ICARM : Infrastructure de Confiance pour les Architectures de Réseaux Mixtes

Mawloud Omar, Yacine Challal, Abdelmadjid Bouabdallah

ICARM : Infrastructure de Confiance pour les Architectures de Réseaux Mixtes

Mawloud OMAR
41, Cité des frères abbad
ouest, Chlef, Algérie.
ReSyD, Béjaia.
+21373451360
mawloud.omar@gmail.com

Yacine CHALLAL
BP 20529, 60205
Compiègne CEDEX France.
UTC, Lab Heudiasyc.
+33344234423
ychallal@utc.fr

Abdelmadjid BOUABDALLAH
BP 20529, 60205
Compiègne CEDEX France.
UTC, Lab Heudiasyc.
+33344234423
bouabdal@utc.fr

Résumé : *Dans cet article, nous proposons un modèle de confiance dans le cadre des architectures réseaux mixtes. Le modèle, repose sur l'utilisation de deux autorités de certification particulières, qui assurent la gestion des certificats X509v3. Des autorités centrales CCA (Central Certification Authority) qui se trouvent au niveau des réseaux ayant une infrastructure pré–existante (comme les réseaux filaires, mobiles cellulaires, etc.), et des autorités mobiles MCA (Mobile Certification Authority) au niveau du réseau ad hoc. Les MCA émulent le rôle du service de certification en utilisant la cryptographie à seuil. Tandis que les CCA délèguent le pouvoir de signature aux MCA. Cette solution décentralisée et partiellement distribuée, supporte la mobilité des noeuds et la défaillance de jusqu'à n-k+1 parmi n autorités MCA. Les résultats de simulation et l'évaluation de performances démontrent l'adéquation de cette solution aux réseaux mixtes : avec et sans infrastructures.*

Mots–clés : *Confiance, Architectures Mixtes, Autorité de Certification, Simulations.*

I. Introduction

L'utilisation des applications sur Internet pour des objectifs commerciaux a orienté de nombreux travaux de recherche sur la nécessité d'offrir des services de sécurité tels l'authentification, l'intégrité et la confidentialité des données, et a donné naissance à plusieurs groupes de travail. D'un autre coté, le besoin à plus de mobilité a rendu très répandu la notion de réseau sans infrastructure ou réseaux ad hoc. Les réseaux ad hoc comporte un ensemble de noeuds sans fils qui forment temporairement un réseau sans l'aide d'une infrastructure centralisée. En effet, les réseaux purement ad hoc ont peu d'applications, et qu'en général on trouve des architectures de réseaux mixtes (avec et sans infrastructures).

Pour assurer les services de sécurité, on doit s'appuyer essentiellement à un modèle de confiance. Beaucoup de travaux ont été proposés tel que les modèles qui reposent sur tierce partie de confiance comme avec les PKI [17], ou Kerberos [11]. Plusieurs modèles ont été également proposés pour les réseaux ad hoc. Chacun se base sur un raisonnement particulier, on trouve par exemple le modèle distribué [2], ou le modèle en graphe comme avec PGP [1]. Il y a également des modèles basés sur la cryptographie à seuil [21][13].

Dans cet article, nous proposons un modèle de confiance dans le cadre des architectures réseaux mixtes. Notre modèle, repose sur l'utilisation de deux autorités de certification particulières, qui assurent la gestion des certificats X509*v3* [8]. Des autorités centrales CCA (*Central Certification Authority*) qui se trouvent au niveau des réseaux ayant une infrastructure, et des autorités mobiles MCA (*Mobile Certification Authority*) au niveau du réseau ad hoc. Les MCA émulent le rôle du service de certification en utilisant la cryptographie à seuil. Tandis que les CCA délèguent le pouvoir de signature aux autorités MCA. Cette solution décentralisée et partiellement distribuée, supporte la mobilité des noeuds et la défaillance de jusqu'à n-k+1 parmi n autorités MCA. Les résultats de simulation et l'évaluation de performances démontrent l'adéquation de cette solution aux architectures réseaux mixtes : avec et sans infrastructures.

L'article est composé de quatre sections. Nous présentons dans la première section une introduction générale. Dans la deuxième section, nous présentons un état de l'art des modèles et infrastructures de

confiance existants dans la littérature. Dans la troisième section, nous présentons notre modèle pour les architectures réseaux mixtes, et nous présentons les résultas de simulation. Nous terminons par la quatrième section avec une conclusion générale et quelques perspectives.

II. Etat de l'Art

Un modèle de confiance [14] fournit un cadre de travail pour la construction et l'administration de la relation de confiance entre les nœuds d'un réseau. Selon l'ITU–T X509, le terme confiance est définie comme suit : « *Généralement on peut dire qu'une entité fait confiance à une deuxième entité si seulement si cette dernière se comporte exactement comme la première le prévoit* » [10]. De cette définition, nous pouvons voir que ceci inclut un rapport (ou une relation) entre les deux entités. Nous pouvons imaginer, alors, plusieurs cas de relation de confiance :

- A fait confiance à B, mais B n'a pas besoin de faire confiance à A.
- A fait confiance à B, et B fait confiance à A.
- A fait confiance à C, et B fait confiance à C, donc A et B peuvent faire confiance à C.

Les relations de confiance dans la plupart des architectures de sécurité se rentrent dans le troisième cas, où C est une entité spéciale dont tous les nœuds font confiance (pour l'infrastructure à clé publique, par exemple, on parle d'autorités de certification). Dans ce cas-ci, on dit qu'un tiers est employé pour établir la relation de confiance. Il y a aussi certains schémas qui reposent sur l'aspect collaboratif entre les nœuds pour établir la confiance dans le réseau, où la confiance change suivant le comportement des utilisateurs.

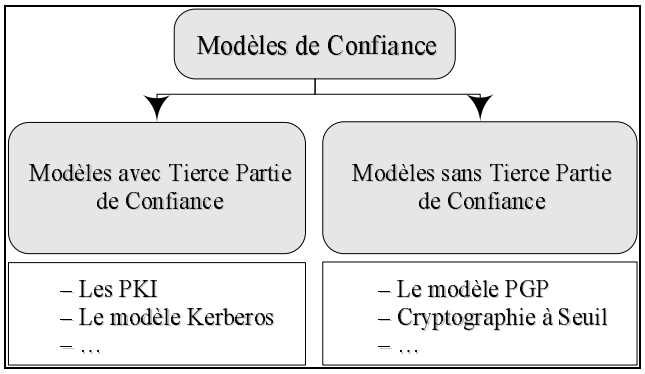


Fig.1 : Classification des Modèles de Confiance

Les modèles de confiance sont employés, généralement, pour établir l'authentification, l'identification, la confidentialité, l'intégrité, et l'autorisation. Plusieurs travaux de recherche ont été menés au sujet des modèles de confiance, et ont abouti à deux types d'approches : Modèles avec tierce partie de confiance, et Modèles sans ou faible dépendants à une tierce partie de confiance (cf. Fig.1).

A. Modèles avec Tierce Partie de Confiance

Bien qu'il soit possible de définir des protocoles de sécurité entre des entités paires, tous les protocoles utilisés en pratique s'appuient sur un troisième partenaire. Celui–ci possède deux propriétés : il est le gardien des données qui permettent d'authentifier les participants d'un échange (clé symétrique, clé publique, mot de passe, … etc.), et il certifie la validité de l'association entre un nom d'identité et la donnée correspondante. Ce partenaire s'appel, dans les systèmes à chiffrement symétrique, le *gardien des clés*, et dans les systèmes à clés publiques, l'autorité de certification. Les PKI (*Public Key Infrastructure*) [17][3], et Kerberos [15][11] sont parmi les modèles qui s'appuient sur une tierce partie de confiance.

