AMASAFEGUARD SECURITY SOLUTION

DOSSIER D'EXPLOITATION

Ce présent document détaille les étapes de la conception à la mise en production de l'application AMAsafeguard, une solution d'envoi de fichiers sécurisés

Avant-propos

CONTEXTE

Pour les besoins des utilisateurs de smartphone fonctionnant sous Android, trois étudiants en développement Mobile, Alexis LEPAGE, Maxime GEOFFROY, Antoine TROUVE, ont imaginé et conçu une solution d'application mobile permettant le transfert de données sécurisées d'un téléphone Mobile à un serveur ftp authentifié.

CONFIDENTIALITE

Toutes les informations contenues dans ce présent document sont soumises à la loi sur la protection de la propriété intellectuelle Elles sont donc confidentielles.

Table des matières

Partie I – Le cadrage du projet

L'EQUIPE

Le groupe d'étudiant en Conception et Développement de Solutions Mobiles, concepteur et créateur du projet **AMAsafeguard** se compose de :

- **Alexis LEPAGE**, Développeur à Martin 3D
- **Maxime GEOFFROY**, Développeur à Technology Everywhere
- **Antoine TROUVE**, Analyste Développeur à Adventi Informatique

LE BESOIN

Le besoin initial

Les utilisateur d'une application mobile Android doivent pouvoir transférer le contenu d'un dossier sur un serveur distant. L'échange des données doit-être sécurisé.

Les contraintes

- L'application doit être authentifiée auprès du serveur afin de lui envoyer des données ;
- Les communications entre l'application et le serveur seront protégées en confidentialité et en intégrité
- Le serveur doit être capable de supporter plusieurs connexions simultanées
- Si le contenu d'un fichier d'un répertoire surveillé change, il faut le mettre à jour sur le serveur
- Si un fichier est supprimé, il faut le supprimer sur le serveur ;

LA SOLUTION AMASAFEGUARD

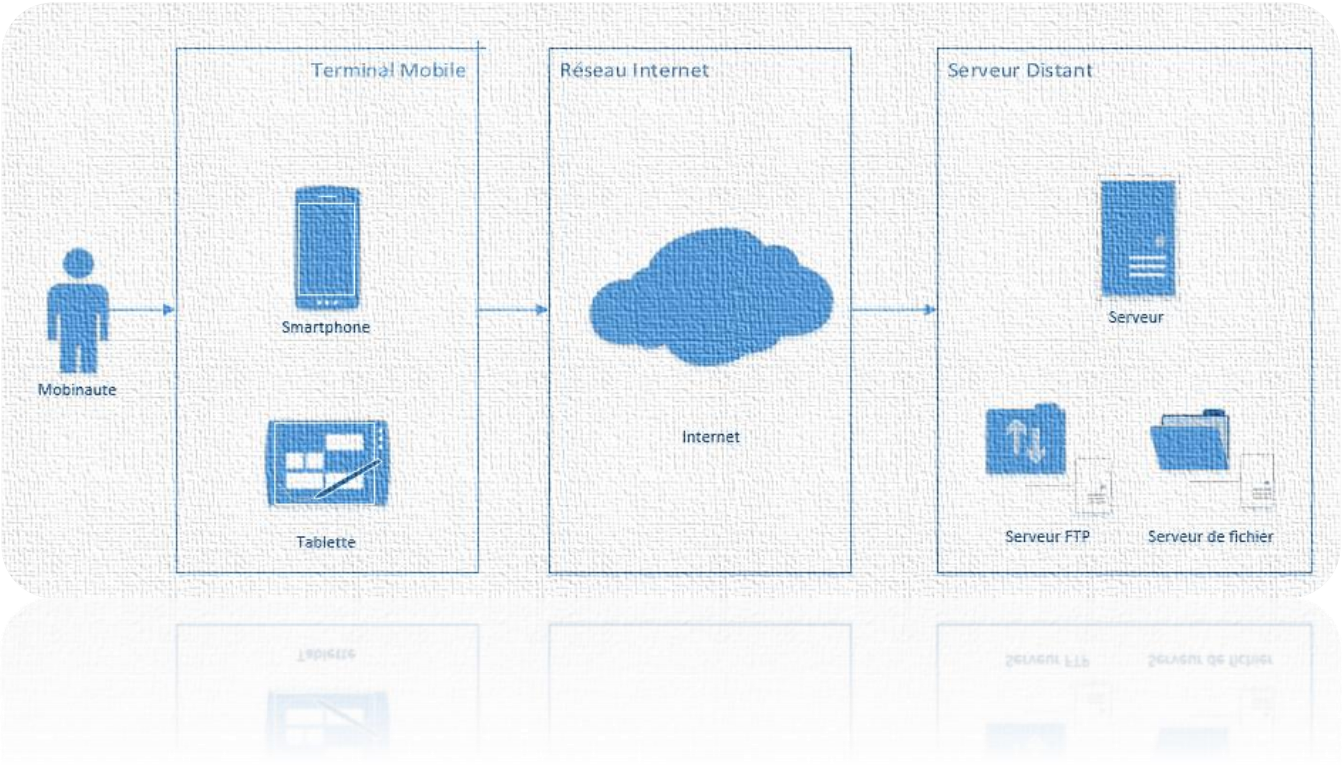
- Une solution mobile sous Android,
- Un échange de données sécurisées comprenant le chiffrement des données transférées ainsi qu'un système d'authentification au serveur distant.
- Un accès sécurisé à l'application.

