Comparaison fonctionnelle et expérimentale d'une PKI sur mobile et de Kerberos sur mobile. Mise en oeuvre et comparaison rigoureuse des résultats expérimentaux. Automatisation du déploiement.

Jean-Philippe Blaise, Willian Jouot

Master SSIC, Metz University,
Ile du Saulcy, 57045 Metz, France
jeanphilippe.blaise@umail.univ-metz.fr
william.jouot@umail.univ-metz.fr

Résumé De nos jours, les téléphones mobiles sont partout, et la population de smartphone est en plein essor. Ces types d'appareils peuvent faire de plus en plus de choses, et ainsi égaler nos ordinateurs. Ainsi en leur fournissant une multitude de moyens de communication, nous devons nous poser la question de la sécurité des données, telle que nous la connaissons pour nos ordinateurs. Les smartphones, étant de plus en plus puissant, peuvent parfaitement gérer un ou plusieurs protocole d'authentification. De ce fait, nous allons développer une solution de PKI et une solution Kerberos pour smartphones utilisants le celebre Operating System de Google, Android. Dans cet article, nous allons comparer quelques solutions déjà existantes, puis proposer notre méthode.

Mots-clés : PKI, Kerberos, Android, mobile, mise en oeuvre, déploiement automatique

1 Problématique

A l'heure actuelle, les téléphones ainsi que les smartphones se démocratisent de plus en plus, et il y a donc de plus en plus de données qui transitent via ces appareils, que cela soit via le wifi, bluetooth, ou directement le réseau téléphonique. On commence ainsi à faire des achats directement depuis son téléphone, où encore envoyer des informations personnels, comme des mots de passe, ou des documents que l'on veut garder secret. Malheureusement, toutes ces données transitent plus ou moins en clair dans l'air, à la portée de n'importe qui. Il faut donc sécuriser les données. Quelles solutions existent ? Des systèmes comme PKI ont déjà fait leurs preuves sur nos ordinateurs, mais existe-t-il des solutions concrètes pour nos mobiles ?

2 Travaux existants dans la littérature scientifique

2.1 PKI pour site marchand

Il existe de nombreux travaux dans le domaine, notamment pour la PKI. Le premier article étudié[1], est très intéressant. Datant de 2007, cet article se pose une excellente

question. Comment sécuriser des données transitant d'un téléphone mobile a un site marchand ? Ainsi on peut trouver une solution assez intéressante, la création des clés d'authentification utilisées se fait en local, c'est a dire que le téléphone est capable de générer ses propres clés. C'est un bon point. Les auteurs utilisent la Crypto librairie[2] pour la signature de données transmises. Cependant cette solution connait quelques faiblesses, que nous détaillerons plus tard.

2.2 système de paiement sécurisé

Le second article[3] nous propose un autre systèmes de paiement sécurisé pour mobile via une PKI. Pour leurs tests, le centre d'enregistrement finlandais s'est chargé de leur fournir la PKI.