1) Les PKI (*Public Key Infrastructure*)

On appelle PKI (*Public Key Infrastructure*, ou infrastructure à clé publique : ICP, parfois infrastructure de gestion de clés : IGC) l'ensemble des solutions techniques basées sur la cryptographie à clé publique [17][3]. En effet, les systèmes de cryptage à clés publiques permettent de s'affranchir de la nécessité d'avoir recours régulièrement à un canal sécurisé pour s'échanger les clés. En revanche, la publication de la clé publique à grande échelle doit se faire en toute confiance pour assurer que : la clé publique est bien celle de son propriétaire, le propriétaire de la clé est digne de confiance, et la clé est toujours valide. Ainsi, il est nécessaire d'associer aux paires de clés *(publique/privée)* un certificat délivré par une tierce partie de confiance. Le tiers de confiance est une entité appelée autorité de certification (*Certification Authority*, ou CA) chargée d'assurer la véracité des informations contenues dans le certificat à clé publique et de sa validité. Pour ce faire, l'autorité signe le certificat de clé publique à l'aide de sa propre

clé privée en utilisant le principe de signature numérique. Le rôle de PKI est multiple : Enregistrer les requêtes des clés en vérifiant l'identité des demandeurs, Générer les paires de clés *(publique/privée)*, Garantir la confidentialité des clés privées, Certifier l'association entre chaque utilisateur et sa clé publique, et Révoquer des clés (perte par son propriétaire, expiration ou de compromission).

Une PKI est en règle générale composée de deux entités distinctes :

- **Autorité d'enregistrement :** Chargée de la gestion des requêtes d'utilisateurs.
- **Autorité de certification :** Chargée de la création et la signature de certificats. Elle est chargée, également, pour la signature des listes de révocations CRL (*Certificate Revocation List*).

2) Le Modèle de Kerberos

Kerberos [15][11] (Cerbère, le chien à trois têtes, gardien de l'enfer) est un protocole d'authentification à tierce partie de confiance. Un service Kerberos, résidant dans le réseau, agit comme un arbitre de confiance. Le système est basé sur l'utilisation de la cryptographie à clé symétrique. Kerberos partage une clé secrète différente avec chaque entité dans le réseau. Comme Kerberos connaît la clé secrète de tout le monde, il peut créer des messages pour convaincre une entité de l'identité d'une autre identité. Kerberos permet aussi de créer des clés de session qui sont données aux clients et aux serveurs. En effet, Kerberos constitue actuellement le standard des systèmes de distribution des clés symétriques, et il est très utilisé dans Internet. Le système Kerberos se compose en deux éléments, d'une part un serveur Kerberos et d'autre part un service de délivrance de ticket. Les deux éléments communiquent par une liaison sûre. Un client demande au serveur Kerberos un ticket pour accéder au service de délivrance de tickets TGS (*Ticket Granting Service*). Ce ticket est appelé TGT (*Ticket Granting Ticket*), Kerberos le chiffre avec la clé secrète du client. Le client demande ensuite au TGS un ticket pour un serveur particulier. Si le client a le droit d'accès à ce serveur, le TGS lui retourne le ticket demandé. Un ticket de service est valable pour un seul serveur et un seul client. Il contient le nom du client, son adresse réseau, le nom du serveur, une datation, et une clé de session. Il est chiffré avec la clé secrète du serveur. Le client ne peut pas déchiffrer ce ticket, mais il l'utilise chaque fois qu'il désire accéder au serveur jusqu'à ce que sa date de validité soit expirée. Le serveur en recevant le ticket peut alors vérifier l'identité du client de façon sûre.

B. Modèles sans Tierce Partie de Confiance

Avec les modèles centralisés, les utilisateurs appuient sur une partie digne de confiance, qui calcule des valeurs de confiance pour chaque utilisateur dans le système. Tous les utilisateurs du système sollicitent cette partie de confiance pour leur fournir des informations sur d'autres utilisateurs. De ce fait, le serveur de la confiance devient un point d'échec, s'il est compromis alors tout le système est compromis. La version décentralisée du problème correspond à chaque utilisateur étant *le centre de son propre monde*. C'est-à-dire, la fonction de confiance est distribuée entre plusieurs entités dans le réseau, comme le cas du modèle PGP [1][4], et les modèles basant sur la cryptographie à seuil [21][13].

1) Le Modèle PGP (*Pretty Good Privacy*)

PGP (*Pretty Good Privacy*) [1][4] a été crée dans un but précis, offrir à tout le monde un moyen de préserver la confidentialité des informations. Celles–ci, peuvent être des messages de courriers électroniques (L'usage le plus fréquent de PGP), des fichiers que l'on souhaite archiver, ou encore des documents dont on souhaite garantir. Il a été créé par P. Zimmermann [20][6]. En effet, plusieurs modèles ont été proposés basés sur PGP. J. Hubaux est parmi qui ont beaucoup contribués pour assurer la confiance pour les réseaux ad hoc en basant sur le modèle PGP. Il a proposé une solution [7] de gestion de clés publique, dans un sens où les certificats sont délivrés par les utilisateurs eux–mêmes. Ceci sans participation de n'importe quelle autorité de certification centrale. A la différence des solutions à base de clé publique, celle–ci est conçue pour les réseaux ad hoc, où les noeuds n'ont aucun rapport antérieur. Chaque utilisateur a les possibilités de certifier des clés publiques à d'autres utilisateurs. Cependant, les auteurs ne reposent pas dans leur approche sur les annuaires des certificats. Au lieu de cela, les certificats sont stockés et distribués par les utilisateurs eux-mêmes. Chaque utilisateur maintient un dépôt local de certificats, qui contient un nombre limité de certificats choisis par l'utilisateur selon un algorithme. Ils ont

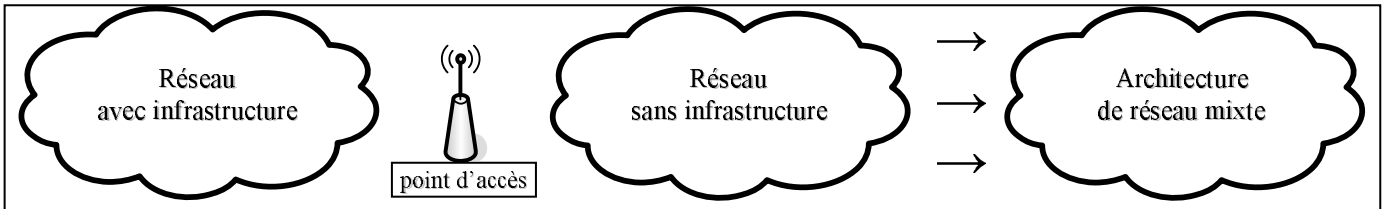
modélisés leur approche via un graphe orienté *G(V,E)*, appelé graphe de confiance (*Trust Graph*). Les relations entre les nœuds sont présentées par des certificats. *V* et *E* représentent, respectivement, les sommets et les arcs du graphe. Une chaîne de certificats de nœud A vers un nœud B, est représentée par un chemin du sommet A vers le sommet B dans *G*. Chaque utilisateur maintient un dépôt local de certificats, qui contient : des certificats crées par l'utilisateur lui–même, et des certificats sélectionnés crées par d'autres utilisateurs dans le système. Chaque nœud possède un graphe local (représente le dépôt local). Quand A désir vérifier la clé de B, A et B fusionnent leurs dépôts locaux (leurs graphes locaux), et A essaye de trouver une chaîne de certificats de A vers B dans le dépôt fusionné (graphe fusionné).

