



Sécurité web

3IW ESGI





Louis HARANG

Ancien IW à l'ESGI

Développeur Fullstack chez Wizards Technologies
Prof de Laravel, Vuejs, Sécurité Web

Pour me contacter par mail :

cours@narah.io

(avec en objet dans le mail [ESGI])





Présentez vous !

Ton prénom

Tu travailles dans quelle entreprise ?

Qu'est ce que tu connais sur la sécurité web ?

Qu'est ce que tu attends de ce cours ?



Sommaire

- **Partie I : Théorie**
Sécurité web, HTTPS, SSL, SSH, l'OWASP, RGPD, les outils de sécurité etc
- **Partie II : Application**
Les failles les plus courantes, comment les reconnaître et les patcher.
- **Partie III : A vous de jouer**
Projet de groupe





Notation :

Contrôle continu : QCM de ~30 min à mi-parcours pour valider les acquis

Partiel : Projet de groupe avec soutenance vendredi





Pour le bon déroulement du cours :

- Mettez la caméra quand vous parlez
- Coupez le micro quand vous ne parlez pas
- Téléchargez docker ou LAMP/MAMP/WAMP qui fonctionne avec php 7.4
- Ne pas hésitez à intervenir à la moindre question ou à la moindre difficulté





Avant propos comment je me protège en tant qu'utilisateur ?

- Toujours utiliser un password manager : [Bitwarden](#), [Lastpass](#) etc
- Utiliser un mot de passe complexe (+20 caractères, maj, min, caractères spéciaux etc)
- Utiliser un mot de passe par site / application
- Activer le 2fa dès que possible
- Toujours verrouiller son pc quand on ne l'utilise pas



I. La théorie

Sécurité web, HTTPS, SSL, SSH, l'OWASP





Sécurité informatique ?

“La sécurité des systèmes d'information (SSI) ou plus simplement sécurité informatique, est l'ensemble des moyens techniques, organisationnels, juridiques et humains nécessaires à la mise en place de moyens visant à empêcher l'utilisation non autorisée, le mauvais usage, la modification ou le détournement du système d'information. Assurer la sécurité du système d'information est une activité du management du système d'information.”





Et du coup c'est quoi la sécurité web ?

La sécurité web est une branche de la sécurité informatique spécifiquement liée au web, impliquant la sécurité d'une application web côté client et côté serveur.

Dans ce cours nous parlerons donc de tout l'aspect sécurité d'un point de vue d'un concepteur d'application web.





Pourquoi se protéger ?

Riques :

- Vols de données
- Altérations des données
- Application non disponible
- Application non résiliente
- Vol de propriété intellectuelle

Impacts :

- Perte d'image
- Perte de crédibilité
- Perte de confiance
- Responsabilité civile et pénale
- Perte financière





Et la performance dans tout ça ?

La performance d'une application web est liée à la sécurité de cette application. Si votre application n'est pas performante et résiliente face à une attaque, les conséquences seront identiques à une faille de sécurité (crash, fuite de donnée etc).

Une application résiliente et performante = une application + sécurisée

Nous verrons donc pendant le cours des notions liées performance web comme le **caching**, le **throttling** ou encore le **rate limiting**.





Ne JAMAIS faire confiance à l'utilisateur





HTTP & HTTPS

HTTP : L'Hypertext Transfer Protocol (HTTP, littéralement « protocole de transfert hypertexte ») est un protocole de communication client-serveur développé pour le World Wide Web.

HTTPS : L'Hypertext Transfer Protocol Secure (HTTPS, littéralement « protocole de transfert hypertexte sécurisé ») est la combinaison du HTTP avec une couche de chiffrement comme SSL ou TLS.

- permet au visiteur de vérifier l'identité du site web, via un certificat d'authentification émis par une autorité tierce, réputée fiable.
- Il garantit **théoriquement** la confidentialité et l'intégrité des données envoyées par l'utilisateur.
- Initialement utilisé pour les transactions financières en ligne : commerce électronique, banque en ligne, courtage en ligne, etc. Il est aussi utilisé pour la consultation de données privées, comme les courriers électroniques, par exemple.

A savoir : Le SSL est un critère SEO pour Google





HTTP

Rappels sur HTTP

[HTTP a plusieurs verbes \(HTTP request methods\)](#), les plus courants sont :

- **GET** : pour récupérer une ressource
- **POST** : pour créer une ressource
- **PATCH** : pour mettre à jour partiellement une ressource
- **PUT** : pour mettre à jour totalement (remplacer) une ressource
- **DELETE** : pour détruire une ressource





HTTP

Rappels sur HTTP

HTTP possède aussi de nombreux code :

- **200:** OK
- **201:** CREATED
- **204:** EMPTY
- **301:** MOVED PERMANENTLY
- **302:** FOUND
- **400:** BAD REQUEST
- **401:** UNAUTHORIZED
- **403:** FORBIDDEN
- **404:** NOT FOUND
- **405:** METHOD NOT ALLOWED
- **500:** INTERNAL ERROR
- **502:** BAD GATEWAY
- **503:** SERVICE UNAVAILABLE






Certificat SSL


- Signé par un tiers de confiance qui assure le lien entre l'entité numérique (le site) et l'entité physique (la société)
- Active le chiffrement des données au travers du protocole HTTPS
- 2 méthodes de chiffrement : clés asymétriques et clés symétriques

< Connection Security for fr.wikipedia.org

 You are securely connected to this site.

Verified by: Let's Encrypt

< Connection Security for www.google.com

 You are securely connected to this site.

Verified by: Google Trust Services





Ok mais je fais comment pour avoir un site en HTTPS ?

Certains hébergeurs vous permettent de créer gratuitement un certificat SSL pour votre site internet. Des services plus simples comme [Let's Encrypt](#) existent et ils sont 100% gratuits.