PARTIE II – L'ARCHITECTURE DE L'ENSEMBLE DE LA SOLUTION

Partie II – L'architecture de l'ensemble de la solution

La solution AMAsafeguard a été pensée de manière à avoir une communication entre le terminal mobile intégrant la solution applicative et un serveur FTP distant.

La communication passe donc par un réseau. Son utilisation en mode hors connexion n'est pas gérée.

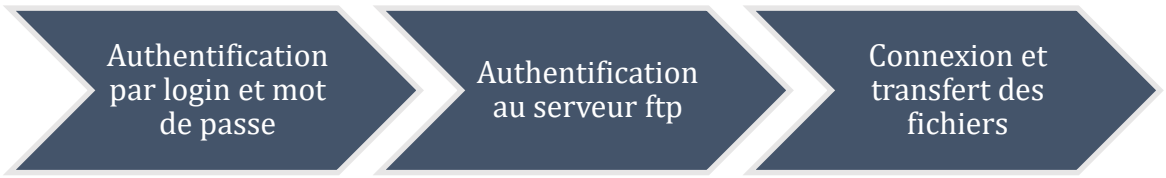


Partie III – AMASAFEGUARD, une solution mobile

L'APPLICATION MOBILE

Sécurité et Authentification

L'accès à l'application mobile requiert une authentification par login et mot de passe stockés en base de données locale. Cette authentification est nécessaire pour se connecter au serveur ftp.



Gestion des inscriptions

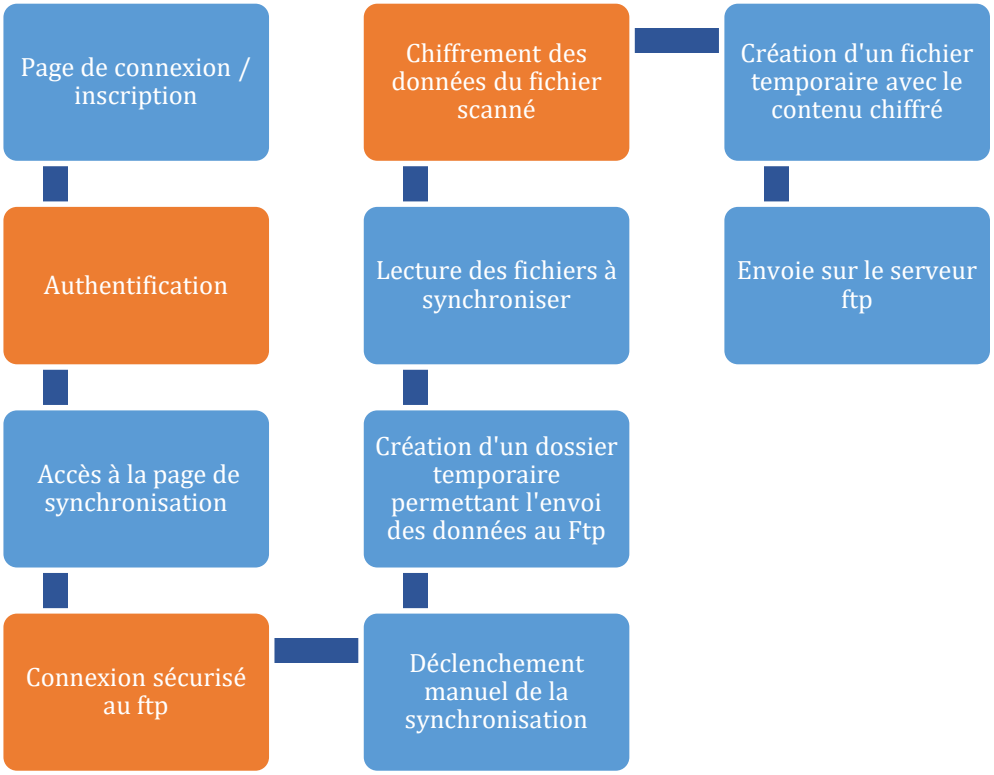
Chaque utilisateur a la possibilité de s'inscrire afin d'obtenir un identifiant et un mot de passe chiffré afin de se connecter à l'application.