2) La Confiance avec la Cryptographie à Seuil

Parmi les méthodes de sécurité proposés pour les réseaux ad hoc, il existe une méthode basée sur un principe de cryptographie apparu dans les années soixante dix, la cryptographie à seuil [9]. Le principe est purement mathématique, et a été combiné avec d'autres techniques pour obtenir un modèle de sécurité pour les réseaux ad hoc. La cryptographie à seuil, propose un schéma de gestion de clés par distribution de confiance sur un agrégat de noeuds. Dans ce modèle, le service de gestion de clés a une configuration d'un schéma de cryptographie à seuil (*k*,*n*). Ceci représente un système de *n* noeuds serveurs qui partagent le pouvoir de signer des certificats aux utilisateurs. La clé privée du service est divisée en *n* parts. Pour signer un certificat, chaque serveur génère une signature en utilisant sa part de clé, et le tout est soumis à un combinateur qui est en mesure de calculer la signature du certificat. Un serveur compromis peut générer une signature partielle incorrecte. L'utilisation d'une telle signature partielle génère une signature invalide du service. Le combinateur peut vérifier la validité d'une signature calculée, en utilisant la clé publique du service. Plusieurs travaux reposent sur l'utilisation de la cryptographie à seuil pour établir la confiance dans le réseau ad hoc, on cite COCA [21], MOCA [19], DICTATE [12].

C. Evaluations & Motivation

Si on arrive hybrider deux solutions on pourra aboutir un modèle qui emploi à la fois deux politiques de confiance différentes pour les deux types d'architectures de réseau. Dans ce cas, un protocole doit être établie pour gérer le passage d'un modèle vers l'autre. Dans le cadre de ce but, nous avons étudié certaines possibilités d'hybridations entres les modèles cités à l'état de l'art (cf. Fig.2).



Solutions centralisées	Hybridation	Solutions distribuées	Le passage à une solution mixte
Kerberos	⊕	PGP	(–) Le chiffrement asymétrique avec PGP alourdit le système dans le réseau ad hoc. (–) Passage d'un système à chiffrement symétrique vers un système à chiffrement asymétrique et vise versa.
PKI	⊕	PGP	(–) Le chiffrement asymétrique avec PGP alourdit le système dans le réseau ad hoc.
Kerberos	⊕	Cryptographie à Seuil	(–) Passage d'un système à chiffrement symétrique vers un système à chiffrement asymétrique et vise versa.
PKI	⊕	Cryptographie à Seuil	(+) La charge de calcul de partage d'un secret au niveau du réseau ad hoc est remédiée par l'utilisation d'une autorité de certification du modèle PKI.

Fig.2 : Evaluations & Motivation

III. Notre Modèle

L'approche la plus simple est de mettre en oeuvre un modèle centralisé qui introduit un serveur dédié pour assurer la confiance. La mise en oeuvre d'un tel modèle introduit des points de vulnérabilités dans le système, car si l'entité centrale est corrompue tout le service est corrompu. De ce fait, le modèle proposé pour les architectures réseau mixtes est défini à la base d'une architecture décentralisée. L'objectif de base est de définir un modèle de confiance qui satisfait les propriétés suivantes :

- **Décentralisation du Service :** Les services de sécurité doivent être garantis en absence d'une entité centrale. Le modèle de confiance peut qu'il soit centralisé au niveau du réseau filaire. Cependant, au niveau du réseau ad hoc le service doit être décentralisé en utilisant plusieurs serveurs.
- **Support de Mobilité :** Le coté du réseau ad hoc est caractérisé par sa dynamicité. Ainsi, le modèle de confiance doit alors s'adapter pour supporter la mobilité et le dynamisme des utilisateurs.
- **Disponibilité du Service :** En effet, le coté du réseau ad hoc est vulnérable et sujet aux pertes de connexions et aux partitions du réseau, et pour cela il faut mettre en place plusieurs serveurs pour assurer la disponibilité du service de sécurité.
- **Sécurité et Facteur d'Echelle :** Le modèle doit assurer les propriétés de sécurité tel que : l'authentification, l'intégrité, la confidentialité, et la non répudiation. Un autre critère qui met en valeur le modèle c'est le facteur d'échelle (ou scalabilité), qui interprète la capacité du modèle à s'adapter à l'évolution des utilisateurs dans le système.

A. Description du Modèle

1) Aperçu du Modèle

L'architecture centralisée représente une grande vulnérabilité aux attaques (attaques passives, attaques actives). Ainsi, notre modèle essaye de réduire cette vulnérabilité en faisant appel à une solution distribuée, où plusieurs serveurs assurent le service de l'autorité de certification. Pour cela, le modèle de confiance proposé fait l'objet d'un rôle distribué. Les tâches de l'autorité de certification sont réparties sur deux types de serveurs :

- Autorités mobiles MCA (***M**obile **C**ertification **A**uthority*).
- Autorités centrales CCA (***C**entral **C**ertification **A**uthority*).

Les autorités MCA partagent le pouvoir de signer les certificats en utilisant les techniques de Cryptographie à Seuil. La clé privée du service de certification est partagée en n parts, où chaque autorité MCA en détiendra une. De cette manière, pour signer un certificat, chaque MCA génère une signature partielle du certificat en utilisant sa part de clé. Un ensemble d'autorités MCA est capable de générer une signature complète du certificat avec la clé privée du service, et ceci à partir de la combinaison des signatures partielles, et sans qu'aucun d'eux ne connaisse nettement la clé privée du service.

Un schéma de (k,n) est utilisé, ceci signifie que pour avoir un certificat il faut au minimum k certificats partiels auprès des autorités MCA (Plus de n-k+1 serveurs MCA doivent être compromis pour pouvoir compromettre tout le système). Les autorités CCA jouent, dans le modèle, un double rôle. Le premier rôle, c'est de se comporter exactement comme une autorité de certification habituelle, alors sa responsabilité sera de signer, révoquer, mettre à jour les certificats pour les utilisateurs. Le deuxième rôle sera le commandement sur les autorités MCA.

A la phase d'initialisation, les autorités CCA distribuent les n parts de la clé privée du service aux autorités MCA. Cependant, les serveurs MCA vont émuler le rôle de l'autorité de certification pour les utilisateurs n'ayant pas l'accès aux autorités CCA à travers les points d'accès au réseau filaire. Au niveau du réseau ad hoc, les protocoles de certification sont exécutés avec la participation de n autorités MCA, qui peuvent être corrects ou compromis. Un serveur compromis peut, en effet, arrêter ou d'éviter son exécution, et/ou rendre l'information enregistrée dans le serveur inaccessible. Compromettre n-k+1 autorités MCA se fait dans un temps limité, appelé *fenêtre de vulnérabilité*. La période de rafraîchissement des partages doit être estimée sur la base qu'au maximum n-k+1 autorités MCA soit compromises. La Fig.3 nous montre un aperçu global de notre modèle.

