|  |
| --- |
| HEIG-VD |
| Modélisation de menaces |
| STI – Projet 2 |

|  |
| --- |
| Gabriel Roch & Cassandre Wojciechowski  22/01/2021 |

Table des matières

[1. Introduction 2](#_Toc60671458)

[2. Description du système, identification des actifs 2](#_Toc60671459)

[2.1 DFD (data-flow diagrams) 3](#_Toc60671460)

[2.2 Identification des biens (éléments du système / actifs / assets) 4](#_Toc60671461)

[2.3 Définition du périmètre de sécurisation 4](#_Toc60671462)

[3. Identification des sources de menaces 4](#_Toc60671463)

[3.1 Eléments du système attaqué 4](#_Toc60671464)

[3.2 Motivation(s) 4](#_Toc60671465)

[3.3 Scénario(s) d’attaque 4](#_Toc60671466)

[3.4 STRIDE (permet de classer les menaces selon différentes catégories) 4](#_Toc60671467)

[4. Identifier les contre-mesures 5](#_Toc60671468)

[4.1 En fonction des scénarios d’attaques 5](#_Toc60671469)

[5. Sécurisation de l’application 5](#_Toc60671470)

[6. Conclusion 5](#_Toc60671471)

# Introduction

Ce rapport contient une modélisation de menaces réalisées pour une application de messagerie Web réalisée pour le cours STI (Sécurité des Technologies Internet) lors de notre cinquième semestre d’étude à la HEIG-VD.

But du document : rendre visible les menaces, l’exposition aux menaces et réduire le risque en prenant des contre-mesures

# Description du système, identification des actifs

Objectifs du système ?

* Echange de messages privés (confidentiels -> seuls les utilisateurs concernés doivent y avoir accès) entre utilisateurs de l’application
* Réputation : avoir une bonne réputation grâce à une bonne confidentialité
* Disponibilité : le système reste disponible 99.9% du temps

Hypothèses de sécurité ?

* administrateurs de confiance
* OS et serveur Web de confiance

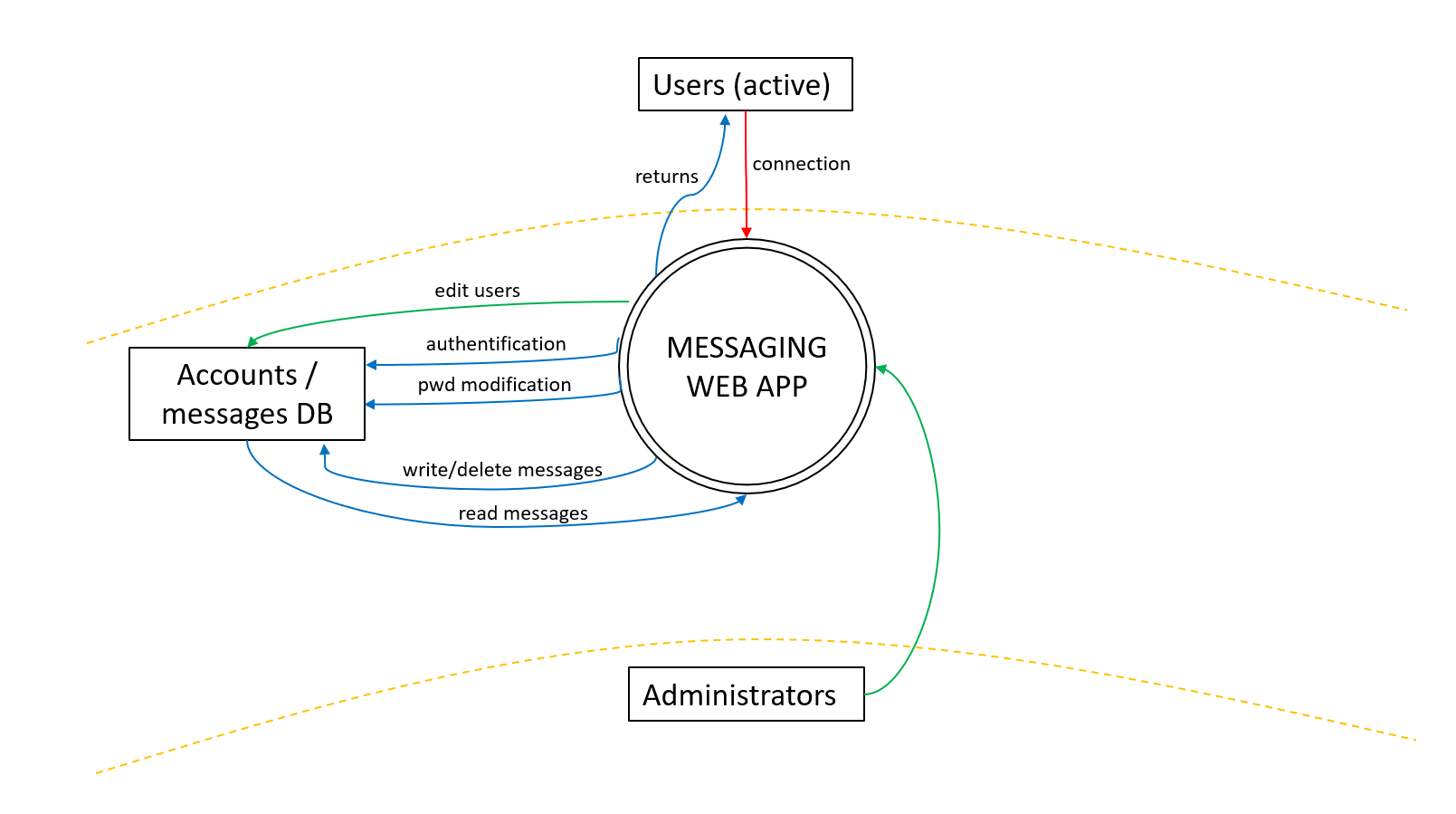
Exigences de sécurité du système ?

