Web Security 101

Ce qu'il faut retenir du cours

<u>Ne jamais faire confiance à l'utilisateur et encore moins à ce qu'il saisit dans votre application web</u>

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.

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.

Pourquoi se protéger son SI : Si on ne protège pas son SI on risque des vols, altérations ou pertes de données, que l'application ne soit pas disponible ou qu'elle ne soit pas résiliente **,** on risque le vol de notre propriété intellectuelle. Ces risques ont des conséquences : perte d'image, perte de confiance, perte de crédibilité, responsabilité civile et/ou pénale et enfin perte financière.

Pourquoi la performance compte aussi : une application performante sera + résiliente, si elle est plus résiliente elle est + sécurisée.

HTTP : Hypertext Transfer Protocol est un protocole de communication client-serveur développé pour le World Wide Web.

HTTPS : Hypertext Transfer Protocol Secure 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.
- Garantit **théoriquement** la confidentialité et l'intégrité des données envoyées par l'utilisateur.

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

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.

Clés symétriques : 2 utilisateurs possèdent la même clef pour déchiffrer un message.

- Avantage : Simple et performant
- Inconvénient : Si quelqu'un récupère la clef il pourra décoder le message

Clés asymétriques : Chaque utilisateur possède une clef publique et une clef privée. Si un utilisateur A veut envoyer un message secret à l'utilisateur B, l'utilisateur A doit chiffrer le message avec la clef publique de l'utilisateur B. L'utilisateur B pourra ensuite décoder le message avec sa clef privée.

- **Avantages :** Permet de distribuer la clef publique sans risque qu'un message puisse être décodé (uniquement la clef privée A peut déchiffrer un message chiffré avec la clef publique A)
- **Inconvénients :** Il est impossible de retrouver une clef privée grâce à une clef publique

SSH : Secure Shell 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.

L'OWASP : L'Open Web Application Security Project® est une fondation à but non lucratif qui œuvre pour améliorer la sécurité des logiciels. Chaque année l'OWASP publie son "Top 10", 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) est un règlement qui encadre le traitement et la circulation des données à caractère personnelles sur le territoire européen. Il est entré en vigueur le 25 mai 2018. Il a pour volonté de créer un cadre juridique unifié entre les pays de l'UE. Il s'applique à tous les organismes traitants des données personnelles, dès lors qu'ils sont établis en europe. Il stipule que l'employeur à la responsabilité de la mise en oeuvre et du contrôle de ce cadre juridique.

Pentest : Le test d'intrusion (ou Pentest) est une méthode qui 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.

Fingerprinting : 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.

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 (avec longueur identique à chaque fois) servant à identifier rapidement la donnée initiale.

Authentification 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.

Authentification via Jeton : 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.

Captcha : 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.

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.

Rate limiting : Il 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. Il peut être implémenté au niveau applicatif.

Throttling : 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

Caching : Cette technique, permet 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. Ex : on peut "cacher" la page d'accueil d'un blog, à chaque fois qu'on rajoute un nouvel article, on va recréer le cache.

Brute force : La 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.

Authentification 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.

Validation des données : 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.

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.
- **Comment s'en protéger :** Vérifier tout ce que saisi l'utilisateur et empêcher l'entrée fournie par l'utilisateur qui contient du SQL malveillant d'affecter la logique de la requête exécutée.

Reflected Cross-Site Scripting :

- **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.
- **Comment s'en protéger :** Toujours remplacer les caractères spéciaux HTML par leur entité HTML

Permanent Cross-Site Scripting :

- **Définition :** 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 le informations. Le XSS persistant est également parfois appelé XSS stocké
- **Comment s'en protéger :** Toujours remplacer les caractères spéciaux HTML par leur entité HTML

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.
- **Comment s'en protéger :** Il faut 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.

Recommandations 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
- Toujours vérifier qu'un site est en HTTPS avant d'envoyer un formulaire

Recommandations en tant de développeur web :

- Toujours valider ce qu'envoie les utilisateurs
- Utiliser des requêtes préparées
- Hasher les mots de passe avec bcrypt
- Utiliser un token stocké en session pour vos formulaires
- Ne stockez pas d'information sensibles dans vos cookies
- Utilisez des certificats SSL pour vos sites
- Utilisez des captchas, du rate limiting ou encore du throttling pour protéger votre service web des attaques automatisées
- Mettez à jour vos dépendances aussi souvent que possible et utilisez des solutions comme dependabot pour vous alerter en cas de faille
- Logger et monitorer vos applications pour pouvoir réagir rapidement en cas d'intrusion