OWASP ET LA SÉCURITÉ DES SITES WEB

BY ANTHONY SEMAL

POURQUOI SÉCURISÉ UN SITE WEB

Protéger les données

Renforcer la confiance

Améliorer le référencement naturel

OWASP

The Open Web Application Security Project

- Fondée en 2001
- Communauté en ligne travaillant sur la sécurité des sites web
- Sa philosophie est d'être à la fois libre et ouverte à tous.
- Publier des recommandations et outils de référence permettant de contrôler le niveau de sécurisation des applications web.

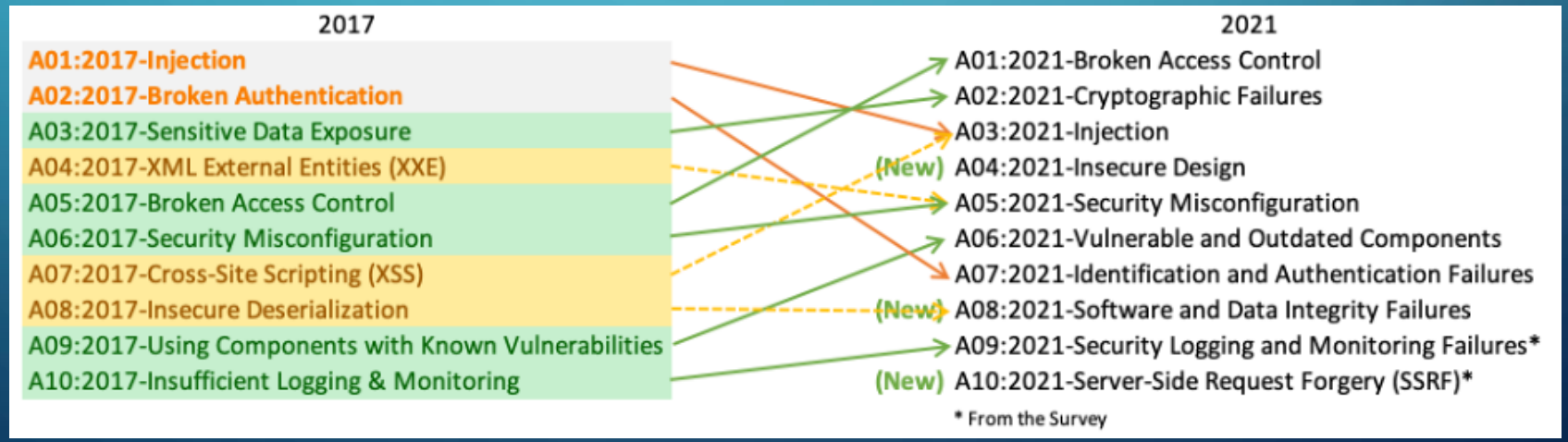
ORGANISATION DU TABLEAU

Calculs du score moyen d'exploitabilité et d'impact -> Incidence

CVE : Common Vulnerability Exposure

CVSS : Common Vulnerability Scoring System