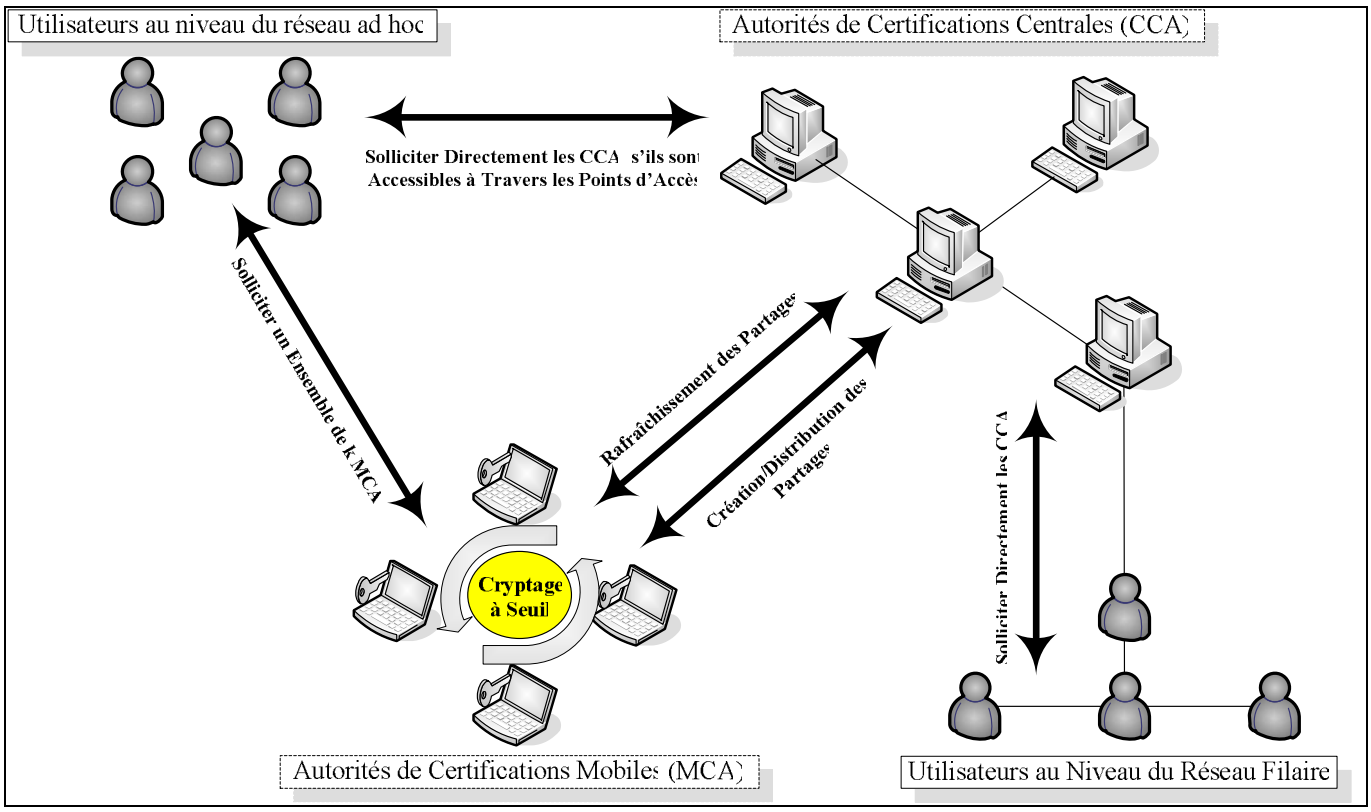


Fig.3 : Aperçu du Modèle

Dans notre modèle, les autorités MCA et CCA exécutent la même politique de certification pour la construction des certificats. Ceci signifie, que les certificats partiels délivrés par chacune des autorités MCA sont identiques, pour un utilisateur donné. Le rôle du client consiste, seulement, à vérifier la signature numérique de l'autorité. Si le client fait partie du réseau ad hoc, et aucun serveur CCA n'est lui accessible, il doit combiner au minimum *k* certificats partiels auprès des autorités MCA. La combinaison des certificats partiels se fait en se basant sur les techniques d'interpolations inspirées du schéma de partage de Shamir [18], et cela pour vérifier la signature numérique du service de certification.

2) Traitement des Requêtes de Certification

Le format des certificats utilisé dans notre modèle est conforme au certificat X509*v3* [8]. En effet, la signature des certificats se fait par, soit une autorité CCA, soit par un ensemble de *k* autorités MCA. Pour ne pas créer les conflits au niveau de la vérification de la signature, le signataire doit indiquer la nature du certificat. Avec cette façon, les utilisateurs peuvent faire la différence entre un certificat et un certificat partiel. Dans ce contexte, la partie « *extensions* » est employée pour enregistrer les indications concernant le signataire. Le format des certificats utilisé dans le cadre notre modèle :

- **Version :** Contient la valeur *v3* (Certificat X509*v3*).
- **ID :** Identificateur unique du certificat.
- **ID Algorithme de Signature :** Le nom de l'algorithme de signature.
- **Emetteur :** Le nom unique de l'émetteur du certificat.
- **Sujet :** Le nom unique du détenteur du certificat.
- **Clé Publique :** La clé publique du détenteur du certificat.
- **Période de Validité :** La date de début et d'expiration du certificat.
- **Attributs :** Extensions optionnelles.
- **Extensions :** Si le certificat est signé par une MCA, on doit indiquer ici que c'est un certificat partiel.
- **Signature :** Contient la signature de l'autorité de certification. Si on trouve sur la partie « *extensions* » qu'il s'agit d'un certificat partiel, le détenteur doit conserver ce certificat jusqu'à avoir *k* certificats partiels, pour pouvoir ensuite les combiner et avoir un certificat signé par la clé privée du service.

Au niveau du réseau ad hoc, pour récupérer un certificat, une requête est envoyée en diffusion aux autorités MCA (cf. Fig.4). Chaque serveur génère une signature partielle du certificat en utilisant sa part

privée et envoie le certificat partiel au client, et ce dernier combine l'ensemble des signatures partielles pour avoir la signature complète. Lorsque le client obtient un sous ensemble de *k* signatures correctes, il devient capable de générer un certificat signé par la clé privée du service de l'autorité de certification.

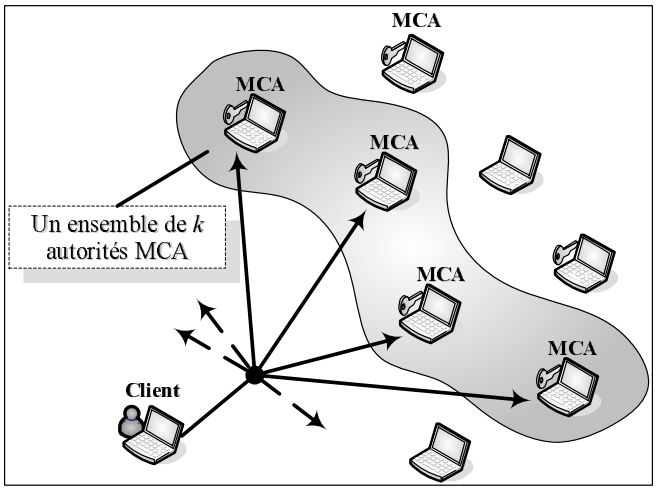


Fig.4 : Traitement des Requêtes par les MCA

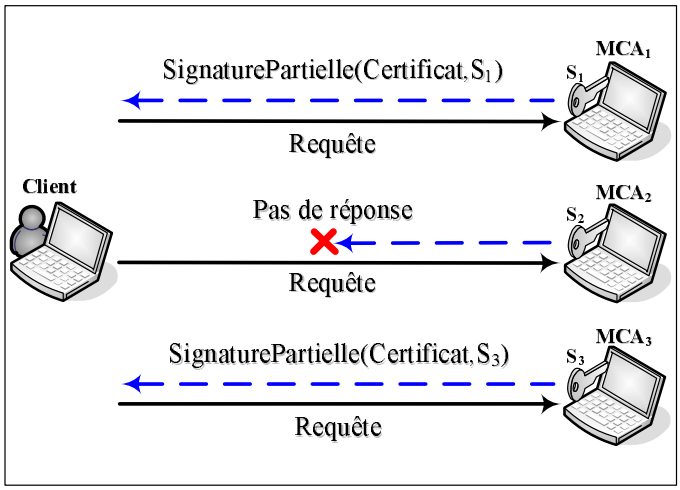


Fig.5 : Un Schéma de Cryptographie à Seuil (2,3)

Les serveurs compromis ne peuvent pas générer correctement des signatures. Sur la Fig.5, on prend un exemple d'un schéma de (2,3) du modèle. Cela signifie que le système comporte trois autorités MCA pour k=2. Chaque autorité MCA_i dispose un partage S_i de la clé privée du service. L'autorité MCA_i peut générer une signature partielle *SignaturePartielle(Certificat,*S_i*)* en utilisant sa part S_i. MCA_1 et MCA_3 génèrent des signatures partielles et les envoient au client. Bien que le serveur MCA_2 n'arrive pas à envoyer sa signature partielle (pour une raison ou une autre), le client est capable de générer la signature *SignaturePartielle(Certificat,S)* avec la clé privée du service, car seulement deux signatures partielles correctes suffissent pour avoir la signature du service. Avec cette façon, notre modèle résiste au comportement des serveurs compromis. En effet, un serveur compromis peut générer une signature non valide que le client peut vérifier en utilisant la clé publique du service (La clé publique de l'autorité de certification est connue par tous le monde). Dans le cas où la vérification échoue, le client doit choisit un autre ensemble de *k* certificat partiels.