- [Protocole ACME](https://ietf-wg-acme.github.io/acme/) (<https://ietf-wg-acme.github.io/acme/>)
- Installe un agent de gestion de certificat
- Système de vérification de domaine (à travers le DNS ou la création de fichier à la racine de votre application web)
- Installation encore plus simple grâce au [Certbot](#)

Limites :

- Limites d'émission de certificat ([Voir ici](#))
- Pas de Organization Validation ou de Extended Validation





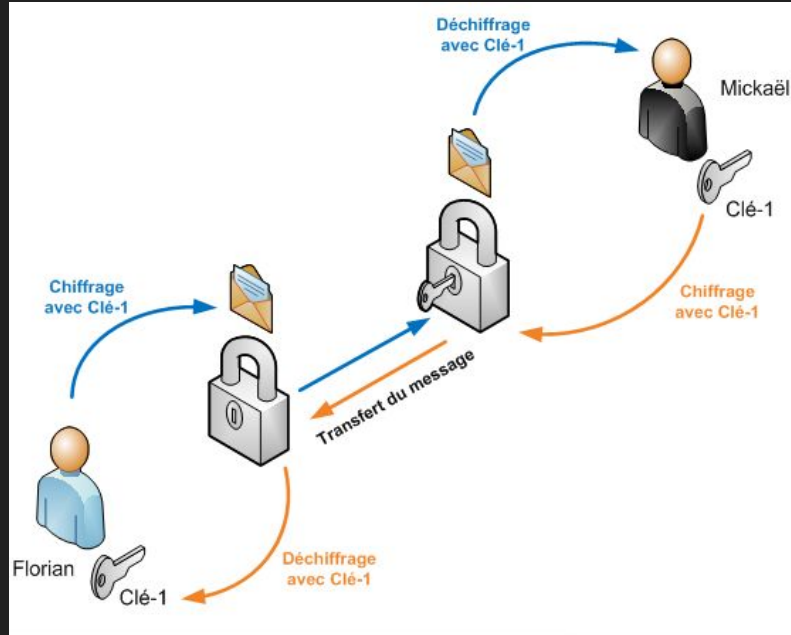
Le chiffrement

Le chiffrement est un procédé de cryptographie grâce auquel on souhaite rendre la compréhension d'un document impossible à toute personne qui n'a pas la clé de chiffrement. Ce principe est généralement lié au principe d'accès conditionnel. (Wikipedia)





Clés symétriques



Florian & Michaël possèdent la même clef qui permet de décoder le message chiffré.

Avantages :

- Simplicité de fonctionnement
- Système rapide et performant

Inconvénients :

- La personne qui récupère la clef pourra chiffrer des messages
- Elle pourra aussi déchiffrer les messages chiffrés à partir de cette clé = faille de sécurité.

source : it-connect.fr





Clés asymétriques

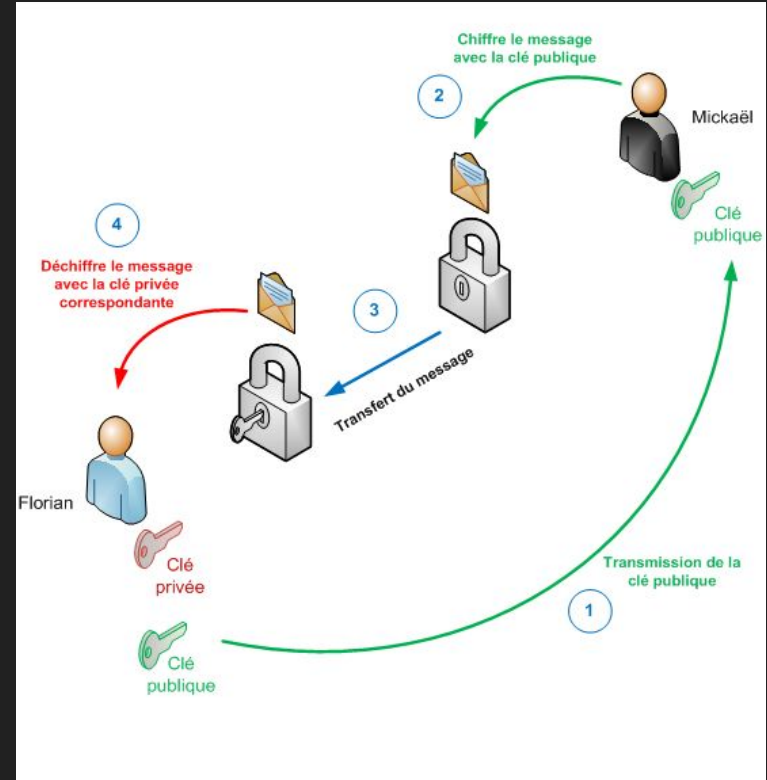
Florian possède une paire de clés asymétrique. Pour pouvoir recevoir un message chiffré de Mickaël, il doit transmettre sa clé publique pour que Michael puisse lui transmettre des messages chiffrés. Mickaël peut ensuite chiffrer un message avec la clé publique reçue, puis envoyer le message à Florian. Florian peut ensuite déchiffrer le message grâce à sa clé privée qui est la seule à pouvoir déchiffrer le message.

Avantages :

- Permet de distribuer la clé publique sans risque que les messages soit déchiffrés avec = très sécurisé

Inconvénients :

- Il est impossible de retrouver une clé privée à partir d'un clé publique





SSH

Secure Shell (SSH) est à la fois un programme informatique et un protocole de communication sécurisé. Le protocole de connexion impose un échange de clés de chiffrement asymétrique en début de connexion. Par la suite, tous les segments TCP sont authentifiés et chiffrés. Il devient donc impossible d'utiliser un sniffer (comme Wireshark) pour voir ce que fait l'utilisateur. (wikipedia)





L'OWASP

L'Open Web Application Security Project® (OWASP) est une fondation à but non lucratif qui œuvre pour améliorer la sécurité des logiciels. Grâce à des projets de logiciels open source menés par la communauté, des centaines de sections locales dans le monde entier, des dizaines de milliers de membres et des conférences d'éducation et de formation dirigées, la Fondation OWASP est la source pour les développeurs pour sécuriser le Web.