Transfert des données

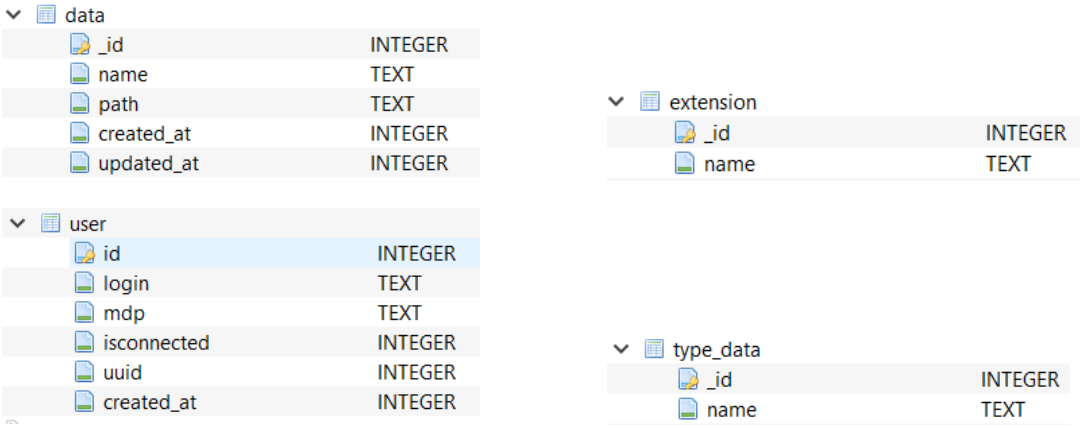
L'utilisateur authentifié a la possibilité de synchroniser et d'envoyer les fichiers vers un serveur ftp distant à la demande. L'utilisateur pourra, à partir d'un fichier de configuration, renseigner les dossiers à synchroniser. L'application va alors scanner et envoyer les fichiers dans le Ftp uniquement si ceux-ci ont été modifiés, supprimés ou ajoutés.

PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

Le flux



La base de données : amasafeguard.sqlite



La base de données est conçue de manière à rester simple et légère tout en permettant la gestion des utilisateurs, l'extension des fichiers, le type de fichier ainsi que le fichier lui-même. Ces informations sont primordiales pour la sauvegarde des données chiffrées ainsi que leur envoi par ftp.

PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

LA STRATEGIE DE SURVEILLANCE DES REPERTOIRES / FICHIERS

Le fichier de configuration

A l'installation de l'application, un **fichier de configuration** est **créé** et **stocké** dans le terminal mobile. Il sera présent dans le stockage interne du terminal Mobile dans le répertoire **AMAsafeguard/Configuration**.

Ce fichier est très important dans le fonctionnement de l'application puisqu'il **détermine les chemins des répertoires à scanner** pour le chiffrement et l'envoi des données au serveur ftp.

L'utilisateur peut alors **modifier ce fichier** de configuration afin d'ajouter les répertoires qui seront soumis au scanne de l'application.