3) Rafraîchissement des Partages

Notre modèle utilise le rafraîchissement des partages pour tolérer les attaques des *Adversaires Mobiles*. La notion d'adversaire mobile a été initialement étudiée dans [16] pour caractériser les adversaires qui, temporairement, corrompent un serveur et passent à un autre serveur victime (par exemple, un virus injecté dans le réseau). De ce fait, un adversaire est capable de corrompre tous les serveurs MCA au bout d'un certain temps, cette période est appelée fenêtre de vulnérabilité. Bien que les serveurs compromis, dans notre modèle, soient détectés seront exclus du service, l'adversaire peut continuer dans le temps à collecter plus de *k* parts à partir de nouveaux serveurs compromis. De cette manière, il devient capable de reconstruire la clé privée du service et de signer des certificats incorrects. Le rafraîchissement dans notre modèle se fait par le biais des autorités CCA qui vont recréer et redistribuer les parts d'une nouvelle clé privée aux autorités MCA dans une période définit de temps. Le nouveau partage constitue également un partage (*k*,*n*). Après le rafraîchissement, les autorités MCA suppriment les anciens partages et vont utiliser les nouveaux pour la signature des certificats. Les nouveaux partages doivent êtres indépendants des anciens. L'adversaire ne doit pas être capable de combiner les nouveaux et les anciens partages pour découvrir la clé privée du service. Par conséquent, l'adversaire est obligé de corrompre *k* autorités MCA dans le temps relatif à la période de rafraîchissement.

B. Résultats de Simulations

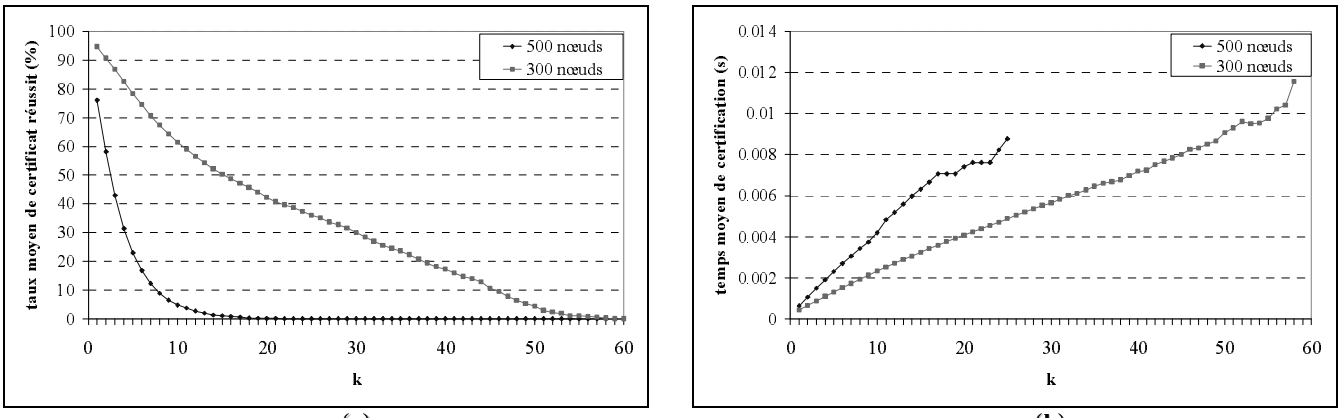
Nous allons à présent, présenter les simulations que nous avons effectué pour évaluer les performances de notre modèle. Nous avons opté pour une durée de simulation d'une heure. Les requêtes des clients arrivent aux autorités de certifications selon une loi de *Poisson* avec une moyenne de 10sec. Pour la

mobilité, le changement de topologie suit également une loi de *Poisson* avec une moyenne de 5mnt. Le simulateur estime si un lien radio existe entre deux noeuds quelconques en fonction de la distance qui lès sépare. Chaque noeud possède une portée de signale de 35m, et se déplace sur une surface rectangulaire de 1km². Les nœuds sont configurés par des interfaces de communication sans fil avec un débit de 22Mbs. La vitesse de déplacement est un paramètre variable entre 0 et 20m/s. La simulation est faite sur la base d'un nombre de noeuds allant de 300 à 500 nœuds, leurs positions initiales étant aléatoire sur la grille. Nous avons utilisé un modèle de mobilité particulier, hérité du modèle *Random Waypoint* [5], ou les nœuds se déplacent d'une façon probabiliste de telle sorte qu'on garde le mouvement condensé au niveau du centre de la surface. Pour des raisons de simplicité, nous supposons que tous les nœuds ont les mêmes caractéristiques matérielles et mêmes puissances de traitement.

Nous avons simulé le modèle en suivant deux axes différents. En premier lieu, nous optons pour un modèle de simulation où le service soit détaché des autorités CCA. Nous allons examiner le comportement du modèle, seulement, avec les autorités MCA au niveau du réseau ad hoc. Ensuite, nous allons mesurer, à nouveau, les performances avec la mise en jeu des autorités CCA. Les critères visés sont : le nombre moyen de certificats réussis, et le délai moyen de réponse.

1) Modèle de Simulation sans Participation des CCA

Dans un premier temps, nous nous sommes intéressés à étudier l'impact du paramètre k sur les performances du modèle. Rappelons qu'un schéma de (k,n) comporte n autorités MCA. Les utilisateurs peuvent satisfaire leurs requêtes de certification, si et seulement s'ils peuvent récupérer k réponses correctes. L'objectif à travers ce test, est de découvrir comment doit-on choisir le nombre de serveurs k^* optimal suffisant pour la participation à la signature. Pour cela, nous avons fait varier la valeur de k et nous avons exécuté d'intensives simulations pour chaque valeur.



(a) **(b)**

Fig.6 : Impact de k sur (a) le Taux Moyen de Certification. (b) le Délai Moyen de Certification. n=60

Comme nous pouvons bien le constater dans Fig.6 (a), lorsqu'on augmente la valeur de k, le taux de certification diminue. Au fait, lorsqu'on augmente k, les clients sont contrariés à récupérer un nombre augmenté de certificats partiels. Cela implique que les clients vont être obligé à solliciter beaucoup plus d'autorités MCA. A une certaine grandeur de k, les clients risquent à ne pas pouvoir récupérer k certificats partiels, et cela est dû à l'éventuelle indisponibilité des MCA. Par contre, si on réduit k, les clients vont solliciter un nombre étroit de serveurs, et ils auront beaucoup plus de chances pour leurs requêtes.

