STI – Projet partie 2

Sécurité des Technologies Internet

Table des matières

Introduction STI – Projet 2

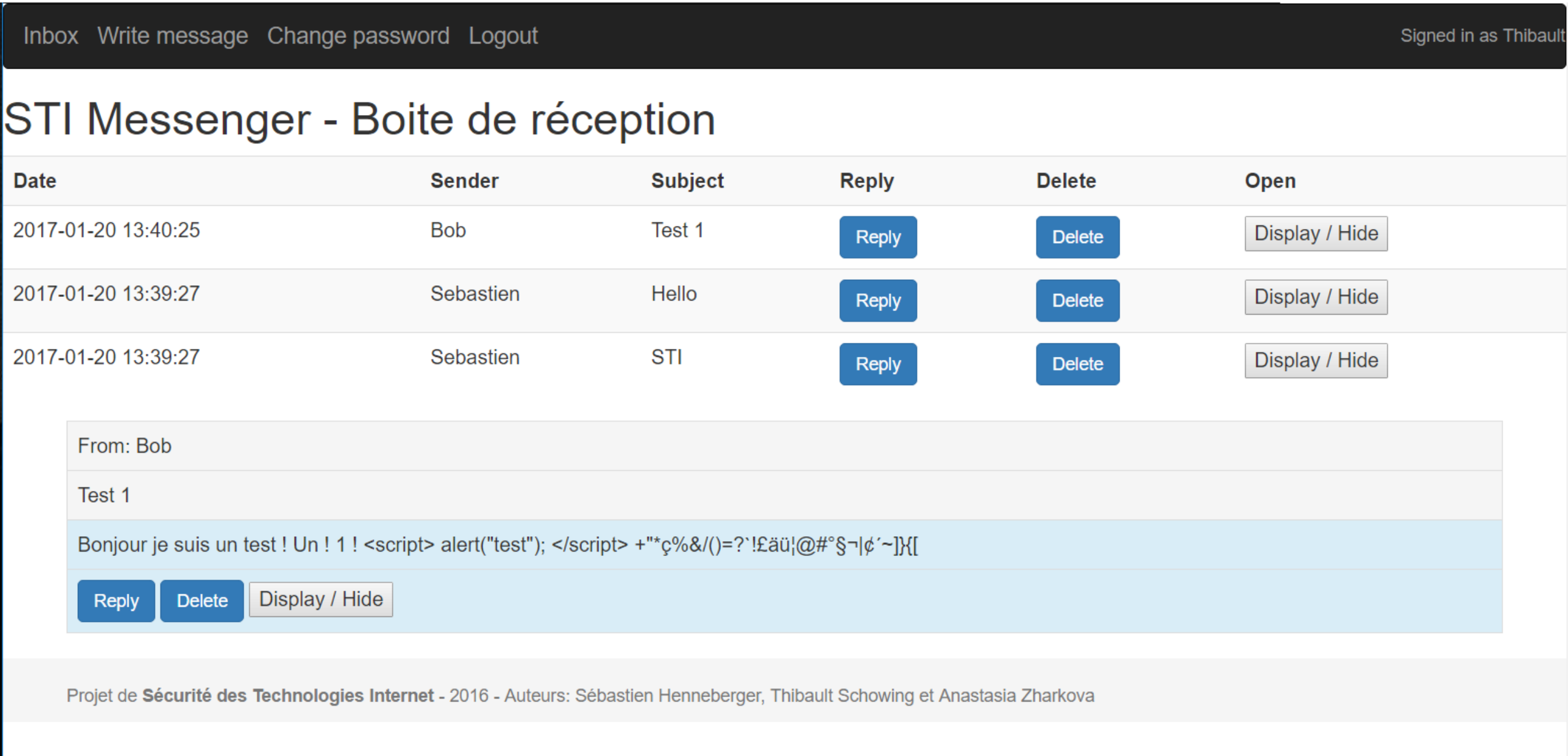
- **Rappel projet 1:** Application de messagerie
- **But:** implémenter la sécurité sur l'application du projet 1
- **Répartition des tâches:** par fonctionnalité et vulnérabilité connues

État des lieux

- Aucune entrée utilisateur n’est protégée
- Tous les caractères spéciaux sont interprétés
- Les formulaires n’ont pas de jeton anti-CSRF
- Aucune politique de mots de passe

- Les restrictions d’accès aux pages sont déjà efficaces
- La génération des hashs est déjà sûre