- Outils et ressources
- Communauté et réseautage
- Éducation et formation

Chaque année l'OWASP publie son "Top Ten", un document de sensibilisation standard destiné aux développeurs et à la sécurité des applications Web. Il représente un large consensus sur les risques de sécurité les plus critiques pour les applications Web.





RGPD

Le [RGPD](#) (Règlement général sur la protection des données) encadre le traitement et la circulation des données à caractère personnelles sur le territoire européen.

- Entré en vigueur le 25 mai 2018
- Volonté de créer un cadre juridique unifié entre les pays de l'UE
- S'applique à tous les organismes traitants des données personnelles, dès lors qu'ils sont établis en europe.
- L'employeur à la responsabilité de la mise en oeuvre et du contrôle de ce cadre juridique.





RGPD

R pour Registre de traitements :

- modèle de registre sur le site de la CNIL (<https://www.cnil.fr/fr/RGDP-le-registre-des-activites-de-traitement>)
- https://www.cnil.fr/sites/default/files/atoms/files/registre_rgpd_basique.pdf

G pour Gouvernance :

- Principe de responsabilité : mise en place de mécanisme permettant de démontrer le respect des règles relatives à la protection de données
- Impose le respect du **privacy by design** : prise en considération de la vie privée dès la conception
- Impose le respect du **privacy by default**
- Traitements des réclamations dans les plus brefs délais





RGPD

P pour Preuve :

- Mettre en place des processus permettant de prouver le respect des nouvelles obligations
- Prouver la véracité des informations, la légalité du processus d'obtention des données

D pour DPO :

- Définir un DPO ([Délégué de la protection des données](#))
- Le DPO supervise le respect de la conformité
- Sensibilise les employés au respect du RGPD





Pentest

Le test d'intrusion

La méthode consiste généralement à analyser l'infrastructure d'un réseau informatique, afin de simuler l'attaque d'un utilisateur mal intentionné, voire d'un logiciel malveillant (« malware »).

Le pentester analyse alors les risques potentiels dus à une mauvaise configuration d'un système d'information, d'un défaut de configuration, de programmation informatique ou encore d'une vulnérabilité liée à la solution testée.

source : wikipedia





Fingerprinting

La collecte d'information

La collection d'informations (fingerprinting) permet de récupérer des informations publiques pouvant faciliter l'attaquer sur le système informatique d'une entreprise ou d'un institution.

C'est en général la première phase d'un pentest.





Fingerprinting

Les outils

- Outils :
 - Wappalizer : <https://www.wappalyzer.com/>
 - nmap : <https://nmap.org/>
 - httpprint : <http://www.net-square.com/httpprint.html>
- Outils online :
 - <https://www.netcraft.com/>
 - <https://www.shodan.io/>





Authentification

Les algorithmes

Hachage : On nomme fonction de hachage, de l'anglais hash function par analogie avec la cuisine, une fonction particulière qui, à partir d'une donnée fournie en entrée, calcule une empreinte numérique servant à identifier rapidement la donnée initiale

Source : wikipedia

Exemples d'algorithme :

- SHA-1
- MD5
- MD1
- bcrypt (recommandé)

Les algorithmes en rouge sont très facilement décryptable. A n'utiliser sous aucun prétexte !





Authentification

Authentication via Session

Dans ce type d'authentification, le serveur est responsable de la création d'une session pour l'utilisateur lorsqu'il se connecte. L'id de la session est ensuite stocké dans un cookie sur le navigateur de l'utilisateur.

Pour chaque requête faite par l'utilisateur, le cookie sera également envoyé. Le serveur peut ensuite comparer l'identifiant de session du cookie avec les informations de session stockées sur le serveur afin que l'identité de l'utilisateur soit vérifiée.

Le session peut être sous plusieurs format, principalement sous forme de fichier sur le serveur ou en base de donnée (table "sessions" par exemple)





Authentification

Authentification via Jeton (ou token)

Avec l'authentification par Token, le serveur crée un JSON Web Token (JWT) et envoie le jeton au client.

Le JWT est généralement stocké dans un stockage local ou dans le cookie, et il est inclus dans chaque demande faite par l'utilisateur.

Le serveur validera le JWT. L'authentification par JWT est une approche plus moderne utilisée dans les nouvelles applications Web et pour les appareils mobiles. L'état de l'utilisateur n'est pas stocké sur le serveur, il est stocké dans le jeton.





Authentification

JWT

JSON Web Token (JWT) est une norme ouverte (RFC 7519) qui définit un moyen compact et autonome de transmettre en toute sécurité des informations entre les parties en tant qu'objet JSON. Ces informations peuvent être vérifiées et fiables car elles sont signées numériquement. Les JWT peuvent être signés à l'aide d'un secret (avec l'algorithme HMAC) ou d'une paire de clés publique / privée à l'aide de RSA ou ECDSA.

Bien que les JWT puissent être chiffrés pour assurer également la confidentialité entre les parties, nous nous concentrerons sur les jetons signés. Les jetons signés peuvent vérifier l'intégrité des revendications qu'ils contiennent, tandis que les jetons chiffrés cachent ces revendications à d'autres parties. Lorsque les jetons sont signés à l'aide de paires de clés publiques / privées, la signature certifie également que seule la partie détenant la clé privée est celle qui l'a signée.

