acceis

SECURITE APPLICATIVE

Yves Duchesne

yves@acceis.fr

Sécurité applicative ?

La sécurité, c’est quoi ?

Une application sécurisée, c’est une application qui :

- fait ce qu’elle dit
- protège ses données sensibles
- implémente un contrôle d’accès robuste
- ne comporte pas de vulnérabilité technique

La sécurité, c'est quoi ?

Il y a les **mécanismes de sécurité** :

- **Authentification**
- **Contrôle d'accès**
- **Cryptographie** (…)

… et la **sécurité des mécanismes**

- **Vulnérabilités techniques**
- **Mauvaises configurations** (…)

L’insécurité, c’est quoi ?

Les défauts dans les mécanismes de sécurité :

… peuvent être identifiés par des tests fonctionnels

- Défaut de contrôle d’accès
- Biais d’authentification

… ou pas

- Cryptographie mal implémentée
- Défauts de protection des sessions (…)

L’insécurité, c’est quoi ?

Des défauts de sécurité dans l’implémentation des mécanismes, en revanche sont (à peu près) **indétectables par des tests fonctionnels :**

- Injections SQL
- XSS
- CSRF
- XXE
- File Upload/Download/inclusion
- Etc.

L'insécurité, c'est quoi ?

Parfois même les problèmes de sécurité identifiés lors de campagnes de tests passent inaperçus

- « *j'obtiens une erreur SQL quand je teste l'application avec le profil de Jean d'Ormesson* » (marche aussi avec « l'agence tous risques »)
- « *C'est un bug* »

➢ Non. C'est une **vulnérabilité** !

L’insécurité, c’est quoi ?

En somme, une application peut être

- Bien écrite
- Fonctionnelle
- Performante

… mais **vulnérable**

Webapp
Security
Audit
your
code
Damn i wish i could read

~~L’audit de sécurité~~

Les audits de sécurité

Un audit, c’est quoi ?

Un audit de sécurité, c’est avant tout un audit

- C’est donc un **état des lieux**
- Sur une **cible**
- Vis-à-vis d’un **référentiel**

L’objectif : Evaluer le degré de conformité d’une cible à un référentiel et faire l’inventaire des non-conformités

Comment ça se passe ?

Un audit, c'est donc un « **contrat d'intervention** »

- Un peu comme une interface en Java

Il peut être mené de différentes manières.

Il doit simplement respecter le contrat : **permettre d'identifier les non-conformités à un référentiel**

Comment ça se passe ?

Traditionnellement, on utilise une terminologie tirée de la théorie des systèmes pour désigner les approches d'audit

- **Boîte blanche** : on connaît le fonctionnement interne du système
- **Boîte noire** : pas

On a également une voie médiane : la **boîte grise**.

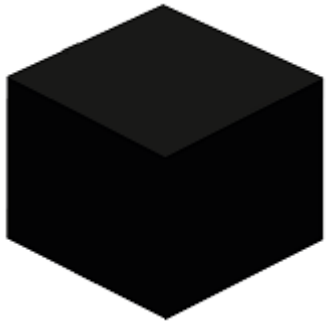
Comment ça se passe ?

Il y a deux erreurs dans la réalisation d'un audit

- Travailler **uniquement** avec des outils automatisés
- Travailler **sans** outil automatisé

L'une comme l'autre mène à un audit **incomplet**

La boîte noire



L'audit en boîte noire, c'est le **test d'intrusion**

- Le « fameux » **pentest**

Objectif :

- Simuler le comportement d'un hacker
- Mener des attaques sur l'application pour identifier les vulnérabilités

La boîte noire

L'outil principal, c'est l'expertise et l'expérience de l'auditeur

- Chaque situation est différente
- On travaille « à l'aveugle », en inférant le fonctionnement de la cible

