VEILLE TECHNOLOGIQUE

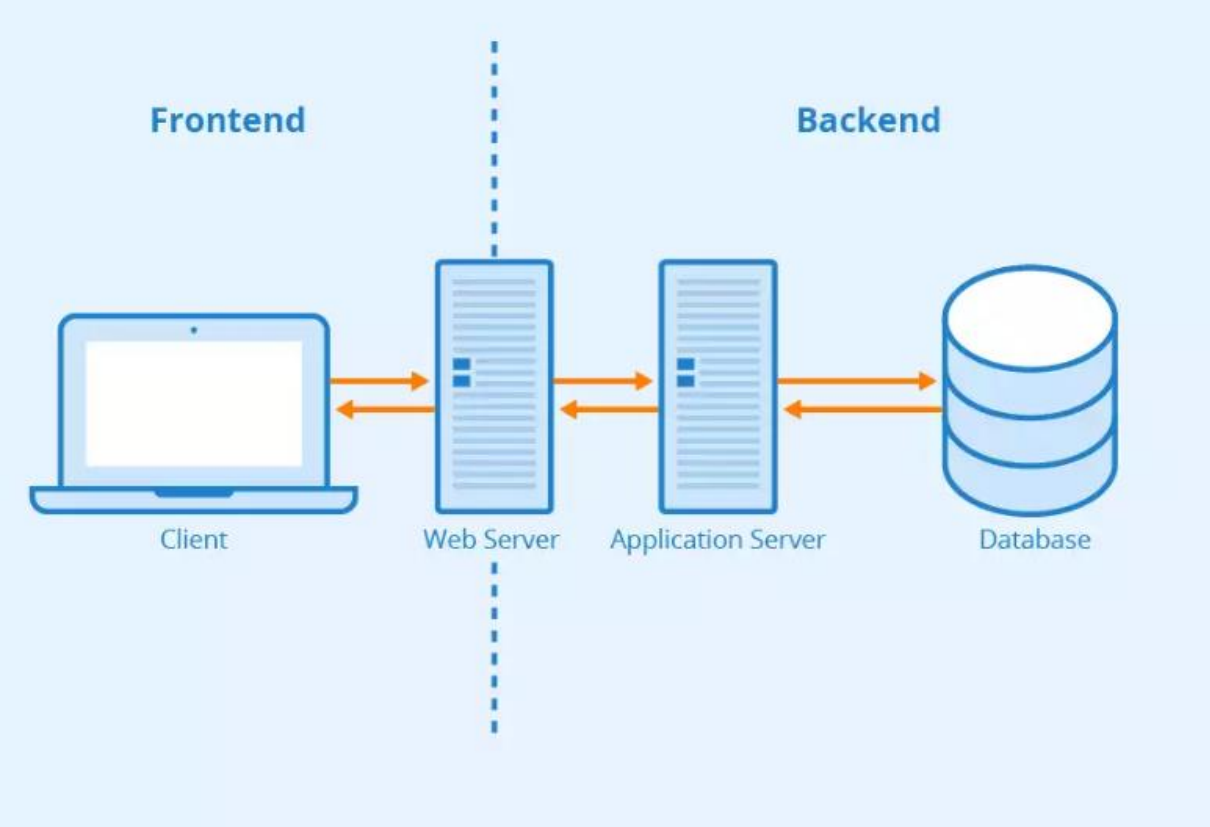
PROTECTION DES DONNÉES D'UN SITE WEB

Par Ilia Choumitzky – SIO SLAM

Sommaire

QU'EST-CE QU'UNE APPLICATION WEB ?

Une application web est une extension dynamique d'un serveur web. Une application web est formée d'un ensemble de composants web, de ressources statiques
de bibliothèques et de classes. Les composants web fournissent cette capacité d'extension.



Architecture

Pour l'architecture d'un site web nous avons plusieurs couches, Couche de présentation, il s'agit une partie de l'application responsable de la présentation, et la communication avec le client, souvent cette interface et représenter d'un code HTML/CSS pour le côté design. Pour la Couche accès aux données, le client ne le vois pas car généralement ce sont des requêtes entre le client et le serveur comme par exemple un formulaire. La couche métier est responsable de la gestion des données, permet d'accéder aux données.