CWE : Common Weakness Enumeration



10 FALSIFICATION DE REQUÊTE CÔTÉ SERVEUR (SSRF)

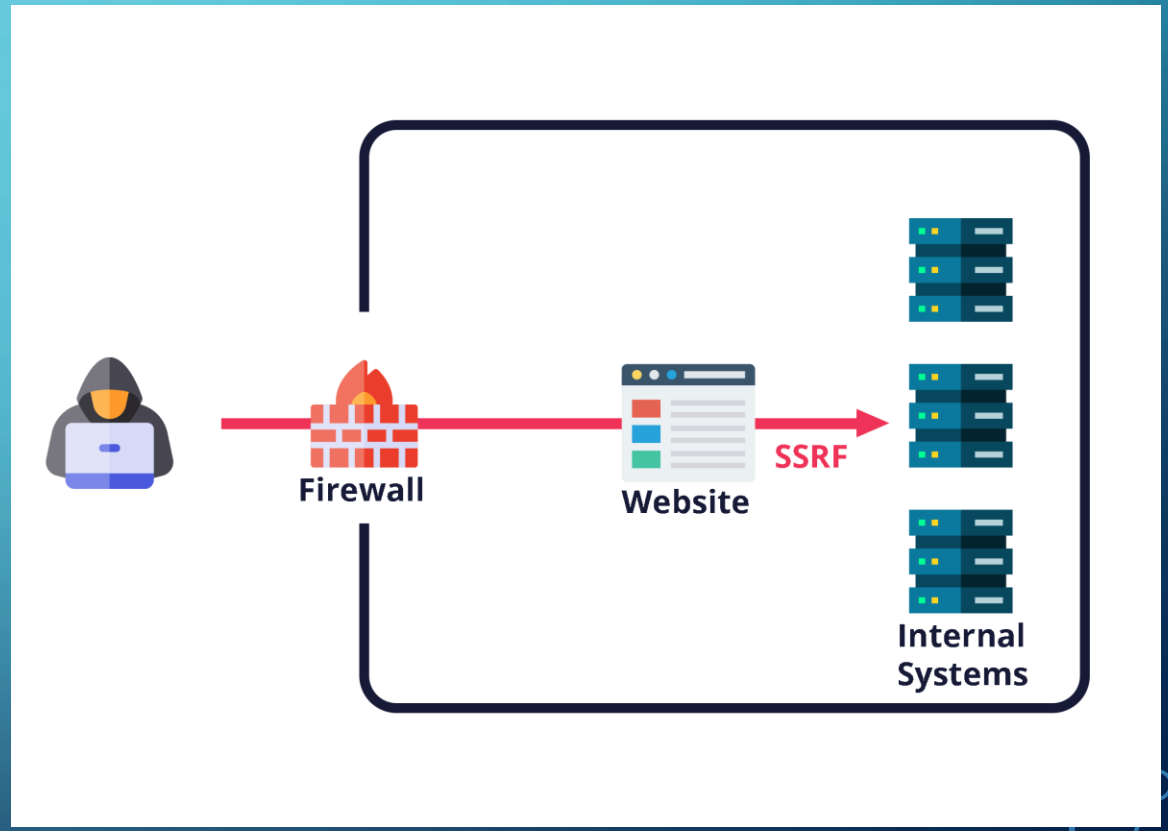
public.example.com héberge un service proxy
-> public.example.com/proxy
-> récupère la page Web spécifiée dans le paramètre URL affiche à l'utilisateur.
lorsque l'utilisateur accède à l'URL :
https://public.example.com/proxy?url=google.com

Sans protection ssrf:
https://public.example.com/proxy?url=admin_panel.example.com

Valider les données fournies

Désactiver les redirections HTTP

Ne pas envoyer des réponses brutes au clients



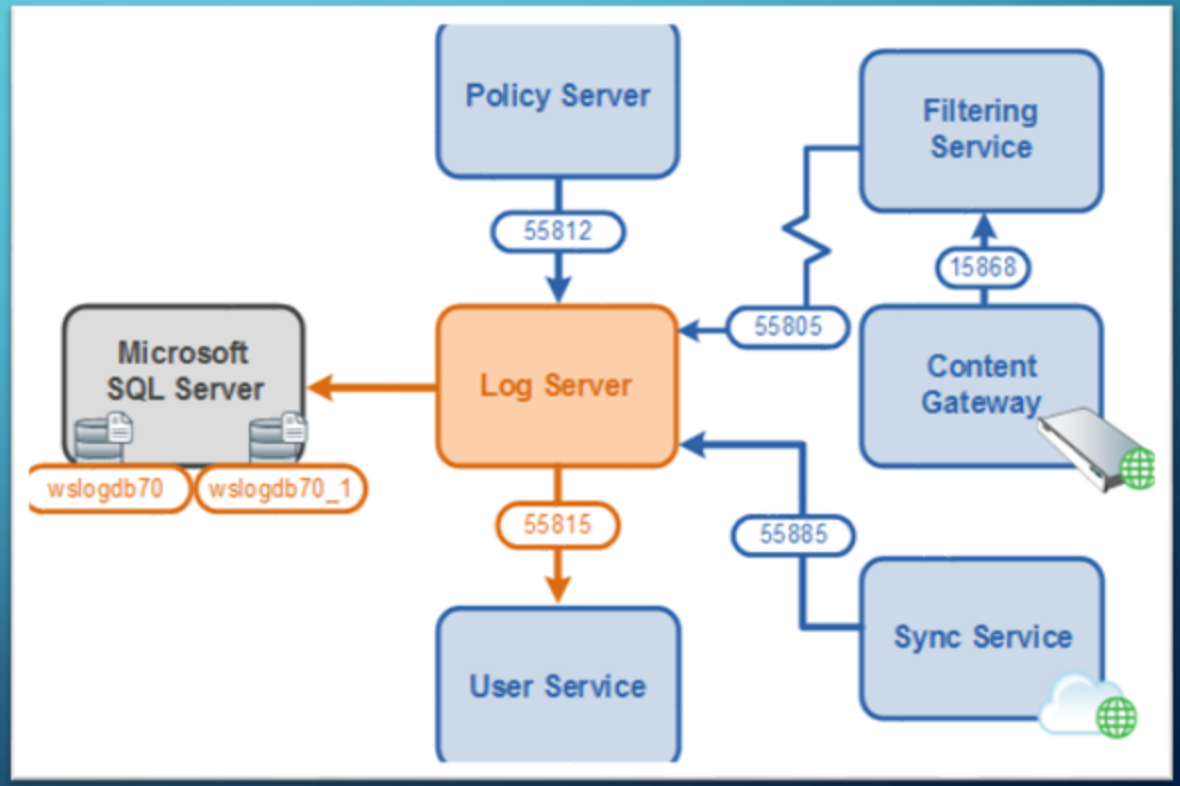
9. CARENCE DES SYSTÈMES DE CONTRÔLE ET DE JOURNALISATION

