Dans de nombreux cas, un attaquant peut modifier ou supprimer ces données, entraînant des modifications persistantes du contenu ou du comportement de l'application.



- De nombreux sites stockent des données dans des bases de données. C’est le cas notamment des sites bancaires, de vente en ligne mais aussi des forums, des newsletters ou des moteurs de recherche, la liste est infinie.
- Dans la majorité des cas, pour dialoguer avec ces bases de données qui sont situées sur un serveur on utilise le langage **SQL**.
- A partir de son navigateur, l’internaute envoie une demande d’informations ou d’inscriptions à un serveur. Cette demande s’effectue sous la forme d’une requête très structurée. Cette requête est analysée par le Serveur de Base de Données et si,elle est compréhensible,(je ne dis pas conforme volontairement), leSBD traite la requête et renvoie les informations au serveur de l’internaute qui les affiche conformément à la programmation de sa page internet

apcpedagogie.com

via un langage comme ASP, JS, PHP,etc...

- Une injection SQL est une insertion de code SQL via des données transmises depuis un site web.
- Une injection réussie et correctement exploitée permet de récupérer des informations sensibles d'une base de données ou encore de modifier/supprimer/ajouter des données.
- D'une manière générale, toutes les actions liées à une base de données sont possibles. Habituellement ce type d'injection concerne PHP avec une base SQL mais d'autres langages comme ASP peuvent aussi être concernés.

Prérequis

- Pour que ces exercices puissent fonctionner, vous aurez besoin d'un serveur web en état de marche comprenant le PHP ainsi que les bases de données **MySQL**. **WAMP** (ou LAMP / MAMP selon votre OS) feront très bien l'affaire mais libre à vous d'en utiliser un autre.

Test de l'existence d'un SQL Injection

- **Étape 1:** Vérifier si l'application Web se connecte à un serveur de base de données afin d'accéder à certaines données.
- **Étape 2:** Lister tous les champs de saisie, les champs cachés et les demandes de poste dont les valeurs pourraient être utilisées dans l'élaboration d'une requête SQL.
- **Étape 3:** Tenter d'injecter du code dans les champs d'entrée pour générer une erreur
- **Étape 4:** Essayer d'insérer une valeur de chaîne où un numéro est prévu dans le champ de saisie
- **Étape 5:** L'opérateur d'union est utilisé dans les injections SQL pour se joindre à une requête à la requête d'origine
- **Étape 6:** Messages d'erreur détaillés fournissent des informations à un attaquant afin d'exécuter SQL Injection.

Les types d'injection SQL

1. **Injection SQL basée sur les erreurs**

- Dans ce type d'injection, le pirate informatique analyse des différentes opérations et trouve le motif d'erreur dans la base de données. Ensuite, il y accède pour satisfaire son désir.

apcpedagogie.com

- **Injection SQL classique**
  - Dans cette technique, le pirate utilise les résultats de la base de données et pirate la base de données pour faire avancer les choses. On l'appelle aussi injection SQL intrabande.

- **Injection SQL basée sur l'union**
  - Dans cette technique, l'utilisateur combine des requêtes en utilisant la commande UNION de SQL et récupère le résultat sous forme de réponse HTTP partielle.
  - La question qui se pose c'est peut-on avoir accès aux données d'une table autre que celle visée par la requête?
    ```
    1 ' and 2=3 UNION SELECT (SELECT GROUP_CONCAT(column_name) FROM information_schema.columns WHERE table_name='users'),1 #
    ```

- **Injection de Blind SQL**
  - Cette technique d'injection SQL est utilisée par les pirates pour mettre les charges utiles. Ici, pirate donne le temps à la base de données pour exécuter la requête. Cela rend cette attaque lente dans la nature.
  - Une **Blind SQL Injection** (ou injection SQL en aveugle en français), c'est un peu un jeu de devinettes : la personne que vous interrogez ne peut répondre que par oui ou non. Il faudrait donc un certain nombre de questions avant de tomber sur l'information que vous souhaiteriez, contrairement à une requête "classique" qui nous retournerait directement le résultat.
  - L'idée avec ce genre d'injection est de se servir d'opération logiques pour tester la réaction du serveur est le retour qui est fait après la requête. L'exemple typique serait de comparer les retours de **' OR 1=1 -- et ' OR 1=2 --** et si vous observez une différence dans le rendu de la page c'est qu'il y a surement une injection. Si c'est le cas il faudra utiliser des fonction telles que substring() pour comparer lettre à lettre ce que vous cherchez.
  - Voici une liste d'autre valeur pouvant faire office de test en cas de protection sur **1=1** :
    - SELECT * FROM table WHERE 'uuu'='uuu'
    - SELECT * FROM table WHERE 1<>2
    - SELECT * FROM table WHERE 3>2
    - SELECT * FROM table WHERE 2<3
    - SELECT * FROM table WHERE 1
    - SELECT * FROM table WHERE 1+1
    - SELECT * FROM table WHERE 1--1
    - SELECT * FROM table WHERE ISNULL(NULL)

apcpedagogie.com

- SELECT * FROM table WHERE ISNULL(COT(0))
- SELECT * FROM table WHERE 1 IS NOT NULL
- SELECT * FROM table WHERE NULL IS NULL
- SELECT * FROM table WHERE 2 BETWEEN 1 AND 3
- SELECT * FROM table WHERE 'b' BETWEEN 'a' AND 'c'
- SELECT * FROM table WHERE 2 IN (0,1,2)
- SELECT * FROM table WHERE CASE WHEN 1>0 THEN 1 END

- **Injection SQL hors bande**

  - Lorsqu'un attaquant exploite l'injection SQL, l'application Web affiche parfois des messages d'erreur provenant de la base de données, affichant que la syntaxe de la requête SQL est incorrecte. L'injection SQL aveugle est presque identique à l'injection SQL normale, la seule différence étant la façon dont les données sont extraites de la base de données.
  - Lorsque la base de données ne renvoie pas de données sur la page Web, un attaquant est obligé de voler des données en lui demandant une série de questions vraies ou fausses. Cela rend plus difficile, mais pas impossible, l'exploitation de la vulnérabilité d'injection SQL.