LES PRINCIPALES FAILLES DE SÉCURUTÉS

Le plus connu des failles c’est l’injection sql le pirate utilise un morceau de code sql qui permet de manipuler une base de données et accéder à des informations potentiellement importantes. C'est l'un des types d'attaques les plus répandus et menaçants, car il peut potentiellement être utilisé pour nuire à n'importe quelle application Web.

La violation de gestion d’authentification et de session, permet à un cybercriminel de capturer ou contourner les méthodes d’authentification utilisées par une application web. Les attaques de ce type sont dues à des défauts de conception lors de la création de l’application.

Falsification de requête : Une attaque de type CSRF force le navigateur d'une victime ayant une session ouverte sur une application web vulnérable à effectuer une requête sur cette dernière application. Cela implique que le navigateur de la victime va effectuer une action sur un site vulnérable à la place de la victime.

XSS : ça permet d’injecter du contenu dans une page, provoquant ainsi des actions sur les navigateurs web visitant la page. Les possibilités des XSS sont très larges puisque l’attaquant peut utiliser tous les langages pris en charge par le navigateur.

Un exemple d'injection SQL :

Lors de l'éxécution du script login.php le code génère la requête suivante via une simple concaténation.

```
$query = "SELECT * FROM users WHERE email ='".$_POST["email"]."' AND password
='".$_POST["password"]."'";
```

Soit la requête suivante :

```
SELECT * FROM users WHERE email ='toto123@hotmail.fr' AND password
='superpassword';
```

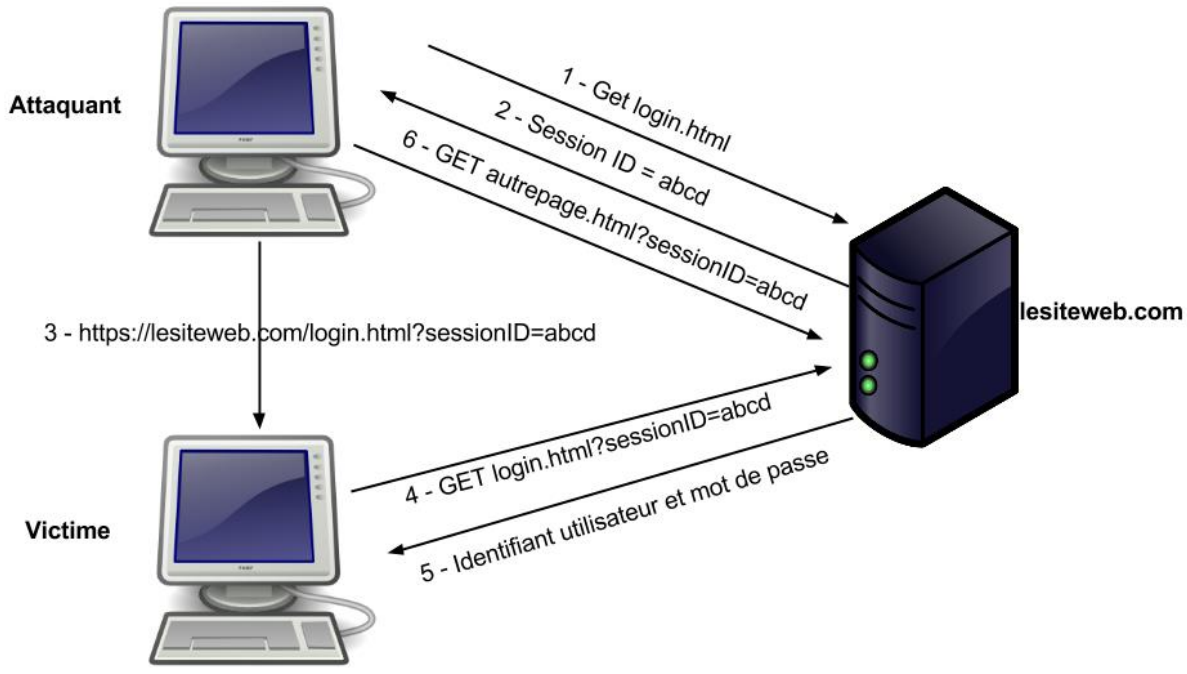
Maintenant je veux forcer les paramètres dans le formulaire :

```
email : " toto123@hotmail.fr '#",
password : "
```

