Sécurité des applications : Vulnérabilités courantes (comme l'injection SQL, cross-site scripting, etc.)

Introduction

Salut à tous ! Lorsque nous naviguons sur le web ou utilisons une application, nous supposons souvent qu'elle est sécurisée. Mais, malheureusement, ce n'est pas toujours le cas. Aujourd'hui, nous allons explorer certaines des vulnérabilités courantes que l'on peut rencontrer dans le monde des applications, comprendre comment elles fonctionnent et comment les éviter.

1. Injection SQL

Qu'est-ce que c'est ?

Une injection SQL se produit lorsqu'un attaquant peut "injecter" ou insérer des commandes SQL malveillantes via une entrée utilisateur, affectant ainsi la base de données d'une application.

Exemple concret :

Supposons un formulaire de connexion où vous entrez votre nom d'utilisateur et votre mot de passe. Un attaquant pourrait entrer quelque chose comme ' OR '1'='1';-- pour se connecter sans connaître un mot de passe valide.

Comment s'en protéger ?

- Utilisez des requêtes préparées ou des "placeholders" pour s'assurer que les données entrées par l'utilisateur sont traitées comme de simples données et non comme du code SQL.
- Évitez de construire des requêtes SQL par concaténation de chaînes.

2. Cross-Site Scripting (XSS)

Qu'est-ce que c'est ?

Le XSS permet à un attaquant d'injecter du code JavaScript malveillant dans une page web, qui est ensuite exécuté dans le navigateur d'un autre utilisateur.

Exemple concret :

Un forum où vous pouvez poster des commentaires. Si le site n'est pas sécurisé, un attaquant pourrait poster un commentaire contenant du JavaScript, affectant ainsi tous les utilisateurs qui voient ce commentaire.

Comment s'en protéger ?

- Validez, filtrez et échappez à toutes les entrées et sorties.
- Utilisez des en-têtes de sécurité HTTP comme Content Security Policy (CSP).

3. Cross-Site Request Forgery (CSRF)

Qu'est-ce que c'est ?

Le CSRF trompe un utilisateur pour qu'il effectue une action non intentionnelle sur un site où il est actuellement authentifié.

Exemple concret :

Imaginons que vous soyez connecté à votre banque en ligne. Si vous visitez un autre site malveillant, il pourrait déclencher une demande pour transférer de l'argent sans que vous le sachiez.

Comment s'en protéger ?

- Utilisez un token CSRF unique pour chaque session et chaque action.
- Assurez-vous que les actions sensibles utilisent la méthode POST plutôt que GET.

4. Inclusion de fichiers distants

Qu'est-ce que c'est ?

Cela se produit lorsque un attaquant peut faire inclure un fichier distant, souvent malveillant, dans une application web.

Exemple concret :

Un site qui permet à l'utilisateur de choisir un modèle de page via une URL. Un attaquant pourrait potentiellement changer l'URL pour pointer vers un fichier malveillant.

Comment s'en protéger ?

- Évitez d'inclure des fichiers basés sur les entrées des utilisateurs.
- Limitez les permissions pour éviter l'exécution de code malveillant.

Je retiens

- Les vulnérabilités courantes sont des failles que les attaquants exploitent pour compromettre les applications.
- Injection SQL : évitez de construire des requêtes SQL par concaténation de chaînes.
- XSS : filtrez et échappez à toutes les entrées et sorties.
- CSRF : utilisez un token CSRF unique pour chaque session.