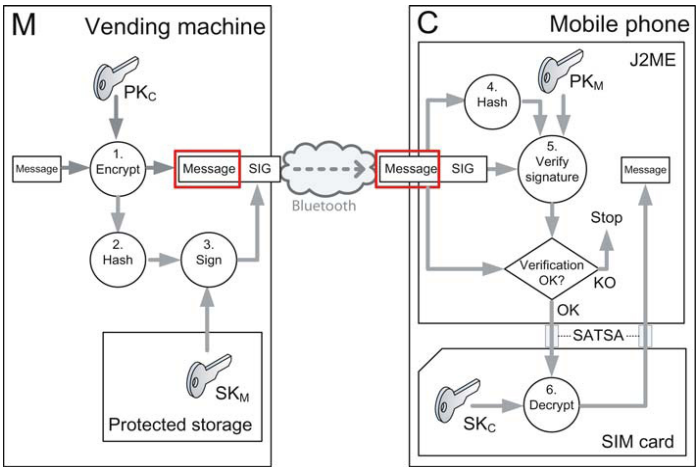

FIGURE 1. Exemple d'échange de messages securisés

Pour faire simple, les clés privées sont stockées dans la carte SIM. Etant infalsifiable, les clés privées ne peuvent pas être compromises. Chaque carte SIM se voit attribuer un unique certificat, délivré par le centre d'enregistrement finlandais. Cette carte SIM est capable de signer les données, mais le cryptage/décryptage se fait sur le téléphone. Un machine en ligne (banques, vente en ligne, etc.) crypte son message, et le signe. Le message est ensuite envoyé au téléphone mobile. Celui-ci vérifie la signature, dans le cas ou elle est correcte, le téléphone consulte sa clé privée, stockée dans la carte SIM, et décrypte le message.

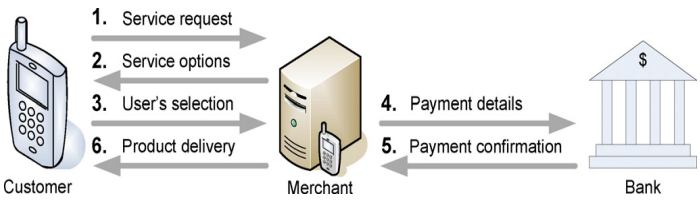

FIGURE 2. Modèle de paiement virtuel

Cette figure schématise le fonctionnement du paiement via téléphone mobile. 6 étapes sont nécessaires.

2.3 PKI pour (très vieux) mobiles

Le troisième article sur une PKI pour mobile [4]. Leur travail se reporte à d'anciens téléphones, avec très peu de mémoire vive et même un processeur très faible. Ainsi la génération de clés RSA leur est impossible. Ils décidèrent d'utiliser un autre algorithme de signature, plus adapté a leur type de téléphone, le ECDSA (Elliptic Curve Digital Signature Algorithm), avec des clés de 163 bits, équivalent à une clé RSA de 1024 bits, en terme de sécurité. De ce fait, les certificats ainsi générés sont eux aussi, plus petits.

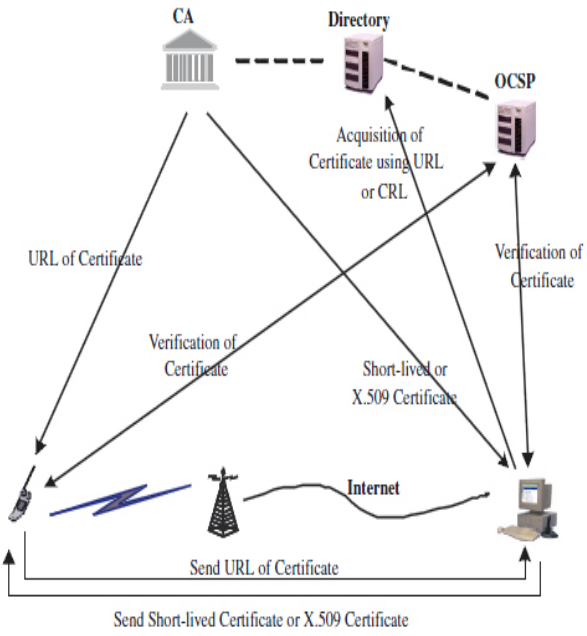

FIGURE 3. Schéma de leur système de PKI pour mobile

2.4 Kerberos et la non répudiation

Pour le premier article, offrant une solution Kerberos [5], met en avant la non-répudiation, en utilisant une méthode de hashage fonctionnant plus rapidement que les algorithmes de signatures numériques. Leur procédé fonctionne exactement comme le protocole Kerberos que nous utilisons sur nos ordinateurs. Le téléphone mobile demande à s'authenfier auprès du centre d'authentification. Puis aprèes quelques opérations, on fournit au téléphone un ticket, lui permettant d'accéder au service demandé au tout debut. Cependant, il est possible de réutiliser un ticket pour accéder au même service, a plusieurs reprises. Au niveau de la sécurité, il n'est pas possible de créer un ticket, car la signature de l'authorité de certifications n'est pas falsifiable. Un attaquant ne peux pas utiliser ou modifier un ticket émis. Les clés simétriques, utilisées lors de l'authentification sont regénérées à chaque session.

