République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderahmane Mira de Béjaia
Faculté des Sciences Exactes
Département d'Informatique

Ecole Doctorale Réseaux et Systèmes Distribués (ReSyD)

Mémoire de Magistère En Informatique

Option
Réseaux et Systèmes Distribués

Thème

Sécurité dans les Réseaux Ad Hoc

Soutenu par

HAMOUID Khaled

Devant le jury composé de :

Président de jury : M. Abdelnasser DAHMANI, Professeur, Université A. Mira de Béjaïa, Algérie.

Rapporteur : M. Kamel ADI, Professeur, Université du Québec en Outaouais (Qc), Canada.

Examinateur : M. Mohamed Ahmed-Nacer, Professeur, USTHB d'Alger, Algérie.

Examinateur : M. Luigi Logrippo, Professeur, Université du Québec en Outaouais (Qc), Canada.

Promotion 2006-2007

Promotion 2006-2007

A mes très chers parents, nulle dédicace n'est susceptible de vous exprimer ma profonde affection

A mon frère et mes sœurs

A ma future femme

A tous mes amis

Remerciement

Tout d'abord, je remercie ALLAH pour la volonté, la force, la santé et la patience qu'il m'a donné afin de réaliser ce travail.

Je tiens à adresser mes plus chaleureux remerciements à ADI Kamel, professeur à l'UQO (Québec, CANADA) et lui exprimer toute ma reconnaissance pour son encadrement, ses conseils, son soutien constant, sa confiance et sa patience, ainsi que pour ses remarques pertinentes et ses contributions considérables tout au long de la réalisation de ce travail. J'ai eu l'honneur et le plaisir de travailler sous sa direction pondant mon stage de Magister et j'espère pouvoir continuer à travailler avec lui dans le futur.

Je remercie profondément TARI Kamel, Chef du département d'informatique et responsable de l'école doctorale ReSyD, pour l'occupation qu'il accorde aux étudiants pour leur fournir les moyens favorables et pour ses encouragements et son soutien. Mes vifs remerciements s'adressent également à tous les enseignants de l'école doctorale qui ont contribué à sa création et sa réussite.

Mes plus sincères remerciements vont également à OMAR Mawloud pour sa précieuse contribution, ses conseils et son aide.

Je remercie tout particulièrement les membres de jury qui ont accepté de juger ce travail.

Je souhaite aussi exprimer ma profonde gratitude à mes chers parents et le reste de la famille pour m'avoir apporté du réconfort et pour m'avoir soutenu dans tous mes efforts. Je remercie également toute l'équipe de ma promo Resyd 4 pour leur soutien moral, ainsi que tous mes chers amis.

Enfin, je remercie Toute personne qui a à des degrés divers, contribué sur le plan intellectuel, technique, moral, affectif ou encore matériel à l'achèvement de ce travail.

Résumé

De nos jours, les réseaux sans fil connaissent un grand succès, avec notamment la possibilité d'accéder à l'information et aux services indépendamment de sa position géographique. Parmi ces types de réseaux, les Réseaux Mobile Ad Hoc (MANET) ont montrés leurs importances car ils sont mieux adaptés que leurs homologue filaire pour de nombreuses utilisations civiles ou militaires. Cependant, la prolifération de ce paradigme de réseau dépend fortement du facteur de sécurité. Un système efficace et sécurisé de gestion de clés est un élément primordial pour fournir la plupart des services de sécurité au sein d'un réseau ad hoc. Cependant cette exigence constitue encore, de nos jours, un défi majeur pour la communauté des chercheurs. Dans ce contexte, nous proposons un schéma de gestion de clés publiques complètement distribué, pour faciliter la gestion de certification dans les MANETs. Dans notre nouveau schéma, la clé privée de signature est partagée entre les nœuds du réseau en se basant sur la cryptographie à seuil, ces nœuds forment donc une autorité de certification distribuée et les opérations de gestion de clés et de certification sont accomplies par collaboration de ces nœuds. Notre schéma dénommé SRKM (pour Secure and Robust Key Management), améliore à la fois la sécurité et la robustesse de la gestion de clés contre les nœuds compromis qui peuvent conspirer pour révéler la clé de signature partagée, notamment si leur nombre dépasse le seuil de vulnérabilité prédéfini. De plus, SRKM fournit une flexibilité pour s'adapter au passage à large échelle de la taille du réseau. Les résultats de simulation montrent l'efficacité, la robustesse et la sécurité de notre nouveau schéma comparé aux autres techniques publiées dans la littérature.

Mots Clés : *MANET, Sécurité, Gestion de clés, PKI, Cryptographie à seuil, Certification.*

Abstract

Nowadays, wireless networks are very popular, especially with the ability to access information and services regardless of their location. Among these types of networks, Mobile Ad Hoc Networks (MANET), have shown their significance because they are better than their wired counterparts in many civil and military uses. However, the proliferation of these networks strongly depends on the security factor. An efficient and secure key management is crucial for providing most of security services in an ad- hoc network. However, this requirement is still today a major challenge for the research community in this domain. Hence, we propose a new fully distributed scheme for public key management providing certification management in MANETs. In our scheme, the signing private key is shared between nodes of the network, based on threshold cryptography; these nodes form therefore a distributed certification authority. In this way, key management and certification operations are performed by collaboration of these nodes. Our scheme called SRKM (for Secure and Robust Key Management) addresses both security and robustness of the key management against compromised nodes that can conspire to reveal the shared key, especially if their number exceeds the threshold of vulnerability. In addition, SRKM provides flexibility to scale with the network size. Simulation results show the efficiency, robustness and security of our scheme compared to other techniques in the literature.

Key words :- *MANET, Security, Key Management, PKI, Threshold cryptography, Certification.*

Table des matières

Chapitre 2 : Mécanismes de sécurité pour les réseaux ad hoc

Chapitre 3 : La gestion de clés dans les réseaux ad hoc : Etat de l'art

Chapitre 4 : Schéma Robuste et Sécurisé pour la Gestion de clés publiques dans les réseaux ad hoc

Liste des Figures

Liste des Tableaux

Liste des Abréviations

AKM	Autonomous Key Management
ARAN	Authenticated Routing for Ad hoc Networks
BSS	Basic Service Set
CA	Certification Authority
CH	Cluster Head
CBA	Cryptography-Based Address
CBID	Crypto Based Identifier
DoS	Denial Of Service attack
DMC	Délai Moyen de Certification
EKE	Enkrypted Key Exchange
IDS	Intrusion Detection System
IBSS	Independent Basic Service Set
IBC	ID-based cryptography
KDC	Key Distribution Centre
MANET	Mobile Ad Hoc NETwork
MAC	Message Authentication Code
NAC	Nodes Apportionment Coefficient
PGP	Pretty Good Privacy
PKG	Private Key Generator
PSS	Proactive Secret Sharing
PKI	Public Key Infrastructure
SRKM	Secure and Robust Key Management for MANETs
SRP	Secure Routing Protocol
SA	Security Association
SEAD	Secure Efficient Ad hoc Distance vector routing protocol
SUCV	Statistically Unique and Cryptographically Verifiable
TESLA	Timed Efficient Stream Loss-tolerant Authentication
TTP	Trusted Third Party
VSS	Verifiable Secret Sharing
WSN	Wireless Sensor Networks

Introduction générale

1. Contexte du travail

Les réseaux mobiles sans fil ont connu un très fort développement ces dernières années pour répondre à la hausse constante des besoins en mobilité. Dans un future proche, ces technologies constitueront le socle d'environnements pervasifs (ubiquitaires) dans lesquels les utilisateurs pourront accéder à des services, communiquer, travailler avec d'autres usagers en tout lieu, à tout instant et depuis n'importe quel équipement mobile.

Les réseaux mobiles Ad Hoc (MANET) représentent une composante clé de cette évolution et leurs fondements seront inévitablement intégrés aux futures générations de réseaux sans-fil. Ces réseaux auto-organisés, peuvent être formés spontanément à partir d'un ensemble d'entités mobiles communicantes, sans nécessiter la présence d'une infrastructure fixe préexistante. Les entités mobiles qui constituent le réseau peuvent être de formes variées : ordinateurs portables, téléphones mobiles, PDA, capteurs, etc., et présentent par conséquent des capacités non homogènes en termes de communication, de puissance de calcul et de stockage. Elles sont libres de se déplacer de manière aléatoire et de s'organiser arbitrairement, si bien que la topologie du réseau est fortement dynamique dans le temps et dans l'espace. Les entités mobiles peuvent intervenir en tant que routeurs pour assurer l'acheminement des paquets entre elles par sauts successifs. Elles communiquent donc soit directement lorsqu'elles se trouvent dans un même voisinage, soit par communication multi-sauts le cas échéant en faisant appel à des nœuds intermédiaires. Cette technologie constitue donc une solution très attractive pour construire des réseaux dynamique de terminaux mobiles qui allient la simplicité de construction à un coût relativement modeste.

L'utilisation des réseaux ad hoc est de plus en plus répandue et les applications de cette technologie sont multiples : déploiement d'un réseau en cas de sinistre majeur rendant inopérable l'infrastructure réseautique existante, déploiement d'un réseau militaire dans une zone d'opération, déploiement d'un réseau de capteurs, etc. Cependant, l'apparente simplicité du concept cache de nombreux défis scientifiques et techniques. En particulier, la sécurité dans les MANET constitue l'un des principaux obstacles à un large déploiement de ces réseaux. En effet, la sécurité est devenue le sujet d'une activité de recherche soutenue qui vise à développer des solutions afin de fournir la protection des communications entre les nœuds dans un environnement potentiellement hostile, tout en assurant les services de base de la

sécurité tels que la confidentialité, l'authentification, l'intégrité, la disponibilité et la non-répudiation.