Enregistré les événements

Détection / Réponse trop tard

Vérifié les enregistrements

Données correctement enregistrées -> système de gestion de logs



8. MANQUE D'INTÉGRITÉ DES DONNÉES ET DU LOGICIEL

Sources non fiables -> Vulnérable

Exemple: color.js / faker.js et Log4shell

Signature numérique -> bonne source non modifié

Processus de révision de code -> vérifié les changements

7. IDENTIFICATION ET AUTHENTIFICATION DE MAUVAISE QUALITÉ

identification par défaut -> login: admin / mdp: 1234

Utiliser des mots de passe en clair / faiblement haché

Questions secrète

Réutilisation de l'identifiant de session

Authentification multifacteurs

Mot de passe fort

Gestionnaire de session

6. COMPOSANTS VULNÉRABLES ET OBSOLÈTE

Logiciels, dépendances non mises à jour

Veille de sécurité non faite

Faire des mises à jour

Supprimer dépendances / fonctionnalités inutiles

Surveiller les CVE

5. SECURITY MISCONFIGURATION

Paramètres par défaut -> mots de passes, certificats, etc...

Bases de données non protégées

Messages d'erreur affichant des informations sensibles

Fonctions inutiles

Processus automatisé -> vérifier l'éfficacité des configurations

Audit de sécurité - Test d'intrusions

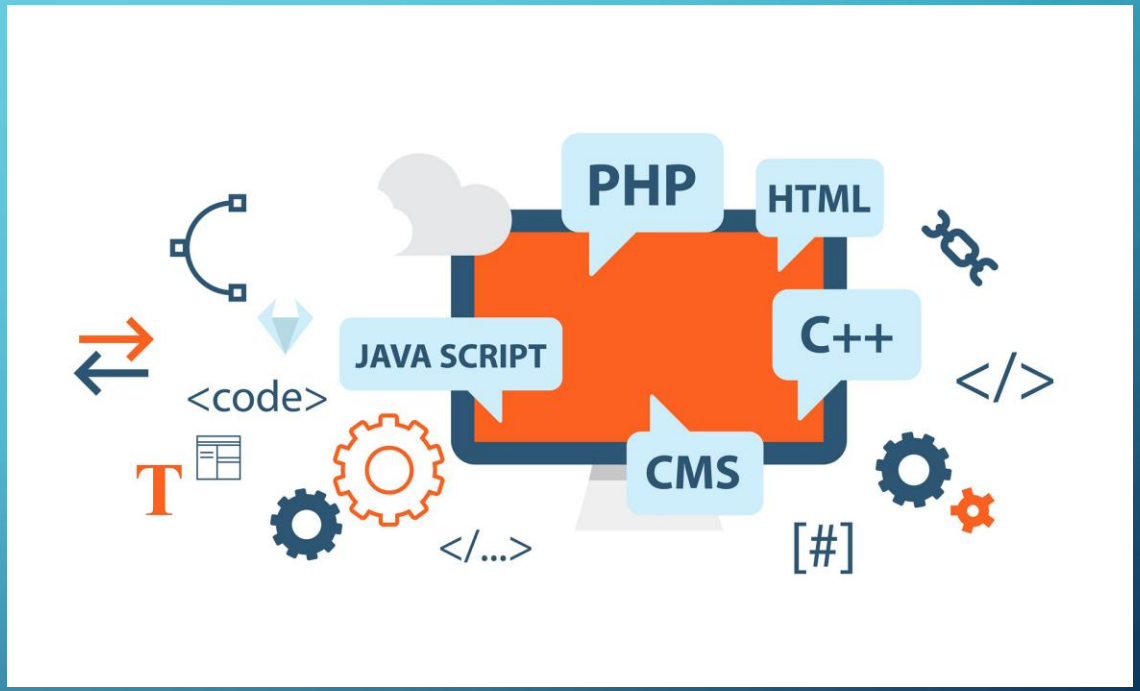
Plateforme minimale -> Fonctionnalités utiles