2.5 Kerberos et ses tickets transportables

Le second article offre lui une solution permettant à un téléphone ayant en sa possession un ticket d'itinérance, d'avoir accès aux services auquels il veut avoir accès, sans communiquer avec le serveur d'authentification à chaque fois.

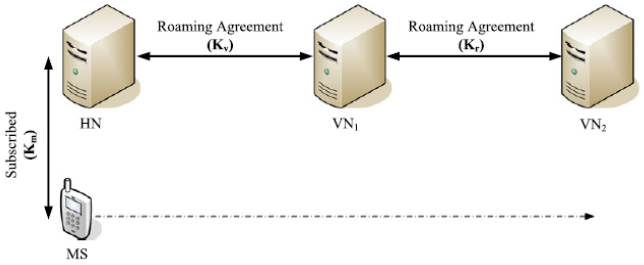

FIGURE 4. Représentation de l'itinérance

Pour ce faire, le téléphone s'authentifie auprès d'un réseau "visiteur", qui lui meme authentifie le téléphone auprès de son home réseau.

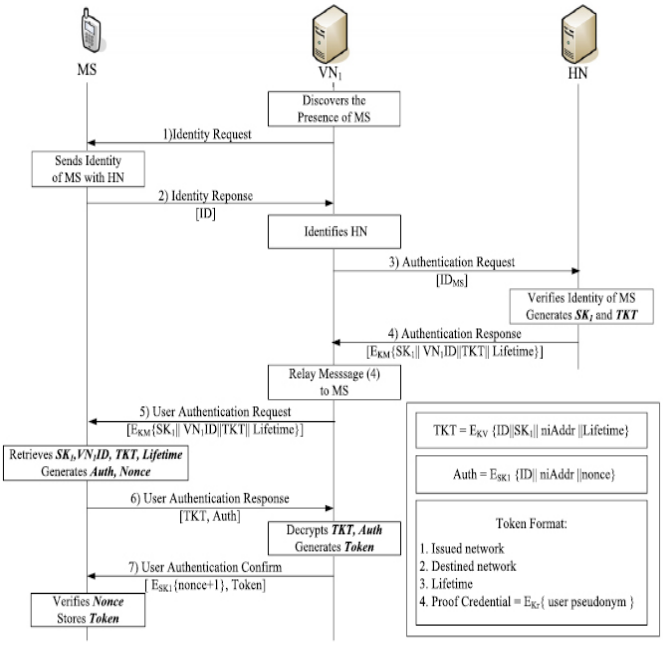

FIGURE 5. Authentification du téléphone auprès du réseau "visiteur"

3 Premières critiques des travaux existants

Premier constat, aucune solution de PKI ne supporte l'échange de données de téléphone à téléphone, seuls les échanges de données entre le téléphone et un serveur (banque, site de vente en ligne, etc.) sont gérés. D'un autre coté, les 3 solutions de PKI présentées, sont relativement complètes. Les systèmes de signatures sont réfléchis, les clés privées sont innaccessibles, même en cas de vol, puisque les clés sont stockées dans la carte SIM, qui est elle même protégée par un code PIN. De leur coté, les solutions Kerberos, présentées plus tôt ont eux aussi ce soucis de ne pas gérer les échanges entre deux téléphones.

Evidemment, chacunes des techniques PKI ou Kerberos présentent des désavantages qui sont propres à leur fonctionnement. Par exemple, lors d'un système Kerberos, les mots de passe utilisateurs sont enregistrés sur le serveur de manière non hashé, car le serveur a besoin du mot de passe en clair pour crypter la réponse à envoyer au client. De plus, Kerberos, de part son fonctionnement un serveur, peut poser problème si pour une raison ou une autre le serveur d'authentification n'est pas accessible. Dans le cas par exemple où le serveur a planté.

