INJECTION SQL

L'injection SQL, ou SQLi, est un type d'attaque sur une application web qui permet à un attaquant d'insérer des instructions SQL malveillantes dans l'application web, pouvant potentiellement accéder à des données sensibles dans la base de données ou détruire ces données.l'injection SQL a été découverte pour la première fois par Jeff Forristal en 1998.

Au cours des deux décennies qui ont suivi sa découverte, l'injection SQL a toujours été la principale priorité de développeurs web lors de la conception d'applications.

Barclaycard a estimé en 2012 que 97 % des violations de données commencent par une attaque par injection SQL. L'injection SQL est encore très répandue aujourd'hui et la gravité des attaques par injection dans une application Web est largement reconnue. C'est l'un des top 10 des risques de sécurité des applications Web les plus critiques selon l'OWASP.

De manière générale, il est difficile de déterminer si une chaîne utilisateur est malveillante ou non. Par conséquent, la meilleure façon de procéder est d'échapper les caractères spéciaux dans la saisie de l'utilisateur.

Ce processus vous évite une attaque par injection SQL. Vous pouvez échapper une chaîne de caractères avant de construire la requête dans PHP en utilisant la fonction `mysql_escape_string()`. Vous pouvez aussi échapper une chaîne de caractères dans MySQL en utilisant la fonction `mysqli_real_escape_string()`.

Lors de l'affichage de la sortie en HTML, vous devrez également convertir la chaîne de caractères pour vous assurer que les caractères spéciaux n'interfèrent pas avec le balisage HTML. Vous pouvez convertir les caractères spéciaux en PHP en utilisant la fonction `htmlspecialchars()`.

Vous pouvez également utiliser des déclarations préparées pour éviter les injections SQL. Une instruction préparée est un modèle de requête SQL, dans lequel vous spécifiez des paramètres à un stade ultérieur pour l'exécuter. Voici un exemple d'une déclaration préparée en PHP et MySQLi.

```
$query = $mysql_connection->prepare("select * from user_table where username = ? and password = ?");
$query->execute(array($username, $password));
```

L’étape suivante pour atténuer cette vulnérabilité est de limiter l’accès à la base de données au strict nécessaire.

Par exemple, connectez votre application Web au DBMS en utilisant un utilisateur spécifique qui n’a accès qu’à la base de données pertinente.

Restreindre l’accès de la base de données utilisateur à tous les autres emplacements du serveur. Vous pouvez également bloquer certains mots-clés SQL dans votre URL via votre serveur Web. Si vous utilisez Apache en tant que serveur web, vous pouvez utiliser les lignes de code suivantes dans votre .htaccess pour afficher une erreur 403 Forbidden à un attaquant potentiel.

Vous devez être prudent avant d’utiliser cette technique car Apache affichera une erreur à un lecteur si l’URL contient ces mots-clés.