Adrien Peguiron
Stefan Simeunovic

STI – Analyse de menace

Table des matières

1. Description du système

a. Objectifs du système

- Permettre aux utilisateurs de communiquer par message
- Réputation : les messages doivent arriver au destinataire voulu et être visibles uniquement par le/les destinataires.

b. Hypothèses de sécurité

- Réseau interne et administrateurs de confiance
- Système d'exploitation et serveur Web de confiance

c. Exigences de sécurité

- Le site Web doit être disponible à 99% du temps (disponibilité)
- Les informations des utilisateurs doivent être scrupuleusement protégées (privacy)
- Les informations accédées par les utilisateurs ne doivent pas pouvoir être tracées (privacy)
- Seul un administrateur doit pouvoir ajouter/modifier/supprimer un utilisateur
- Les mails reçus par un utilisateur ne doivent être visibles que pour lui (privacy)

d. Eléments du système

- Base de données des utilisateurs
- Base de données des messages
- Application Web

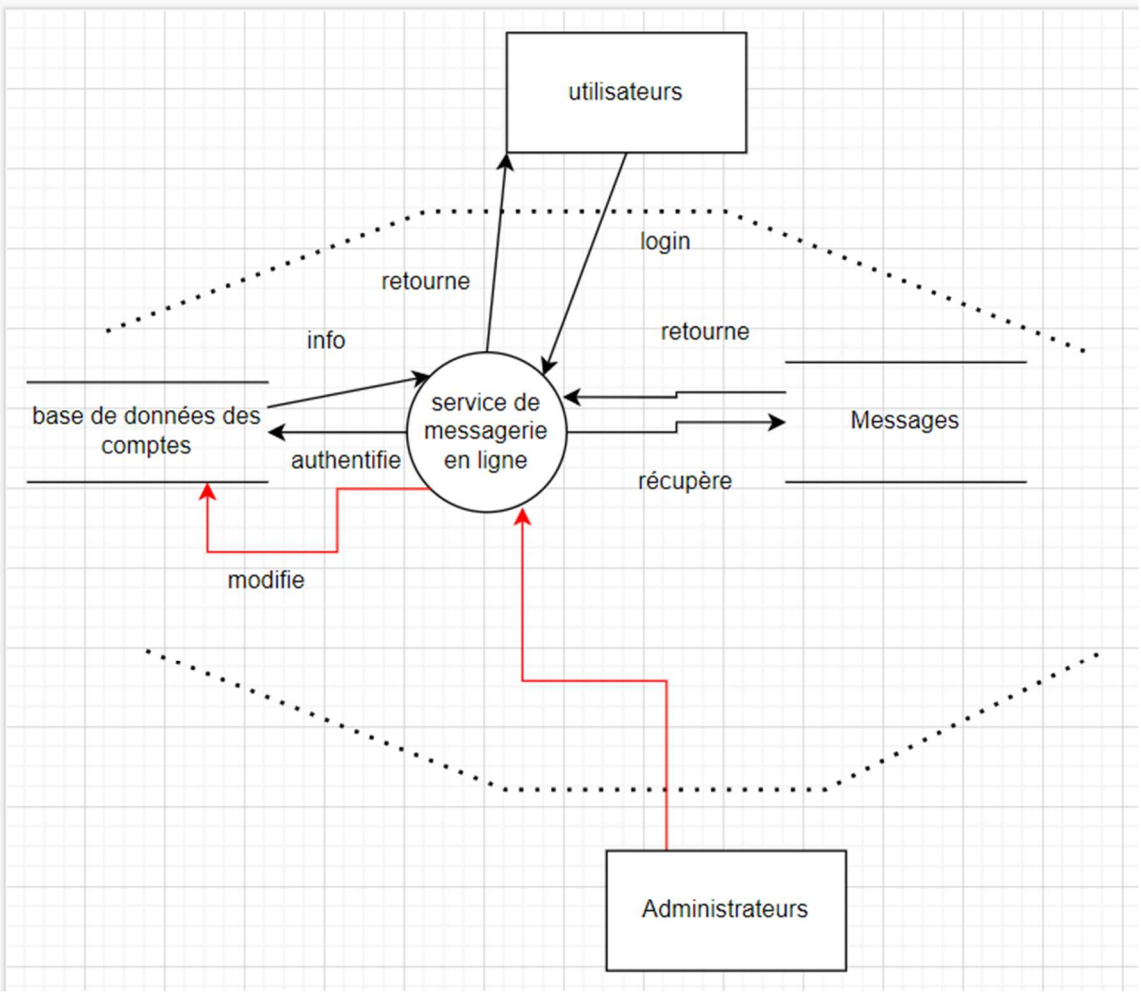
e. Rôles des utilisateurs

- Collaborateur
- Administrateur (même droits que collaborateurs + options administrateur)

f. Actifs à haute valeur :