4. CONCEPTION NON SÉCURISÉE

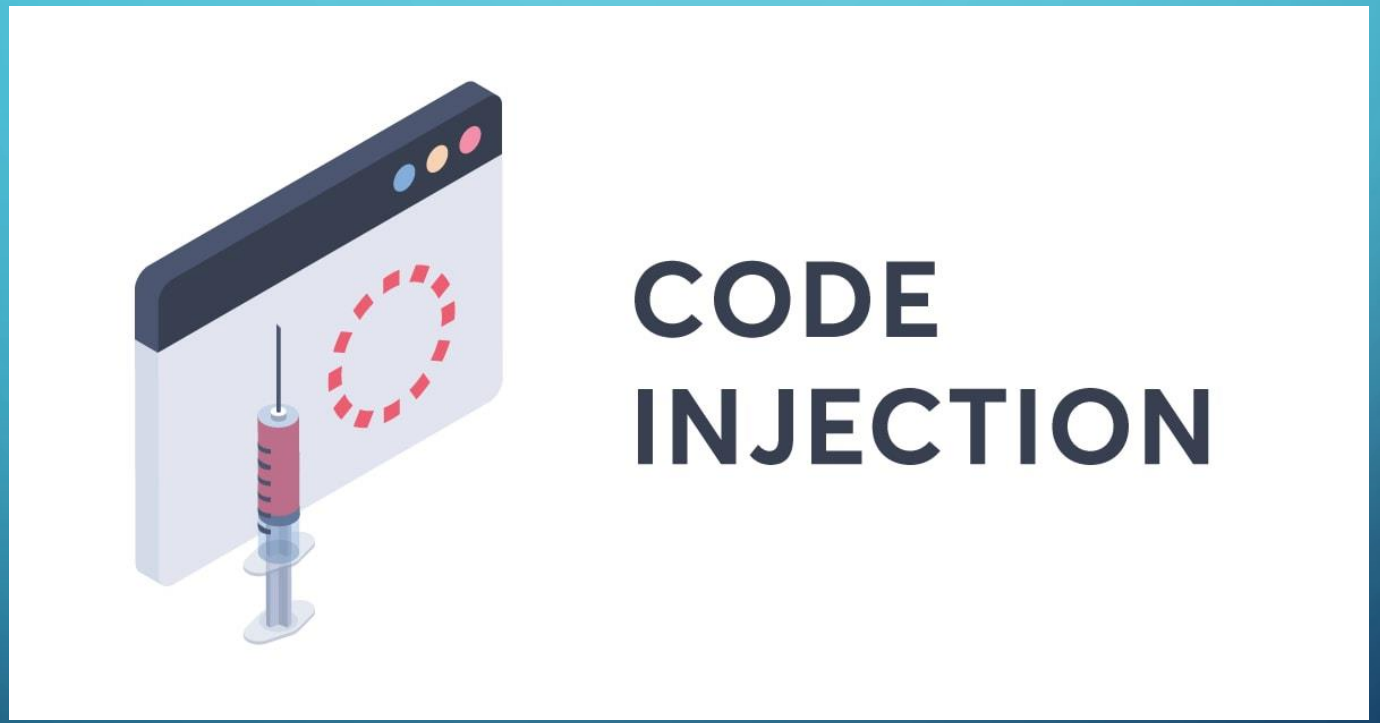
Mauvaise analyse des risques

Mecanisme anti-bots

Définir des règles de sécurité pour chaque cas d'utilisations

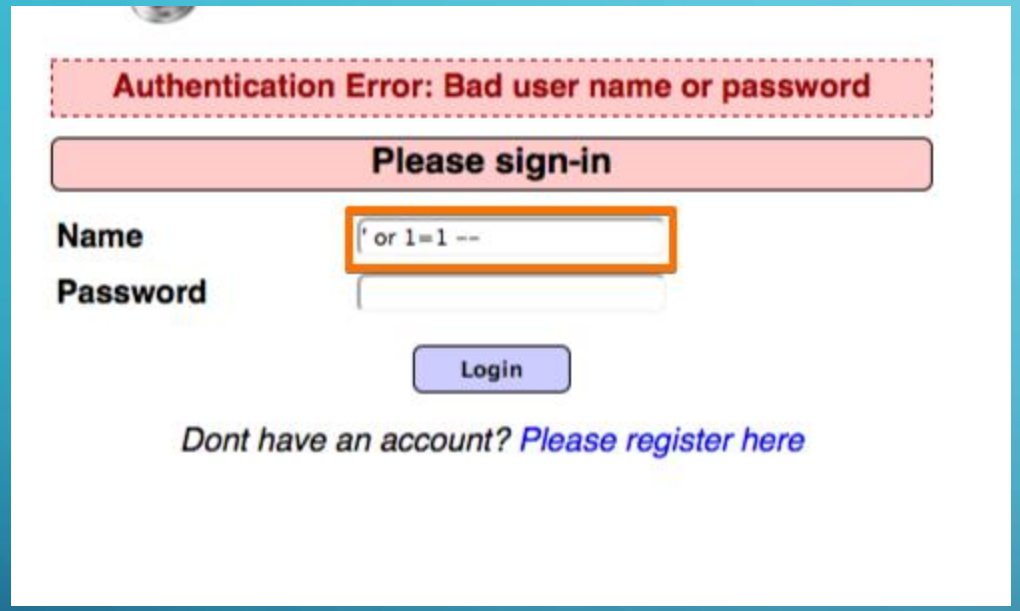


3. LES INJECTIONS DE CODE



3. LES INJECTIONS DE CODE

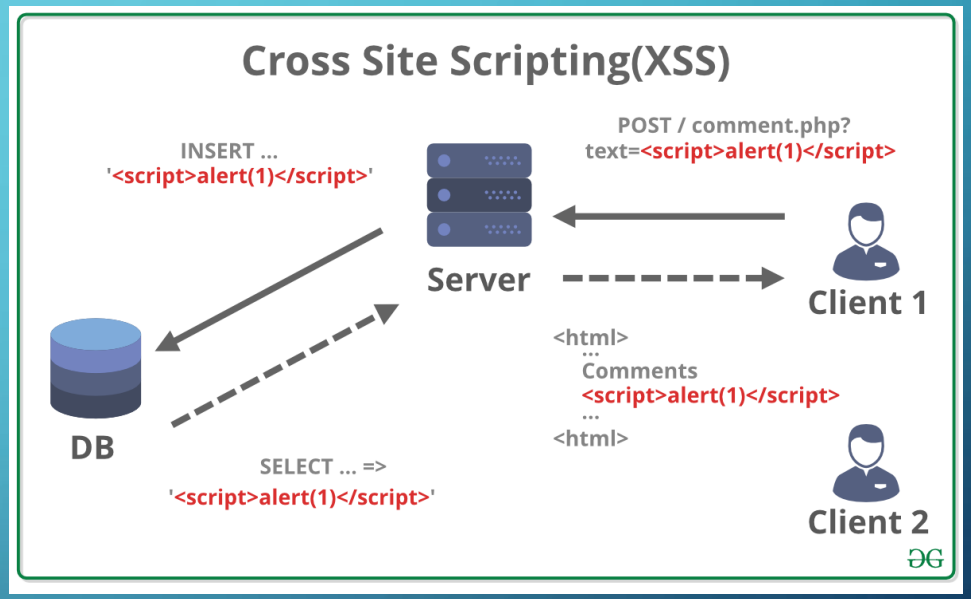
Injection SQL



Mysqli_real_escape_string()