Pour un nombre de 300 nœuds, les certificats délivrés sont beaucoup plus réussis. Cependant, pour 500 nœuds, les performances s'affaiblissent. Nous constatons alors, que le choix de k doit être relatif au nombre d'autorités MCA et à la taille du réseau. Sur la Fig.6 (b) on voit que le délai d'attente agrandit lorsqu'on augmente k. Au faite, lorsque le client envoi sa requête, il doit attendre la réponse au minimum de k autorités MCA. Par conséquent, le temps de réponse augmente lorsque le nombre de serveurs à solliciter est important. D'autre part, quand on réduit k, le nombre de certificats réussis augmente et le délai de réponse diminue. Donc, pour rendre le modèle plus performant il faut réduire la valeur de k.

Ensuite, nous avons étudié l'impact de k sur la disponibilité du service de certification et sur le niveau sa solidité (confiance). La disponibilité du service est interprétée par la capacité à répondre aux requêtes

d'utilisateurs. Nous avons lancé une simulation sur des centaines de topologies de réseaux, ou nous avons mesuré pour chaque nœud, la disponibilité du service de certification en lui calculant le nombre d'autorités MCA accessibles. Si, au minimum, *k* MCA sont accessibles, alors le service est lui disponible.

La Fig.7 nous montre que le paramètre *k* influence sur la disponibilité du service et sur le niveau de sa solidité. Si *k* est petit, ça sera relativement moins difficile à compromettre les parts de la clé du service de certification. Dans ce cas, le niveau de la confiance diminue, mais la disponibilité du service devient plus en plus forte. Si *k* est grand, l'adversaire devra compromettre beaucoup plus d'autorités MCA pour pouvoir compromettre tout le système (Si *k* augmente alors la confiance augmente ; Si *k* diminue alors la confiance diminue), et alors, le niveau de la confiance augmente. Cependant les clients devront solliciter beaucoup plus d'autorités MCA pour pouvoir satisfaire leurs requêtes, alors la disponibilité du service s'affaibli. On constate finalement, que le choix de *k* doit accomplir un compromis entre la disponibilité du service et le niveau de la confiance.

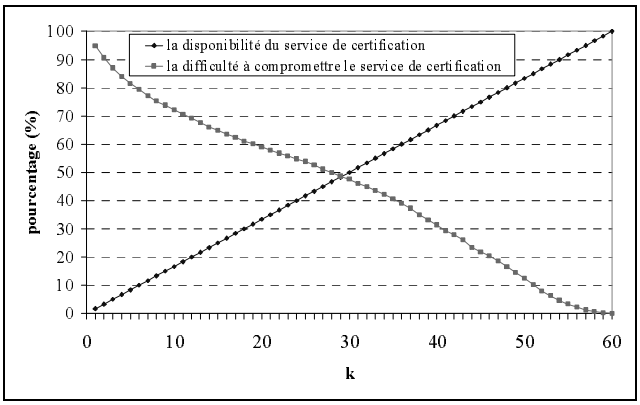


Fig.7 : Impact de *k* sur la Disponibilité et la Solidité du Service de Certification. *n*=60

Dans un deuxième temps, nous nous sommes intéressés à étudier l'impact de *n*. Nous avons fait varier *n* jusqu'à le nombre total des nœuds, et nous avons mesuré les performances pour *k*=*n*/3 et pour *k*=2*n*/3. Les résultats trouvés sur la Fig.8 (a) nous montrent que la relativité de *k* par rapport à *n* influe d'une façon très claire sur les performances du modèle. Le nombre de certificats réussis reste, approximativement, stable durant toute la variation de *n*. Cela veut dire, que le paramètre ayant plus le poids c'est surtout le rapport qui se trouve entre *k* et *n*. Même chose pour *k*=2*n*/3, mais avec des performances faibles à celle de *k*=*n*/3. Egalement sur la Fig.8 (b), les performances en temps de réponses se stabilisent, approximativement, durant toute la variation de *n*. Les résultats trouvés pour *k*=*n*/3 sont plus favorables que celle pour *k*=2*n*/3.

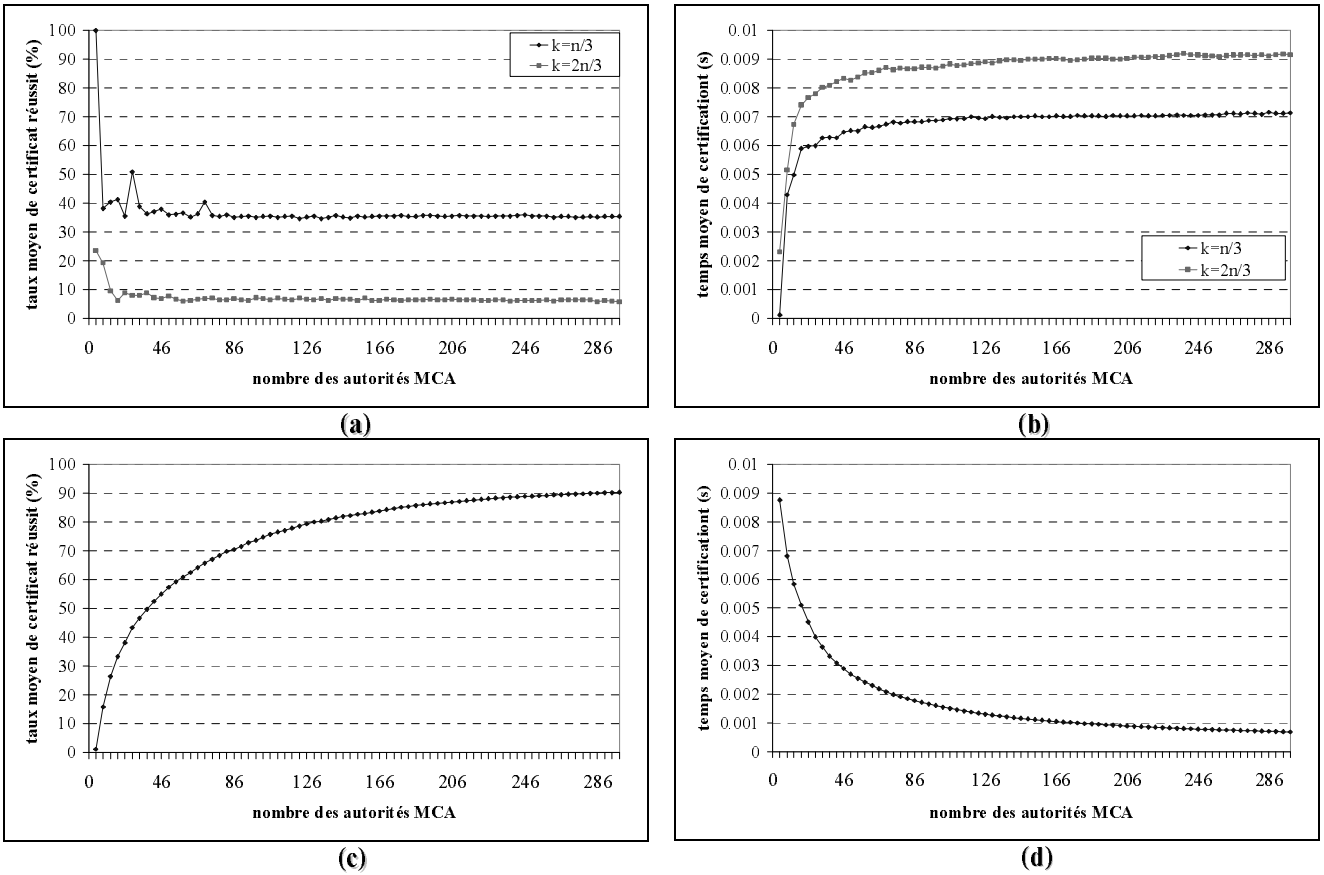
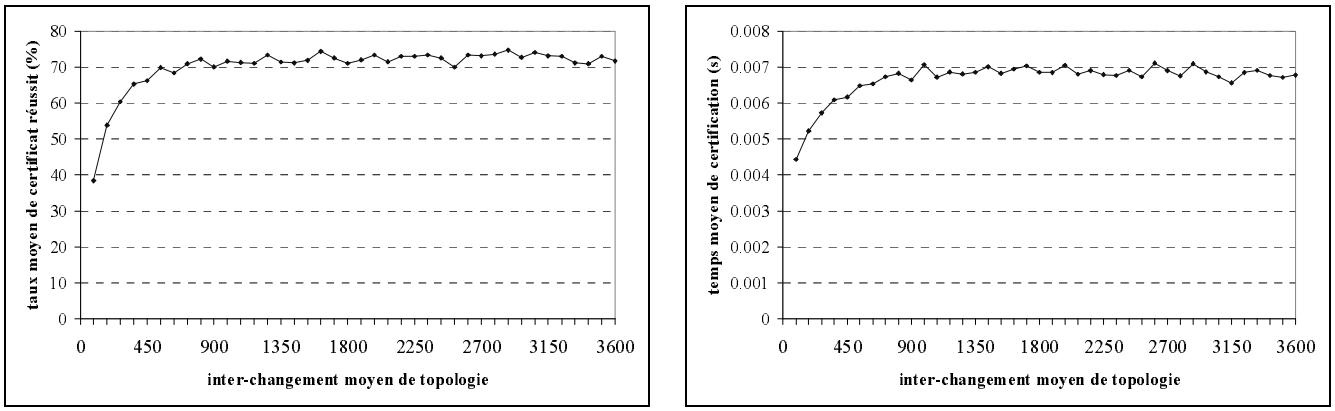


