

LES ESSENTIELS

SE PROTÉGER DES FAILLES DE TYPE IDOR

L'*Insecure Direct Object Reference* (**IDOR**) est une vulnérabilité due à un manque de vérification de l'autorisation d'accès d'un utilisateur à une ressource côté serveur. Elle est répandue sur les applications web, mais également sur toutes les applications reposant sur des [API](#) (web, mobile, native, IA, etc.). Elle est déclinée en sous types selon qu'elle concerne le droit d'accès à une fonction ([BFLA](#)), à un objet ([BOLA](#)), ou à une propriété ([BOPLA](#)).

Ces failles facilement exploitables occupent la première place du [Top 10 OWASP 2025](#) sur les risques les plus critiques pour la sécurité des applications web. Elles peuvent devenir dévastatrices en permettant à des attaquants de récupérer une quantité massive d'informations parfois sensibles (voir [CVE-2025-53640](#)), ou d'agir à la place d'un autre utilisateur (voir [CVE-2023-3287](#)).

Retrouvez dans ce document les bonnes pratiques pour vous en protéger et les détecter lors du cycle de développement ou en production.

1/ SE PROTÉGER DÈS LA CONCEPTION D'UN SERVICE

- **Sensibiliser les développeurs** au besoin de vérifier systématiquement l'autorisation de l'utilisateur courant lorsqu'il accède à une ressource ou réalise une action. Que l'utilisateur soit authentifié n'est jamais suffisant.
- **Contrôler le droit d'accès systématiquement du côté serveur.** Un contrôle côté client est insuffisant.
- **Simplifier l'implémentation de la vérification des droits d'accès** en utilisant un cadre de référence d'autorisation connu, et former l'équipe de développement à l'utiliser correctement.

- **Mettre en œuvre de la revue de code par les pairs** avec une liste de vérifications incluant le contrôle des droits d'accès.
- **Inclure des tests de cas non passant d'autorisation dans la stratégie de test.** Ces tests viennent en complément des tests de cas passant effectués avec l'utilisateur A, et permettent de vérifier que l'utilisateur B est bien refusé. Ils doivent être réalisés au niveau de tous les points de terminaison HTTP (API ou contrôleur MVC).

2/ RECHERCHER DES FAILLES IDOR AU COURS DU CYCLE DE VIE

- **Automatiser les tests dans la chaîne d'intégration et de déploiement continus ([CI/CD](#))**, grâce à l'utilisation de plusieurs outils d'analyse de différents types :
 - > au plus tôt avec les outils d'analyse statique de code (SAST) ;
 - > après le déploiement en qualification avec les outils d'analyse dynamique (DAST) ;
 - > certains éditeurs de SAST / DAST proposent des outils d'analyse IA qui sont plus pertinents pour suivre la logique métier et tester les IDOR en réduisant le taux de faux positifs.
- **Effectuer des tests de pénétration et des primes aux bogues (*bug bounty*) périodiques de l'application.** Les IDOR étant faciles à trouver et à exploiter, ces méthodes sont particulièrement efficaces pour les détecter.

3/ SUPERVISER EN PRODUCTION

→ **Mettre en œuvre des règles de détection spécifiques pour détecter l'exploitation de possibles vulnérabilités IDOR :**

- > détection d'un pic d'utilisation d'un point de terminaison HTTP dans un laps de temps limité avec un seuil en nombre de requêtes défini en fonction du volume courant connu (à mettre à jour régulièrement) ;
- > vérifier si une bonne partie du volume du pic est effectué depuis la même adresse IP et/ou par le même compte authentifié, puis déterminer si cela ressemble à une énumération séquentielle d'identifiants.

→ **Alerter et réagir en cas d'attaque** en bloquant l'adresse IP et/ou le compte d'un utilisateur malveillant ou détourné par un acteur malveillant, puis **corriger l'IDOR rapidement**.