* Seuls les utilisateurs peuvent se connecter et envoyer des messages (Contrôle d’accès)
* Les utilisateurs ne doivent pas accéder aux fonctions d’administrateur (Contrôle d’accès)
* Les messages et les données personnelles ne doivent être modifiés que par l’utilisateur concerné (Intégrité)
* L’application doit être disponible 99.9% du temps pour ses utilisateurs (Disponibilité)
* Les messages et les informations personnelles des utilisateurs doivent être protégés (Privacy)

Comment est constitué le système ? (users, machines, flux, …)

* Utilisateurs « lambdas »
* Administrateurs de l’application

## 2.1 DFD (data-flow diagrams)



Petit texte d’explications …

## 2.2 Identification des biens (éléments du système / actifs / assets)

Eléments du système :

* Base de données avec les utilisateurs et leurs messages envoyés / reçus (données)
  + confidentialité / sphère privée des utilisateurs
  + en cas de pb : réputation de l’application
* Application Web (infrastructure)
  + intégrité et disponibilité
  + en cas de pb : possible de nuire à la disponibilité et/ou à la réputation (les utilisateurs ne voudront plus utiliser une application qui n’est pas disponible ou qui donne accès à leurs informations personnelles)

Rôles des utilisateurs :

* utilisateurs de base
  + actifs
  + inactifs
* administrateurs

## 2.3 Définition du périmètre de sécurisation

Ce qu’il faut absolument sécuriser dans le cadre de ce projet, c’est donc l’application PHP ainsi que la base de données SQLite qui lui est reliée.

# Identification des sources de menaces

## 3.1 Eléments du système attaqué

Cibles potentielles (système ou rebonds)

Ici certainement la base de données ou l’application elle-même

## 3.2 Motivation(s)

Lecture des messages privés d’utilisateurs de la messagerie

Envoi de messages en usurpant l’identité d’autres utilisateurs

Récupérations des informations personnelles des utilisateurs

Empêcher l’utilisation de l’application (Dos, DDos)

* Proposé dans les slides :

• S'amuser (souvent sans autre but).

• Prise de contrôle, pouvoir, ego.

• Acquérir des connaissances techniques.

• Morales : politique, écologie, social, "robin des bois", etc.

• Ressources gratuites : accès, machines, etc.

• Argent, escroqueries, espionnage industriel/économique.

• Terrorisme, espionnage, guerre informatique.

## 3.3 Scénario(s) d’attaque

Sources : script kiddies (s’amuser), hackers (s’amuser), spies (accéder aux infos / messages des users), …

Réfléchir à :

• Vols d'informations

• confidentialité, compétition, ...

• Destruction information

• Modification information or systems

• Arrêt de processus

• Infection des systèmes des utilisateurs

• Usurpations d'identités

• Accès aux services payants

Surtout :

Threat scenario 2: users database stealing

• Business impact: medium (reputation, loss of assets)

• Threat source: organized cybercrime, competition

• Motivation: financial

• Targeted asset(s): user accounts database

• Attack scenario(s):

• Code injection (request tampering or malicious file injection)

• Authorization bypass (access to user profile details)

• Control(s):

• Input validation

• File input validation

• Strong access control to user details

• Defense-in-depth: secure password storage, account data encryption

## 3.4 STRIDE (permet de classer les menaces selon différentes catégories)

• Spoofing

• Example: authenticating to the application using a stolen password

• Countermeasure: strong authentication, secure data transport

• Tampering

• Example: using SQL injection to modify or delete records of a data base

• Countermeasure: use of prepared statements, escaping user input

• Repudiation

• Example: Modify a user shipping address on an e-commerce

• Countermeasure: request address confirmation and additional authentication to confirm

• Information disclosure

• Example: intercept clear-text browser traffic in a public wifi

• Countermeasures: traffic encryption

• Denial of service

• Example: allocate session memory based on user provided values

• Countermeasures: validate size before allocating (input validation)

• Elevation of privileges

• Example: copy/paste an administrative URL within a normal user session

• Countermeasures: authorization mechanism

# Identifier les contre-mesures

## 4.1 En fonction des scénarios d’attaques

Pour chaque scénario imaginé, chercher une solution / contre-mesure pour éviter le scénario

# Sécurisation de l’application

Contrôles / mécanismes recommandés pour diminuer les risques :

• Validate input for all request parameters

• Use parameterized SQL queries

• Ensure access control when displaying user profiles

• Prevent from request automation

• Deploy random URLs for accessing the electronic edition

• Access control when requesting the electronic edition

# Conclusion

Nous avons un moyen de créer des modèles de menaces

• le système est décrit

• les sources de menaces sont identifiées

• les scénarios d'attaques sont identifiés

• les mécanismes de mitigation des risques sont identifiés

La modélisation de menaces permet :

• d'identifier des menaces, détecter des scénarios d'attaques, déterminer des contremesures

• d'accroître le niveau de confiance en terme de sécurité

• créer un modèle réutilisable