Implémentation générale



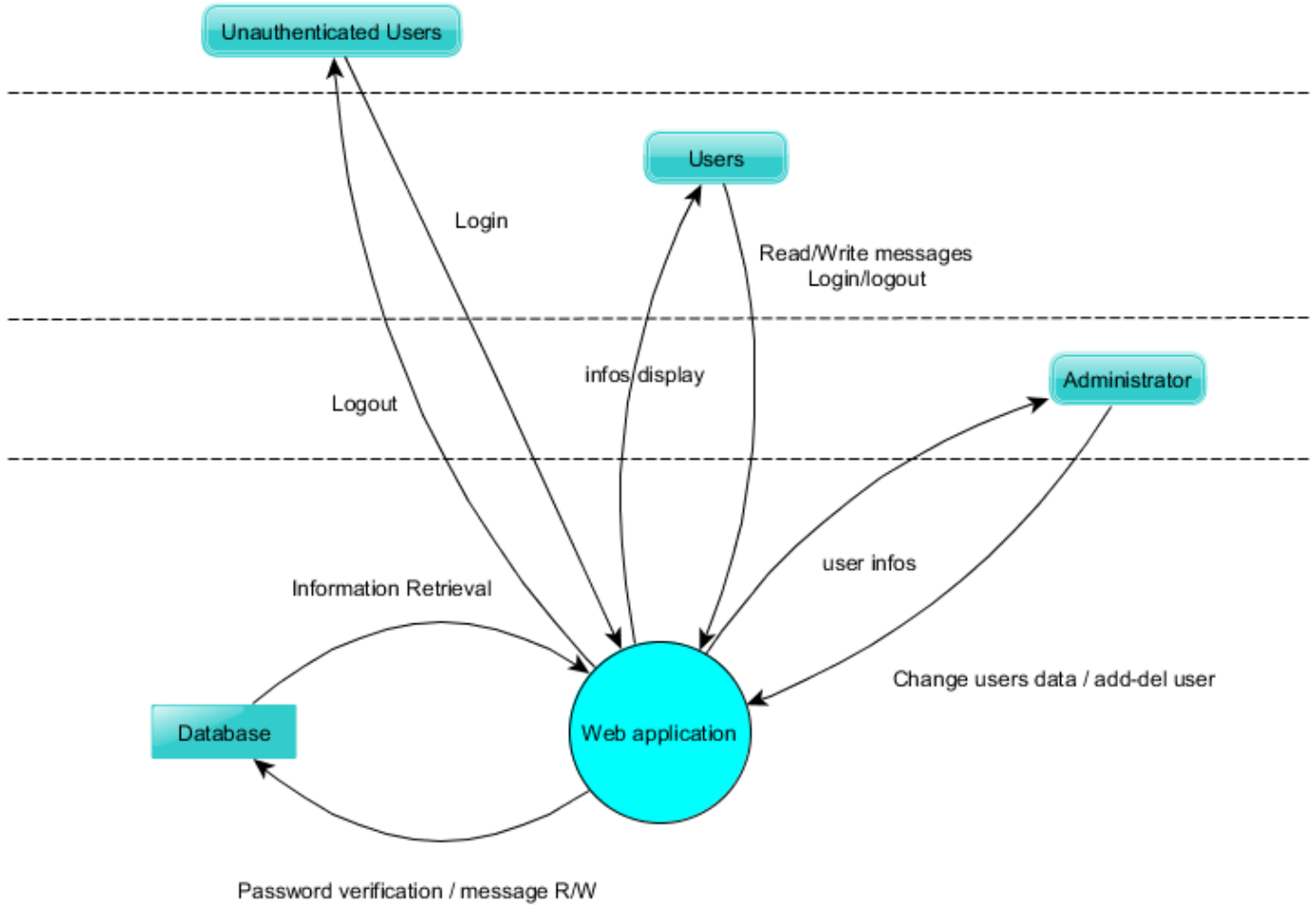
Analyse de menaces

- Description du système
- Sources de menaces
- Scénario d’attaques
- Contremesures

Analyse de menaces – description du système

- **Objectifs**: fournir un moyen de communication interne à une entreprise
- **Exigences**: uniquement les personnes autorisées ont accès. Non répudiation des données.
- **Constitué de:** Utilisateurs / base de données / serveur / application.

