OWASP Top 10 2021

Sujets

- Sujets Généraux
  - Qu’est-ce que le Top 10
  - Méthodologies
  - Le Top 10 (et les changements depuis 2017)
- Mon opinion
  - Les changements du Top 10 sont bien alignés avec les changements du risque dans mon entreprise

Qu’est-ce que le Top 10 d’OWASP ?

OWASP

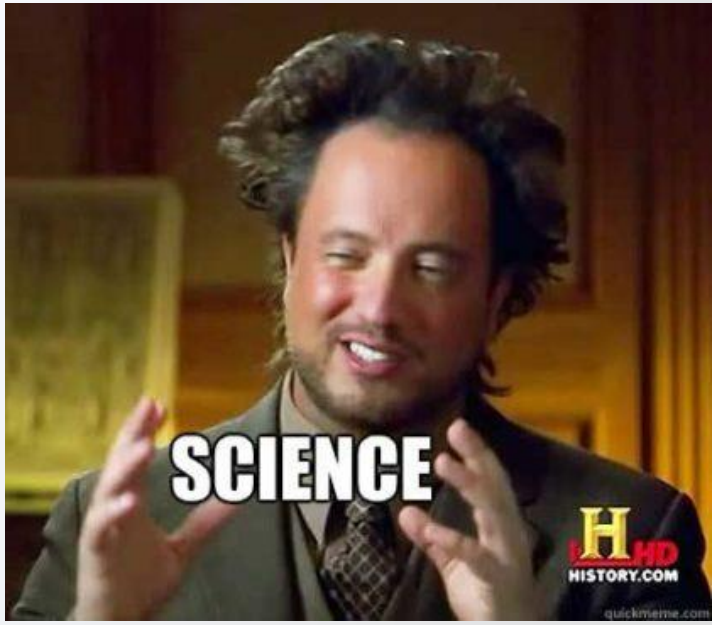
“The Open Web Application Security Project® (OWASP) is a nonprofit foundation that works to improve the security of software. Through community-led open-source software projects, hundreds of local chapters worldwide, tens of thousands of members, and leading educational and training conferences, the OWASP Foundation is the source for developers and technologists to secure the web.”

Qu’est-ce que le Top 10

- Publié en 2003 pour la première fois et mise à jour au 3-4 ans
- Top 10 en ordre de risque
- Basé sur des données réelles (8) et sur les votes de la communauté (2) pour conserver son efficacité au fil des année
- https://owasp.org/Top10/