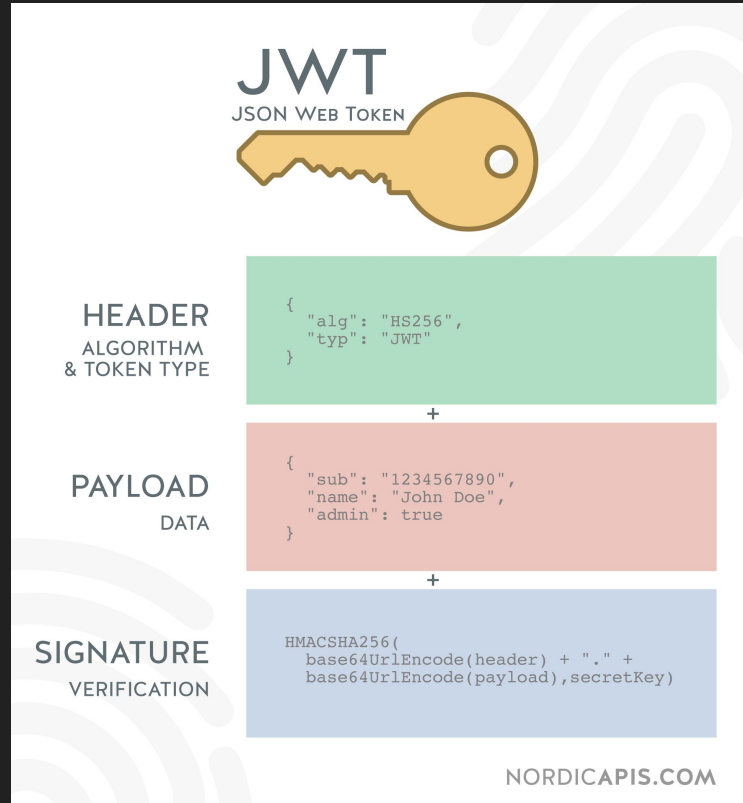
Source : <https://jwt.io/>





Authentication

JWT





Captcha

Définition

Un CAPTCHA est conçu pour déterminer si un utilisateur en ligne est vraiment un humain et non un robot. CAPTCHA est un acronyme qui signifie «**Completely Automated Public Turing test to tell Computers and Humans Apart**».

Ces tests sont un moyen de gérer l'activité des robots. Ils sont particulièrement utiles pour empêcher les robots malveillants de créer automatiquement des données dans votre application, pour empêcher le web-scraping.

Bien que les CAPTCHA soient conçus pour bloquer les robots automatisés, les CAPTCHA sont eux-mêmes automatisés. Ils sont programmés pour apparaître à certains endroits sur un site Web.





Captcha

reCaptcha

reCAPTCHA est un service gratuit proposé par Google en remplacement des CAPTCHA traditionnels. La technologie reCAPTCHA a été développée par des chercheurs de l'Université Carnegie Mellon, puis acquise par Google en 2009.

C'est maintenant la technologie CAPTCHA la plus utilisée, vous pouvez la trouver partout. Il fournit différents types de CAPTCHA, plus ou moins protecteurs et intrusifs.

- **ReCAPTCHA v3**: algorithme basé sur le score qui vous permet de gérer le résultat de la vérification comme vous le souhaitez
- **ReCAPTCHA v2**: "Je ne suis pas un robot": nécessite de cliquer sur une case à cocher et mettra potentiellement l'utilisateur au défi
- **«Invisible»**: géré avec Javascript et moins intrusif





Captcha

reCaptcha : comment ça marche

Vous intégrez le CAPTCHA dans votre application.

Lors de la soumission d'un formulaire, vous déclencherez le CAPTCHA qui générera un jeton et l'injectera dans la demande de soumission.

Lorsque vous recevez la demande d'envoi de formulaire, vous devez vérifier auprès des services Google que le jeton est valide.

Si le jeton est valide, vous acceptez la demande.

Pour intégrer le CAPTCHA dans votre frontend, vous avez besoin d'une clé de site. Pour vérifier le jeton, vous devez fournir une clé secrète.

Ces clés sont fournies par Google lors de la configuration d'un ReCAPTCHA dans la console d'administration:

<https://www.google.com/u/0/recaptcha/admin>





Captcha

reCaptcha : A vous de jouer

1. Télécharger le repo du cours : <https://github.com/LouisHrg/web-security-esgi>

```
cd web-security-esgi
cd recaptcha

# Pour docker
docker-compose build
docker-compose up

composer install
```

3. Allez sur <http://localhost:9999/> !



2. Ouvrez le projet et installer les dépendances



Captcha

reCaptcha : A vous de jouer

Protéger votre application avec un “Je ne suis pas un robot” ReCAPTCHA v2.

Vous devez créer une clef api ici : <https://www.google.com/u/0/recaptcha/admin>

La documentation : <https://developers.google.com/recaptcha/docs/display>

Pour vérifier le token côté serveur vous aurez besoin de [Guzzle](#)

Une fois fini, vous pouvez essayer avec le ReCAPTCHA v3.





DDOS

Une attaque par déni de service (abr. DoS attack pour Denial of Service attack en anglais) est une attaque informatique ayant pour but de rendre indisponible un service, d'empêcher les utilisateurs légitimes d'un service de l'utiliser. À l'heure actuelle la grande majorité de ces attaques se font à partir de plusieurs sources, on parle alors d'attaque par déni de service distribuée (abr. DDoS attack pour Distributed Denial of Service attack).