PDO + bindparam

Exemple simple: ‘ or ‘1’ =’1

Injection XSS



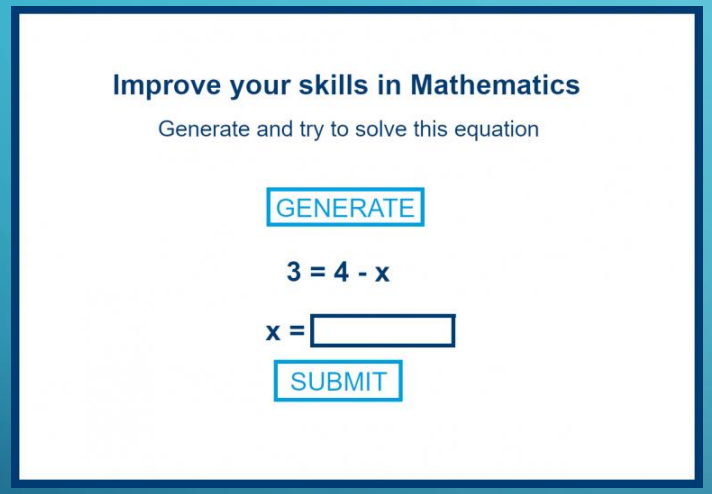
http://localhost/xss.php?keyword=<script>window.alert("Mes cookies sont : " + document.cookie);</script> = non persistant

Forum -> <h1> Salut</h1> = persistant

Htmlspecialchars / htmlentities

3. LES INJECTIONS DE CODE

- Faille OS Command