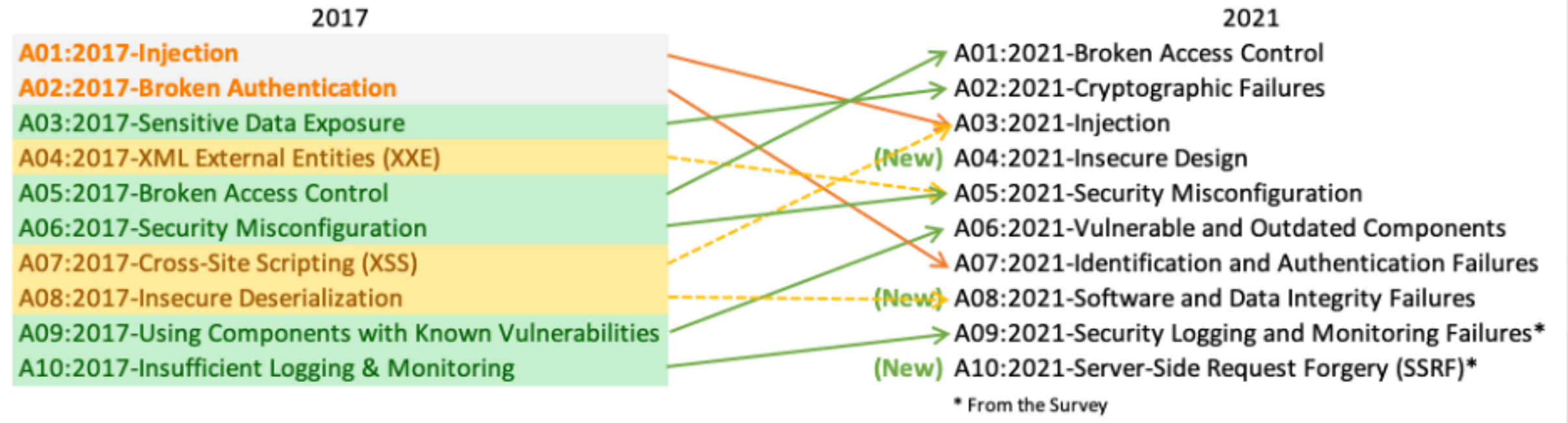
Méthodologie

1. Demander des données à la communauté
   - Reçu ≈500,000 rapports (scan automatique et Pentest manuel)
2. Normaliser les données
   - E.g. il y a moins de données manuelles qu'automatique
3. Mettre les CWEs des vulnérabilités en groupe
4. Mesurer la fréquence de chaque groupe (probabilité) dans les rapports
5. Mesurer les impacts (exploitabilité) en utilisant le CVSS vector pour le CVE de chaque groupe
6. Calculer le risque (probabilité X impact) et prendre les 8 premiers groupes
7. Analyser les votes de la communauté, prendre les 2 plus populaire qui ne sont pas dans les 8 groupes qui proviennent des données



Top 10 2021 et changements depuis 2017

De 2017 à 2021



Orange = Moins important qu'en 2017
Jaune = combinaison avec un autre élément
Vert = Plus haute importance qu'en 2017

Notes

- J’ai raccourci les « includes » d’OWASP pour que ça entre dans la présentation
- Je n’ai pas tout traduit
- Les « includes » d’OWASP ne sont pas consistent : parfois ce sont des vulnérabilités, parfois ce sont des mécanismes, etc.

A01:2021-Broken Access Control

- Sommaire : Problème d’implémentation du contrôles des accès
- Includes :
  - Violation of the principle of least privilege or deny by default.
  - Bypassing access control by URL or DOM tampering.
  - Insecure direct object references.
  - Accessing API with missing access controls for POST, PUT and DELETE.
  - Elevation of privilege.
  - Tampering with JWT access control token, or a cookie or hidden field.
  - CORS misconfiguration
  - Force browsing to authenticated pages as an unauthenticated user.

A02:2021-Cryptographic Failures

- Sommaire : L'ancien nom était « sensitive data exposure ». C'était plus clair.
- Includes :
  - Data transmitted in clear text.
  - Crypto keys checked into source code repositories.
  - Old or weak cryptographic algorithms or protocols used either by default or in older code.
  - Default keys in use, weak crypto keys generated, or improper key rotation mechanism.
  - HTTP headers (browser) security directives missing (e.g. HSTS).
  - Trust chain improperly validated.
  - Ignored initialization vectors or not generated sufficiently secure for the cryptographic mode of operation.
  - Passwords being used as cryptographic keys (without key derivation function).
  - Deprecated hash functions such as MD5 or SHA1, or non-cryptographic hash functions used
  - Cryptographic error messages or side channel information exploitable.

A03:2021-Injection

- Sommaire: Quand une entrée d'un utilisateur change de contexte, des donnée deviennent du code, et que l'exécution est compromise
- XSS était un élément séparé mais a été combiné dans injection
- Includes:
  - User-supplied data is not validated, filtered, or sanitized by the application.
  - Dynamic queries or non-parameterized calls without context-aware escaping.
  - Hostile data used within object-relational mapping (ORM) search parameters.
  - Hostile data is directly used or concatenated. The SQL or command contains the structure and malicious data in dynamic queries, commands, or stored procedures.
  - Common injections are : SQL, NoSQL, OS command, Object Relational Mapping (ORM), LDAP, and Expression Language (EL) or Object Graph Navigation Library (OGNL) injection, GraphQL.

A04:2021-Insecure Design

- Nouvelle catégorie, importante pour « shift-left » et les architectures micro services.
- Description:
  - “A secure design can still have implementation defects”
  - “An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks”
  - “lack of business risk profiling inherent in the software or system being developed, and thus the failure to determine what level of security design is required.”

A05:2021-Security Misconfiguration

- Sommaire: configuration insécure de n’importe quelle composante
- Includes:
  - Missing appropriate security hardening across any part of the application stack.
  - Unnecessary features are enabled or installed (e.g., unnecessary ports, accounts).
  - Default accounts and their passwords are still enabled and unchanged.
  - Error handling reveals stack traces.
  - For upgraded systems, the latest security features are disabled or not configured.
  - The security settings in the application servers, application frameworks (e.g., Struts, Spring, ASP.NET), libraries, databases, etc., are not set to secure values.
    - **“XML External Entities (XXE)” was merged into this one**
  - The server does not send security headers or they are not set to secure values.

