**Rapport de séminaire**

**Web security**

Présenté par M. Valentin Lecomte

1. Introduction

Présentation de NVISO, des équipes de travail

1. Security in the News

Les grandes entreprises essaient d’éviter la diffusion d’informations sur les failles qui ont été exploitées.

1. Fondamentaux

Bugs vs failles :

* Bugs : Injections SQL, XSS, redirection (phishing), injection de commandes (commandes exécutées sur le serveur, violation de vie privée.
* Failles : Authentification manquante, se reposer sur la validation côté client, créer sa propre librairie de cryptographie, authentification mais sans autorisation, management des erreurs inconsistantes, violation de vie privée.

~50% de bugs et failles. Il est donc important de faire attention à TOUT quand on développe / teste une application.

Côté client vs serveur :

* Ne jamais faire confiance aux données envoyées par un utilisateur. Tous les uploads, update de données, etc.

1. OWASP Top Ten

* Organisme open source développant des projets pour la sécurité.
* Lutte contre les injections, le XSS, le passage d’informations par headers XML, la configuration non sécurisée du serveur, le logging insuffisant, authentification cassée, exposition de données sensibles, etc.
* Top 10 des failles les plus courantes, mises à jour tous les ~3 ans.

Les attaques arrivent à tous les niveaux, on peut injecter du code dans des entrées selon la requête :

* SQL
* LDAP
* PATH
* SMTP
* XML
* HTML
* JS

1. Validation de données

Il ne faut **JAMAIS** modifier la valeur d’un champ après l’avoir validé.

Il faut faire de l’encodage de données afin d’éviter l’injection de code (à tous les niveaux).

Ne jamais concaténer les informations de login avec la requête SQL -> on peut alors juste demander un username administrateur et mettre le reste de la requête en commentaire => retourne un token avec la sessionID de l’admin.

1. Cross-site Scripting

Injecter du code HTML et JavaScript malicieux côté client. Et changer le comportement du code.

* Reflective XSS : pas stocké serveur, événement lorsqu’on clique / interagit.
* Persistent XSS : stockage niveau serveur, tous les utilisateurs seront affectés par le code.

Protéger l’application grâce à l’input validation et l’output encoding

1. Access Control

Vérifier les contrôles d’accès pour éviter d’avoir accès à des liens.

1. Partie pratique

Recherche et exploitation de failles sur Juice Shop OWASP