Il peut s'agir de :

- l'inondation d'un réseau afin d'empêcher son fonctionnement ;
- la perturbation des connexions entre deux machines, empêchant l'accès à un service particulier ;
- l'obstruction d'accès à un service pour une personne en particulier ;
- également le fait d'envoyer des milliards d'octets à une box internet.

source : wikipedia

L'outil le plus connu pour faire un DDOS : <https://sourceforge.net/projects/loic0/>





Rate limiting & Throttling

- Le **rate limiting** est utilisé pour contrôler le débit du trafic envoyé ou reçu sur un serveur. Le trafic inférieur ou égal à la limite spécifiée est envoyé, tandis que le trafic excédant cette limite est rejeté ou retardé. Le rate-limiting peut être aussi implémenté au niveau applicatif.
- La limitation de bande passante (bandwidth throttling en anglais) est une méthode pour empêcher un utilisateur demandant une bande passante importante (ex : un utilisateur qui télécharge des fichiers) de dépasser un certain débit en limitant la quantité de données transmise et/ou reçue pendant un certain laps de temps. Elle peut aussi être appliquée au niveau du serveur web et aussi au niveau applicatif.

source : wikipedia





Caching

Plus un site Web est visité, plus il est important de trouver des techniques pour délivrer des pages rapidement.

On peut acheter de nouveaux serveurs pour optimiser la performance de notre application, cependant cela reste coûteux. Une solution alternative consiste dans le caching des pages. Cette technique, permettant de conserver sous une forme statique - en quelque sorte à portée de main - de l'information potentiellement voulue par l'utilisateur dans le but de la lui fournir plus rapidement quand il la demande.

Par exemple pour un blog, l'information ne change pas à chaque requête de l'utilisateur, on peut donc “cacher” les données à chaque nouvel article sur le blog. Tous les articles seront alors disponible sous forme statique.

source : journaldunet





Brute force

La recherche exhaustive ou recherche par force brute est une méthode algorithmique qui consiste principalement à essayer toutes les solutions possibles. Par exemple pour trouver le maximum d'un certain ensemble de valeurs, on consulte toutes les valeurs. En cryptanalyse on parle d'attaque par force brute, ou par recherche exhaustive pour les attaques utilisant cette méthode.

source : wikipedia

Outils pour bruteforce : Aircrack-ng





Validation des données

Définition

La validation d'entrée est effectuée pour s'assurer que seules des données correctement formées entrent dans un système d'information, empêchant les données malformées d'être insérées dans la base de données et de déclencher un dysfonctionnement.

La validation des entrées doit avoir lieu le plus tôt possible dans le flux de données, de préférence dès que les données sont reçues de la partie externe.

Les données de toutes les sources potentiellement non fiables doivent être soumises à une validation d'entrée, non seulement les clients Web connectés à Internet, mais également les flux backend sur des extranets, provenant de fournisseurs, partenaires, vendeurs ou régulateurs, chacun d'entre eux pouvant être compromis.





Validation des données

Comment faire pour tester la validation ?

Pour tester la validation de formulaire en faisant des requêtes POST on peut utiliser un client API comme

<https://insomnia.rest/download>

ou encore

<https://www.postman.com>





Validation des données

A vous de jouer

1. Télécharger le repo du cours : <https://github.com/LouisHrg/web-security-esgi>

```
cd web-security-esgi
cd validation

# Pour docker
docker-compose build
docker-compose up

composer install
```

3. Allez sur <http://localhost:9999/> !



2. Ouvrez le projet et installer les dépendances



Validation des données

A vous de jouer

Exercice 1 :

Créer une route /exo-1 qui doit suivre les champs avec les règles suivantes :

- **firstname**, chaîne de caractère, obligatoire, 3 caractères min, 30 caractères max
- **lastname**, chaîne de caractère, obligatoire, 3 caractères min, 50 caractères max
- **age**, integer, obligatoire, supérieur à 18 et inférieur à 120
- **github**, url, facultatif

Une fois les 4 champs validés, l'api retourne les 4 champs.





Validation des données

A vous de jouer

Exercice 2 :

Créer une route /exo-2 qui doit suivre les champs avec les règles suivantes :

- **phone**, string, obligatoire, doit être un numéro de téléphone au format français
- **description**, obligatoire, chaîne de caractère, 250 caractères max, supprimé les espaces au début et à la fin de la chaîne de caractère
- **type**, obligatoire, doit être égal à "contact"

Une fois les champs validés, l'api retourne les champs validé et traités.





Validation des données

A vous de jouer

Exercice 3 :

Créer une route /exo-3 qui doit suivre les champs avec les règles suivantes :

- **email**, chaîne de caractère, obligatoire, doit être un email valide au bon format.
- **password**, chaîne de caractère, obligatoire, doit être faire min 8 car, max 255, doit contenir une majuscule, une minuscule, un caractère spécial, un chiffre.
- **password_confirmation**, obligatoire, doit être égal à **password**
- **birthdate**, obligatoire, chaîne de caractère, doit être au format "dd/mm/yyyy"

Une fois les champs validés, l'api retourne les champs validé sauf **password** et **password_confirmation**.





Les outils

Pour rechercher des failles

- L'OWASP Zed Attack Proxy : <https://www.zaproxy.org/>
- SQLmap : <http://sqlmap.org/>
- wpscan : <https://wpscan.com/wordpress-security-scanner>





Les outils

Base de donnée de vulnérabilité

- Common vulnerability and Exposure : <https://www.cvedetails.com/>
- National Vulnerability Database : <https://nvd.nist.gov/>
- Security Focus : <https://www.securityfocus.com/vulnerabilities>





Les outils

Exploits

Un exploit est un bout de code permettant d'exploiter une faille de sécurité qu'elle soit distance ou locale :

- <https://www.exploit-db.com/>
- <https://packetstormsecurity.com/>
- <https://0day.today/>





Les outils

Pour se protéger

- <https://haveibeenpwned.com/>
- <https://www.cloudflare.com/>
- [Agence nationale de la sécurité des systèmes d'information](#)



III. Application

Les failles les plus courantes, comment les reconnaître et les patcher.





