Analyse d'Architecture de Sécurité - Client Web

Équipe de Sécurité Front-End

3 novembre 2025

1 Introduction et Périmètre

Ce document présente l'analyse de sécurité pour l'architecture client web (Front-End) du projet `demperm/src/client/web/`. L'objectif est d'identifier les composants critiques, les principaux risques et les exigences de sécurité que l'équipe Client Web doit gérer ou exiger du Back-End.

— **Périmètre :** Code Client (React/TypeScript), Configuration de Build, Interactions API.
— **Focus :** Vulnérabilités Client-Side (*XSS, Clickjacking, IDOR vectorisé par le client*).

2 Composants Critiques et Responsabilités de l'Équipe Client

L'architecture expose des points d'interaction majeurs avec des implications de sécurité directes pour l'équipe Front-End.

2.1 Secrets et Configuration (`.env` et Fichiers de Build)

Responsabilité Client : S'assurer qu'aucun secret (clés API privées, chaînes de connexion) n'est exposé dans le bundle final de l'application.

— **Contrôle :** Le fichier `.gitignore` doit exclure `.env`. Les outils de build (`vite.config.ts / next.config.mjs`) doivent limiter l'injection de variables d'environnement au strict minimum `PUBLIC_`.

2.2 Gestion des Entrées/Sorties (`src/shared/ui/` et `src/app/`)

Responsabilité Client : Protéger le navigateur de l'utilisateur contre le code malveillant.

— **Entrées (Validation) :** Valider la forme et le type des données utilisateur *avant* l'envoi aux API. (Première ligne de défense contre l'Injection).
— **Sorties (Encodage/Nettoyage) : Encodage par défaut** de toutes les données générées par l'utilisateur (*User Generated Content*) pour prévenir le **Cross-Site Scripting (XSS)**.

3 Analyse des Menaces Majeures (Client-Side)

3.1 1. Cross-Site Scripting (XSS)

Menace : L'injection de scripts malveillants exécutés dans le navigateur de la victime (vol de cookies, usurpation de session).

— **Zone à risque :** Affichage des données sociales (`social/`) et tout champ de formulaire.
— **Exigence Fondamentale :** Implémenter une **Content Security Policy (CSP)** stricte pour le Front-End. (*Exigence à l'équipe Back-End/Infra pour l'en-tête HTTP*).

3.2 2. Falsification de Requête Inter-Sites (CSRF)

Menace : Forcer un utilisateur authentifié à exécuter une action indésirable (ex : changer son mot de passe) sans son consentement.

- **Zone à risque :** Requêtes `POST` ou `PUT` dans `domains/*/services/`.
- **Exigence Fondamentale :** Utiliser des **jetons Anti-CSRF** (le client doit les obtenir et les inclure) et s'assurer que les cookies de session ont l'attribut `SameSite=Strict` ou `Lax`.

3.3 3. Référence d'Objet Directe Non Sécurisée (IDOR)

Menace : Le client manipule un ID (`[id].tsx`) pour accéder à une ressource qu'il n'est pas autorisé à voir (ex : la boîte mail d'un autre utilisateur).

- **Zone à risque :** Routage dynamique (`social/users/[id].tsx`, `social/mailbox/[id].tsx`).
- **Exigence Fondamentale :** L'équipe Client doit **uniquement** utiliser les IDs. L'équipe Back-End doit **systématiquement** vérifier les droits de l'utilisateur pour l'ID demandé. (*Le Front-End ne doit jamais implémenter le contrôle d'accès lui-même*).

4 Recommandations et Contrôles Techniques

Contrôle	Détails d'Implémentation Front-End	Responsable
Cookies de Session	Exiger que les jetons d'authentification soient délivrés via des **cookies HTTP-Only** (*Empêche l'accès JavaScript et le vol via XSS*).	Back-End (Exigence Client)
Composants Tiers	Utiliser des outils pour auditer `package.json` pour les vulnérabilités dans les dépendances (NPM, etc.). (*Éviter les attaques sur la chaîne logistique*).	Client Web
Défenses Navigateur	Exiger la mise en place d'en-têtes de sécurité (CSP, `X-Frame-Options`) au niveau de l'infrastructure pour protéger l'interface client.	Infra/Back-End (Exigence Client)
Règles d'Autorisation	Mettre en place la logique d'affichage/masquage dans `domains/*/guards/`, mais sans jamais envoyer de données sensibles si le contrôle serveur échoue.	Client Web