2. Problématique et objectifs

Un schéma de gestion de clés (*key Management*) efficace et sécurisé est la partie fondamentale de toute architecture de sécurité utilisée pour assurer l'authentification et la confidentialité des communications. Ces conditions de sécurité peuvent être assurées en utilisant la cryptographie à clé publique et les certificats. Ainsi, les clés sont une condition préalable pour initialiser un service réseau sécurisé. Cependant, la gestion de clés dans les MANETs est un problème fondamental qui a été abordé par plusieurs chercheurs aboutissant toutefois à des résultats assez limités. La difficulté accrue pour fournir le matériel cryptographique approprié aux entités du système est due à la conjonction de plusieurs facteurs qui s'étendent du manque de sécurité physique des nœuds au manque des relations de confiance apriori et au manque d'infrastructure et d'une autorité centrale.

Dans les réseaux basés infrastructure, les certificats sont gérés par une infrastructure à clé publique (PKI : Public Key Infrastructure) qui inclut une autorité de certification (CA : Certification Authority) jouant le rôle d'un tiers de confiance (TTP : Trusted Third Party) et qui certifie l'authenticité des clés publique. Toutefois, les modèles de PKI classiques ne sont pas transposables aux MANETs pour les raisons déjà mentionnées. Outre le problème du passage à l'échelle et de disponibilité des nœuds, il n'est pas souhaitable d'avoir une autorité de certification centralisée car cette dernière va constituer un point de vulnérabilité pour les MANETs : la sécurisation des communications ne peut plus être assurée si la CA est indisponible ou compromise.

La technique de cryptographie à seuil semble être une bonne solution au problème de déploiement d'une PKI dans les MANETs, ces modèles permettent de partager la confiance d'une CA entre un ensemble de nœuds. De cette manière les opérations de base de la CA (signature digitale par exemple) sont effectuées par collaboration de ces nœuds qui forment ainsi une CA distribuée. Le principe de base consiste à partager la clé privée de la CA entre n nœuds en utilisant un schéma de partage de secret (n, t) où t nœuds sont nécessaires pour reconstruire la clé privée. Une sécurité proactive est également proposée pour renforcer la robustesse de ces schémas, cela consiste à rafraîchir périodiquement les parts de la clé pour empêcher un adversaire mobile à révéler la clé privée en utilisant les parts de la clé détenues par des nœuds compromis. Ces solutions reposent sur l'hypothèse qu'un adversaire ne doit pas compromettre t nœuds dans l'intervalle qui sépare deux opérations successives de rafraîchissement. Cependant, une limitation majeure que nous trouvons dans ces schémas est que la robustesse n'est pas préservée au passage à

l'échelle du réseau, car la probabilité de compromettre un grand nombre de nœuds augmente avec le nombre de nœuds composant le réseau. En effet, le temps de rafraîchissement des parts augmente avec le nombre de nœuds, de plus les attaques DoS peuvent aussi ralentir l'opération de rafraîchissement, ce qui donne plus de chances à un adversaire mobile de compromettre suffisamment de nœuds pour révéler la clé privée.

Notre contribution vise à améliorer la robustesse et la sécurité dans les opérations de gestion de clés pour les réseaux ad hoc à large échelle. Dans ce travail nous proposons un nouveau schéma de gestion de clés pour les MANETs dénommé SRKM (pour Secure and Robust Key Management). Ce schéma tient compte de l'absence d'infrastructure dans les MANETs, et garantit une flexibilité au passage à l'échelle du réseau, tout en assurant les services requis de sécurité. Pour assurer l'authentification et la distribution sécurisée des clés sans aucune autorité centrale, notre schéma est basé sur la cryptographie à seuil. Il fournit une meilleur robustesse et sécurité aux opérations de gestion de clés et de certification, en assurant que la confidentialité de la clé privée est préservée même si le nombre de nœuds compromis dépasse le seuil de vulnérabilité, ce qui n'est pas possible avec les schémas connus dans la littérature.

3. Organisation du mémoire

Ce mémoire est organisé en cinq chapitres. Dans Le premier chapitre nous définissons, dans un premier temps, des concepts de base des réseaux ad hoc et dans un second temps nous analysons les menaces et défis de sécurité dans ces réseaux. Dans le deuxième chapitre nous présentons un survol général de l'état de l'art sur les solutions proposées pour assurer les différents services de sécurité dans les réseaux ad hoc. Le troisième chapitre a pour but de discuter et de synthétiser les travaux de recherche en termes de gestion de clés dans les réseaux ad hoc. Ainsi, nous présentons dans ce chapitre une taxonomie des schémas de gestion de clés dans les réseaux ad hoc, proposées dans la littérature. Notre contribution est présentée dans le chapitre quatre dans lequel nous décrivons le schéma que nous avons proposé afin d'assurer une gestion robuste et sécurisée de clés adaptée au contexte des MANETs. Dans le chapitre cinq nous présentons la validation expérimentale de notre technique en donnant les mesures de performance de notre nouvelle architecture à travers une simulation réalisée et implémentée à l'aide de l'environnement MATLAB.

Chapitre 1

Les réseaux Ad Hoc et la sécurité

Dans ce chapitre, nous introduisons le concept des réseaux ad hoc et leur contexte d'utilisation et décrivons brièvement les différentes architectures que nous rencontrons dans la littérature. Nous abordons, par la suite, les problèmes et défis de sécurité des réseaux ad hoc, nous énumérons les caractéristiques et les vulnérabilités induites par ces réseaux et présentons quelques attaques de sécurité possibles.

1.1 Introduction

Les réseaux sans fil (Wireless Networks) constituent de plus en plus une technologie émergente permettant à ses utilisateurs un accès à l'information et aux services indépendamment de leurs positions géographiques. Par ailleurs, les réseaux Mobiles Ad Hoc *(MANET)*, caractérisés par une structure dynamique et une facilité et une rapidité de déploiement, suscitent un intérêt particulier. Comparé à un environnement statique, ce nouvel environnement permet aux unités de calcule une libre mobilité et ne pose aucune restriction sur la localisation des usagers.

Néanmoins, la sécurité dans les environnements ad hoc présente un véritable défi technologique pour avoir un large déploiement de ce type de réseau. En effet, la nature des liens sans fils et la topologie dynamique des réseaux ad hoc les rend plus vulnérables aux attaques que les réseaux filaires. De plus, l'absence d'une infrastructure élimine toute possibilité de déploiement d'une ligne de défense ou une autorité de défense centralisée. Un nœud mobile n'opère pas seulement comme un simple hôte mais aussi comme un routeur en acheminant les paquets aux autres noeuds du réseau qui ne peuvent pas être dans la même portée de transmission. Cela expose la couche réseau à plusieurs attaques. Sécuriser un réseau ad hoc revient donc à instaurer les différents services de sécurité dans ce réseau, tout en prenant en compte les caractéristiques propres à ce type de réseaux.

1.2. Les réseaux sans fil

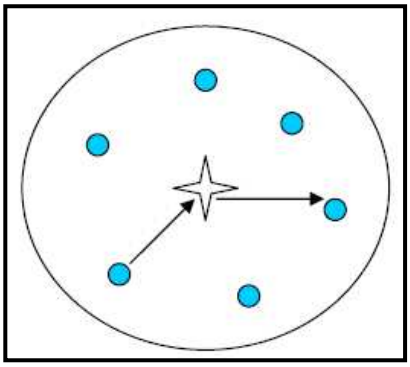
L'évolution récente de la technologie sans fil et l'apparition des unités de calculs portables poussent aujourd'hui les chercheurs à faire des efforts afin de réaliser le but ultime « offrir l'accès à l'information n'importe où et n'importe quand ». Dans les réseaux sans fil les ordinateurs communiquent via les ondes radio, ou au moyen de rayonnement infrarouge. Les ondes radio sont le support le plus utilisé, les fréquences disponibles se trouvent dans la bande des micro-ondes, autour de 2.4 GHz (bande ISM) et 5 GHz (bande U-NII) [67]. Les standards pour le support hertzien, au jour d'aujourd'hui, sont le IEEE 802.11 avec ses branches principales 802.11a (Wi-Fi5), 802.11b (Wi-Fi), 802.11g, Hiper-LAN de l'ETSI, et Bluetooth.

1.2.1. Architectures sans fil

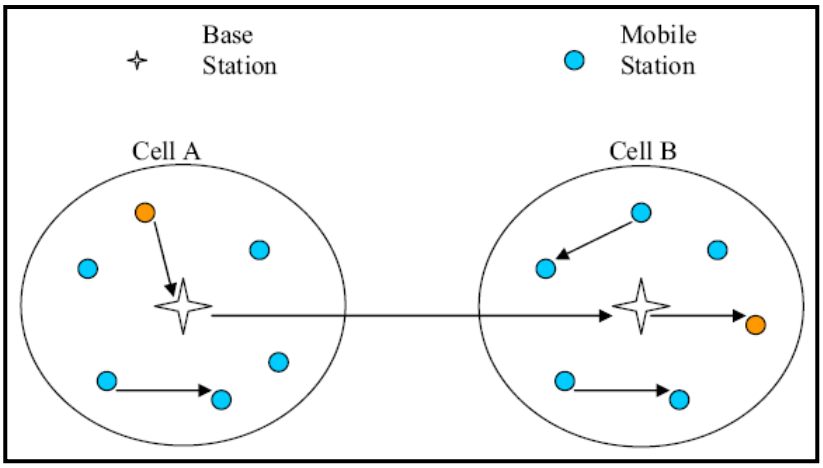
Dans le cadre des réseaux sans fil, le standard IEEE 802.11 définit trois modes de fonctionnement différents :

a) En mode infrastructure ou BSS, les machines (*noeuds*) sont en connexion à travers un point d'accès (voir la figure 1.1 (a)), ce type de réseaux est appelé aussi réseaux cellulaires à simple saut (Single-hop) qui nécessite d'avoir une station de base fixe (ou points d'accès dans le glossaire 802.11) pour contenir un domaine de service, la zone de couverture de ce point d'accès est appelé l'ensemble de services de base BSS (Basic Service Set). Dans une telle architecture les paquets sont routés à travers la station de base.
b) En mode point à point ou IBSS (Independent Basic Service Set), les nœuds mobiles peuvent se communiquer directement les uns avec les autres s'ils sont mutuellement joignables. S'ils ne sont pas dans la même cellule, les paquets sont acheminés à travers la station de base (voir la figure 1.1 (b)).
c) En mode Ad Hoc, ce sont des réseaux ad hoc au vrai sens du mot, il n'existe aucune infrastructure, il y a juste des nœuds mobiles de faible puissance formant un réseau. Dans ce model il n'y a pas de station de base et à cause de la portée de transmission limitée, de multiples sauts peuvent être nécessaires pour que les noeuds communiquent à travers le réseau, les fonctions de routage sont incorporées dans chacun des nœuds.

