Étude de Sécurité – Architecture FastAPI / React / React Native / PostgreSQL

1. Introduction

Cette étude fournit une analyse de sécurité complète pour une architecture comprenant un backend FastAPI, un frontend React, une application mobile React Native et une base de données
PostgreSQL.

2. Analyse des risques et menaces

- FastAPI : injections, mauvaise validation, config CORS trop permissive, autorisations faibles.
- React : XSS, mauvaise gestion du DOM, stockage local non sécurisé.
- React Native : exposition de secrets, MITM si TLS mal configuré.
- PostgreSQL : SQL injection, privilèges excessifs, absence de chiffrement au repos.

3. Vulnérabilités potentielles

- FastAPI : endpoints non protégés, validation Pydantic insuffisante.
- React : gestion incorrecte des entrées utilisateurs.
- React Native : stockage de tokens dans AsyncStorage.
- PostgreSQL : pas de rotation des identifiants, pas de séparation des rôles.

4. Recommandations

Backend FastAPI :
- Activer rate limiting, CORS restrictif, OAuth2/JWT robuste, headers de sécurité.

PostgreSQL :
- Utiliser des rôles séparés, chiffrement TLS, requêtes préparées, audit logging.

Frontend :
- Interdire dangerouslySetInnerHTML, sécuriser cookies, utiliser HTTPS forcé.

Mobile :
- Stockage sécurisé (SecureStore), certificate pinning, pas de secrets embarqués.

Infrastructure :
- Reverse proxy (Nginx/Traefik), WAF, monitoring centralisé, rotation des secrets.

5. Plan d’action priorisé

1. Sécurisation Auth & tokens
2. Chiffrement TLS complet
3. Revue des permissions PostgreSQL
4. Mise en place d’outils CI/SAST/DAST
5. Hardening du frontend et mobile

6. Outils recommandés

- SAST : SonarQube, Bandit, Semgrep
- DAST : OWASP ZAP
- Dépendances : Dependabot, Safety
- Observabilité : Grafana, Loki, ELK

7. Tableau récapitulatif

Menace	Contre-mesure
Injection	Validation stricte, requêtes préparées
XSS	Sanitation, CSP
MITM	TLS + pinning
Fuite de données	Chiffrement & RBAC
Secrets exposés	Vault + rotation