Création des fichiers sur le serveur ftp

La **première étape** consiste à la **création automatique d'un répertoire unique** pour chacun des utilisateurs de l'application.

Une fois le compte créé à partir de la section inscription, l'application crée un répertoire **sur le serveur ftp**.

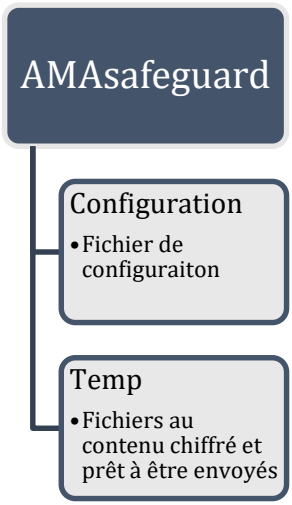
Pour des mesures de sécurité, ce répertoire prendra non pas le nom de l'utilisateur mais celui de son **Universally Unique IDentifier** créé lui aussi lors de son inscription.

Création d'un répertoire temporaire sur le terminal mobile

La **seconde étape** consiste à la **création d'un répertoire temporaire** sur le terminal mobile. Ce répertoire **stocke l'ensemble du contenu chiffré des fichiers scannés**. C'est le contenu de ce répertoire qui est transféré au serveur ftp.

Le répertoire se trouve dans le système de stockage interne du terminal mobile **/AMAsafeguard/Temp.**

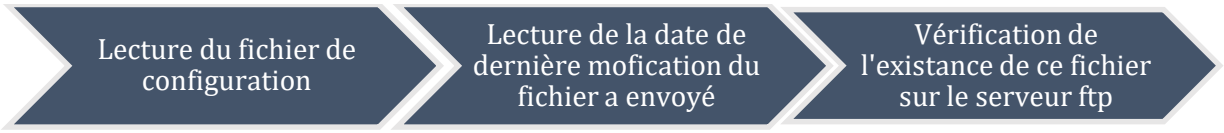
Les répertoires créés par l'application



PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

La stratégie globale de surveillance

DANS TOUS LES CAS :



Scénario 1 : si le fichier n'existe pas sur le serveur ftp :

Scénario 2 : si la date de création est différente de la date de modification :

Scénario 3 : si le fichier existe sur le serveur ftp mais pas dans les répertoires cibles :

PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

LA STRATEGIE DE SECURITE

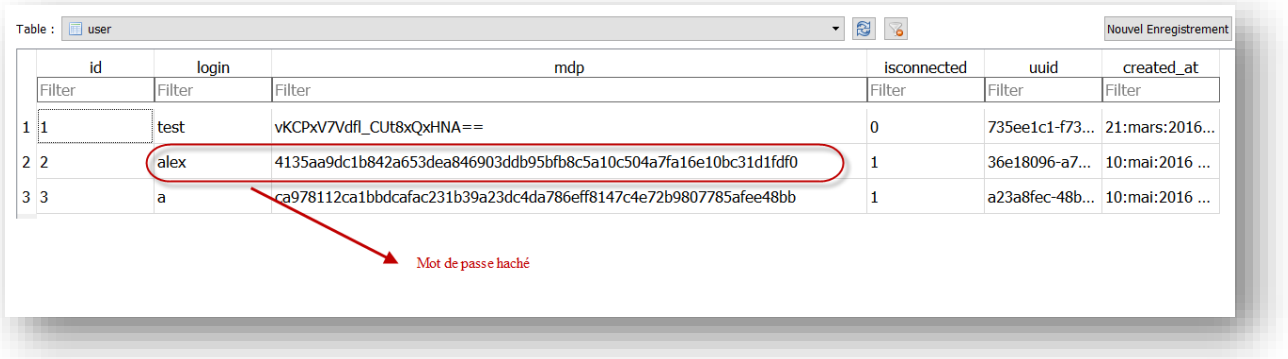
L'accès à l'application

Pour accéder à la fonction de synchronisation des données avec le serveur ftp distant, l'utilisateur doit **s'authentifier** par l'intermédiaire d'un **login** et **mot de passe**.

L'obtention d'un compte se fait par l'intermédiaire d'une inscription en renseignant un identifiant et un mot de passe.