Un réseau sans fil est beaucoup plus souple que son homologue filaire, dans la mesure où les noeuds ne sont pas connectés par des câbles et peuvent être totalement mobiles. Pourtant, un réseau sans fil est beaucoup plus vulnérable pour ce qui concerne la connectivité des noeuds et la sécurité des données.



(a) Mode infrastructure (BSS)



(b) Mode point à point ou (IBSS)

Fig.1.1 Architectures des réseaux sans fil

1.2.2. Vulnérabilités des réseaux sans fil

La caractéristique la plus évidente des réseaux sans fil est que la communication a lieu sur un canal sans fil (qui est habituellement un canal radio), Ce type technologie sans fil souffre d'un certain nombre de vulnérabilités mentionnées ci-après :

- Le canal radio peut être facilement écouté : en plaçant une antenne à un repérage approprié, un adversaire peut intercepter l'information transmise par une victime. L'écoute (eavesdropping) est souvent utilisée pour réaliser des attaques passives, ce genre d'attaque consiste à écouter le réseau et analyser les données capturées sans interaction avec ce dernier.
- Les données peuvent être altérées : un adversaire tente de modifier le contenu des messages échangés entre les entités sans fil, ce type de menaces est classé dans les attaques actives.

- L'absence d'une liaison filaire rend plus facile d'usurper les identités et se faire passer pour des utilisateurs légitimes.
- Le canal peut être brouillé, notamment pour perpétrer une attaque de dénis de service (DoS) et ainsi empêcher les autres entités de transmettre leurs données.
- Généralement dans les réseaux sans fil, les entités qui forment le réseau se caractérisent par une capacité de stockage et puissance de calcul limitée. Ce problème est atténué en réduisant la complexité des opérations effectuées par les stations mobiles, néanmoins, cela peut mener à une dégradation de l'efficacité des protocoles de sécurité utilisés.
- Une sécurité physique limitée : cela est justifiée par le fait que le canal de communication radio est relativement vulnérable et peut être une cible facile pour espionner de manière passive.

1.3. Les réseaux mobiles Ad Hoc (MANET)

1.3.1. Définition

Un réseau mobile ad hoc (MANET pour Mobile Ad hoc NETworks) peut être défini comme une collection d'entités mobiles interconnectées par une technologie sans fil formant un réseau temporaire sans l'aide de toute administration ou de toute infrastructure fixe (voir Figure 1.2) [40]. Aucune supposition ou limitation n'est faite sur la taille du réseau cela veut dire qu'il est possible que le réseau ait une taille très grande. Les réseaux Ad Hoc sont auto organisés, ce qui implique que la connectivité doit être préservée autant que possible automatiquement lorsque la topologie du réseau change suite à l'apparition, la disparition ou au mouvement de certains nœuds.

Un réseau ad hoc peut être multi sauts dans le cas où les nœuds mobiles le constituant ne sont pas localisés dans la même portée (ne sont pas adjacents). Ainsi, le chemin entre un nœud source et un nœud destination peut impliquer plusieurs sauts où d'autres nœuds du réseau servent de relais. En effet, dans ce type de réseau, chaque nœud mobile n'opère pas seulement comme un simple hôte mais aussi comme un routeur, en acheminant les paquets aux autres nœuds du réseau qui ne peuvent pas être dans la même portée de transmission sans fil. Chaque noeud participe à un protocole de routage ad hoc qui lui permet de découvrir des chemins "multi-sauts" à travers le réseau à tout autre noeud.

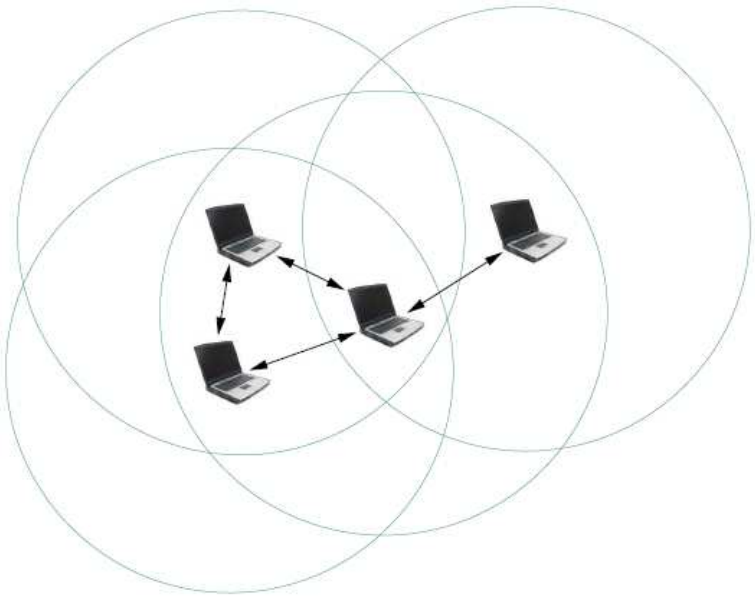


Fig.1.2 Un simple réseau ad hoc

1.3.2. Caractéristiques et contraintes spécifiques aux MANET

Un réseau ad hoc est caractérisé par sa topologie, qui peut évoluer avec le temps et cela en fonction du déplacement des noeuds, de la puissance d'émission et des caractéristiques du signale de transmission. Cependant, ce type de réseaux impose un certain nombre de contraintes liées à plusieurs caractéristiques qui les distinguent des réseaux traditionnels, parmi celles-ci nous pouvons citer :

- ***Liaisons sans fil.*** Le seul moyen de communication entre les noeuds mobiles au sein d'un réseau ad hoc est l'utilisation d'un canal radio. Malgré des progrès très importants, les performances des technologies sans fil restent et resteront en deçà de celles des technologies filaires. En effet, les liaisons sont à débits variables et la bande passante est assez limitée.
- ***Topologie dynamique due à la mobilité.*** La mobilité des noeuds constitue à l'évidence une caractéristique très spécifique des MANET. En effet, les nœuds mobiles du réseau se déplacent de façon libre et arbitraire. Par conséquent, la topologie du réseau peut changer à tout moment.
- ***Autonomie des nœuds***. La consommation d'énergie constitue un problème important pour des équipements fonctionnant grâce à une alimentation électrique autonome. Ces équipements intègrent des modes de gestion d'énergie et il est important que les protocoles mis en place dans les réseaux ad hoc prennent en compte cette caractéristique.

- ***Equivalence des noeuds du réseau.*** La charge du réseau doit être distribuée équitablement entre les éléments en tenant compte de leur capacité respective. Chaque élément d'un réseau ad hoc est autonome et possède à la fois les fonctionnalités de relais et de point de communication.
- ***Faible capacité des nœuds***. Les équipements utilisés dans les réseaux ad hoc ont une faible capacité de stockage et une puissance de calcule limitée. Leur sécurité physique est également faible.
- ***L'absence d'infrastructure***. Les réseaux ad hoc se distinguent des autres réseaux mobiles par l'absence d'infrastructure préexistante et la non disponibilité d'une administration centralisée. Les nœuds mobiles sont responsables de l'établissement et du maintient de la connectivité du réseau de manière continue.
- ***Vulnérabilité.*** Les réseaux sans fil sont par nature plus sensibles aux problèmes de sécurité. Pour les réseaux ad hoc, le principal problème ne se situe pas tant au niveau du support physique mais principalement dans le fait que tous les noeuds sont équivalents et potentiellement nécessaires au fonctionnement du réseau. De plus les mécanismes de sécurité utilisée pour les réseaux traditionnels ne peuvent pas être directement appliqués aux MANET.

1.3.3. Le routage dans les MANET

La gestion de l'acheminement de données ou le routage, consiste à assurer une stratégie qui garantit, à n'importe quel moment, la connexion entre n'importe quelle paire de noeuds appartenant au réseau. La stratégie de routage doit prendre en considération les changements de la topologie ainsi que les autres caractéristiques du réseau ad hoc (bande passante, nombre de liens, ressources du réseau, etc.).

Il existe actuellement trois types de protocoles de routage dans les réseaux ad hoc; les protocoles proactifs, réactifs et hybrides.

Routage proactif

Les méthodes proactives se basent sur l'établissement de routes à l'avance. Les nœuds s'échangent alors de manière périodique leurs tables de routage ou des informations concernant celles-ci. Dans le contexte des réseaux ad hoc les noeuds peuvent apparaître ou disparaître de manière aléatoire et la topologie même du réseau peut changer; cela signifie qu'il faut un échange continu d'informations pour que chaque noeud ait une image à jour du réseau. Les trois protocoles les plus connus dans cette catégorie sont OLSR (Optimized Link State Routing Protocol), GSR (Global State Routing protocol) et DSDV (Destination-Sequenced Distance Vector routing protocol).

