SR03 – TD

Sécurité des applications web

Qu'allons-nous faire durant ce TD ?

1. Préparer un mini-projet qui servira d'étude de cas aux différents mécanismes abordés
2. Tester la vulnérabilité de votre site vis à vis de quelques exemples de failles de sécurité
3. Appliquer les correctifs nécessaires pour sécuriser votre site

Ce TD n'est pas noté et ne donne pas lieu à une soutenance. **Toutefois**, votre présence est OBLIGATOIRE et vous devrez faire valider le travail demandé par le chargé de TD avant de quitter la salle en fin de séance.

A. Votre mini site web

Vous allez mettre en place une petite application web à destination des clients et des employés d'une banque. Les pages seront développées en PHP et votre base de données sera implémentée sous MySQL.

Pour conserver l'intérêt pédagogique du TD, vous n'utiliserez aucun framework CSS, Javascript, PHP, etc...

1. Environnement de travail

Installez sur votre poste les outils nécessaires pour développer en PHP et travailler avec une base de données MySQL, de préférence : WAMP (windows), LAMP (linux) ou MAMP (mac).

2. Base de données

Créez une base nommée **`sr03`** et ajoutez les tables suivantes dans cette base :

- USERS :
  - id_user
  - login
  - mot_de_passe
  - profil_user ('CLIENT', 'CONSEILLER')
  - nom
  - prenom
  - numero_compte
  - solde_compte

- MESSAGES :
  - id_msg
  - id_user_to (identifiant du destinataire du message)
  - id_user_from (identifiant de l'expéditeur du message)
  - sujet_msg,
  - corps_msg

3. Mettez en place les pages suivantes :

- **login.php** : page comportant les champs 'login'et 'password' et un bouton de connexion
- **accueil.php** : page sur laquelle arrive un utilisateur qui a réussi à se connecter; on affiche sur cette page toutes les informations de l'utilisateur connecté; on dispose également des liens suivants sur la page : *messagerie* (tous les profils), *effectuer un virement* (tous les profils), *fiche client* ('CONSEILLER')
- **messagerie.php** : page qui liste tous les messages reçus par l'utilisateur connecté et qui permet de lire un message reçu ou envoyer un message à n'importe quel utilisateur
- **virement.php** : page qui permet d'effectuer un virement d'un compte vers un autre; cette page reçoit en paramètre le numéro de compte qui sera débité. Attention : un client ne peut effectuer un virement que de son propre compte vers un autre compte.

- **ficheClient.php** : page qui donne accès à liste de tous les clients; quand un client est sélectionné dans la liste, on affiche toutes ses informations ainsi qu'un lien *effectuer un virement* qui appelle `virement.php` avec le numéro de compte de ce client en paramètre

Dans cette partie, vous ferez abstraction de l'aspect sécurisation : développez vos pages en vous souciant essentiellement de la bonne conception des fonctionnalités demandées.

Vous trouverez sur Moodle un canevas de projet MVC écrit en PHP : sr03.zip
N'hésitez pas à utiliser ce canevas si vous n'êtes pas suffisamment à l'aise en PHP pour réaliser rapidement cette partie.

B. Etude de quelques failles de sécurité

Voici une liste de failles courantes sur les applications WEB (et présentées en cours) :

- **Injection**
- **Cross-site scripting (XSS)**
- **Violation de gestion d'authentification**
- **Violation de gestion de session**
- **Référence directe non sécurisée**
- **Falsification de requête (CSRF)**
- **Vulnérabilité d'un composant**
- **Stockage non crypté**
- **Accès aux répertoires par HTTP**
- **Scripts de redirection**

1. Etudier la vulnérabilité de votre site par rapport à chacune de ces failles de sécurité.
2. Quand cela est possible, présenter un exemple de scénario d'attaque complet (vous devrez éventuellement concevoir un faux site pirate, des mails frauduleux, etc…).

C. Sécuriser votre site

Pour toutes les failles détectées dans la partie B, proposez un correctif en vous inspirant des bonnes pratiques de développement vues en cours.