

Face Key

Quentin GERARD, Louis L'HARIDON et Matthieu VILAIN
CMI Systèmes Intelligents et Communicants, Université de Cergy-Pontoise
mail@etu.u-cergy.fr

Résumé

abstract of the project

I. SÉCURITÉ

A. Introduction au problème

Lors de l'utilisation de l'application, beaucoup de données transitent entre les différents blocs que composent l'architecture du logiciel, et/ou sont stockés dans la Base de Donnée. Parmi elles, se trouvent des informations extrêmement sensibles concernant les utilisateurs. Pour un utilisateur, nous pouvons y trouver ces informations (liste non ordonnée et non exhaustive) :

- Adresses mails
- Mot de passe (de l'application Face Key)
- Combinaisons Identifiants/Mot de passe de plusieurs sites
- Coordonnées Bancaires
- Et d'autres

Si un utilisateur mal intentionné arriverait à intercepter les messages entre le client et le serveur, en se plaçant entre les deux grâce à une attaque basique de type Man in the Middle (MITM), alors il aurait accès à toutes ces données. De même si la base de donnée arrivait un jour à être compromise, et accessible par un utilisateur mal intentionné, alors il aurait également accès à toutes les données. Il nous faut donc trouver une solution qui nous permet à la fois de sécuriser les envois de données Client/Server, et une solution qui nous permet de rendre illisible les informations dans la base de données à tous les utilisateurs autres que l'utilisateur concerné.

B. RSA

1) *Introduction:* Le RSA est un algorithme de cryptographie dit "asymétrique" (une clé privée et une clé publique, contrairement à la cryptographie symétrique qui utilise la même clé pour crypter et décrypter). Il a été rendu public par R.L. Rivest, A. Shamir, et L. Adleman dans un papier publié en 1977 appelé "*A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*". Il a pour but, à la fois de crypter des messages afin qu'ils puissent être transmis sans risquer d'être lu par une personne tierce, mais aussi de signer les messages. Un des principes du RSA est que la clé privée, permettant de décrypter les messages soit impossible à retrouver à partir de la clé publique, qui elle permet d'encrypter les messages. Nous avons donc décidé d'utiliser cette méthode afin de crypter les messages entre le Client et le Server.

2) *Génération des clés:* La génération des clés appartient au receveur des messages, il doit ensuite communiquer la clé publique, qui n'est pas une information sensible, à l'expéditeur du message. Dans notre cas, le serveur et le client vont chacun générer de leur côté une paire de clés et transmettre à l'autre partie, la clé publique. Ainsi, même si les clés sont interceptées, n'importe qui pourra envoyer des messages au client et au serveur, mais personne ne pourra lire les messages cryptés envoyés, à l'exception du client et du serveur.

Pour créer les clés nous devons d'abord choisir deux nombres premiers p et q et ensuite calculer

$$n = p.q$$

Notons que n sera présent dans la clé publique il est donc important de trouver deux nombres premiers p et q assez grand pour qu'il soit difficile de décomposer n (Il est recommandé d'après les auteurs d'utiliser un nombre premier composé de 100 chiffres), en effet chaque nombre n a une décomposition **unique** en facteurs premiers. On cherche ensuite d tel que :

$$\text{pgcd}(d, (p-1).(q-1)) = 1$$

Nous venons donc de créer notre clé privée (d, n) , pour trouver notre clé publique nous devons calculer e tel que :

$$e.d \equiv 1 \pmod{(p-1).(q-1)}$$

Notre clé publique est donc le couple (e, n) .

3) *Cryptage et Décryptage des messages:* Pour crypter un message M nous avons besoin de notre clé publique (e, n) , notre message crypter C se calcule de la manière suivante :

$$C \equiv M^e \pmod{n}$$

La fonction de décryptage est tout aussi simple, à partir de notre message crypter C , et de notre clé privé (d, n) , nous calculons :

$$M \equiv C^d \pmod{n}$$

La méthode décrite ici est démontrée mathématiquement dans le papier d'origine de R.L. Rivest, A. Shamir, et L. Adleman.