- Base de données des utilisateurs (données)
  - Confidentialité, sphère privée
  - Un incident nuirait à la réputation du site
  - Plainte possible et perte financière
- Base de données des messages (données)
  - Confidentialité (messages privés entre collaborateur)
  - Intégrité (une modification impliquerait une baisse de réputation)
- Infrastructure
  - Intégrité, disponibilité
  - Un incident serait critique et nuirait à la disponibilité/réputation

g. DFD



2. Identification des sources de menaces

a. Sources de menace

- Hackers, script-kiddies
  - Motivation : s'amuser, gloire
  - Cible : n'importe quel élément / actif
  - Potentialité : haute
- Cybercrime (spam, maliciels)
  - Motivation : récupération d'information / nuire aux utilisateurs
  - Cible : vol de credentials des utilisateurs, spam des utilisateur, modification / récupération d'informations
  - Potentialité : haute
- Concurrent
  - Motivation : détruire la réputation de l'entreprise, espionnage industriel.
  - Cible : base de données des utilisateurs / base de données des messages
  - Potentialité : Moyenne
- Employé malicieux
  - Motivation : élévation de privilèges, lecture de messages de collaborateurs
  - Cible : Base de données des messages, compte administrateur
  - Potentialité : faible

3. Identification des scénarios d'attaque

a. Scénario 1 : Vol de compte utilisateur

- ***Business impact*** : Haut (réputation, acquisition d'informations confidentielles, usurpation d'identité)
- ***Sources de menace*** : Toutes
- ***Motivations*** : espionnage industriel (accès aux messages), curiosité d'un utilisateur de l'application, criminalité (chantage pour soutirer de l'argent, vente infos sur darkweb, …), avoir plus de droits pour un utilisateur (avoir accès à compte admin)
- ***Biens ciblés*** : Base de données des utilisateurs ou identité d'un utilisateur particulier
- ***Scénarios d'attaque*** : Mots de passe faible (brute force), session hijacking, injections SQL, injections commandes OS pour dumper la base, vol credentials d'un utilisateur/admin

b. Scénario 2 : Attaque de type injection SQL

- ***Business impact :*** Haut (réputation, acquisition d'informations confidentielles, suppression/modification de compte utilisateurs et mots de passe)
- ***Sources de menace :*** Hacker, cybercrime, criminel infiltré dans l'entreprise (physiquement ou ayant accès une machine du LAN)
- ***Motivations :*** Rendre le site indisponible, accès aux messages (espionnage industriel, utilisateur curieux, vol de données), compromettre l'intégrité des données (notamment le contenu des messages), récupérer des infos stockées dans la base mais dont l'application n'a pas directement accès (mots de passe, …)
- ***Biens ciblés :*** base de données des utilisateurs et des messages.
- ***Scénarios d'attaque :*** injection SQL

c. Scénario 3 : Attaque de type XSS

- ***Business impact :*** Moyen (réputation, récupération du contenu de cookies)
- ***Sources de menace :*** Toutes
- ***Motivations :*** exécuter un script chez la victime.
- ***Biens ciblés :*** Navigateur de la victime.
- ***Scénario d'attaque :*** injection de script dans un message. Lors de son affichage, le script va s'exécuter.

d. Scénario 4 : Attaque de type CSRF

- ***Business impact*** : Moyen (usurpation d'identité)
- ***Source de menace :*** Hacker, cybercrime
- ***Motivations*** : Faire effectuer à la victime une action non désirée sur un site
- ***Biens ciblés :*** Navigateur de la victime, ses données utilisateurs
- ***Scénario d'attaque :*** Requête qui pointe vers le site de messagerie forgée depuis un site malicieux

e. Scénario 5 : Elévation de privilèges

- ***Business impact :*** Moyen (ajout/suppression/modification d'utilisateurs par un collaborateur)
- ***Source de menace :*** Employé malicieux
- ***Motivation :*** Avoir des privilèges administrateur
- ***Biens ciblés :*** Base de données des utilisateurs
- ***Scénario d'attaque :*** Accès aux pages réservées aux admins en mappant le site / devinant le nom de la page (nécessite un accès au site avec un compte utilisateur).