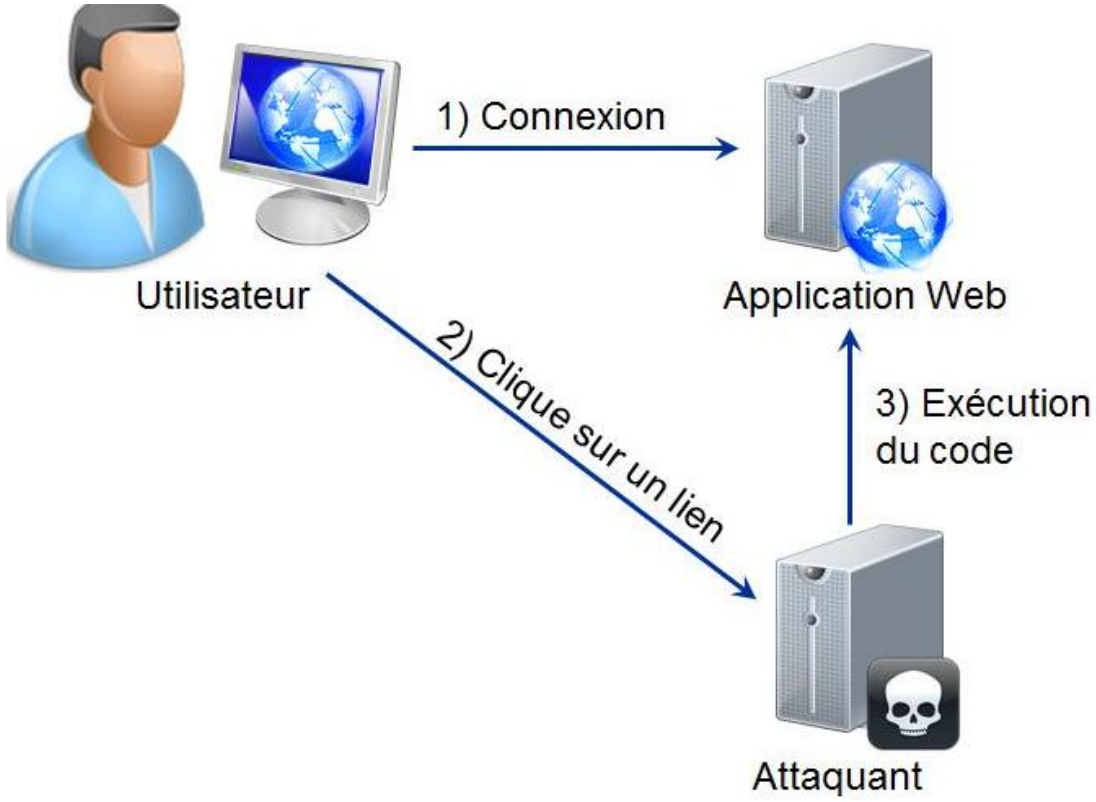
Le serveur va lire de cette manière sans tenir compte de mot de passe :

```
SELECT * FROM users WHERE email ='toto123@hotmail.fr '
```

Violation de gestion d'authentification et de session :



Falsification de requête :



COMMENT SE PROTÉGER DE CE TYPE D'ATTAQUE ?

Empêcher l'injection SQL dans PHP :

Si on reprend l'exemple :

```
"SELECT id FROM users WHERE name = 'Admin' AND password = '".$_POST["password"]."'"
```

On utilise ici ce que l'utilisateur envoie directement dans la requête

Donc la première chose à faire est d'éviter les caractères spéciaux à l'aide de mysqli_real_escape_string() :

```
"SELECT id FROM users WHERE name = 'Admin' AND password =
'".mysqli_real_escape_string($_POST["password"])."'"
```

Violation de gestion d'authentification et de session :

Ça demande une planification rigoureuse et de se plier à de nombreuses règles, dont parmi les plus importantes :

- Utiliser uniquement des mécanismes de gestion de sessions intégrés. N'utiliser sous aucun prétexte des mécanismes secondaires ou faits maison
- Assurez-vous que toutes les pages disposent d'un lien de déconnexion. La déconnexion doit totalement détruire tout état de session côté serveur et cookies côté client. Considérez aussi le facteur humain : ne demandez pas confirmation car les utilisateurs fermeront juste l'onglet ou la fenêtre plutôt que se déconnecter proprement
- Utiliser une période de timeout qui déconnecte automatiquement tout utilisateur après une période d'inactivité
- Vérifier l'ancien mot de passe lorsque l'utilisateur change pour un nouveau mot de passe
- Soyez prudent lors de l'envoi d'informations confidentielles à une adresse email en tant que mécanisme de réinitialisation de mot de passe (cf. RSNAKE01). Utilisez uniquement des nombres aléatoires à durée limitée pour réinitialiser l'accès et envoyer un e-mail de confirmation dès que le mot de passe a été réinitialisé.