Analyse de menaces - DFD



Analyse de menaces – sources de menaces

- Script-kiddies
  - Motivation : s'amuser
  - Cible : n'importe que élément
  - Potentialité : haute
- Cybercrime
  - Motivation : accès aux éléments interdits qui pourrons être utilisées plus tard
  - Cible : vol de credentials et/ou modification d'information
  - Potentialité : moyenne

Analyse de menaces – sources de menaces

- Concurrent
  - Motivation : vol d'information confidentiels, reutilisation du même contenu
  - Cible : serveur local
  - Potentialité : moyenne

Analyse de menaces – STRIDE

Component	S	T	R	I	D	E
External agent	v		v			
Data store	v	v	v	v	v	v
Process		v	v	v	v	
Data flow		v		v	v	

Analyse de menaces - Scénario d'attaques

Scenario d'attaque 1 : User non authentifié essaie de se connecter à l'application

Business impact : élevé (image de l'entreprise, cout de remédiation)

Motivation : challenge, curiosité, accès aux données privés

Scenario d'attaque : injection du code

Contrôle : utilisation de requête de BD, validation des entrées de fichier.

Scenario d'attaque 2 : Vol de base de donné

Business impact : élevé ou moyen (réputation, pertes d'actifs)

Motivation : financière

Scenario d'attaque : injection de code, bypass autorisation

Contrôle : validation des entrées de fichier, défense en profondeur, chiffrement de données

Analyse de menaces - Scénario d’attaques

Scenario d’attaque 3 : bruteforce de login et/ou mot de passe

Business impact : moyen (réputation, pertes d’information)

Motivation : avoir accès aux données privés

Scenario d’attaque : bruteforce

Contrôle : captcha, mot de passe fort

Analyse de menaces - Scénario d'attaques

Scenario d'attaque 4 : user accède aux messages d'un autre user (attaque horizontale)

Business impact : faible (réputation, pertes de confidentialité)

Motivation : avoir accès aux données privés

Scenario d'attaque : modification d'ID personnelle

Contrôle : prepared statement

Scenario d'attaque 5: user accède aux messages/privilèges d'un administrateur (attaque verticale)

Business impact : moyen/faible (pertes d'information/ confidentialité)

Motivation : avoir accès aux données privés, escalade de privilèges

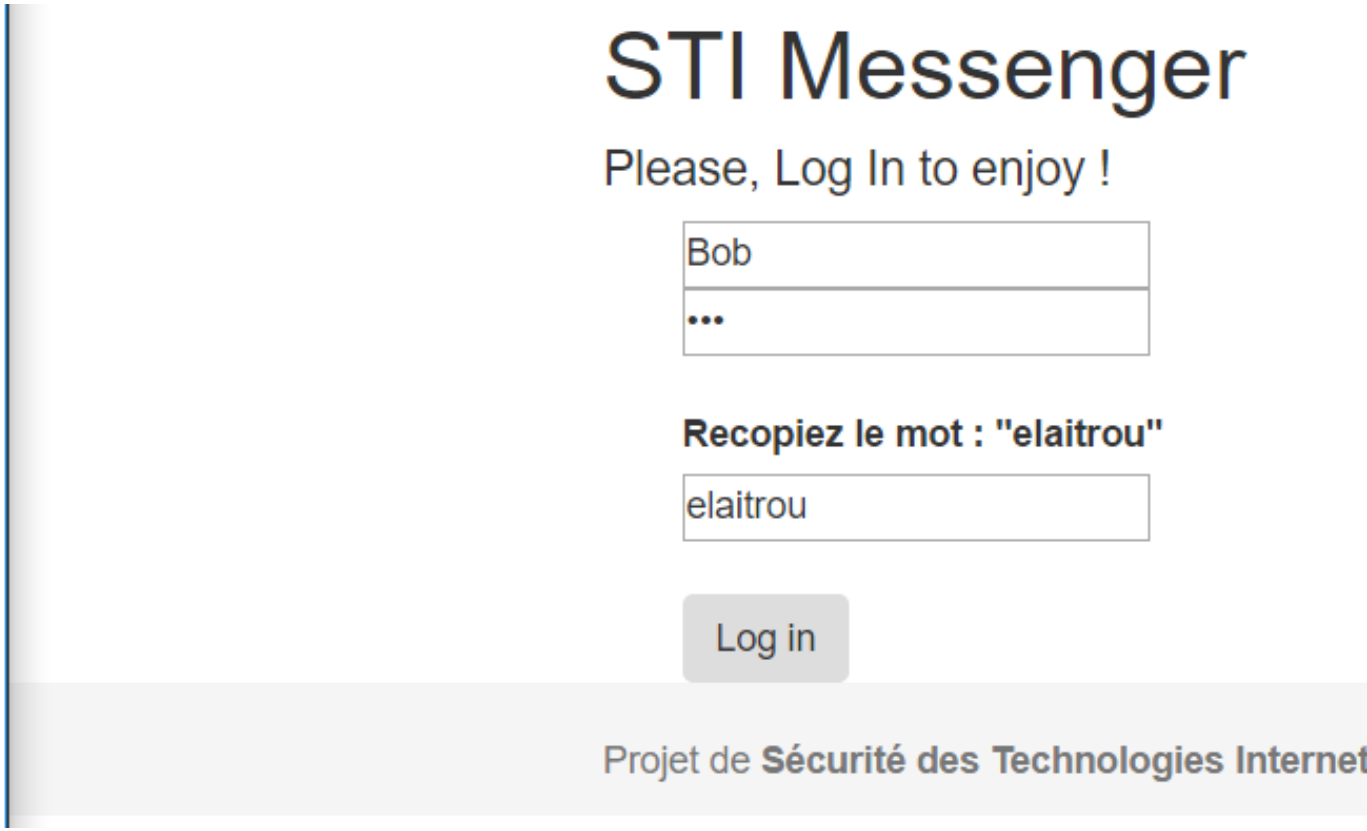
Scenario d'attaque : modification d'ID, d'URL