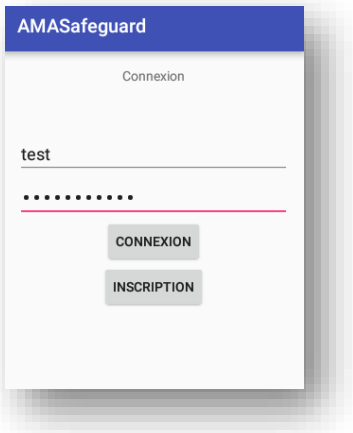
Le stockage haché du mot de passe

Les **mots de passe** sont **stockés** en **base de données**. Bien évidemment, une **fonction de hachage** permet de ne pas rendre le mot de passe lisible en consultation dans la base de données.



Une lisibilité impossible du mot de passe

Lorsque l'utilisateur renseigne son mot de passe, celui-ci n'est pas lisible par l'utilisateur ou par autrui afin de garder un maximum de sécurité.



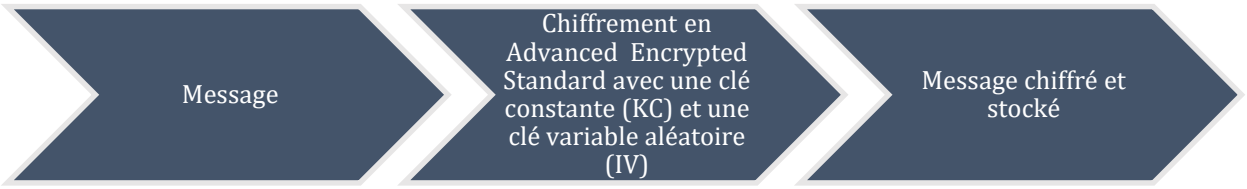
PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

L'accès au ftp

L'accès au ftp se fait directement en interne dans l'application au moment de l'authentification de l'utilisateur.

L'échange des données

Le flux

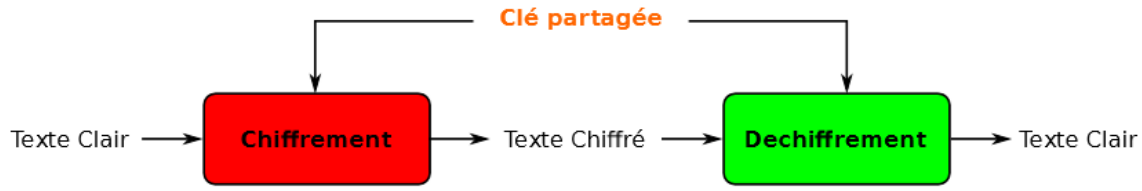


Pour le moment, il n'est pas prévu de déchiffrer.

La **clé de chiffrement** est **symétrique** et de **128 bits**.

Pourquoi une clé symétrique ?

N'omettant pas la possibilité ultérieure de déchiffrer le message lors du téléchargement de celui-ci, l'équipe d'AMAsafeguard a choisi une clé de chiffrement symétrique. Voici le principe de la clé symétrique :



Ce principe permet avec une même clé de chiffrer et de déchiffrer le message.

Prenant en considération également que l'échange de données se fait par le moyen unique du Transfert File Protocol et que par conséquent le serveur n'a pas les outils nécessaires pour déchiffrer lui-même la clé partagée, seule l'application est en mesure de chiffrer et déchiffrer les contenus des fichiers. Modèle qui ne correspondait donc pas à une stratégie de sécurité basée sur une clé asymétrique.

Le stockage des répertoires/fichiers sur le serveur ftp

Chacun des utilisateurs a un répertoire dédié sur le ftp. Ce fichier prendra non pas le nom de l'utilisateur mais l'**Universally Unique Identifier** créé lui aussi lors de son inscription :

[IMAGE]

PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

Liste des fonctions liées à sécurisation des données

- `public static boolean protectSymetricFile (FTPClient client)`
  Permet la génération des clés pour le chiffrement des données
- `public static final byte[] encrypt (final byte[] input, final byte[] key, final byte[] iv)`
  Permet le chiffrement des données
- `public static final byte[] sha256(final String password)`
  Permet le hachage du mot de passe en sha-256
- `public static String toHexString(byte[] data)`
  Convertit un byte en chaine de caractère afin de rendre lisible la donnée.