f. Scénario 6 : Contourner l'authentification

- ***Business impact :*** faible (accès aux fonctionnalités d'un collaborateur sans être authentifié)
- ***Source de menace :*** Hacker, concurrence, cybercrime
- ***Motivation :*** Accéder au site sans compte utilisateur
- ***Biens ciblés :*** Application web
- ***Scénario d'attaque :*** Accès aux pages réservées aux utilisateurs loggés en mappant le site / devinant le nom de la page.

g. Scénario 7 : Lecture/suppression de mail sans en être le destinataire

- ***Business impact :*** Moyen (acquisition/suppression de messages d'autres employés)
- ***Source de menace :*** Employé malicieux (nécessite d'être loggé)
- ***Motivation :*** lire ou supprimer les messages des différents collaborateurs
- ***Biens ciblés :*** base de données des messages
- ***Scénario d'attaque :*** Modifier les requêtes http avec un proxy pour lire ou supprimer les messages contenus sur la base de données.

h. Scénario 8 : Contourner la vérification côté client

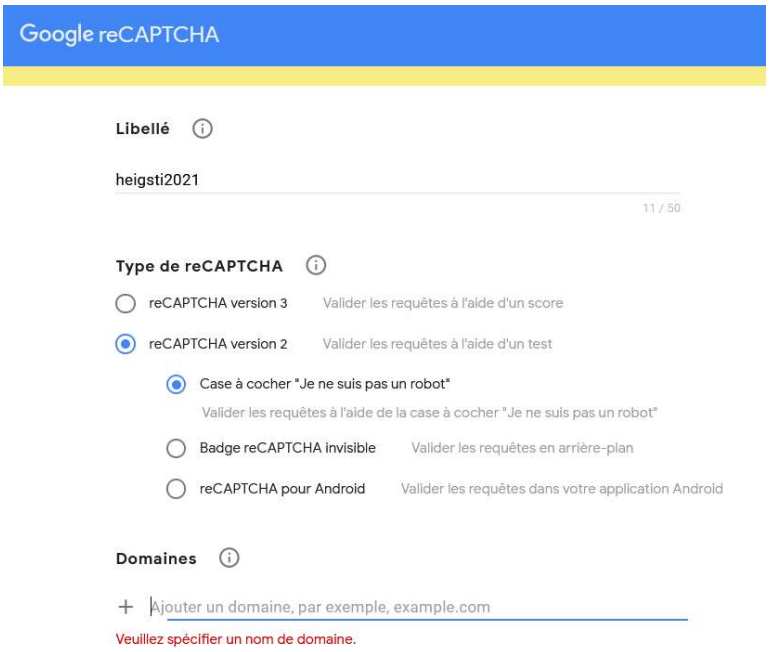
- ***Business impact :*** moyen (utilisateur avec un mot de passe non conforme à la politique de mot de passe)
- ***Source de menace :*** Employé malicieux
- ***Motivation :*** Avoir un mot de passe facile à retenir
- ***Biens ciblés :*** base de données des utilisateurs, application web
- ***Scénario d'attaque :*** Modifier les requêtes http avec un proxy pour envoyer un mot de passe trop faible/pas de mot de passe lors de l'édition de mot de passe.

4. Mitigation des risques

a. Scénario 1 : Vol de compte utilisateur

• Mettre en place des mots de passe fort (la politique appliquée ici est un mot de passe avec au minimum 15 caractères. Cela peu sembler faible mais, même si uniquement des lettres minuscules sont utilisées, cela représente 1,677,259,342,285,726,023,680 combinaisons possibles. Et cela permet à un utilisateur de créer un mot de passe qu'il peut facilement mémoriser et qu'il n'a pas besoin de noter.)
• Utiliser le protocole HTTPS pour éviter le vol de cookies. (Cela n'a pas été mis en place dans ce projet car spécifié dans la consigne comme non nécessaire)
• Mettre en place un système de double identification. Dans le cadre de ce projet, ce n'a pas été possible pour cause de ressources à disposition, mais c'est un moyen efficace dans d'autres circonstances.