4 Objectifs et perspectives du projet de synthèse

Notre objectif était d'implémenter le déploiement automatique d'un système de certification pour mobile. Pour cela, nous devions implémenter une solution de PKI pour

mobile, et une solution Kerberos pour mobile. Ainsi, nous pouvons comparer de manière fonctionnelle les deux systèmes et choisir le plus efficace. Dans le cas de Kerberos, le but est de créer un petit client, capable de se connecter à un centre d'authentification Kerberos pour ensuite avoir un ticket pour un service demandé.

FIGURE 6. Nos début d'implementation

Nous avions commencé à implémenter une solution de PKI. Seules les méthodes de génération de clés, et de signatures étaient présentes. Le but de cette première implémentation était de pouvoir manipuler et vérifier le fonctionnement des fonctions de bases pour les clés publiques et privés.

Coté Kerberos, nous avions réussi à créer un serveur Kerberos, basé uniquement sur des utilisateurs crées manuellement, et à créer un client Java capable de se connecter sur ce serveur pour récupérer, après quelques échanges, un ticket de service. Selon nos espérances, la prochaine étape était de, premièrement, convertir le projet pour qu'il fonctionne sur Android, et ensuite de créer un service, compatible Kerberos, comme par exemple une session SSH, pour que le téléphone puisse se connecter dessus.

Malheureusement, nous n'avons pas pu réaliser cette partie du projet. Nous détaillerons tout cela, un peu plus tard.

5 Critères de comparaison

Notre sujet était d'exporter deux méthodes d'authentification pour machines traditionnelles sur des téléphones mobiles et de les comparer.

Nous avons décidé de tester et comparer ses deux protocoles sur quelques aspects :

– Implémentation
– Rapidité d'execution
– Interaction avec l'humain
– Sécurité du protocole
– Solidité du protocole
– Scalabilité

6 Problèmes rencontrés

6.1 La PKI

Nous n'avons pas rencontré de rééls problèmes pour le développement de notre client PKI. Les classes Java de bases et celle d'Android nous ont permis de réaliser cette partie sans soucis.

6.2 Kerberos

Comme nous l'avions mentionné dans notre premier rapport, nous sommes tombé sur un problème de taille, pour Kerberos. La création d'un serveur Kerberos n'était pas une masse à faire. Nombreux sont les tutoriaux sur Internet, mais aucun ne disait la même chose. Ainsi nous avons eu quelques soucis pour tout mettre en place côté serveur.

Mais ce n'a pas été notre plus grand problème. Malheureusement, nous n'avons pas pu réalisé ce que nous avions prévu, à savoir convertir notre client Java classique pour un client Android. En effet l'API de Kerberos mis à disposition fonctionne très bien avec le Java traditionnel que l'on connait tous, mais elle n'est pas apprécié par le SDK d'Android. Toute une série d'incompatiblité de paquets, de classes s'est révélée, nous empêchant totalement de developper un client Kerberos pour Android.

Nous aurions pu contourner le problème en réécrivant totalement les classes Kerberos que nous devions utiliser, ce qui aurait été un travail considérable.

7 Implémentation

7.1 La PKI

Nous avons du developper notre application de PKI pour android, ainsi qu'un serveur Java sur lequel notre application se connecte. Notre application android est complète, c'est à dire qu'il est capable de générer ses propres clés, les sauvegarder ou bien charger des clés deja stockées. Il est capable d'envoyer sa clé publique à notre serveur. Il peut récupérer les clés publiques de serveur. Et bien sur, il peut signer et crypter un message.

De son côté, le serveur peut réceptionner une clé publique, et ainsi déchiffrer et vérifier la signature du message envoyé via notre application et le message en lui même.