A06:2021-Vulnerable and Outdated Components

- #2 dans le vote de la communauté
- Includes:
  - No inventory of the versions of all components you use
    - This includes nested dependencies and dev-dependencies.
  - If the software is vulnerable, unsupported, or out of date.
  - If you do not scan for vulnerabilities regularly and subscribe to security bulletins
  - If you do not fix or upgrade the underlying platform, frameworks, and dependencies in a risk-based, timely fashion.
    - New exploits typically appears in days, not weeks
  - If software developers do not test the compatibility of updated, upgraded, or patched libraries.
  - If you do not secure the components' configurations after upgrade

A07:2021-Identification and Authentication Failures

- Sommaire: n’importe quel problème dans la protection du login ou de l’identité des utilisateurs
- Includes:
  - Brute-force, password-spray or other automated attacks.
  - Default, weak, or well-known passwords, such as "admin/admin".
  - Weak or ineffective credential recovery and forgot-password processes.
    - such as "knowledge-based answers," which cannot be made safe.
  - Plain text, encrypted, or weakly hashed passwords data stores.
  - Missing or ineffective MFA
  - Session identifier in the URL.
  - Session identifier reused after successful login.
  - Sessions or tokens aren't properly invalidated at logout or after inactivity.

A08:2021-Software and Data Integrity Failures

- Nouvelle catégorie pour les « supply chain attack» comme Solarwind
- Includes:
  - Digital signatures to verify the software or data is from the expected source and has not been altered.
  - Libraries and dependencies, such as npm or Maven, are consuming trusted repositories.
  - Supply chain security tool, e.g. OWASP Dependency Check, is used to verify that components do not contain known vulnerabilities
  - Review process for code and configuration changes to minimize the chance that malicious code or configuration could be introduced into your software pipeline.
  - CI/CD pipeline has proper segregation, configuration, and access control to ensure the integrity of the code flowing through the build and deploy processes.
  - Unsigned or unencrypted serialized data is not sent to untrusted clients without some form of integrity check or digital signature to detect tampering or replay of the serialized data
    - **Merge with the old “Insecure Deserialization”**

A09:2021-Security Logging and Monitoring Failures

- #3 des votes de la communauté
- Includes:
  - Events, such as logins, failed logins, and high-value transactions, are not logged.
  - Warnings and errors generate no, inadequate, or unclear log messages.
  - Logs of applications and APIs are not monitored for suspicious activity.
  - Logs are only stored locally.
  - Alerting thresholds and escalation processes are not in place or effective.
  - Penetration testing and scans by dynamic scanners do not trigger alerts.
  - The application cannot detect, escalate, or alert for active attacks in real-time.

A10:2021-Server-Side Request Forgery

- Nouvelle catégorie
- Sommaire: Un hacker force un serveur à envoyer des requêtes quelque part et ces requêtes peuvent être utilisées pour exfiltrer des données ou leurs réponses peuvent confondre le serveur
- N’est pas dans les données mais est #1 dans le vote de la communauté
- Description:
  - “SSRF flaws occur whenever a web application is fetching a remote resource without validating the user-supplied URL. It allows an attacker to coerce the application to send a crafted request to an unexpected destination, even when protected by a firewall, VPN, or another type of network access control list (ACL). As modern web applications provide end-users with convenient features, fetching a URL becomes a common scenario. As a result, the incidence of SSRF is increasing. Also, the severity of SSRF is becoming higher due to cloud services and the complexity of architectures.”

Discussion:
Les changements du Top 10 sont bien alignés avec les changements du risque dans mon entreprise

Est-ce que c’est la même chose chez vous?