Injection SQL

Définition

Une attaque par injection SQL consiste en l'insertion ou «injection» d'une requête SQL via les données d'entrée du client vers l'application. Un exploit d'injection SQL réussi peut lire des données sensibles de la base de données, modifier les données de la base de données (Insérer / Mettre à jour / Supprimer), exécuter des opérations d'administration sur la base de données (comme l'arrêt du SGBD), récupérer le contenu d'un fichier donné présent sur le fichier SGBD système. Dans certains cas, émettre des commandes vers le système d'exploitation.

Source : owasp





Injection SQL

Démonstration

1. Télécharger le repo du cours : <https://github.com/LouisHrg/web-security-esgi>

```
cd web-security-esgi
cd sql-injection

# Si vous utilisez docker
docker-compose build
docker-compose up

composer install
```

3. Allez sur <http://localhost:9999/login> !

2. Ouvrez le projet et installer les dépendances





Injection SQL

Comment s'en protéger

Ne **JAMAIS** faire confiance à l'utilisateur





Injection SQL

Comment s'en protéger

Il est EXTRÊMEMENT simple d'éviter les injection SQL dans votre code.

Des failles d'injection SQL sont introduites lorsque les développeurs d'application créent des requêtes de base de données dynamiques qui incluent une entrée fournie par l'utilisateur.

Pour éviter les failles d'injection SQL, c'est simple. Les développeurs doivent:

- a) Vérifier tout ce que saisi l'utilisateur**
- b) Empêcher l'entrée fournie par l'utilisateur qui contient du SQL malveillant d'affecter la logique de la requête exécutée.**

source : Owasp





Injection SQL

Comment s'en protéger

En php pour prévenir une attaque sql on peut :

- Utiliser des [requêtes préparées](#)
- Utilisation la fonction [addslashes](#) pour “escape” les quotes (et donc neutraliser une possible injection)





Injection SQL

A vous de jouer

Trouvez la faille et fixez la ! (en modifiant uniquement index.php)





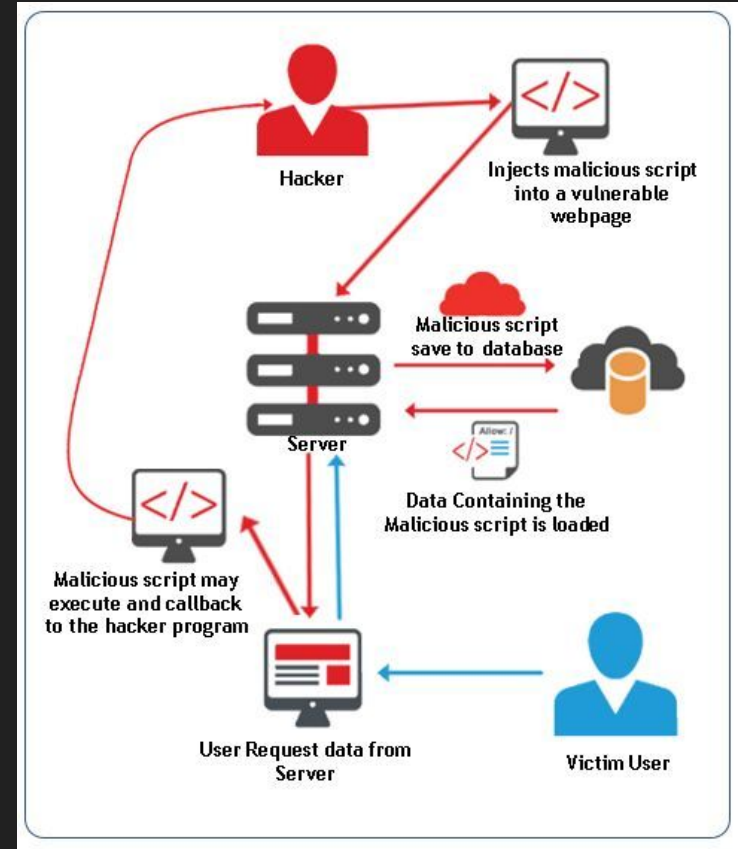
Reflected Cross-Site Scripting(XSS)

Définition

“Les attaques de type Cross-Site Scripting (XSS) réfléchi (ou non permanent) sont un type d'injection, dans lequel des scripts malveillants sont injectés dans des sites Web de confiance. Les attaques XSS se produisent lorsqu'un attaquant utilise une application Web pour envoyer du code malveillant, généralement sous la forme d'un script côté navigateur, à un autre utilisateur final.

Les failles qui permettent à ces attaques de réussir sont assez répandues et se produisent partout où une application Web utilise l'entrée d'un utilisateur dans la sortie qu'elle génère sans la valider ni l'encoder.”

Source : owasp.org





Reflected Cross-Site Scripting (XSS)

Démonstration

```
cd web-security-esgi
cd xss
# Pour docker
docker-compose build
docker-compose up

composer install
```

2. Allez sur <http://localhost:9999> !

1. Ouvrez le projet et installer les dépendances





Reflected Cross-Site Scripting (XSS)

Démonstration

Si vous cherchez bien, vous pouvez voir que avez la possibilité de changer le titre et le contenu de la page via des query string (variables dans l'url) :

<http://localhost:9999/?title=toto&content=mon%20super%20contenu>

C'est une petite faille de sécurité qui n'est pas bien méchante !