Fig.8 : Impact de *n* sur (a) le Taux Moyen de Certification. (b) le Délai Moyen de Certification. (c) le Taux Moyen de Certification avec *k*=10. (d) le Délai Moyen de Certification avec *k*=10. Taille=300 noeuds

On constate alors, que pour rendre le modèle plus performant, il faut qu'on *maximise n en minimisant k*. Pour confirmer, nous avons lancé une nouvelle simulation, et cependant, à cette fois-ci nous avons fixé k=10 serveurs. Nous avons varié n allant de k à 300 serveurs. Pour la même valeur de k (k=10), les résultats de performance se diffèrent.

La Fig.8 (c) nous montre que pour k=10, le nombre de certificats délivrés augmente lorsqu'on augmente le nombre d'autorités MCA. En effet, quand on augmente le nombre d'autorités MCA, les clients auront plus de chances à avoir à leurs disponibilités 10 serveurs MCA accessibles. Egalement, les clients auront plus les chances qu'on trouve des serveurs proches, et alors le délai de certification diminue (cf. Fig.8 (d)). Les résultats trouvés affirment que pour améliorer les performances du modèle, il faut maximiser n et minimiser k. Cependant, la valeur de k ne doit pas être trop petite, sinon on risque d'affaiblir la solidité du service de certification au niveau de la confiance.

Ensuite, nous nous sommes intéressé à étudier l'impact de la mobilité sur les performances du modèle. Rappelons que, tous les nœuds du réseau sont caractérisés par une configuration matérielle identique. Chaque nœud dispose d'une interface de communication sans fil, avec une portée de signale de 35m. Le changement de topologie suit une loi de *Poisson* avec un inter–changement moyen de topologie. Dans ce contexte, nous avons varié la moyenne de l'inter–changement de topologie de 3600sec à 90sec. En effet, si on réduit la valeur de l'inter–changement moyen de topologie, les nœuds vont suivre une forte mobilité.



(a) **(b)**

Fig.9 : Impact de la Mobilité sur (a) le Taux Moyen de Certification. (b) le Délai Moyen de Certification. k=10, n=60, Taille=300 noeuds

Les résultats trouvés sur la Fig.9 nous montrent que les performances du modèle diminuent lorsque la mobilité est trop forte, et en particulier à l'intervalle [90sec,360sec]. Au-delà de 360sec les performances restent, approximativement, stables.

1) Modèle de Simulation avec la Participation des CCA

A cette étape, nous avons introduit les autorités CCA sur le modèle de simulation. Au faite, les autorités CCA sont des autorités de certification centrales. Elles se trouvent au niveau des réseaux ayant une infrastructure préexistante. Les serveurs CCA ont le privilège d'être disponibles, en plus leurs bonnes configurations matérielles rendent le service beaucoup plus performant. Les autorités CCA jouent le rôle d'autorités de certification ordinaire, où ils signent et délivrent les certificats aux utilisateurs. En plus, les serveurs CCA délèguent le pouvoir de signature aux autorités MCA, ou ils leurs délivrent les partages de la clé du service de certification. Les utilisateurs qui se trouvent au niveau du réseau ad hoc peuvent solliciter directement les autorités CCA, si ces derniers sont accessibles à travers les points d'accès. Autrement, les utilisateurs devrant envoyer leurs requêtes aux serveurs MCA.

Nous nous somme intéressé dans cette partie à étudier les performances du modèle avec et sans les participation d'autorités CCA. Nous avons simulé deux modèles, le premier avec un schéma de (k,60). Le deuxième avec un schéma de (k,60) et 5 serveurs CCA, qui peuvent être accessibles ou non à travers les points d'accès. Nous avons varié la valeur de k en observant, respectivement, le taux moyen de certificats réussis et le délai moyen de certification sur les deux cas.

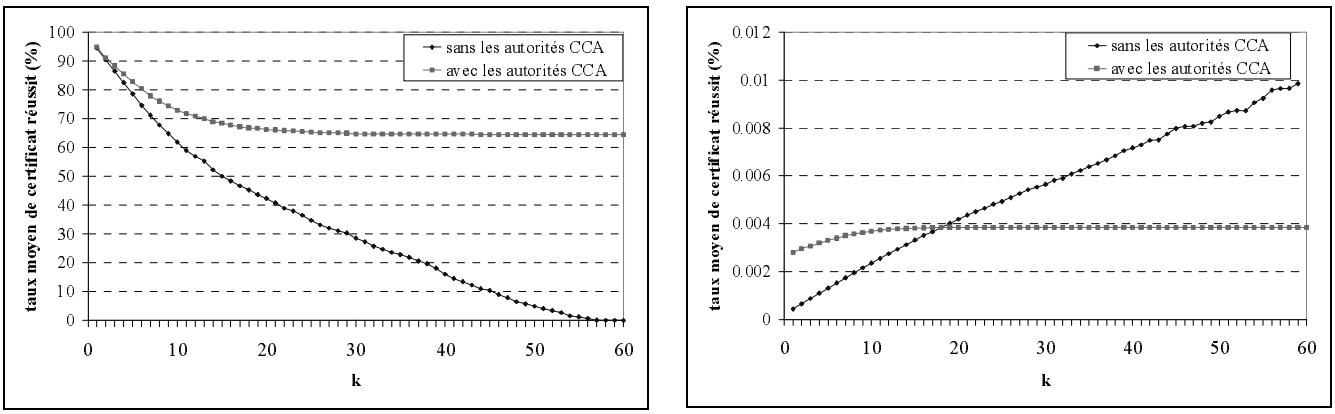


Fig.10 : Comparaison du Service avec et sans Autorités CCA sur (a) le Taux Moyen de Certification. (b) le Délai Moyen de Certification. *n*=60, Taille=300 noeuds

La Fig.10 (a) nous montre qu'avec les autorités CCA l'écart est largement claire en nombre de certificats réussis. Les utilisateurs ayant à leurs disponibilités l'accès à un serveur CCA peuvent y transmettrent leurs requêtes, et cela sans faire le recours aux autorités MCA. Les résultats présentés sur la Fig.10 (b) sont, également, favorables en délai de réponses pour le modèle attaché du service des autorités CCA.