Contrôle : prepared statement

Sécurité de l’application

- Aucune entrée utilisateur n’est protégée -> OK
- Tous les caractères spéciaux sont interprétés -> OK
- Les formulaires n’ont pas de jeton anti-CSRF -> OK
- Aucune politique de mots de passe -> OK
- Génération de hash -> OK
- Gestion des accès -> OK

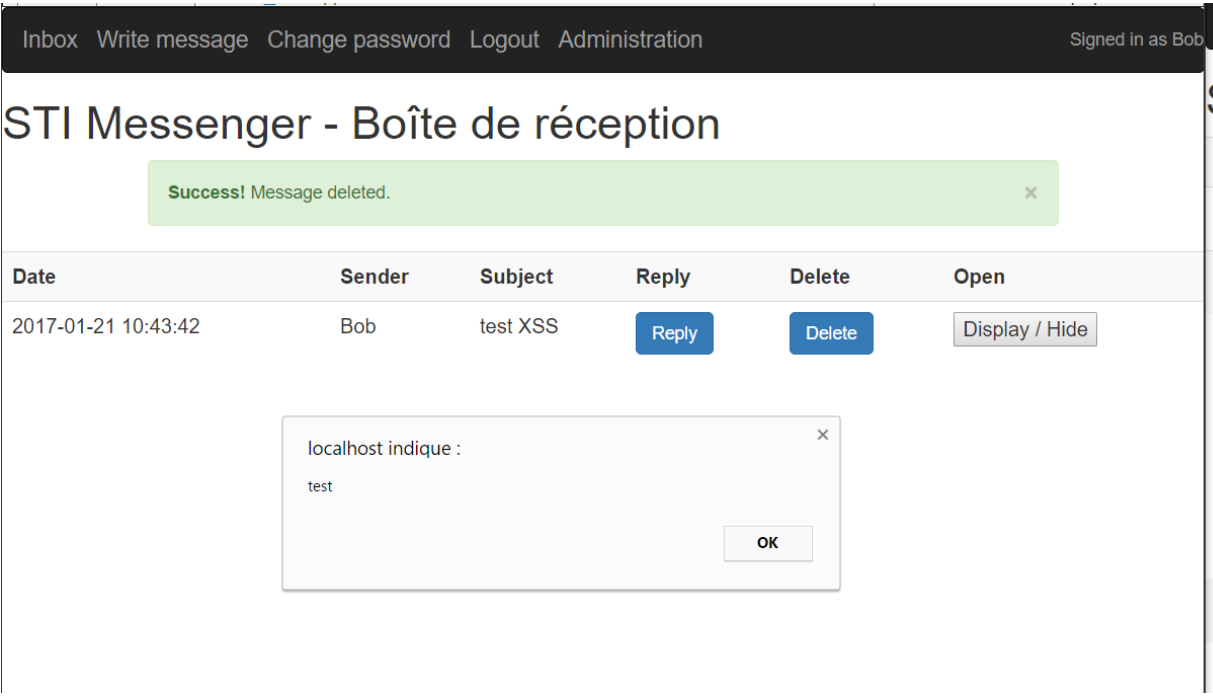
Comparaison

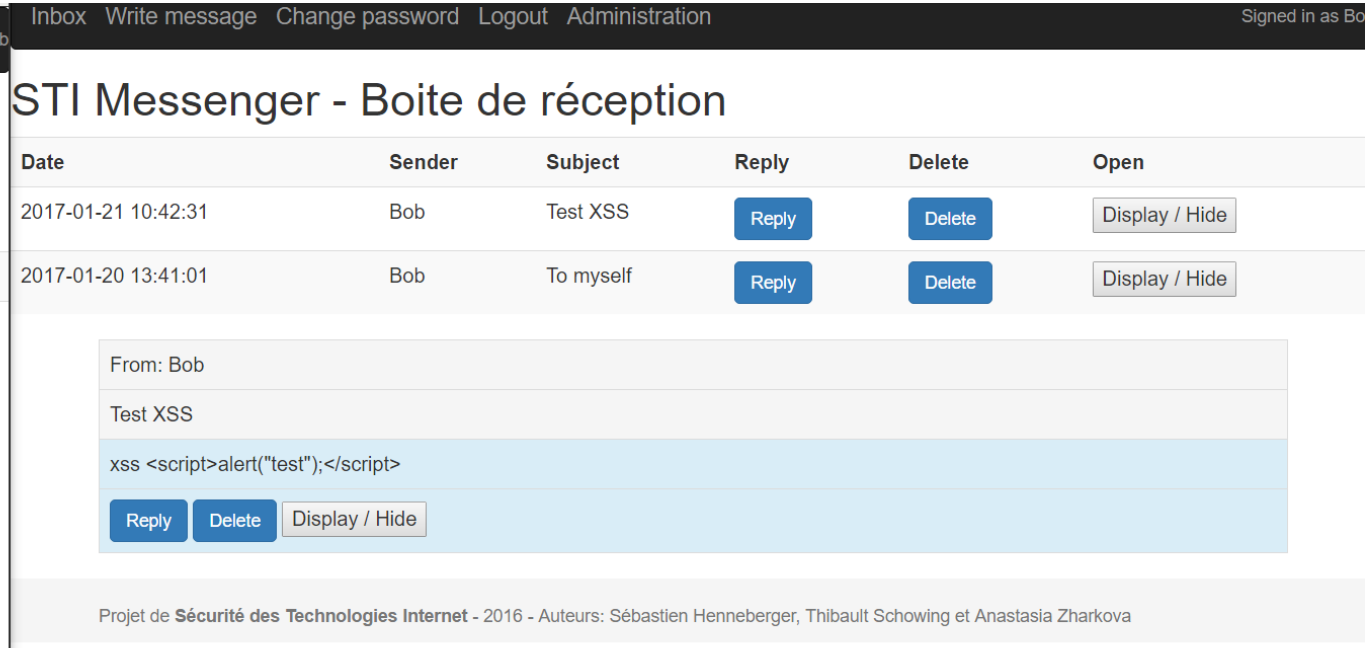


Comparaison

```
<script>alert("test");</script>
```

```
xss &lt;script&gt;alert(&quot;test&quot;);&lt;/script&gt;
```





Un peu de code pour le plaisir ☺

```php
function sendMessage($userIdTo, $subject, $message) {
    include("databaseConnection.php");
    $sql = "INSERT INTO messages (message_subject, message_message, message_sender_id , message_receiver_id)
            VALUES (:subject, :message, :sender, :userIdTo);";
    $sth = $file_db->prepare($sql, array(PDO::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
    $sth->execute(array(
        ':subject' => htmlspecialchars($subject),
        ':message' => htmlspecialchars($message),
        ':sender' => $_SESSION['userId'],
        ':userIdTo' => htmlspecialchars($userIdTo)
    ));
    $file_db = null;
    return;
}
```

À améliorer

- Configurer SSL/TLS (et autre gestion de HTTP et accès niveau serveur)
- Améliorer la politique de mots de passe (caractères, validité, nombre de tentatives fausse, etc ...)
- Captcha plus complexe
- Gestion des redirections lors d'erreur 404 ou d'accès (ne pas divulguer d'informations)

Conclusion

- La sécurité est présente à tous les niveaux (code PHP, configuration serveur, logique machine,…)
- Bon début dans ce projet ☺

Вопросы