Protection contre la Falsification requête :

- S'assurer qu'il n'y a pas de vulnérabilité de type XSS dans l'application

- Pour des données sensibles ou des transactions financières, ré-authentifiez ou utilisez la signature de transaction afin de s'assurer que la requête est authentique. Mettez en place des mécanismes externes tels le courrier électronique ou le contact téléphonique afin de vérifier les demandes ou en notifier l'utilisateur.

- N'utilisez pas de requêtes GET (URLs) pour les données sensibles ou pour effectuer des transactions de valeur. Utiliser uniquement la méthode POST lorsque vous recevez des données de l'utilisateur. En revanche, l'URL peut contenir le token aléatoire car ceci crée une URL unique, ce qui rend une attaque CSRF presque impossible à exécuter.

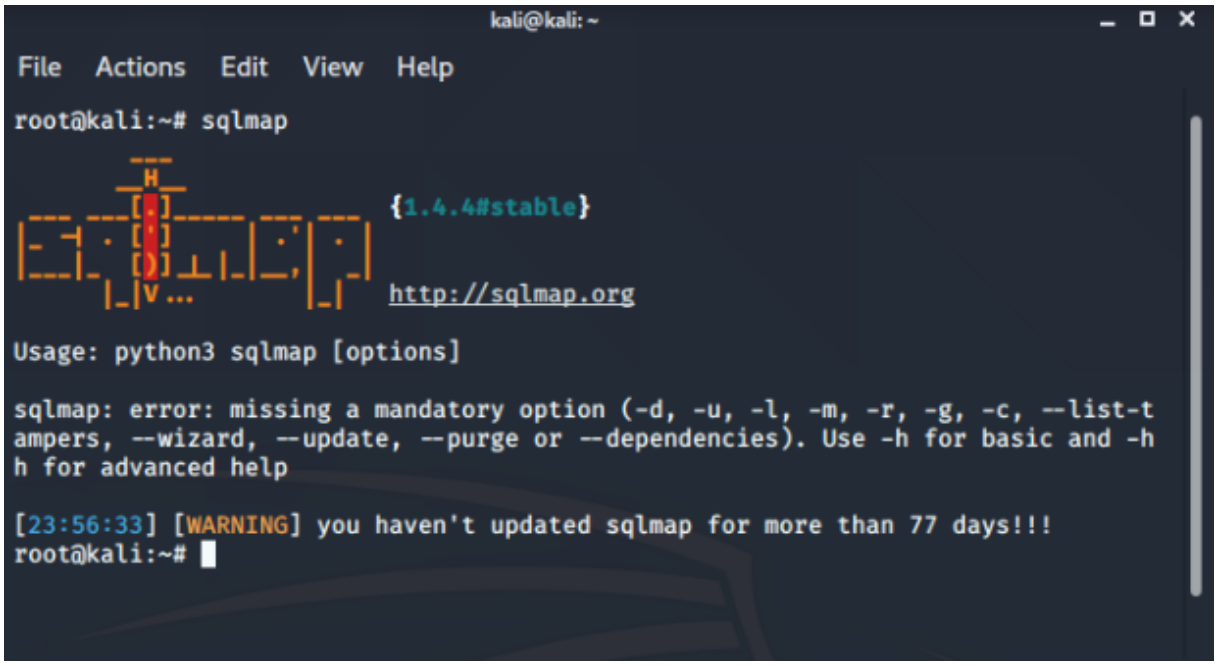
  GET : Les données sont envoyés au serveur sont écrites directement dans l'URL comme ceci :