Routage réactif

Les méthodes réactives se basent sur l'établissement et la maintenance des routes à la demande, et donc uniquement lorsqu'un noeud en a besoin. Une procédure de découverte de route est alors lancée. L'avantage de cette technique est que la charge du réseau est beaucoup moins élevée et que l'information sur la route obtenue est bien souvent de meilleure qualité. En contrepartie, trouver une route prend plus de temps, et il est impossible de connaître au préalable la qualité du chemin (en termes de bande passante ou de délai par exemple). Les protocoles appartenant à cette catégorie sont DSR (Dynamic Source Routing Protocol), AODV (Ad-hoc On-demand Distance Vector) ou TORA (Temporally-Ordered Routing Algorithm).

Routage hybrides

Les méthodes hybrides combinent les deux approches précédemment cités pour tirer profit de leurs avantages. Généralement, le réseau est divisé en deux zones. Le principe est d'utiliser une approche proactive pour avoir des informations sur les voisins les plus proches, qui se trouvent au maximum à deux sauts du noeud mobile et une approche réactive au delà de cette zone prédéfinie afin de chercher des routes. Il existe plusieurs protocoles connus appartenant à cette catégorie de protocoles hybride, citons ZRP (Zone Routing Protocol) et CBRP (Cluster Based Routing Protocol)

1.3.4. Quelques applications des réseaux ad hoc

Les applications ayant recours aux réseaux ad hoc couvrent un large spectre, incluant les applications militaires et tactiques, l'enseignement à distance, les systèmes de fichiers réparties, la simulation distribuée interactive, les applications de calcule distribué ou méta-computing, etc. D'une façon générale, les réseaux ad hoc sont utilisés dans toute application où le déploiement d'une infrastructure est trop contraignant. Voici quelques applications des réseaux ad hoc :

- ***Les réseaux de capteurs***. Les noeuds sont des capteurs en charge de la collecte des informations sur un champ. À titre d'exemple, des données scientifiques peuvent être collectées par les robots sur un volcan et transmis à un centre situé dans une zone sûre pour l'analyse.

- ***Réseaux Inter-véhicules***. Avertissements de secours et de congestion, l'assistance de changement de voie, ou la coopération au niveau des intersections de route peuvent être fournies par une communication multi-saut entre des véhicules.

- ***Les réseaux MESH***. Pour déployer à moindre coût un réseau sans fil à l'échelle d'une ville, on peut utiliser un réseau radio maillé, aussi appelé « mesh network ». Il consiste en un ensemble de stations de base qui utilisent une liaison radio pour communiquer et qui couvrent la zone visée. Ainsi, les stations de base agissent comme des relais radio pour les communications des mobiles. Dans un réseau radio maillé, chaque mobile est rattaché à la station de base la plus proche, avec laquelle il communique exclusivement.

1.4. La Sécurité dans les MANET

La sécurité est un sujet très important à traiter, notamment pour les applications de MANET dites sensibles à la sécurité. Effectivement, les réseaux ad hoc sont connus pour leur manque d'infrastructure préexistante et d'administration centralisée, ils sont de ce fait considérés comme difficiles à sécuriser. Nous avons déjà signalé que les opérations de gestion de communication dans les MANET sont gérées d'une manière très différente que celles des réseaux traditionnels, il en est de même pour leurs sécurité.

1.4.1. Les défis de sécurité

Les réseaux ad hoc constituent un nouveau paradigme dans les systèmes de communications sans fil, ils se distinguent par des propriétés et caractéristiques spécifiques comparativement aux réseaux traditionnels qui sont basés infrastructure. Ces caractéristiques et propriétés amènent aussi de nombreux défis de gestion de la sécurité. Les caractéristiques principales des MANET incluent une topologie dynamique, une rupture de liens fréquente et une perte de données considérables, des contraintes d'énergie, une bande passante et portée de transmission limitées, une protection physique des nœuds relativement faible, une coopération distribuée pour les communications multi saut et l'absence d'une autorité centrale. Avec ces caractéristiques particulières, les réseaux ad hoc sont exposés au risque de violations de la sécurité. Nous présentons ci-après les thèmes critiques qui empêchent les solutions traditionnelles de sécurité d'être directement appliquées aux MANET.

- Un réseau ad hoc est *dynamique* en raison des changements fréquents de sa topologie et de ses membres. Il est ainsi indispensable que les mécanismes de sécurité puissent s'adapter rapidement à ces changements. La topologie dynamique du réseau aboutit à un changement significatif dans les relations de confiance entre les nœuds. Ainsi, l'implémentation de sécurité basée sur la

confiance fait face à de grands défis et les solutions statiques ne s'appliquent pas dans un tel environnement dynamique.

- *La mobilité des noeuds* peut également causer la rupture de liens et la perte fréquentes de données, puisque les noeuds peuvent joindre et quitter le réseau sans aucun avertissement préalable. Ainsi les liaisons entre les noeuds ne seront pas garanties, ce qui peut avoir un impact important sur la communication et qui affecte aussi la mise en œuvre de mécanismes de sécurité.

- Les nœuds mobiles d'un réseau ad hoc incluent typiquement des *équipements portables* tels que des « laptops », des PDA, des téléphones cellulaires et des microcapteurs. Ces dispositifs sont inévitablement alimentés par *batterie*. La durée de vie de la batterie devient, de ce fait, cruciale pour la communication sans fil et le calcul distribué. Cela peut être une cible des nœuds malicieux qui tentent de bloquer les communications et les services via des attaques par *consommation de batteries*. En outre, les solutions de sécurité traditionnelles sont conçues pour une forte capacité des nœuds ce qui ne s'adapte pas aux environnements ad hoc. Ceux-ci exigent donc de nouvelles solutions qui soient économes en termes de puissance de calcul, du coût de l'énergie et de la charge du trafic. De plus, la solution doit être équitable au niveau de la consommation des ressources du réseau.

- Les nœuds ad hoc ont une protection physique relativement limitée. Il y a donc une probabilité non négligeable que ces derniers soient compromis. Ainsi il ne faut pas considérer seulement les attaques provenant de l'extérieur mais aussi de l'intérieur des nœuds compromis. Ces attaques ne peuvent donc pas être prévenues avec de simples mécanismes d'authentification et les schémas de clé seuls ne sont pas suffisants pour y faire face. Par conséquent, des mécanismes plus sophistiqués doivent être conçus. Ainsi, il est très important de considérer des solutions de sécurité spécifiques aux MANET contre les nœuds compromis, car certaines attaques très sophistiquées peuvent être commises par conspiration des nœuds compromis.

- Le *manque d'organisation et de confiance a priori* peut également influencer la sécurité des MANET. En effet, lors du déploiement du réseau les nœuds s'auto-organisent et il n'y a pas une confiance a priori entre ces nœuds. De plus, la gestion et la distribution de clés peuvent être difficiles à réaliser en raison de l'absence d'une autorité centrale. Les solutions de sécurité dans les réseaux traditionnels s'appuient généralement sur des relations de confiance établies par une tierce autorité de confiance. Elles utilisent des primitives cryptographiques pour authentifier les nœuds et sécuriser les échanges de données. Afin d'utiliser ces moyens dans les MANET, nous devons étudier

comment établir des autorités de confiance et/ou des relations de confiance entre les nœuds sans l'aide d'aucune infrastructure ou autorité centrale.

- Exigence de coopération, en raison du manque d'une infrastructure les fonctions de base de gestion du réseau doivent être effectuées d'une façon distribuée par la collaboration d'un ensemble de nœuds ordinaires. Ainsi, l'exécution des opérations de base du réseau peut être fortement affectée par manque malveillant ou accidentel de coopération.

En raison de ces défis de sécurité, Les réseaux ad hoc imposent de nouveaux besoins de sécurité et exigent des mécanismes robustes, tolérants aux pannes et ayant un faible surcoût. De plus, un réseau ad hoc peut se composer de centaines voire de milliers de nœuds, il est donc important que la solution de sécurité proposée permette le passage à l'échelle.

1.4.2. Les vulnérabilités

Tout système peut avoir des faiblesses qui sont due à une mauvaise conception ou à certaines caractéristiques liées à la nature même de ce système. L'exploitation de ces faiblesses et les capacités de l'intrus présentent, évidement, une menace contre ce système. Il est donc évident qu'avant de proposer toute solution de sécurité, il est primordial d'identifier d'abord les vulnérabilités et les failles pouvant être exploitées pour lancer une éventuelle attaque contre le système.

Parmi les vulnérabilités intrinsèques des réseaux ad hoc, certaines résident dans leur mécanisme de routage, d'autres dans l'utilisation des liens sans fil et d'autres encore dans leurs mécanismes d'auto-configuration. Ces fonctionnalités de base reposent sur une confiance complète entre tous les nœuds participants au réseau. Dans le cas du routage, le transport des paquets dans le réseau repose sur la véracité des informations fournies par les autres nœuds. De plus, la livraison d'un paquet vers une destination dans les MANET repose sur un routage multi-sauts, et donc nécessite la coopération totale des nœuds intermédiaires. Un nœud malicieux pourrait, en refusant de coopérer, tout simplement bloquer, modifier, ou ralentir le trafic qui passe par lui.

L'utilisation des liaisons sans fil rend ces réseaux très vulnérables aux attaques s'étendant de l'écoute clandestine passive à l'interférence active. Un attaquant doit juste être dans la portée radio d'un noeud afin d'intercepter le trafic réseau.