7.2 Kerberos

Pour notre partie Kerberos

8 Comparaison des deux systèmes d'authentification

8.1 Pour les PC

Il existe quelques comparaisons pour ces deux systemes que sont la PKI et Kerberos pour les PC. Notamment une comparaison par des tcheques [6]. Pour eux, ils différencient les systèmes cryptographiques en eux même. Pour Kerberos sysmétrique avec des tickets, pour la PKI asymétrique avec des certificats. Au final les concepts sont similaires. Kerberos a absolument besoin d'avoir son serveur d'authentification (KDC) en ligne pour fonctionner, la PKI peut se permettre une non disponibilité de sa CA (certification authority) pendant un instant. Kerberos fonctionne avec des passwords, alors la PKI utilise des clés privées. De ce fait la gestion n'est pas identique. La gestion des clés peut etre source de problèmes, ainsi que la revocation de ces clés, impossible pour Kerberos. En terme de déploiement, c'est un peu plus compliqué pour Kerberos, qui doit enregistrer chaque utilisateur, et gérer les tickets qui ont une durée de vie limitée et qui doivent donc être recrées. Dans la PKI, les clés n'ont pas besoin d'être regénérées, une fois les clés distribuées, il n'y plus de manipulation à faire. PKI a un net avantage en terme de cryptage d'e-mail, qui est commun, et facile à mettre en oeuvre avec Enigmail pour Thunderbird par exemple. D'après eux, la PKI est plus facile à gérer et à mettre en oeuvre, ils la recommandent donc.

Dans une seconde comparaison d'un docteur suisse [7],est synthétisé avec un tableau qui est très explicite.

	Kerberos-based	PKI-based
Security	+	+
Non-repudiation	--	++
Trust requirements	-	+
Complexity	-	o
Scalability	--	+
Interoperability	--	-
Application modifications	--	-
Vendor support	o	+
Future perspectives	-	+

FIGURE 7. Comparaison de Kerberos et de la PKI

Dans ces deux cas, on voit nettement que la PKI est plus intéressante à utiliser. Va-t-on avoir les mêmes résultats pour notre cas, sur mobile ?

8.2 Pour notre problème

Côté PKI, nous avons de très bons résultats. Nos tests ont été réalisés sur l'émulateur android fourni avec l'API et sur une tablette tactile fonctionnant sous android. Nous avons réalisé une série de test, notamment la génération des clés publiques et privée.

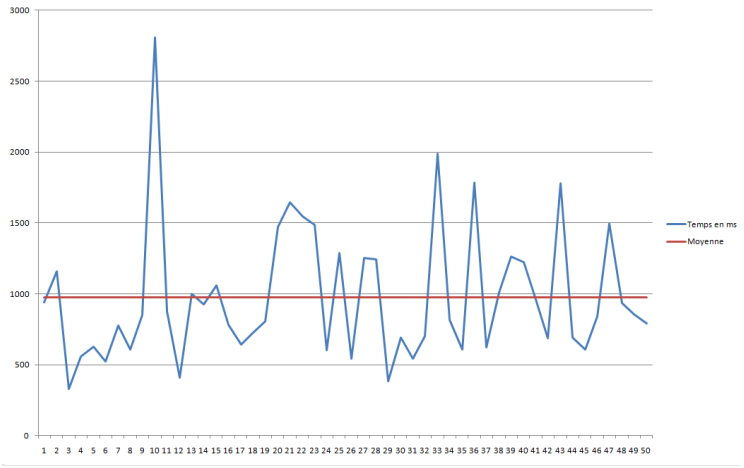

FIGURE 8. Génération des clés sur 50 essais

Cette figure montre sur 50 tentatives, le temps mis pour la génération de clés. En moyenne on s'aperçoit que l'application met un peu moins d'une seconde pour générer ses clés publique et privée.