Notes

- Essayez de partager vos expériences
- Faites attention de ne pas parler de sujets internes de votre entreprise dont vous ne devriez pas parler en publique
  - Il y a toujours moyen de censurer les détails mais d'avoir une discussion intéressante quand même, faites un effort svp ☺

A01:2021-Broken Access Control

- De #5 à #1 dans le top 10
  - Augmentation marquée de la priorité
- Depuis 2017, notre architecture devient de plus en plus complexe.
  - Au lieu de gros monolithe on a plus de (micro-) services
    - Donc plus de « trust boundaries », plus d'acteurs et plus de workflow, etc.
- Plus il y a d'éléments, plus il y a de chance d'avoir des oublis.
  - E.g. une équipe va assumer qu'un autre équipe fait une vérification d'autorisation
- C'est très important de passer en détails tous nos workflows, au travers de toutes les équipes et tous les composantes et de s'assurer qu'il n'y a pas des gaps d'autorisation

A02:2021-Cryptographic Failures

- De #3 à #2
- Depuis 2017, des nouvelles technologies ont gagné beaucoup en popularité, tel que Kubernetes, qui complique la protection des secrets
- Dans la plupart des entreprises, la cadence des releases a augmenté et le besoin d'automatisé le control des secrets aussi
- Le flow des secrets à partir dans un environnement DevOps, à partir des systèmes de builds et des vaults vers les backends est devenu plus compliqué.
- Il y a plusieurs options pour protéger les secrets dans ses flows mais il n'y a rien de parfait, il y a des pour et des contre pour chaque méthode.

A03:2021-Injection

- De #1 à #3
- Les nouveaux frameworks populaire rendent les injections plus difficile
- On utilise moins de DB classique et plus de NoSQL et de SQLite qui sont plus difficile à injecter.
- Le risque est moindre, mais est quand même très important à la position 3

A04:2021-Insecure Design

- De non-existent à #4
- Explosion d'architecture de micro services, Kubernetes, Electron app, etc.
- La complexité et l'ennemi de la sécurité
- Example : World load balancer, Regional APIM, NGINX ingress/egress, Kuberneters router (network policy), pod load balancer, host TCP stacks, etc.
  - E.g. CORS dans une chaine comme ça, chaque couche peut ignorer les headers, passer les headers, modifier les headers ou bloquer les headers…
- On a besoin de bonnes architectures et de bon design pour pouvoir faire un bon threat-model
- « A perfect implementation doesn't solve design flaw »

A05:2021-Security Misconfiguration

- De #6 à #5
- Pour les gens qui font du AppSec pure, le hardening de réseau et de nodes à faire dans un cluster de Kubernetes est nouveau et il y a plus de chance d'avoir des problèmes de configuration
- Un exemple commun avec Electron c'est un XSS qui devient un RCE quand Electron est mal configuré
  - nodeIntegration: true

A06:2021-Vulnerable and Outdated Components

- De #9 à #6
- Le nombre de CVEs découverts chaque année augmente et le nombre de dépendances dans notre code augmente aussi chaque année… on a donc une augmentation exponentiel de CVEs dans notre code
  - Notre code est de plus en plus comme un iceberg: on code un petit bout mais on s'assoit sur des tonnes de dépendances
- On doit être meilleur à automatiser la détection et plus vite a corriger nos logiciels
  - OWASP fait Dependency Check qui est bon pour le code mais ne couvre pas tout
- Inclut les dépendances des outils de développement aussi !

A07:2021-Identification and Authentication Failures

- De #2 à #7
- Nous on est passé d'un système développé à l'interne vers un service, alors notre profile de risque est aligné avec ce qui est arrivé dans le Top 10
  - https://www.pingidentity.com/en/customer-stories/bentley-systems-3546.html

A08:2021-Software and Data Integrity Failures

- De non-existent à #8
- Inclue la sécurisation des chaines de CI/CD
- On a aussi de plus en plus de produits qui font des releases très rapide alors ils doivent automatiser, et sécuriser, leur pipeline
- Il y a eu plusieurs attaques de « supply-chain » dans la dernière année et différent sous-type :
  - SolarWind
  - Colonial Pipeline
  - UA-Parser-JS
- **Aimeriez-vous que je fasse une présentation sur les sous-types de supply-chain attacks dans le future?**

A09:2021-Security Logging and Monitoring Failures

- De #10 à #9
- C’est aussi plus important pour nous cette année qu’en 2017
  - Plus de monitoring et plus d’analyse des logs de sécurité

A10:2021-Server-Side Request Forgery

- C’est vrai qu’il y a un retour du « périmètre » qui était disparu dans les dernières années et un SSRF permet de s’amuser à l’intérieur
- Un SSRF dans un pod qui n’est pas bien isolé permet d’avoir des impacts sur d’autres pods.
- Voyez-vous d’autres raisons pourquoi les SSRF ont été le vote #1 ?

Questions?