Maintenant, nous nous intéressons à mesurer l'impact du nombre d'autorités CCA sur les performances du modèle. Nous avons opté pour un schéma de (10,60), avec un réseau de taille 300 nœuds.

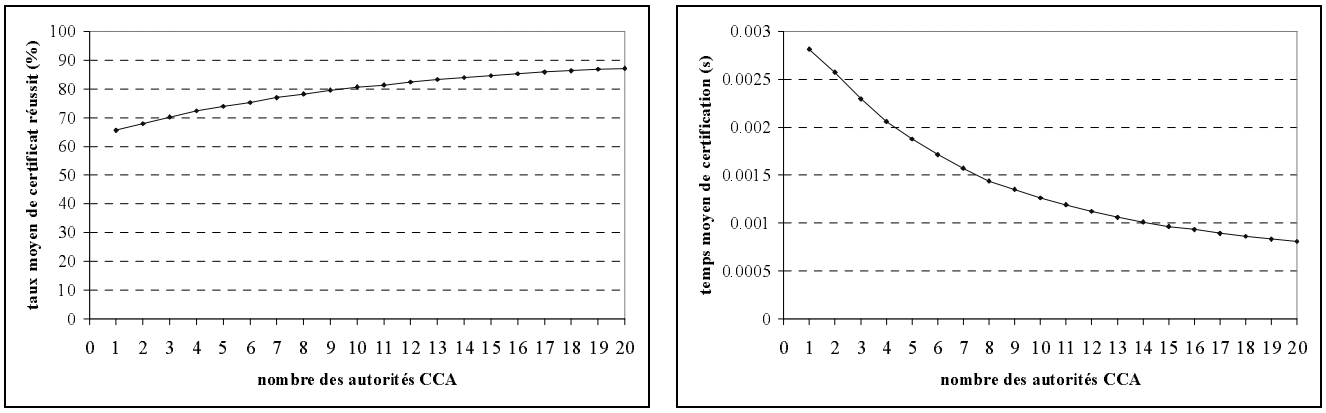


Fig.11 : Impact du Nombre de CCA sur (a) le Taux Moyen de Certification. (b) le Délai Moyen de Certification. *k*=10, *n*=60, Taille=300 noeuds

Comme nous pouvons bien le constater sur la Fig.11 (a), le taux de certificats réussis augmente lorsqu'on augmente le nombre de serveurs CCA, et ceci augment la disponibilité du service. Bien évidemment, le délai de réponse diminue (cf. Fig.11 (b)), car avec un nombre important d'autorités CCA, on augmente les chances qu'ils soient proches.

IV. Conclusions et Perspectives

Dans cet article, nous avons proposé un modèle de confiance pour les architectures de réseau mixtes. Le modèle proposé, repose sur des autorités de certification particulières, qui assurent la gestion des certificats X509*v3*. Nous avons présenté deux types d'autorités de certification distinctives. Des autorités centrales CCA qui se trouvent au niveau des réseaux ayant une infrastructure pré–existante, et des autorités mobiles MCA au niveau du réseau ad hoc. Les MCA émulent le rôle du service de certification en utilisant la cryptographie à seuil. Tandis que les CCA délèguent le pouvoir de signature des certificats aux MCA. Le modèle proposé décentralisé et partiellement distribué, supporte la mobilité des noeuds et la défaillance de jusqu'à n-k+1 parmi n autorités MCA. Pour mettre en valeur les privilèges du modèle, nous avons effectué des simulations intensives. Ces dernières ont montré que notre modèle fournit une grande flexibilité et plusieurs paramètres pour faire adapter l'architecture pour répondre aux besoins de

l'application utilisée, et l'évaluation de performances démontrent l'adéquation de cette solution aux réseaux mixtes : avec et sans infrastructures.

L'inconvénient le plus important dans notre modèle, est l'utilisation de la diffusion comme mécanisme de transmissions des requêtes au niveau du réseau ad hoc. Cet aspect va, en effet, créer une surcharge sur le réseau, et alors les performances se diminuent en ce qui concerne le délai de certification. Nous envisageons, que le prochain pas dans nos futurs travaux est d'adapter notre modèle à qu'il soit performant en utilisant des protocoles de routage plus performant. Nous envisageons, également, à développer une API Java qui permettra d'implémenter les composants et les services de notre modèle de confiance en reposant sur architecture CORBA. En effet, avec une telle architecture on laisse la possibilité d'implémenter les protocoles avec n'importe quel outil de programmation, et avec cette façon, un autre aspect de l'hétérogénéité sera résolu.

Références

[1] A. Abdul-Rahman. ***The PGP trust model***. EDI-Forum: the Journal of Electronic Commerce, 1997.

[2] A. Abdul-Rahman, S. Hailes. ***A Distributed Trust Model***. In Proc 97 New security paradigms, 1997.

[3] C. Adams, S. Lloyd. ***Understanding PKI: Concepts, Standards, and Deployment Considerations***. 2nd Edition, Addison-Wesley, 2002.

[4] D. Atkins, W. Stallings, P. Zimmermann. ***PGP Message Exchange Formats***. RFC 1991, 1996.

[5] C. Bettstetter. ***Topology Properties of Ad Hoc Networks with Random Waypoint Mobility***. ACM Int'l Symposium in Mobile Ad Hoc on Networking and Computing (MobiHoc), 2003.

[6] D. Atkins, W. Stallings, P. Zimmermann. ***PGP Message Exchange Formats***. RFC 1991, 1996.

[7] S. Capkun, L. Buttyan, J.P. Hubaux. ***Self–ogranized public key management for mobile Ad hoc networks***. IEEE Transactions on mobile computing, 2003.

[8] S. Chokhani, W. Ford. ***Internet X509 Public Key Infrastructure Certificate Policy and Certification Practices Framework***. RFC 2527, 1999.

[9] Y. Desmedt. ***Threshold Cryptography***. In CRYPTO'89, Springer Verlag, 1990.

[10] ITU-T Recommendation. ***Public-Key and Attribute Certificate Frameworks***. 4th Edition, 2001.

[11] J. Kohl, B. Neuman. ***The Kerberos network authentication service version 5***. RFC 1510, 1991.

[12] J. Luo, J.P. Hubaux, P.T. Eugster. ***DICTATE: Distributed Certification Authority with probabilistic freshness for Ad hoc Networks***. IEEE Transactions on Dependable and Secure Computing, 2005.

[13] H. Luo, S. Lu. ***Ubiquitous and Robust Authentication Services for Ad hoc Wireless Networks***. Technical Report, UCLA Computer Science, 2000.

[14] M. Ma, C. Meinel. ***A proposal for Trust Models: Independent Trust Intermediary Service***. IADIS www/Internet Proceedings, 2002.

[15] C. Neuman, T. Ts'o. ***Kerberos: an authentication service for computer networks***. IEEE Communications, 1994.

[16] R. Ostrovsky, M. Yung. ***How to withstand mobile virus attacks***. In Proceedings of the 10th ACM Symposium on Principles of Distributed Computing, 1991.

[17] R. Perlman. ***An overview of PKI trusts models***. IEEE Network, 1999.

[18] A. Shamir. ***How to share a secret***. Communications of the ACM, 1979.

[19] S. Yi, R. Kravets. ***MOCA: Mobile Certificate Authority for Wireless Ad hoc Networks***. In Proceedings of 2nd Annual PKI Research Workshop, NIST, Gaithersburg, MD, 2003.

[20] P. Zimmermann. ***The Official PGP User's Guide***. MIT Press, 1995.

[21] L. Zhou, F.B. Schneider, R.V. Renesse. ***COCA: A Secure Distributed Online Certification Authority***. ACM Transaction Computer Systems, 2002.