Nous avons réalisé des tests sur le chiffrement d'un message "CHALLENGE", pour faire des statistiques en terme de performances de temps. Lors de ces tests, nous avons mesuré, le temps de signature du message avec la clé privée de notre application Android, le temps de chiffrement du message avec la clé publique du serveur, et le temps de transmission du message signé et chiffré au serveur.

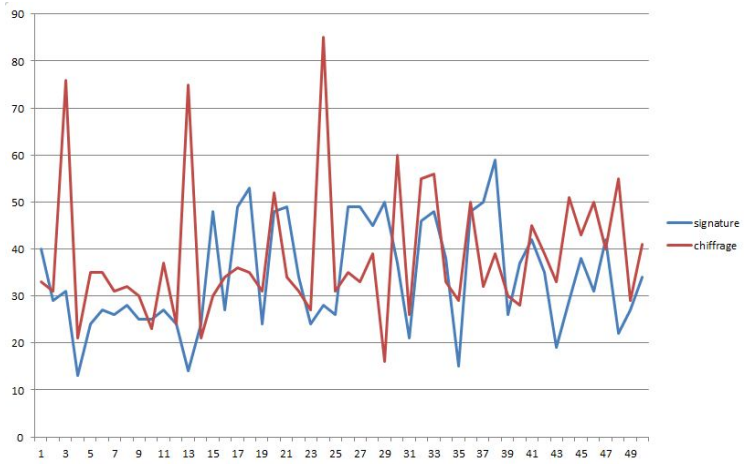

FIGURE 9. 50 essais de signature et de chiffrement du message "CHALLENGE"

Nos tests d'envoie du message au serveur se sont revelés plus excellent, d'une extreme rapidité.

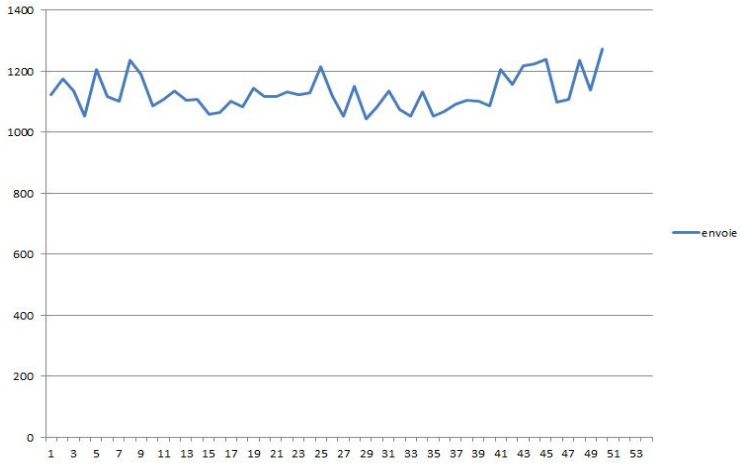

FIGURE 10. 50 essais d'envoie du message

Comme mentionné plus haut, en

Références

1. M. Assora, J. Kadirire, and A. Shirvani, “Using wpki for security of web transaction,” *Lecture Notes in Computer Science, 2007*, vol. 4655, pp. 11–20, 2007.
2. *WMLScript Crypto API Library : WAP-161-WMLScriptCrypto-20010620*.
3. M. Hassinen, K. Hyppönen, and E. Trichina, “Utilizing national public-key infrastructure in mobile payment systems,” *Electronic Commerce Research and Applications*, vol. 7, pp. 214–231, 2008.
4. Y. Lee, J. Lee, and J. Song, “Design and implementation of wireless pki technology suitable for mobile phone in mobile-commerce,” *Computer Communications*, vol. 30, pp. 893–903, 2006.
5. Y. Lei, A. Quintero, and S. Pierre, “Mobile services access and payment through reusable tickets,” *Computer Communications*, vol. 32, pp. 602–610, 2009.
6. D. Kouril, L. Matyska, and M. Prochazka, “Kerberos and pki cooperation,” 2006.
7. R. Oppliger, “Authentication and authorization infrastructures : Kerberos vs. pki,”