➢ 80% du boulot, c'est de comprendre son fonctionnement

La boîte noire

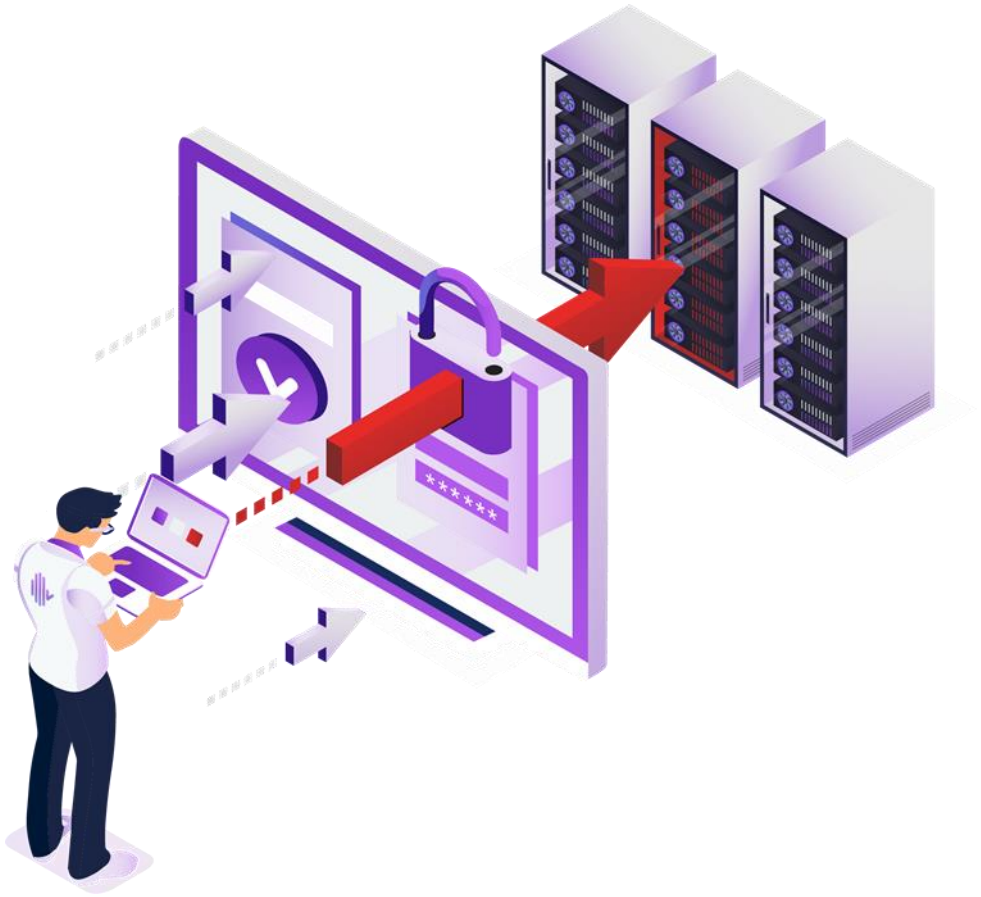
En appui de l'auditeur : l'outillage spécialisé

- **Burp** : un proxy d'interception HTTP, qui inclut de nombreux modules
- **sqlmap** : un outil d'exploitation d'injections SQL
- **Acunetix** : un scanner de vulnérabilités
- Et bien d'autres…

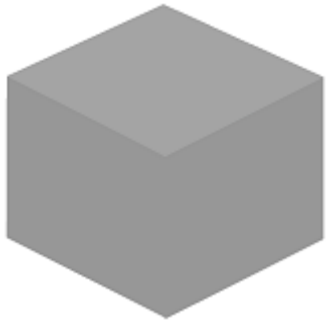
La boîte noire

Le déroulement d'un **test d'intrusion** :

- :1
- Prise d'empreinte : *fingerprinting*
- Recherche de vulnérabilités
- Exploitation des vulnérabilités
- GOTO 1



La boîte grise



L'audit en boîte grise, c'est aussi le **test d'intrusion**

- Même méthode
- Mêmes objectifs
- Même approche offensive
- Mais les auditeurs disposent de comptes d'accès à l'application
- Ils peuvent ainsi auditer l'ensemble du périmètre fonctionnel