Reflected Cross-Site Scripting (XSS)

Démonstration

Mais on peut aller plus loin :

[http://localhost:9999/?title=Hey!&content=<script>alert\('Vous avez été hacké !'\)</script>](http://localhost:9999/?title=Hey!&content=<script>alert('Vous avez été hacké !')</script>)

Et la faille devient beaucoup plus dangereuse





Reflected Cross-Site Scripting (XSS)

Démonstration

On peut même récupérer le cookie du navigateur :

`http://localhost:9999/?title=Hey!&content=<script>alert(document.cookie)</script>`

ou même faire une redirection :

`http://localhost:9999/?title=Hey!&content=<script>redirect:window.location.replace("https://youtu.be/dQw4w9WgXcQ");</script>`





Reflected Cross-Site Scripting (XSS)

Comment s'en protéger

Ne **JAMAIS** faire confiance à l'utilisateur





Reflected Cross-Site Scripting (XSS)

Comment s'en protéger

Vous devez toujours remplacer les caractères spéciaux HTML par leur [entité HTML](#) :

*< devient **‹**;
> devient **›**;
& devient **&**;
etc*

En php vous pouvez utiliser la fonction [htmlspecialchars](#)

```
htmlspecialchars(string $string)
```





Reflected Cross-Site Scripting (XSS)

A vous de jouer

En modifiant uniquement [index.php](#), patchez la faille de sécurité !

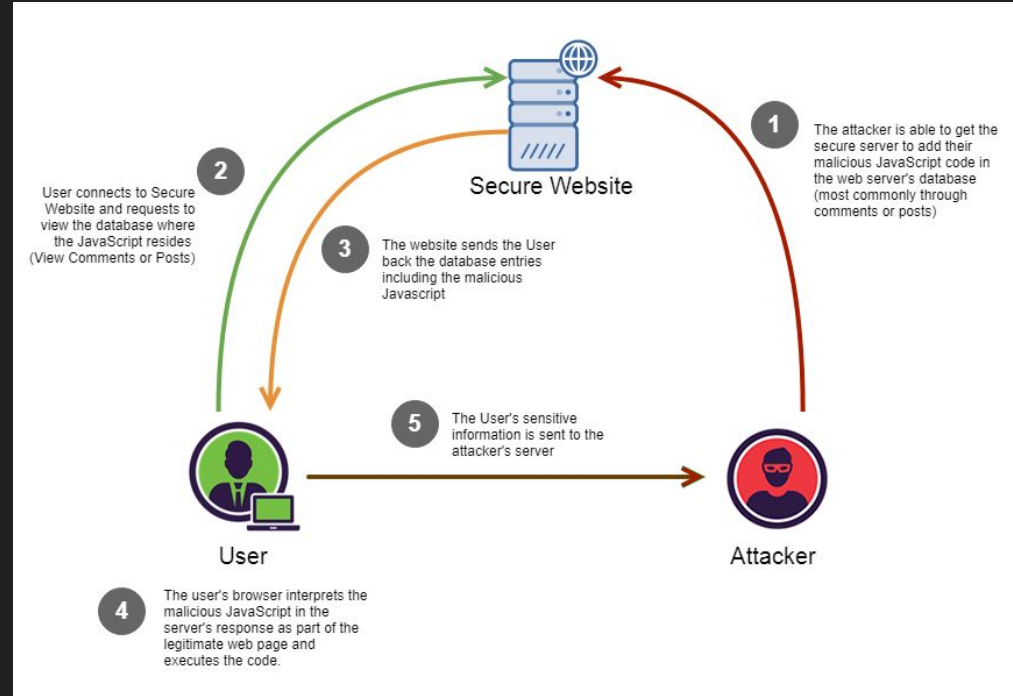




Permanent Cross-Site Scripting (XSS)

Démonstration

“Les attaques persistantes sont celles où le script injecté est stocké en permanence sur les serveurs cibles, comme dans une base de données, dans un forum, un blog, un champ de commentaire, etc. La victime récupère alors le script malveillant du serveur lorsqu'elle demande les informations. Le XSS persistant est également parfois appelé XSS stocké”





Permanent Cross-Site Scripting (XSS)

A vous de jouer

Allez sur <http://localhost:9999/boot> pour initialiser la base de donnée

Naviguer sur <http://localhost:9999/blog>

Trouvez la faille et fixez la ! (en modifiant uniquement index.php)





Cross-site request forgery (CSRF)

Définition

La Cross-site request forgery (CSRF) est une attaque qui force un utilisateur final à exécuter des actions indésirables sur une application Web dans laquelle il est actuellement authentifié. Avec un peu d'ingénierie sociale, un attaquant peut inciter les utilisateurs d'une application Web à exécuter les actions de son choix. Si la victime est un utilisateur normal, une attaque CSRF réussie peut forcer l'utilisateur à effectuer des actions comme des virements, la modification de son adresse e-mail, etc. Si la victime est un compte administratif, le CSRF peut compromettre l'ensemble de l'application Web.