Le manque de frontières sécurisées dans les MANET représente une autre vulnérabilité. En effet, il n'y a pas une ligne de défense claire comparativement aux

réseaux traditionnels où l'intrus doit passer par plusieurs lignes de défense tels que les pare-feux pour visiter le réseau. Cependant, dans les MANET l'intrus n'aura pas besoin d'obtenir un accès physique pour visiter le réseau, il peut communiquer avec n'importe quel nœud dans sa portée de transmission et ainsi rejoindre le réseau automatiquement. Ces réseaux sont donc vulnérables d'être physiquement accessibles, et les nœuds peuvent être facilement compromis, ce qui peut avoir comme conséquence la révélation des clés cryptographiques. Un autre problème pour protéger les MANET est la difficulté de distinguer entre des nœuds compromis et des nœuds saints à partir de leurs comportements. Par exemple, un nœud égoïste ne peut pas être considéré comme nœud malicieux ou compromis, car ce nœud n'effectue aucune attaque mais il est possible qu'il refuse de coopérer pour préserver son l'énergie électrique. De plus, l'absence d'une administration centralisée peut conduire à un état de vulnérabilité. En effet, l'établissement de confiance entre les nœuds, la gestion et la distribution des clés cryptographiques est une tâche critique sans aucune administration centralisée et la plupart des modèles de confiance utilisés dans les réseaux traditionnels sont basés sur l'existence d'une infrastructure et d'une administration centralisée. L'application directe de ces modèles dans les MANET peut générer des vulnérabilités considérables. La bande passante limitée peut être également un point de vulnérabilité et une cible à des attaques malveillantes telles que le déni de service (DoS), un nœud malicieux peut envoyer massivement des requêtes afin de consommer la bande passante et de rendre les services indisponibles aux autres nœuds. Outre les contraintes de bande passante, la portée de transmission des nœuds est relativement faible. Ainsi, de multiples sauts à travers des nœuds intermédiaires sont nécessaires pour faire passer les messages. Cette coopération distribuée est basée sur l'hypothèse que les nœuds intermédiaires sont dignes de confiance et n'ont pas un comportement malveillant. Toutefois, cette hypothèse n'est pas fiable, car il n'y a aucune garantie que ces nœuds se comporteront comme prévu.

Pour récapituler, un réseau ad hoc est vulnérable en raison de ses caractéristiques de milieu ouvert, des changements dynamiques de la topologie, des algorithmes basés sur la coopération, du manque de point centralisé de contrôle et de gestion et du manque d'une ligne de défense claire.

1.4.3. Les Menaces de sécurité

Une approche pragmatique pour établir un système de sécurité est de considérer les menaces auxquelles le système pourrait faire face après déploiement. Une classification des menaces dans les réseaux ad hoc est suggérée dans [68] selon laquelle on distingue deux types de menaces :

1. Les menaces externes, où l'attaquant est limité à l'écoute et l'analyse du trafic échangé. De plus l'attaquant vise à provoquer la congestion ou de perturbation des noeuds durant la prestation de services.

2. Les menaces internes où l'attaquant se donne les moyens pour gagner l'accès normal au réseau et participer aux activités de ce dernier, soit par une usurpation d'identité pour obtenir l'accès au réseau comme nouveau noeud, ou de compromettre directement un nœud et l'utiliser comme base pour mener des attaques malveillantes.

1.4.3.1 Les menaces externes

En présence d'un protocole d'authentification pour protéger les couches supérieures du réseau, des menaces externes sont dirigées aux couches liaison de données et physique. La sécurité de la couche physique est intrinsèquement difficile à fournir en raison de la nature mobile des noeuds ad hoc. Nous divisons des menaces externes en deux grandes catégories : (i) l'écoute clandestine passive, où l'adversaire écoute simplement les signaux transmis et (ii) l'interférence active, où l'adversaire envoie des signaux ou des données destinés à perturber le réseau d'une certaine manière.

a) L'écoute clandestine passive

Ceci peut permettre aux nœuds non autorisés d'écouter et de recevoir des messages comprenant des informations de routage ou même des clés cryptographiques. Les noeuds non autorisés sont en mesure de recueillir des données qui peuvent être utilisées pour déduire la topologie du réseau, ou des informations sur les clés cryptographiques qui seront utilisées pour faire de la cryptanalyse. Par conséquent, il est nécessaire de développer des techniques pour cacher ces informations. L'écoute clandestine est également une menace pour la confidentialité de la localisation d'un nœud.

b) L'interférence active

La menace principale dans cette catégorie est l'attaque de déni de service (DoS) provoquée, en bloquant la voie de transmission sans fil, ou en déformant les communications. L'effet de cette attaque dépend de sa durée, et le protocole de routage utilisé ainsi que la robustesse des services de sécurité. Le type le plus sérieux de l'attaque DoS s'appelle « sleep deprivation » [25]. Dans cette attaque, l'énergie des nœuds est délibérément gaspillée. Avec la puissance et les ressources limitées des nœuds ad hoc, la prévention de telles attaques est de plus grande importance. Il y a également des menaces pour l'intégrité, par exemple où un attaquant externe peut essayer de rejouer d'anciens messages, ou de changer l'ordre des messages.

1.4.3.2 Les menaces internes

Les menaces posées par les nœuds internes sont très graves, car ces derniers internes disposent de l'information nécessaire pour participer à des opérations distribuées. Les nœuds internes peuvent mal se comporter dans une variété de façons; nous identifions quatre catégories de comportements malveillants des nœuds : les noeuds échoués, les noeuds égoïstes, les nœuds compromis et les noeuds malicieux.

a) Les nœuds échoués

Les noeuds échoués sont simplement ceux incapables d'effectuer une opération; ceci peut être dû à plusieurs facteurs, comprenant des problèmes de manque d'énergie et ceux liés aux événements environnementaux. Un défi majeur réside dans le routage ad hoc où certains nœuds échouent à acheminer des paquets de données ou même des messages de routage afin de mettre à jour les structures de donnés. Cette information est cruciale et peut affecter les mécanismes de sécurité mis en place, telle que l'authentification. La menace d'avoir des noeuds échoués est plus sérieuse si ces derniers sont nécessaires en tant qu'élément participant d'une manière distribuée à un service de sécurité.

b) Les nœuds compromis

Les nœuds compromis possèdent les mêmes caractéristiques que les nœuds échoués, comme la non-transmission ou le non-acheminement des paquets de données reçus. En plus, ils peuvent également envoyer de fausses informations pour perturber certaines fonctions de base du réseau comme le routage, ce qui présente une menace à l'intégrité du réseau. Les nœuds compromis peuvent aussi divulguer des informations confidentielles comme par exemple des clés privées, ou conspirer pour lancer une attaque active. Les mécanismes de sécurité proposés pour les environnements ad hoc doivent être en mesure d'identifier et d'isoler ces nœuds compromis.

c) Les nœuds égoïstes

Dans certains cas, des nœuds ad hoc présentent un comportement égoïste, ils peuvent refuser de transférer des données ou de coopérer avec d'autres nœuds pour garantir les opérations et les services de base du réseau. Les nœuds égoïstes se comportent ainsi pour des raisons de performances ou pour préserver de l'énergie à cause de la capacité limitée des batteries et du manque de ressources dans les MANETs. Il est important de préciser que les nœuds égoïstes n'effectuent aucune

action pour compromettre l'intégrité du réseau par injection active d'information de contrôle erronée.

d) Les nœuds malicieux

Les nœuds malicieux visent à perturber délibérément le fonctionnement correct des différentes opérations de base de gestion du réseau tels que le routage, compromettre les nœuds du réseau ou rendre indisponible les services du réseau si possible. Ils peuvent lancer différentes attaques sur les mécanismes de base du réseau ou même sur les mécanismes de sécurité. L'impact des actions des nœuds malicieux devient plus important lorsque ces nœuds sont le seul lien entre des groupes de nœuds voisins.

1.4.4. Les attaques sur les MANET

Les attaques contre un réseau ad hoc peuvent venir de noeuds malicieux qui ne sont pas une partie valide du réseau et qui tentent de joindre le réseau sans autorisation. Ces noeuds sont généralement appelés des nœuds externes. Les réseaux sont généralement protégés contre les nœuds malicieux grâce à l'utilisation de techniques cryptographiques. Ces techniques permettent aux nœuds de vérifier d'une manière sécurisée l'identité des autres noeuds, et peuvent donc essayer de prévenir tout dommage causé par les nœuds malicieux. Nous considérons également des attaques à partir des noeuds qui sont autorisés pour faire partie du réseau, on les appelle nœuds internes. Des noeuds internes peuvent lancer des attaques parce qu'ils ont été compromis par des nœuds malicieux.

Le succès d'une attaque dépend de la vulnérabilité du système et l'efficacité des contre-mesures. Les attaques peuvent être divisées en deux catégories principales :

- ***Les attaques passives***. Dans ce type d'attaques, un adversaire écoute passivement le trafic échangé sans modifier ou insérer des données. Ces attaques visent principalement la confidentialité du système. Toutefois, ce processus de collecte d'informations peut conduire à des attaques actives plus tard.

- ***Les attaques actives***. Sont des attaques où l'adversaire effectue une action malveillante en plus d'écouter passivement le trafic. Par exemple un adversaire pourrait choisir de modifier des paquets, d'injecter des paquets, ou même de perturber des services réseau.

Une attaque qu'elle soit passive ou active peut être classée dans l'une des deux catégories suivantes :

- Attaques sur les mécanismes de base de gestion du réseau tel que le routage. La prévention de ces attaques nécessite des mécanismes de sécurité qui reposent souvent sur des algorithmes cryptographiques.
- Attaques sur les mécanismes de sécurité notamment les mécanismes de gestion de clés. En raison des particularités des MANET, les solutions, nous verrons dans le reste de ce document, exigent une attention particulière.

Dans ce qui suit, nous dressons la liste d'attaques les plus probables contre les réseaux ad hoc.