Exemples d'utilisation

Contournement d'un formulaire d'authentification

- Pour tester un champ spécifique dans une application, il suffit le plus souvent d'y insérer des caractères spéciaux du SQL, comme "'", """, etc. En général, si le champ est sensible à l'injection SQL, un message d'erreur va être affiché. Très souvent, ce message contient la requête qui a été exécutée, ce qui facilite le travail de l'attaquant en lui donnant des indices sur le schéma de la base.
- Il est également possible de tester tout un site, soit manuellement en regardant dans le code source de l'application si les accès à la base de données sont sécurisés, soit de l'extérieur en utilisant des applications comme **IBM Rational AppScan**.

- **Réalisation**

1. Créer un formulaire de connexion, un formulaire simple sans tenir compte de design
2. Nommer votre page en connexion.php

3. Tester votre travail en entrant des noms d’utilisateur et mots de passe existant dans la base de données. Notant que les mots de passe sont en clair et ce n’est "pas bien" mais c’est pour l’exemple, il est clair qu’il vaut toujours mieux les chiffrer.
4. Tester le travail en entrant des noms d’utilisateur et mots de passe qui n'existent pas dans la base de données.
5. Ré-entrer un mot de passe quelconque avec le **bon nom d’utilisateur**.
6. La requete sera:

   ```
   SELECT id, username FROM users WHERE username = 'admin' AND password = 'injectionsql'
   ```
7. Le même message sera affiché, et c'est le bug

- Transformer maintenant la requête initiale pour non plus lui faire dire "il faut le bon nom d’utilisateur et le bon mot de passe" mais simplement "il faut le bon nom d’utilisateur ".
- Comment faire cela ? C’est tout simple : placer toute la partie concernant le mot de passe en commentaire.
- **Résultat :** cette condition-là ne sera plus du tout prise en compte, seul le nom d’utilisateur le sera, et comme c’est le bon... ;-)
- Dans le cas de MySQL, le caractère pour commenter une ligne est #.

- La requête avec commentaire sera:

  ```
  SELECT id, username FROM users WHERE username = 'admin#' AND password = 'injectionsql'
  ```

  - Ce qui correspond en fait à ceci (tout ce qui sera après le # sera ignoré).

    ```
    SELECT id, username FROM users WHERE username = 'admin'
    ```

- Pour se sécuriser, il y a plusieurs façons de faire.On peut par exemple utiliser l'échappement des chaînes de caractères.

  ```
  mysqli_real_escape_string($db, $_GET['username']); $password = mysqli_real_escape_string($db, $_GET['password']); ?> span class="bleu">mysqli_real_escape_string
  ```

  - La fonction **mysqli_real_escape_string ()** échappe aux caractères spéciaux d'une chaîne pour les utiliser dans une instruction SQL.
  - Syntaxe:mysqli_real_escape_string(connection,escapestring);

- **Autres exemples**

apcpedagogie.com

- Situation normale

```
string.Format("SELECT * FROM Utilisateur WHERE NomUtilisateur='{0}' AND MotDePasse='{1}' AND EstAdmin=1;", nomUtilisateur, HasherMotDePasse(motDePasse));
```

```
SELECT * FROM Utilisateur WHERE NomUtilisateur='<nom>' AND MotDePasse='<hash>' AND EstAdmin=1;
```

- Avec la chaîne « ' OR 1=1 -- »

```
SELECT * FROM Utilisateur WHERE NomUtilisateur='' OR 1=1; -- AND MotDePasse='<hash>' AND EstAdmin=1;
```

- **Utilisation de la commande ORDER BY**

  - Une des difficultés pour accéder à une base de données est de définir le nombre de colonne dans une table. Pour cela on utilise la commande ORDER BY. Supposons que nous ayons une url sous la forme /id=4. On effectue une injection sous la forme:

    `/id=4 ORDER BY 1 /id=4 ORDER BY 3 /id=4 ORDER BY 4 /id=4 ORDER BY 5`
  - Etc ...Jusqu'à obtenir une erreur par exemple en 5. Il y a donc 4 colonnes.Il existe d'autres méthodes pour obtenir ces informations.

Source:

- http://www.thehackademy.net/article.phpstory_id=95&section_id=57
- http://www.spidynamics.com/papers/SQLInjectionWhitePaper.pdf
- http://www.nextgenss.com/papers/advanced_sql_injection.pdf
- http://www.phpfrance.com/

Sommaire du cours Administration et sécurité des sites web

Source : https://apcpedagogie.com/les-injections-sql-classiques · téléchargé le 04/11/2026 · © APC Pédagogie · apcpedagogie.com