La boîte blanche

L’audit en boîte blanche c’est **l’audit de code source**

- Mêmes objectifs : identifier les vulnérabilités
- Méthode radicalement différente : analyse du code source
- Objectif complémentaire : identifier les mauvaises pratiques

La boîte blanche

L'audit de code source s'attachera donc à effectuer un travail plus large

- Analyse de la **qualité du code source**
- Analyse des **bonnes pratiques de sécurité** des développements
- Etude de la **dette technique**
- Eventuellement, avis d'expert sur le **profil technologique** de la cible

La boîte blanche

Comme pour le pentest, il faut :

- Un **auditeur expérimenté**

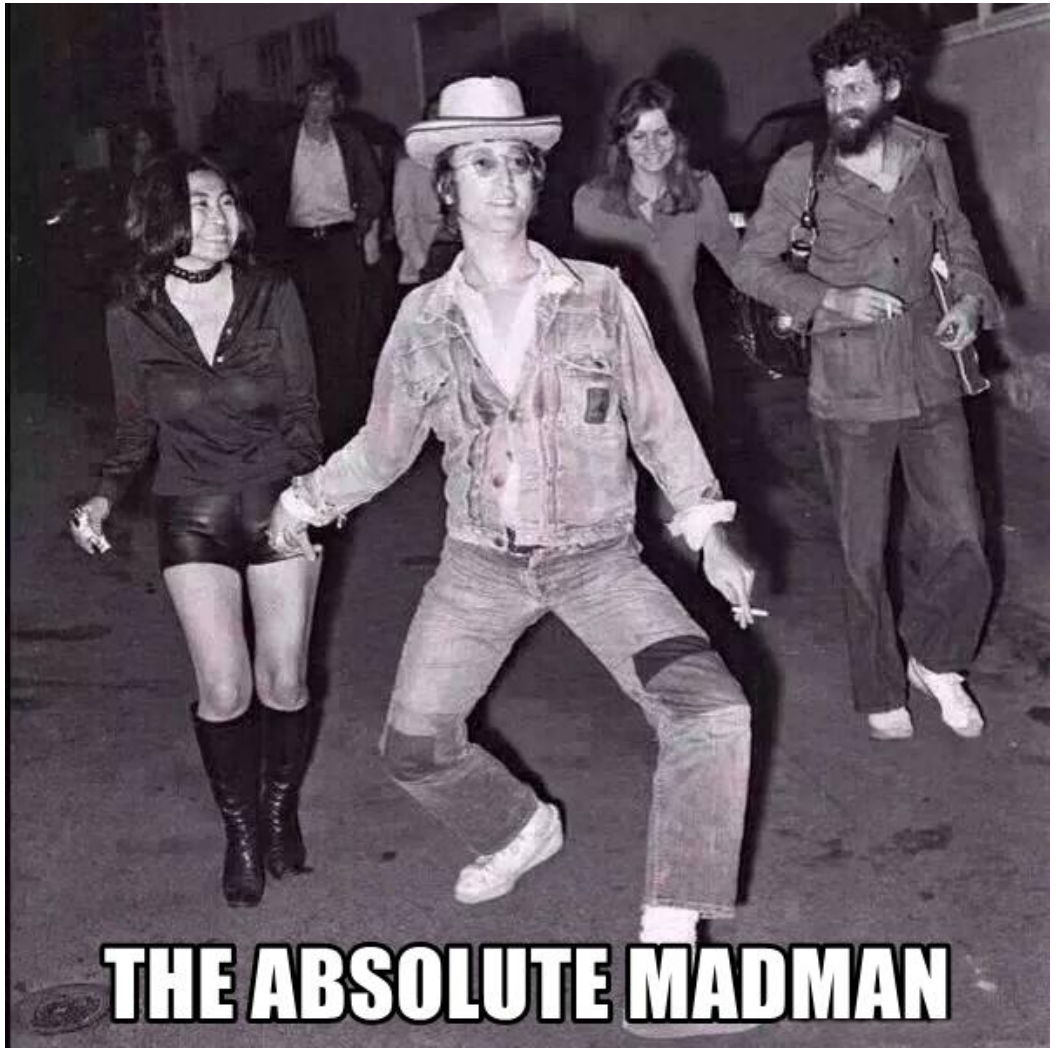
➢ L'exercice est difficile et cette expertise est peu commune

Des outils

- Un bon IDE
- Eventuellement, un scanner de vulnérabilités
- Des outils complémentaires (indicateurs, complexité, etc.)