1.4.4.1 Attaques de déni de services (DoS)

Les attaques DoS constituent une menace sévère de sécurité dans n'importe quel système distribué. Dans les MANETs ces attaques visent à mettre en péril la disponibilité des nœuds du réseau et plus particulièrement des entités qui jouent le rôle de serveurs ou de contrôleurs au sein du réseau ad hoc. Les attaques DoS sont possibles à diverses couches, à savoir, couche physique, couche MAC, couche réseau et également sur les applications s'exécutant dans de tels réseaux.

Les modèles de dénis de services qui suivent s'appliquent plus particulièrement dans le cas de réseau ad hoc [69] :

- Brouillage du canal radio pour empêcher toute communication.
- Tentative de débordement des tables de routage des nœuds servant de relais.
- La non coopération des nœuds au bon fonctionnement du réseau pour préserver de l'énergie. Cette attaque est connue sous le nom de « *Selfishness* » et peut être détectée grâce à des mécanismes de réputation et de détection des comportements égoïstes.
- Tentative de gaspillage de l'énergie des nœuds ayant une autonomie de batterie faible. Cette attaque est connue sous le nom de « *Sleep Deprivation* » et consiste à faire en sorte que le nœud soit obligé de rester en état d'activité et ainsi de lui faire consommer toute son énergie.
- Attaques sur les mécanismes de sécurité eux même.

1.4.4.2 L'attaque de consomation de ressources (Sleep Deprivation)

Habituellement cette attaque [25] est possible seulement dans les réseaux ad hoc lorsque la durée de vie de la batterie est un paramètre critique. Les nœuds fonctionnant à batterie essayent de conserver l'énergie en transmettant seulement si

absolument nécessaire. Dans cette attaque, un nœud malicieux agit sur un nœud cible avec l'intention d'épuiser la batterie de ce nœud victime. Par exemple, un adversaire peut essayer de consommer la puissance de batterie d'un nœud en lui envoyant des requêtes de routage, ou en expédiant des paquets inutiles à ce nœud, ou en falsifiant le routage pour acheminer une quantité massive de trafic vers ce nœud.

1.4.4.3 Attaque par usurpation de l'identité d'un nœud

Cette attaque appelée aussi *Spoofing* constitue une menace sévère de sécurité dans les MANETs. Si une authentification appropriée des parties n'est pas pris en charge, un nœud malicieux peut se faire passer pour un nœud légitime auprès des autres nœuds afin de récupérer illégalement des informations confidentielles ou d'injecter des messages erronés dans le réseau. Une partie malveillante peut avoir sa clé publique certifiée même sans qualifications appropriées. Ce type d'attaque met en péril l'authentification et le contrôle d'accès des membres du réseau ad hoc.

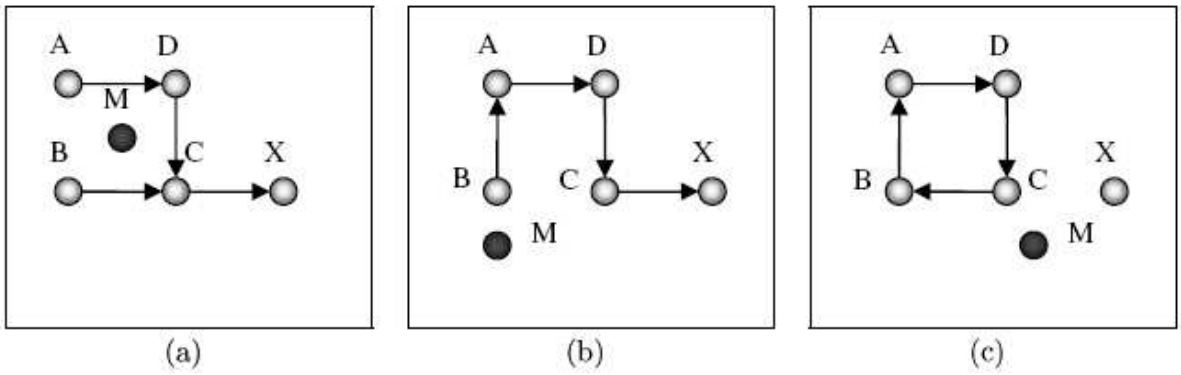


Fig.1.3 Une topologie en boucle formée par une attaque de *spoofing* par le nœud malicieux *M*

La figure 1.3 illustre un exemple d'une attaque de *spoofing* qui a pour conséquence de créer une topologie en boucle dans un réseau ad hoc. On suppose qu'il y a un chemin entre les quatre nœuds et la destination *X*, dans un processus de découverte de route, le nœud malicieux *M* peut falsifier ces chemins comme suit; M se fait passer pour *A* ensuite il se rapproche à *B*, il envoie à *B* un RREP qu'il contient un saut vers *X* ce qui est moins de celui envoyé par *C*, *B* change donc sa route vers *X* en traversant par *A* comme c'est illustré dans la figure 1.3 (b). Ensuite, *M* se fait passer pour *B* et se déplace pour se rapprocher de *C* et lui envoyer un RREP contenant un nombre de sauts vers *X* inférieur à celui envoyé par *X* lui-même. *C* change alors sa route vers *B*. À ce point une boucle est crée et le nœud *X* est isolé.

1.4.4.4 Les attaques passives d'écoute clandestine et d'analyse de trafic

Ces attaques appelées aussi « *sniffing* », consistent à écouter le réseau dans lequel transitent des paquets de données. Ces données, qui peuvent être confidentielles, sont récupérés à la volé et de manière illégale. Les attaques d'écoute et d'analyse de trafic sont plus dangereuses dans les environnements ad hoc. En effet, les ondes radio ont intrinsèquement une grande capacité à se propager dans toutes les directions avec une portée relativement grande, facilitant ainsi à une personne non autorisée d'écouter le réseau, décoder les messages, envoyés et obtenir des informations sensibles (Fig.1.4). Ces attaques constituent une menace certaine pour la confidentialité des données ainsi que pour l'anonymat des utilisateurs. L'émetteur n'est pas en mesure de détecter que la transmission a été interceptée. Toutefois, cette attaque peut être évitée en employant un schéma cryptographique pour protéger les données transmises. Cela, néanmoins, nécessite des stratégies efficaces de distribution de clés pour qu'elles soient transmises à tous les nœuds d'une manière sécurisée.

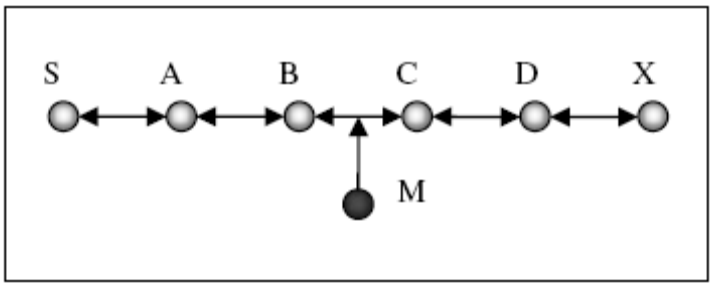


Fig.1.4 Un simple réseau ad hoc avec un nœud malicieux M

1.4.4.5 Les attaques sur le mécanisme de routage

Le routage est l'une des fonctions les plus critiques dans les MANET, il est également l'une des principales cibles des adversaires. Dans les MANET, les attaques contre le routage sont généralement classées en deux catégories : attaques sur les protocoles de routage et attaques sur l'expédition/livraison de paquets [70]. Les attaques sur les protocoles visent à bloquer la propagation des informations de contrôle de routage. Les attaques sur la transmission de paquets essayent de perturber la livraison de paquets le long des chemins de communication.

Nous présentons ci-dessous quelques attaques visant à compromettre le mécanisme de routage :

- ***L'attaque Black hole***. Dans cette attaque, un nœud malicieux utilise un protocole de routage pour annoncer aux autres qu'il est le nœud ayant le plus court chemin vers la destination [31]. Le nœud malicieux recevra donc le trafic

envoyé vers la destination et peut donc choisir de tuer ce trafic pour réaliser un déni de service.

- ***Packet Dropping***. Un nœud malicieux peut tuer des paquets contenant des informations de contrôle de routage destinés aux autres nœuds valides du réseau.
- ***L'attaque wormhole***. L'attaque du trou de vers (wormhole) [71] consiste à capturer le trafic à partir d'un point du réseau et le rejouer à partir d'un autre point distant. Dans cette attaque, un nœud malicieux envoie le trafic capturé à un autre nœud malicieux situé à plusieurs sauts, ce dernier nœud injecte de nouveau le trafic reçu dans le réseau créant ainsi un tunnel de trou de ver entre les deux nœuds malicieux pour faire croire aux autres nœuds que les deux extrémités du tunnel sont voisine. La sévérité de l'attaque de trou de ver vient du fait qu'il est difficile de la détecter, et elle est efficace même dans un réseau où la confidentialité, l'intégrité, l'authentification et la non-répudiation sont préservées.
- ***Rushing Attack.*** Cette attaque affecte les protocoles de routage réactifs, dans ce type de routage un nœud ne transmet que le premier message de type *Route Request* reçu et rejette le reste. L'adversaire peut exploiter cette caractéristique du routage réactif, il transmet rapidement toutes les *Route Request*. En conséquence, les noeuds qui reçoivent ces requêtes précipitées, les retransmettent et rejettent les autres *Route Request* qui arrivent bien après. En résultat, les routes vont passer par l'adversaire, ce qui le met dans une position avantageuse.

1.4.4.6 L'attaque Sybil

Dans cette attaque [9] un nœud malicieux pratique une *usurpation d'identité multiple*. Les identités multiples peuvent être obtenues soit par l'usurpation de l'identité d'autres noeuds, soit en faisant usage de fausses identités. Ces identités peuvent être utilisées simultanément ou étalé sur une période de temps. Cette attaque peut affecter plusieurs services dans un réseau ad hoc. Par exemple elle peut affecter le routage par trajets multiples, où un ensemble de chemins disjoins peuvent tous passer par le même noeud malveillant qui emploie plusieurs identités Sybil. Les mécanismes d'attribution équitable de ressource seront également affectés puisqu'un noeud malicieux peut réclamer plus que sa part en employant les diverses identités de Sybil. Les mécanismes basés sur la confiance peuvent également être touchés par cette attaque. Une approche simple pour détecter les identités de Sybil pourrait être de fournir un certificat de clé publique à chacune des identités. Le problème, cependant, est le besoin d'autorité centrale qui distribue les certificats.