1;System("ls");

eval('$result = 4 –1;
system("ls");');

index.php style.css

script.js routes.txt

src

2. CRYPTOGRAPHIC FAILURE

Algorithme obsolète -> md5, sha1,...

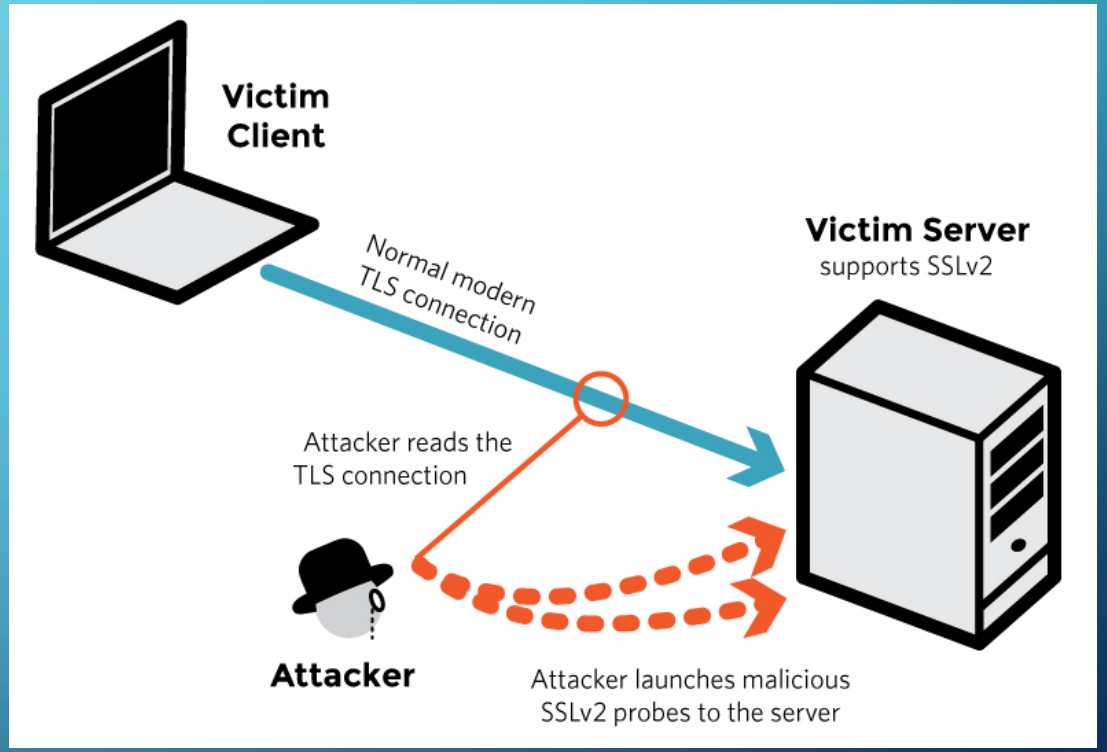
Certificats non valides

Données sensibles en clair -> HTTP, FTP,...

Mot de passe codé en dur

HTTPS (HSTS)

Hachage salé -> Argon2, bcrypt,...