• Assigner l'attribut Secure aux cookies de session pour les transmettre uniquement via HTTPS (HTTPS n'ayant pas été mis en place, cela n'a pas été fait.)
• Regénérer l'identifiant de session après une connexion/déconnexion
• Ralentir le processus de connexion utilisateur (avec un captcha)
  ○ Nous avons essayé d'intégrer le captcha de Google sur notre site mais il faut le lier avec un nom de domaine. Nous avons donc omis la mise en place d'un captcha créé par nos soins de A à Z car nous avons jugé ceci trop exigeant en matière de temps.



• Ne pas stocker les mots de passe en clair dans la base

b. Scénario 2 : Attaque de type injection SQL

• Utiliser PDO pour faire des requêtes paramétrées. Cela permet à l'application de pouvoir faire la différence entre le code SQL et les données entrées par l'utilisateur.

c. Scénario 3 : Attaque de type XSS

• Assainir les entrées utilisateurs avec la fonction PHP htmlspecialchars[1]. Cette fonction converti les caractères spéciaux en entité HTML et leur permet d'être affichés normalement. Les caractères doivent êtres assainis dans le corps du message mais aussi dans le sujet.
• Assigner l'attribut HTTPOnly aux cookies de session pour les empêcher d'être récupérés pas un script.

d. Scénario 4 : Attaque de type CSRF

• Utiliser des tokens anti CSRF généré aléatoirement pour empêcher la création de requête valide par un attaquant.

[1] https://www.php.net/manual/en/function.htmlspecialchars

e. Scénario 5 : Elévation de privilèges

• Les credentials administrateurs sont probablement la paire la plus utilisée et celle testée en premier par toute personne souhaitant acquérir un compte administrateur : « admin » comme username et comme password. Le mot de passe doit impérativement être changé par un mot de passe fort et le nom d'utilisateur peut être changé pour rendre une attaque plus complexe.
• À la suite d'un accès à une page réservée à un administrateur, un contrôle doit être effectué afin d'être sûr que l'utilisateur possède bien les privilège requis pour y accéder. Cela permet d'éviter qu'un collaborateur devinant le nom d'une page (par exemple newUser) puisse y accéder en l'entrant dans l'URL.

f. Scénario 6 : Contourner l'authentification

• Mettre en place un contrôle de login sur les différentes pages pour vérifier si l'utilisateur est authentifié

g. Scénario 7 : Lecture /suppression de mail sans en être le destinataire

• Avant d'effectuer une action sur un message, contrôler qu'il appartienne bien à l'utilisateur loggé.

h. Scénario 8 : Contourner la vérification côté client

• Vérifier du côté du serveur si le mot de passe respecte bien les politiques de sécurité en vigueur. Si l'utilisateur arrive à faire passer un mot de passe non valide au serveur, son mot de passe restera inchangé dans la base de données. Il n'est pas nécessaire de vérifier les deux mots de passe. En effet, s'ils ne sont pas identiques, la modification n'a pas lieu.

i. Bonnes pratiques générales valables pour tous les scénarios

• Eviter les messages d'erreur trop verbeux pouvant communiquer des informations sur la structure interne.
• Appliquer le principe du moindre privilège pour tous les droits. Droits utilisateurs, droits sur les fichiers que l'application utilise (ici le fichier .sqlite notamment).
• Ne pas avoir un compte de type « Super administrateur » possédant tous les droits sur l'application. Si un attaquant venait à en prendre possession, il devient alors maître du site.
• Les actions exécutables uniquement par les administrateurs ne sont pas sécurisées contre le contournement des restrictions (par exemple créer un nouvel utilisateur sans username en modifiant la requête http). En effet, les administrateurs sont supposés être de confiance, ce n'est donc pas nécessaire.
• Pour se prémunir des attaques visant les logiciels qui exécutent l'application web (Apache, PHP, MySQL, …), il faut toujours faire l'effort de maintenir leur version à la dernière stable disponible. Dans ce projet, ce n'est pas le cas car l'application tourne dans un container Docker fournit par l'enseignant.

5. STRIDE

Composant	Spoofing	Tampering	Repudiation	Information disclosure	Denial of service	Elevation of privileges
Utilisateur	X		X			X
Administrateur	X		X			
Invité (utilisateur sans compte)	X					
Base de données des utilisateurs	X	X	X	X	X	X
Base de données des messages	X	X	X	X	X	
Application web	X	X	X	X	X	X