L’approche croisée

Il est même possible de croiser ces approches



Comparaison

Le test d'intrusion

- ✓ Approche Pragmatique
- ✓ Efficacité (Bon ROI)
- ✓ Bonne qualification des impacts
- ✗ Travail de surface

L'audit de code source

- ✗ + coûteux
- ✗ Mauvaise qualification des impacts
- ✓ Travail en profondeur
- ✓ Travail plus large

L'approche croisée

- ✓ Absolute madman

Le piège de la boîte noire

On peut être tenté d'opter pour un test en boîte noire

- « C'est un meilleur indicateur »
- « Pour savoir ce qu'un attaquant peut faire »
- C'est (souvent) une erreur
- Au final cela revient à gêner l'audit

La stratégie de sécurité

Comment procéder ?

Il faut **définir une stratégie en matière de sécurité applicative**

- On audite quoi ?
- On audite comment ?
- On audite quand ?

Comment procéder ?

On audite quoi ?

Il n’est pas nécessaire d’auditer toutes les applications.

- Un site web institutionnel hébergé de façon externe

En revanche, certaines applications doivent être auditées :

- Hébergement interne
- Fonction(s) sensible(s)
- Données sensibles

L’approche par les risques

Il faut donc déterminer ce qu’on audite ➔ **Comment ?**

- Selon quels critères ?
- La meilleure approche est par **les risques**
- On déclenche en fonction des risques portés par l’application

L’approche par les risques

C’est une démarche centrale en cybersécurité : l’analyse des risques

- Lister les **biens sensibles** (données, fonctions)
- Lister les **sources de menace**
- **Etablir des scénarios de risque** en croisant ces éléments
- **Attribuer une gravité**, dépendant
  - De la probabilité d’occurrence
  - De l’impact

L’approche par les risques

Ce processus repose sur une **matrice des risques**

		Impact			
		Négligeable	**Mineur**	**Important**	**Critique**
Probabilité	**Improbable**	Bas	Bas	Bas	Significatif
	Possible	Bas	Significatif	Elevé	Elevé
	Probable	Significatif	Significatif	Elevé	Critique
	Certain	Significatif	Elevé	Critique	Critique

Et la stratégie ?

En face de ces risques, il faut mettre des stratégies.

On ne gère pas la sécurité d'une application portant un risque critique de la même façon que la sécurité d'une application portant un risque négligeable

- Mais comment gère-t-on la sécurité des développements ?
- C'est quoi une stratégie de sécurité ?

Et la stratégie ?

Comme toujours en cybersécurité, la réponse est multiple

La cybersécurité repose sur **4 piliers** :

- Organisationnel
- (Juridique)
- Humain
- Technique

L’aspect organisationnel

La cybersécurité est **avant tout** une organisation et des procédures.

- Référentiels
- Responsables / référents
- Procédures
- Matrices des risques
- Etc.

L’aspect technique

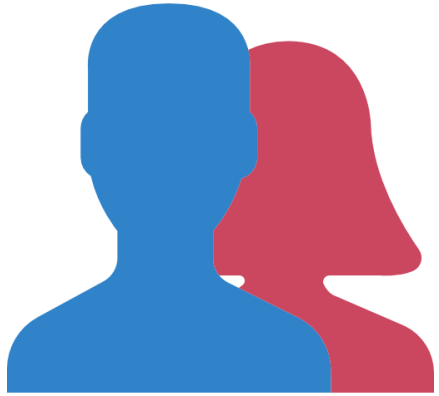
La technique constitue une brique fondamentale de la sécurité au sein des développements

- **Audits de sécurité**
- **Outillage des développeurs** : plugins d’analyse de code
- **Outillage de l’intégration continue** (ex. SonarQube)
- **Outillage du delivery** (ex. scanner de code source)
- Etc.