  www.example.com/register.php?firstname=peter&name=miller&age=55&gender=male

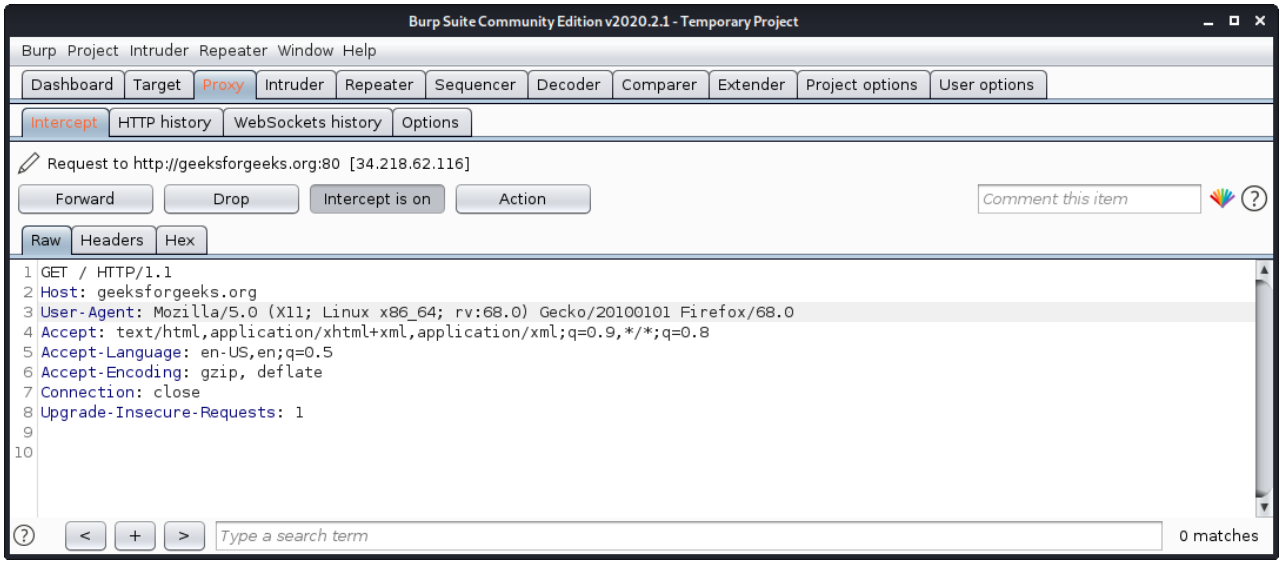
  POST : La méthode POST écrit les paramètres URL dans la requête HTTP pour le serveur. Les paramètres ne sont donc pas visibles pour les utilisateurs et la portée des requêtes POST est illimitée

QUELQUES OUTILS POUR TESTER LES VULNÉRABILITÉS D'UN SITE :

SQLMap : SQLMap est un outil open source utilisé pour automatiser le processus d'injection SQL. Il détecte et exploite lui-même les paramètres d'injection SQL



Suite Burp : Burp Suite et un logiciel de test de sécurité pour les applications Web. Il teste des vulnérabilités telles que XSS ou SQLi ou même toute vulnérabilité liée au Web.



CONCLUSION

Bien que de nouvelles failles émergent avec le Web, les applications Web sont toujours vulnérables aux failles les plus anciennes. Cela démontre que le comportement des développeurs n'a pas changé. Pressés par des contraintes de temps, ils ne font pas l'effort d'appliquer quelques règles de bases qui permettent de se défendre contre les attaques les plus dangereuses.

Sources

https://www.oracle.com/fr/security/injection-sql-attaque.html

https://www.websecurity.digicert.com/fr/fr/security-topics/what-is-ssl-tls-https#:~:text=SSL%20signifie%20couche%20des%20sockets,donn%C3%A9es%20envoy%C3%A9es%20sur%20l'Internet

https://www.isdecisions.fr/problemes-securite-authentification-unique-sso/

http://www.web-france.info/pages/falsification-de-requete-intersites-csrf.html

https://support.ptc.com/help/thingworx/platform/r9/fr/index.html#page/ThingWorx/Help/REST_API/UpdatingtheRequestMethodandContentTypeFilteringforCSRFProtection.html

https://fr.acervolima.com/comment-empecher-l-injection-sql-en-php/

https://www.delftstack.com/fr/howto/php/prevent-sql-injection-php/

http://www.web-france.info/pages/violation-de-gestion-d-authentification-et-de-session.html

https://docs.microsoft.com/fr-fr/microsoft-365/security/defender-endpoint/prevent-changes-to-security-settings-with-tamper-protection?view=o365-worldwide

https://www.ibm.com/docs/fr/order-management?topic=ssgtjf-com-ibm-help-sfs-cpqsolution-doc-customization-c-wcc-crosssiterequestforgery-html