LE CHOIX DES TECHNOLOGIES

Android

L'application est exécutable sur Android comme le stipule les contraintes énoncées plus haut.

Compatibilité

Compatibilité des applications : Android 4.1 -> 6.0. Par cette compatibilité, les concepteurs veulent s'assurer un maximum d'utilisateur potentiel.

Serveur

Le système d'exploitation du serveur ftp :

GNOME (**GNU** **N**etwork **O**bject **M**odel **E**nvironnent). Ce système d'exploitation est un **environnement de bureau libre** dont l'objectif est de rendre **accessible** l'utilisation du système d'exploitation GNU **au plus grand nombre**.

Cette interface fonctionne sur les systèmes GNU/Linux et mais également sur les systèmes de type UNIX. **Pas gourmant en ressource**, il permet également d'offrir une interface graphique facile d'utilisation pour les utilisateurs.

L'accès à ce système d'exploitation permet ainsi d'avoir une **infrastructure sans frais hormis la location du serveur distant**.

PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

Le service ftp :

Le service FTP a été choisi en **fonction du système d'exploitation**. Egalement pour **amoindrir les coûts du projet**.

La solution retenue est donc celle du **ftp libre** et **fonctionnant sous un système Linux** : **ProFTPd Version 1.3.5a.** Ce dernier est connu pour offrir des transferts de données performant et sécurisé.

PARTIE III – AMASAFEGUARD, UNE SOLUTION MOBILE

LES EVOLUTIONS POSSIBLES

- Pouvoir **modifier le fichier de configuration via l'application mobile**
- **Améliorer la sécurité de l'application** en permettant un **chiffrement des données de connexion au serveur ftp**.
    - La solution sécurisée envisagée :
        - Création d'un fichier de connexion à la racine du répertoire AMAsafeguard\Configuration\connexion.config
        - Indiquer dans ce fichier les chaîne de connexion ainsi que les identifiants et mot de passe de connexion au ftp.
        - Chiffrer le contenu de ce fichier.
        - Implémenter les fonctions permettant la gestion de ce fichier et la connexion au serveur ftp en lisant les informations de ce fichier.
    - La **gestion du fichier de connexion via l'application mobile**
        - Pouvoir renseigner les informations directement depuis l'application mobile

Partie IV – L'installation de l'application

INSTALLATION D'UN SERVEUR VIRTUEL

Monter une machine virtuelle avec VirtualBox

Installer un système d'exploitation GNU Linux avec ou non interface graphique

L'équipe AMAsafeguard a mis en place une machine virtuelle de test fonctionnant sous le système d'exploitation GNOME. En voici la configuration si vous souhaitez reproduire l'environnement de test.

INSTALLER LE SERVEUR FTP PROFTPD

Sur la machine virtuelle, installer le service FTP ProFtpd.

Vous trouverez toutes les informations d'installations à l'adresse suivante : http://www.proftpd.org/

INSTALLER L'APPLICATION AMASAFEGUARD SUR LE TERMINAL MOBILE ANDROID

Partie V – L'utilisation de l'application

CREER UN COMPTE POUR LA PREMIERE UTILISATION

[IMAGE]

PARAMETRER LE FICHIER DE CONFIGURATION

[IMAGE]

SE CONNECTER AVEC L'IDENTIFIANT ET LE MOT DE PASSE DE VOTRE COMPTE

[IMAGE]

SYNCHRONISER LES REPERTOIRES / DOSSIERS

[IMAGE]

TESTER LE RESULTAT

Tester le chiffrement des données des fichiers transférés sur le serveur

- Se connecter au serveur FTP
- Consulter le répertoire unique dédié à l'utilisateur. Ce répertoire est situé sur les serveur à cet endroit : **[COMPLETER]**

Tester la lisibilité du mot de passe dans la base de données

- Récupérer la base de données locales au moyen d'Android Studio
- Ouvrir la base de données : **amasafeguard.sqlite**
- Pour ouvrir la base de données vous pouvez télécharger l'utilitaire gratuit **DBSqliteBrowser**