L’aspect humain

Enfin, les aspects humains sont primordiaux

- Formation des développeurs
- Création / animation d’une communauté
- Piqûres de rappels



Et la stratégie ?

L'erreur classique : se baser sur un seul levier

- Outiller les développeurs sans les former
- Installer un outil de détection sans organiser son utilisation
- Installer des outils sans référentiel pour les configurer
- Etc.

Les référentiels

Un problème récurrent en audit de cybersécurité concerne les référentiels utilisés

Un audit classique se base sur un référentiels très arrêté

- Ex. audit de conformité ISO/CEI 27.002:2013

En sécurité offensive, de nouvelles vulnérabilités ou techniques d'attaque peuvent voir le jour régulièrement

- On travaille sur « l'état de l'art »

Les référentiels

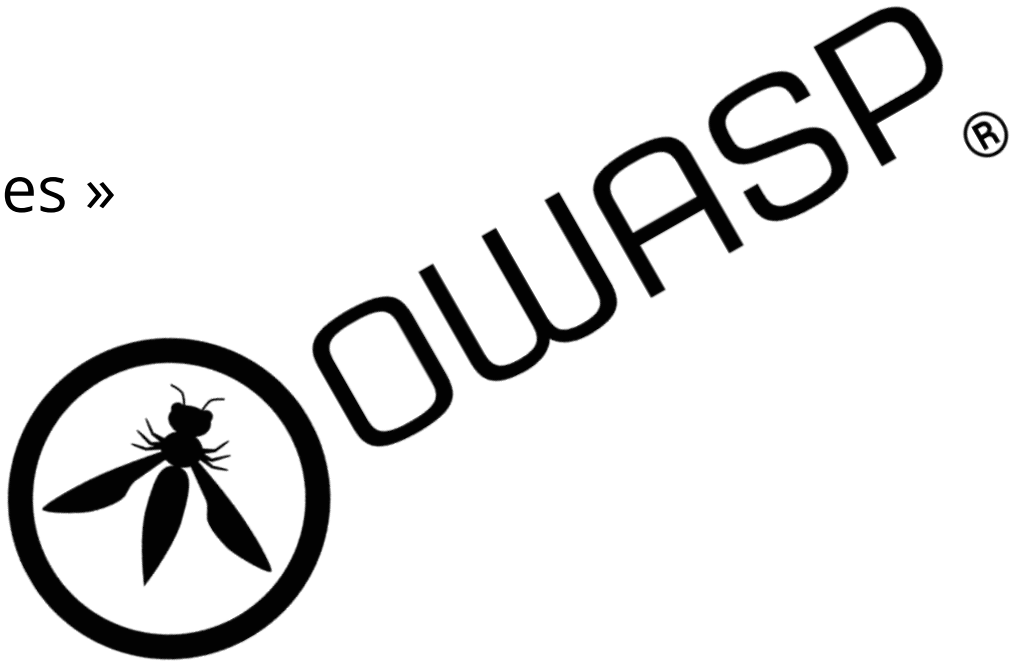
Néanmoins, des référentiels existent sur la sécurité applicative

- OWASP TOP 10
- Mitre CWE TOP 25

Les référentiels

OWASP TOP 10

- Open Web Application Security Project
- Les « 10 vulnérabilités les plus communes »
- Pas très bien fait
- Et surtout incomplet
- Ce N’EST PAS un référentiel d’audit



Les référentiels

Mitre CWE TOP 25

- Common Weakness Enumeration
- Les « 25 principales faiblesses logicielles »
- Plus cohérent
- Néanmoins assez orienté bas niveau
- Il existe énormément de CWE !

acceis

Merci de votre attention.

Yves Duchesne
yves@acceis.fr