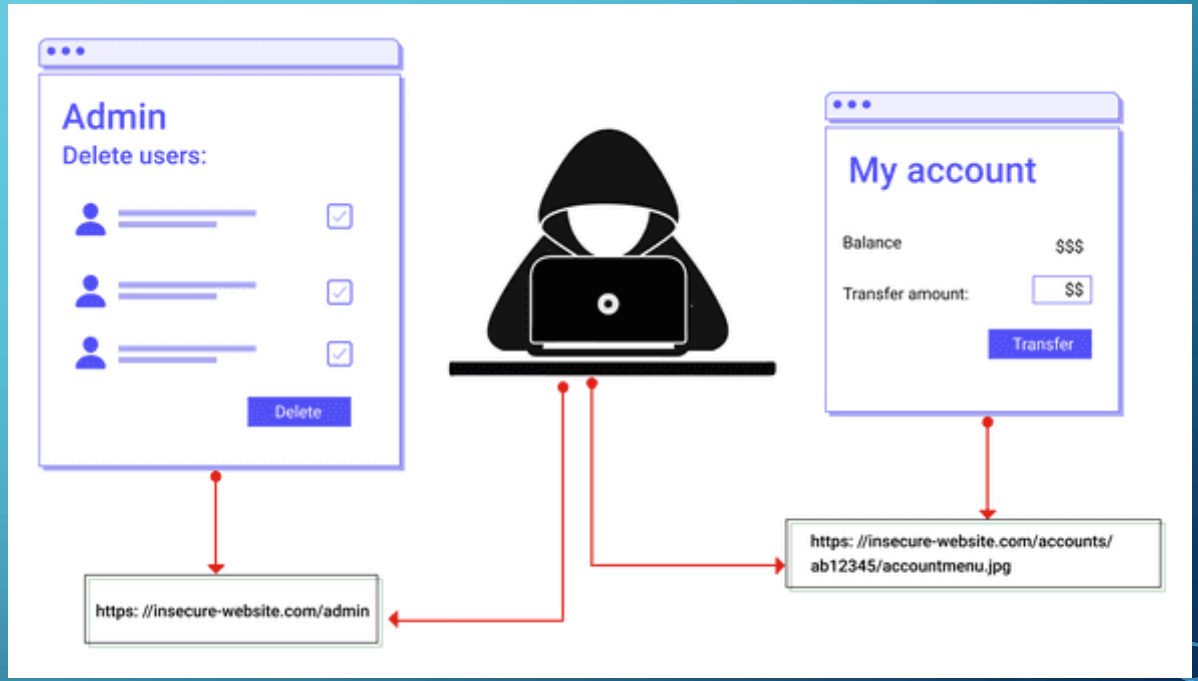
1. BROKEN ACCESS CONTROL

Mecanisme de sécurité -> restreindre l'accès

Elévation de privilèges

Désactiver le listing des dossier sur le server web -> htaccess

Activer le contrôle d'accès sur les rôles et permissions



POST-SCRIPTUM EN APARTÉ ET SANS DISCRETION BREF CONCLUSION

c'est un jeu de mot

- Résumer des différentes catégories par l'owasp
- Redondance de donnée dans les catégories -> arrachage de tête pour résumer sans dupliqué
- Renseignements pour les admin/dev -> problèmes existants -> Correction

SOURCE

- OWASP: https://owasp.org/Top10/
- Crash-test security: https://crashtest-security.com/category/vulnerability-prevention/
- La sécurité d'un site web: https://developer.mozilla.org/fr/docs/Learn/Server-side/First_steps/Website_security
- Geeksforgeeks.org: How to Prevent Broken Access Control? - GeeksforGeeks OWASP Top 10 Vulnerabilities And Preventions - GeeksforGeeks https://www.geeksforgeeks.org/what-is-cross-site-scripting-xss/
- Portswigger.net: https://portswigger.net/web-security

- https://zestedesavoir.com/articles/232/les-failles-xss/
- https://www.leblogduhacker.fr/se-proteger-de-l-injection-sql/
- https://www.vaadata.com/blog/fr/comprendre-la-vulnerabilite-web-server-side-request-forgery-1/
- https://www.carnetdebord.info/owasp-mieux-proteger-applis-web-menaces-identifiees/#Des_controles_drsquoacces_casses
- https://www.owasp-risk-rating.com/
- https://www.dailysecurity.fr/server-side-request-forgery/