1.4.4.7 Attaque par chantage

Elle est connue sous le nom anglais de *Blackmail attack*. Un noeud malicieux fait annoncer qu'un autre noeud légitime est malicieux pour éliminer ce dernier du réseau. Si le noeud malicieux arrive à attaquer un nombre important de noeuds, il pourra perturber le fonctionnement du réseau.

1.4.4.8 Les attaques physiques d'un nœud valide du réseau

Ces attaques compromettent des nœuds valides du réseau (destruction, altération ou changement physique d'un composant) et s'avèrent être des attaques particulièrement dangereuses.

1.5. Conclusion

Les réseaux ad hoc offrent une grande flexibilité de déploiement et sont de plus en plus utilisés. Ces réseaux représentent également un environnement hostile qui apporte plusieurs défis de sécurité, dus à leurs caractéristiques et spécificités (liens sans fil, capacité de calcul limitée, absence d'infrastructure, etc.). Dans ce chapitre, nous avons d'abord décrit brièvement les réseaux ad hoc et leurs caractéristiques particulières, ensuite, nous avons présenté les défis majeurs en sécurité relevés dans ce domaine. Nous avons aussi présenté les vulnérabilités des réseaux ad hoc induites par leurs caractéristiques ainsi que les attaques potentielles. Ces dernières ciblent principalement les fonctions du réseau tel que le routage et les mécanismes de sécurité. Il est clair que les mécanismes de sécurité utilisés dans les réseaux traditionnels ne peuvent pas être directement appliqués aux réseaux ad hoc vu de la différence architecturale qui existe entre ces deux environnements. Les réseaux ad hoc nécessitent donc le développement de nouveaux mécanismes robustes, qui s'adaptent à leur nature et tiennent compte leurs différentes caractéristiques et leurs vulnérabilités. Dans le chapitre suivant, nous présentons les mécanismes et solutions de sécurité des réseaux ad hoc proposés dans la littérature.

Chapitre 2

Mécanismes de sécurité pour les réseaux Ad Hoc

De nombreuses approches ont été proposées dans la littérature pour traiter une variété de mécanismes de sécurité et de contre-mesures afin de fournir les services de base de sécurité aux réseaux ad hoc. Ce chapitre aborde les principaux schémas et solutions de sécurité visant la conception de mécanismes fonctionnels de base pour sécuriser les différentes fonctions d'un réseau ad hoc. Les solutions présentées dans ce chapitre incluent des mécanismes pour l'authentification des nœuds, la sécurité des protocoles de routage, la gestion de clés, et autres mécanismes complémentaires.

2.1. Introduction

Les réseaux ad hoc sont actuellement au cœur d'une activité de recherche soutenue. En particulier, la sécurité dans les réseaux ad hoc constitue encore aujourd'hui l'un des principaux obstacles à un large déploiement de ces réseaux. En effet, la sécurité est devenue une préoccupation primaire afin de fournir la protection des communications entre les nœuds dans un environnement potentiellement hostile. Bien que la sécurité ait longtemps été un sujet de recherche dans les réseaux filaires, les caractéristiques uniques des MANETs présentent de nouveaux défis de sécurité qu'il faut absolument relever. Le but ultime des solutions de sécurité pour les MANETs consiste à fournir aux nœuds mobiles du réseau, les services de base de sécurité tels que la confidentialité, l'authentification, l'intégrité, la disponibilité et la non-répudiation. Afin d'atteindre cet objectif, la solution de sécurité doit fournir une protection complète couvrant toute la pile de protocoles.

Les solutions de sécurité pour les réseaux ad hoc sont particulièrement difficiles à fournir dû à la conjonction de plusieurs facteurs. En raison du manque d'une relation de *confiance apriori* entre les nœuds du réseau, les modèles classiques de sécurité basés sur l'établissement apriori de la confiance sont inadaptés et ne

peuvent donc être directement appliqués aux réseaux ad hoc. De plus, les solutions de sécurité qui supposent l'existence d'un un point central tels que une tierce partie de confiance ou un serveur de clés, ne sont pas compatibles avec la définition de base d'un réseau ad hoc où une infrastructure prédéfinie n'est pas disponible.

Les défis fondamentaux de sécurité présentés dans le chapitre précèdent indiquent les nouveaux besoins de sécurité et la nécessité de développer de nouveaux mécanismes adéquats pour l'architecture ouverte de ce paradigme de réseaux. Des études ont été consacrées à la sécurisation des réseaux ad hoc. Toutefois, les techniques proposées offrent des solutions partielles et adaptées à un type particulier d'applications ou de protocoles.

2.2. Les services de base de la sécurité

La sécurité est une question importante pour les réseaux ad hoc et toute conception de sécurité pour ces réseaux doit fournir les services de base de sécurité qui sont *la confidentialité, l'authentification, l'intégrité, la disponibilité* et *la non répudiation*. Bien que les besoins de base en sécurité sont identiques à ceux des réseaux traditionnels, garantir ces services soulève de nouveau défis.

2.2.1. La confidentialité

Un nœud malicieux ne devrait pas pouvoir accéder à des informations confidentielles en transit entre les nœuds du réseau. Ainsi, un mécanisme qui assure ce service de base que la transmission de l'information ne soit interceptée que par l'entité prévue. Dans le cas des réseaux ad hoc, la confidentialité est un critère important pour protéger la transmission des informations sensibles. Ceci est particulièrement crucial dans les MANET compte tenu du fait que les liaisons sans fil sont facilement vulnérables à l'écoute clandestine. Les informations relatives au trafic de contrôle comme le routage doivent rester secrètes dans certaines situations. Généralement, les techniques cryptographiques sont employées pour assurer la confidentialité.

2.2.2. L'intégrité

L'intégrité garantit que les données ne seront pas modifiées d'une manière non autorisée au cours de la transmission. Principalement, l'intégrité peut être compromise de deux façons; modification malveillante ou modification accidentelle. Un message peut être retiré, rejoué ou révisée par un adversaire ce qui est considéré comme modification malveillante. On utilise souvent les fonctions de hachage pour assurer l'intégrité.

2.2.3. L'authentification

S'assurer de l'identité des nœuds en cours de communication. Il existe deux types d'authentification, l'authentification des entités et l'authentification des données. L'authentification des entités consiste à vérifier l'identité de l'autre partie communicante. Sans mécanisme d'authentification, un nœud malicieux peut se usurper l'identité d'un nœud valide pour obtenir un accès aux ressources. D'autre part, l'authentification des données se focalise de fournir des garanties quant à l'origine des données.

2.2.4. La disponibilité

La disponibilité signifie que les services fournis par un nœud continuent à être fournis en présence d'attaques et de nœuds compromis. En d'autres termes, cela assure la survie des services indépendamment des attaques de déni de services. Ces attaques peuvent se présenter au niveau de différentes couches d'un réseau ad hoc. La disponibilité donne aussi une assurance sur la réactivité et le temps de réponse du réseau.

2.2.5. La non-répudiation

Empêcher un noeud de nier l'envoi ou bien la réception d'un message. Ceci est utile surtout lorsque on doit distinguer si un noeud avec certain comportement anormal est compromis ou non.

2.3. Les nouveaux besoins de sécurité dans les réseaux ad hoc

Les problèmes de sécurité présentés dans le chapitre précédent montrent à quel point les réseaux ad hoc sont vulnérables aux attaques malveillantes et que la sécurité est de grande importance et présente un véritable défi dans les réseaux ad hoc. Ainsi, les MANET apportent de nouveaux besoins en terme de sécurité, et de nouvelles solutions adéquates doivent être développées. Nous présentons ci-après quelques nouveaux besoins de sécurité liés aux MANET :

- Dans un réseau ad hoc, les nœuds compromis ne peuvent pas être détecté par simple mécanisme d'authentification, ce problème ne peut pas être résolu en utilisant la cryptographie. Donc, outre les mécanismes d'authentifications, il faut considérer de nouvelles solutions permettant la détection des comportements malveillants des nœuds et l'isolation de ces nœuds.

- Les solutions de sécurité traditionnelles sont conçues pour des nœuds relativement puissants, donc elles ne sont pas adaptées aux MANETs. Par conséquent, les nouvelles solutions destinées aux réseaux ad hoc doivent prendre en compte la faible capacité des nœuds.

- Le problème d'égoïsme n'existe pas dans les réseaux traditionnels [] où les nœuds ne dépendent pas les uns des autres mais les fonctions de base du réseau sont assurées par des éléments dédiés comme les routeurs. Donc, de nouveaux mécanisme doivent être conçu garantir la coopération des nœuds dans les réseaux ad hoc.

- La plupart des mécanismes de sécurité traditionnels sont basés sur l'établissement a-priori de la confiance ou sur une autorité centrale de confiance. Ces mécanismes incluent des cryptosystèmes symétriques et/ou asymétriques pour authentifier les nœuds et sécuriser les échanges. Ainsi, les réseaux ad hoc ont besoin de nouvelles solutions pour établir des relations de confiance d'une manière distribuée et sans l'aide d'aucun point central.

- Les mécanismes de sécurité sont eux-mêmes un sujet aux attaques. Ainsi, les solutions proposées pour sécuriser les réseaux ad hoc doivent être robustes et capables à se protéger contre les nœuds compromis.