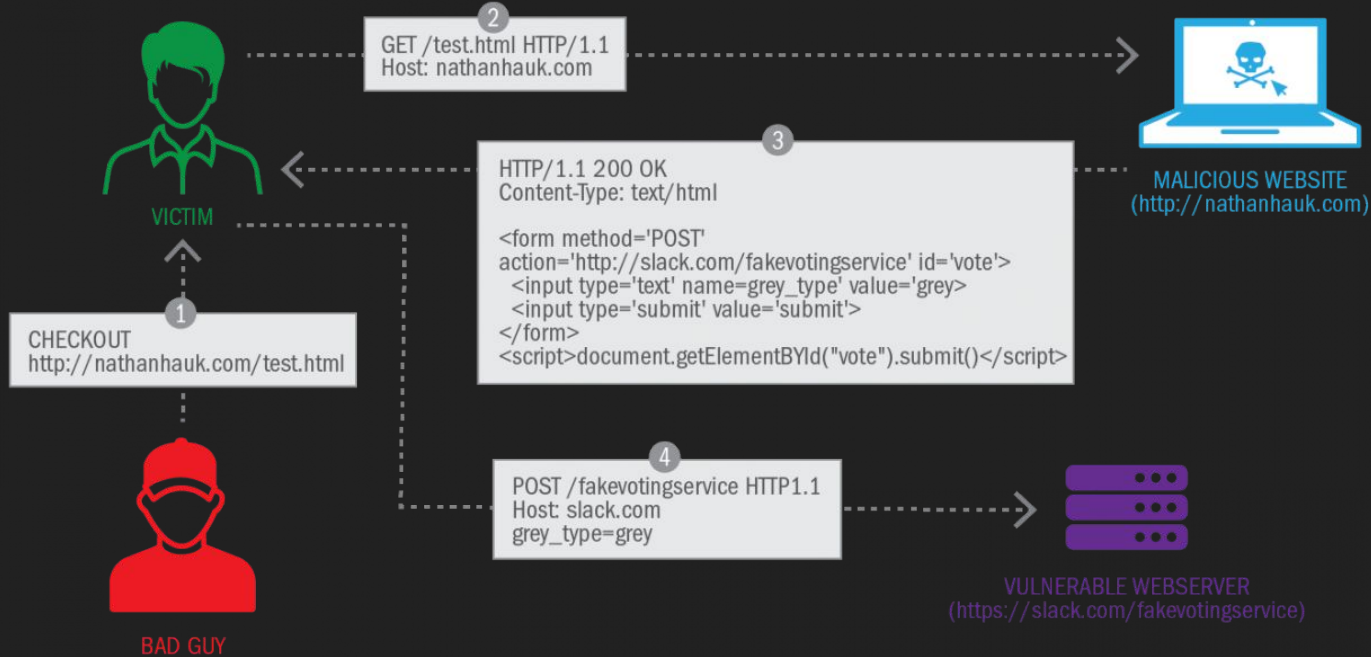
Source : owasp





Cross-site request forgery (CSRF)

Définition





Cross-site request forgery (CSRF)

Démonstration

```
cd web-security-esgi  
cd csrf
```

```
# Pour docker  
docker-compose build  
docker-compose up  
  
composer install
```

2. Allez sur <http://localhost:9999> !

1. Ouvrez le projet et installer les dépendances





Cross-site request forgery (CSRF)

Comment s'en protéger

Pour se protéger des failles CSRF nous allons utiliser un token stocké en session. Cette défense est la plus populaire et recommandée pour se protéger de ce genre de faille. Le token sera généré à l'affichage de la page, stocké en session et inséré dans le formulaire. A la validation on va tout simplement comparer le token du formulaire et celui en session.





Cross-site request forgery (CSRF)

A vous de jouer

Trouvez la faille et fixez la ! (en modifiant uniquement index.php)





Authentification faillible

Présentation

“Les fonctions applicatives liées à l'authentification et à la gestion de session sont souvent mal implémentées, ce qui permet aux attaquants de compromettre les mots de passe, les clés ou les jetons de session, ou d'exploiter d'autres failles d'implémentation pour assumer temporairement ou définitivement l'identité des autres utilisateurs.”

Source : owasp





Authentification faillible

A vous de jouer

```
cd web-security-esgi
cd bad-auth

# Pour docker
docker-compose build
docker-compose up

composer install
```

2. Allez sur <http://localhost:9999> !

1. Ouvrez le projet et installer les dépendances





Authentication faillible

A vous de jouer

Trouvez la faille et fixez la ! (en modifiant uniquement index.php)





Exposition de données sensibles

“De nombreuses applications Web et API ne protègent pas correctement les données sensibles, telles que les données financières, les soins de santé et les informations personnelles. Les attaquants peuvent voler ou modifier ces données faiblement protégées pour mener une fraude par carte de crédit, un vol d'identité ou d'autres crimes. Les données sensibles peuvent être compromises sans protection supplémentaire, telle que le cryptage au repos ou en transit, et nécessitent des précautions particulières lorsqu'elles sont échangées avec le navigateur.”

Source : owasp





Access Control (ACL) faillible

“Les restrictions (ACL) sur ce que les utilisateurs authentifiés sont autorisés à faire ne sont souvent pas correctement appliquées.

Les attaquants peuvent exploiter ces failles pour accéder à des fonctionnalités et / ou des données non autorisées, telles que l'accès aux comptes d'autres utilisateurs, afficher des fichiers sensibles, modifier les données d'autres utilisateurs, modifier les droits d'accès, etc.”

Source : owasp





Utilisation de composants avec des vulnérabilités connues

Les composants, tels que les bibliothèques, les frameworks et d'autres modules logiciels, s'exécutent avec les mêmes privilèges que l'application. Si un composant vulnérable est exploité, une telle attaque peut faciliter de graves pertes de données ou une prise de contrôle du serveur. Les applications et les API utilisant des composants avec des vulnérabilités connues peuvent saper les défenses des applications et permettre diverses attaques et impacts.

Source : owasp

[Exemple d'une lib nodejs avec une faille de sécurité](#)





Logging et Monitoring insuffisants

Un logging et du monitoring insuffisants, associées à une intégration manquante ou inefficace avec la réponse aux incidents, permettent aux attaquants d'attaquer davantage les systèmes, de maintenir la persistance, de basculer vers plus de systèmes et de falsifier, extraire ou détruire les données.

La plupart des études sur les violations montrent que le délai de détection d'une violation est supérieur à 200 jours, généralement détecté par des parties externes plutôt que par des processus ou une surveillance internes.

Source : owasp



III. A vous de jouer

Projet de groupe





Projet de groupe

Un client important veut que vous réalisiez l'audit de son site internet et que vous trouviez les failles.

Avec le cours et les connaissances acquises pendant le cours, trouvez les failles du site du client et faite un audit (document pdf listant les failles de sécurité et comment y remédier).

Chaque groupe passera en soutenance (20 min) pour me présenter son audit le dernier jour de la semaine thématique.

Bonus :

- Fixez les failles vous même

Le site du client est disponible sur ce repo : <https://github.com/LouisHrg/unsecure-app>

