Liste des vérifications OWASP

L'OWASP Top 10 est la liste des vulnérabilités de sécurité des applications web les plus couramment rencontrées dans les applications et les API existantes. Les risques sont classés de A1 à A10, A1 étant le risque le plus fréquent. Voici le top 10 des vulnérabilités actuelles :

1. Broken Object Level Authorization (Injection) (violation d'autorisation au niveau de l'objet).
2. Broken Authentication (violation d'authentification).
3. Excessive Data Exposure (exposition excessive des données).
4. Lack of Resources & Rate Limiting (manque de ressources et limitation du débit).
5. Broken Function Level Authorization (violation d'autorisation au niveau de la fonction).
6. Mass Assignment (attribution en masse).
7. Security Misconfiguration (mauvaise configuration de la sécurité).
8. Injection.
9. Improper Assets Management (mauvaise gestion des ressources).
10. Insufficient Logging & Monitoring (connexion et surveillance insuffisantes).

Toutes ces vulnérabilités peuvent être encore regroupées selon le processus **AAA** *(Authentication, Authorization, Accounting)* (authentification, autorisation, traçabilité) des protocoles réseau.

Nous allons donc traiter ces 3 A en mettant en œuvre ces bibliothèques ou méthodes dans notre application :

1. **Authentification :** utilisez JWT (JSON Web Token) pour le back-end d'authentification du framework Django REST. Cela vise à couvrir les cas d'utilisation de JWT les plus répandus, en offrant par défaut un jeu de fonctionnalités prudent. Pour cela, vous pouvez utiliser cette ressource.
2. **Autorisation :** la deuxième étape est l'autorisation, dans laquelle le back-end décide si *l'utilisateur authentifié* est autorisé à accéder à une ressource. Dans notre application, un utilisateur ne doit pas être autorisé à accéder à un projet pour lequel il n'est pas ajouté en tant que contributeur. De même, un contributeur ou un utilisateur doit toujours être connecté pour accéder à une fonctionnalité.
    1. Seuls les utilisateurs authentifiés doivent être en mesure d'accéder à quoi que ce soit dans l'application. Utilisez Django REST APIView. Vous devez utiliser la classe d'autorisation pour vérifier que l'utilisateur est authentifié. Ressource : https://django-rest-framework.org/api-guide/permissions/

    2. Vous aurez besoin d'ajouter des autorisations à vos modèles en spécifiant l'auteur (clé étrangère de l'utilisateur) de ce projet/problème/commentaire.
3. **Accès :** même après que l'utilisateur a été authentifié et autorisé à accéder à une ressource, la ressource elle-même peut nécessiter des autorisations spéciales, par exemple pour actualiser ou supprimer un commentaire.
    1. Les commentaires doivent être visibles par tous les contributeurs au projet et par le responsable du projet, mais seul leur auteur peut les actualiser ou les supprimer.
    2. Un problème ne peut être actualisé ou supprimé que par son auteur, mais il doit rester visible par tous les contributeurs au projet.
    3. Il est interdit à tout utilisateur autre que l'auteur de demander une mise à jour et de supprimer des demandes sur une question/un projet/un commentaire.