2.3.1. Fonctions et données sensibles à protéger

Les fonctions sensibles des noeuds d'un réseau ad hoc sont principalement le routage, et les mécanismes de sécurité en particulier la gestion de clés. La plupart des données sensibles sont directement liées à ces fonctions puisqu'il s'agit des données relatives au routage (tables de routage et données de configuration des mécanismes de routage) et d'autre coté des données relatives à la sécurité (clés cryptographiques, mots de passe, certificats...).

2.3.1.1 Le routage

Les protocoles de routage des réseaux ad hoc s'adaptent bien au changement dynamique de la topologie du réseau. Cependant, leur conception ne tient pas compte un aspect très important qui est la sécurité contre des attaques malveillantes. Un protocole de routage sécurisé doit satisfaire les conditions suivantes pour qu'une route établie entre la source et la destination soit correcte même en présence des nœuds malicieux :

- Des messages de routage fabriqués ne peuvent pas être injectés dans le réseau.
- Des messages de routage en transit ne peuvent pas être modifiés.

- Des boucles de routage ne peuvent pas être formées par une action malveillante.
- Le trafic lié au routage doit être authentifié.

En plus de ces conditions de base, un protocole de routage sécurisé repose sur le déploiement d'un schéma de gestion de clés pour la distribution et la gestion des clés cryptographiques nécessaires durant l'exécution du protocole de routage.

2.3.1.2 La gestion de clés

Des schémas cryptographiques tels que les signatures digitales sont souvent employés afin de protéger à la fois des informations liées au routage ainsi que le trafic lié aux données des utilisateurs. Les systèmes à clés publiques sont généralement adoptés en raison de leur flexibilité dans la distribution des clés. Dans une infrastructure à clés publique (PKI), chaque nœud possède une paire de clés privée/publique. Les clés publiques sont publiquement connues par tous les participants du réseau, alors que les clés privées sont conservés aux noeuds eux-mêmes et doivent rester confidentielles.

Généralement les schémas de gestion de clés publiques sont basés sur un tiers de confiance appelé *Autorité de Certification* (CA). Une CA possède également sa propre paire de clés privée/publique, son rôle principal consiste à signer des certificats de clés publiques aux autres nœuds pour lier une clé publique à son véritable détenteur. La CA doit rester en ligne pour fournir les services de base de certification (Mise à jour des certificats, révocation, création,…). Cependant, vu des caractéristiques des réseaux ad hoc, les architectures centralisées de PKI ne sont pas appropriées à ce paradigme de réseau, se baser sur une CA centrale pour fournir tout le service de certification conduit à un état de vulnérabilité. Si la CA est indisponible, les nœuds du réseau ne peuvent plus avoir les clés publiques des autres nœuds pour établir une communication sécurisée entre eux. De plus si la CA se trouve compromise, un adversaire peut signer de faux certificats. Une réplication naïve de la CA peut rendre le système plus vulnérable, du moment que la compromission d'une seule CA peut faire échouer le système. Ainsi, il est plus prudent de distribuer la confiance à un ensemble de nœuds en permettant à ces nœuds de partager la responsabilité de la gestion de clés.

2.4. Schémas et mécanismes de sécurité dans les réseaux ad hoc

Pour faire face aux obstacles de sécurité discutés précédemment, plusieurs approches et mécanismes de sécurité pour les réseaux ad hoc ont été proposés dans

la littérature. La plupart de ces mécanismes sont principalement classés en trois catégories :

- **Mécanismes pour l'intégrité et l'authentification**. Pour assurer l'intégrité et l'authentification des messages échangés entre les nœuds du réseau ad hoc.
- **Mécanismes pour la sécurité du routage**. Ces mécanismes assurent la confidentialité, l'intégrité, l'authentification et la non-répudiation dans les deux phases de routage : la découverte de topologie et l'acheminement des données.
- **Mécanismes pour la gestion de clés**. Ces mécanismes traitent l'identification et toutes les opérations sur les clés (création, distribution, révocation, rafraîchissement…)

Nous discutons dans cette partie les solutions pertinentes de sécurité dans ces trois catégories. D'autres mécanismes complémentaires comme la détection des comportements malveillants et le renforcement de coopération sont également présentés dans cette partie.

2.4.1. Mécanismes pour l'intégrité et l'authentification des messages

Les moyens classiques pour assurer l'intégrité et l'authentification des messages échangés entre les noeuds d'un réseau sont basés sur l'utilisation des schémas cryptographiques symétrique, asymétrique ou de MACs (Message Authentication Code) [40]. Une approche typique utilisant la cryptographie asymétrique est basée sur l'utilisation de signatures numériques. Cependant, cette approche est onéreuse, particulièrement en terme de calcule, la génération et la vérification et les communications associés aux signatures numériques engendre un coût important. Ainsi, les algorithmes de signature utilisés dans ces approches doivent être sophistiqués et prennent en compte la faible capacité des nœuds ad hoc.

Une autre solution consiste à utiliser des MACs. Il s'agit de fonctions mathématiques à sens unique dépendant d'une clé secrète qui à partir du message original, produisent un condensé de ce message. Ces fonctions mathématiques sont telles qu'il est difficile de retrouver un message à partir de son condensé ou de produire deux messages ayant le même condensé. De plus, la moindre modification du message original entraîne un changement dans le condensé. L'inconvénient de cette solution est qu'il est nécessaire au préalable d'établir autant de clés secrètes que de paires de nœuds susceptibles de vouloir communiquer ensemble. D'autres solutions doivent donc être envisagées pour les réseaux ad hoc.

2.4.1.1 Le protocole TESLA

Perrig *et al* proposent **TESLA** (**T**imed **E**fficient **S**tream **L**oss-tolerant **A**uthentication) [45] qui utilise uniquement des primitives de cryptographie symétrique. TESLA permet d'authentifier les messages avec un MAC dépendant d'une clé secrète qui n'est divulguée par l'émetteur du message qu'après un délai d'attente *d*, La valeur *d* est calculée de manière à ce qu'on soit sûr que le destinataire a reçu le message avant la divulgation de la clé, cette condition garantie l'intégrité du message.

La clé secrète utilisée pour générer le MAC est issue d'une chaîne de clés. L'élément initial K_n est choisi aléatoirement par l'émetteur, Ensuite les autres éléments de la chaîne sont calculés récursivement de la manière suivante : $K_{i-1} = H(K_i)$ où *H* est une fonction de hachage non réversible. L'émetteur va utiliser ces clés dans l'ordre inverse de leur génération c'est à dire dans l'ordre $K_0, K_1, \ldots, K_n$. En réception, le destinataire vérifie la relation $K_{i-1} = H(K_i)$ où K_i est la clé dernièrement reçue et K_{i-1} la clé précédente. Cette condition permet de vérifier que la clé K_i fait bien partie de la chaîne de clés de l'émetteur, ce qui assure la propriété d'authentification de la source. Le déroulement plus détaillé du protocole TESLA peut être illustré dans la figure 2.1. L'émetteur choisit une clé secrète puis procède à la génération d'une chaîne de clés. La dernière valeur de la chaîne de clés est envoyée au récepteur. La longueur de la chaîne détermine la durée maximale de transmission avant qu'une nouvelle chaîne soit générée. L'émetteur définit des intervalles de temps de durée uniforme puis attribue à chaque intervalle *i* une clé K_i de la chaîne. Durant chaque intervalle l'émetteur calcule le MAC de chaque message en utilisant la clé correspondante à cet intervalle, le message transmis contient les données originales, le MAC calculé et la valeur récente de la chaîne de clé qui puisse être révélée, donc l'émetteur divulgue les clés après le *time interval* de leur utilisation. En réception, le récepteur vérifie que la clé utilisée pour calculer le MAC sur le message reçu est encore secrète, dans ce cas il enregistre ce message jusqu'à ce que la clé soit révélée, il peut donc vérifier la validité de la clé et ainsi authentifier le message

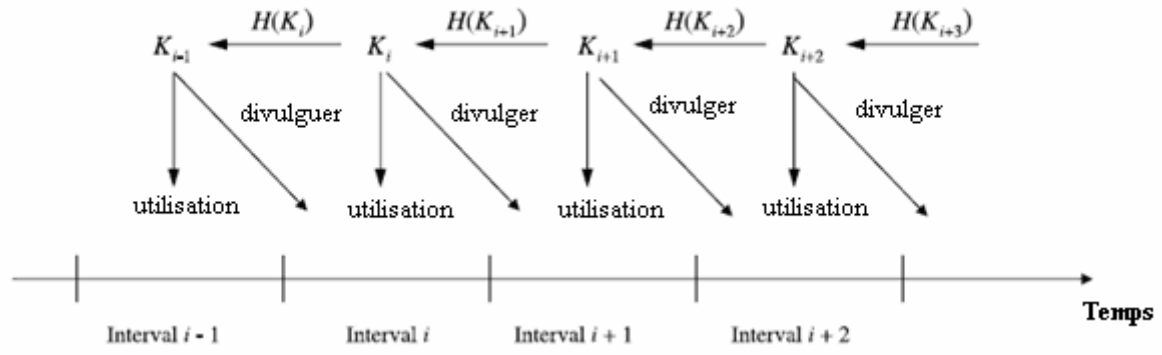


Fig.2.1 Fonctionnement de TESLA

2.4.2. Mécanismes de sécurité de routage dans les réseaux ad hoc

Dans un réseau purement ad hoc les nœuds mobiles doivent coopérer pour fournir les services et les fonctions de base du réseau, en particulier le routage. Les protocoles de routage actuels proposés (comme par exemple OLSR, DSR, AODV…etc.) ne tiennent pas compte de l'aspect sécurité et sont exposés à certain nombre d'attaques actives qui visent à affecter le mécanisme de routage.

Les mécanismes de routage dans les réseaux ad hoc sont très sensible aux problèmes de sécurité : chaque noeud peut être un routeur et les protocoles de routage conçus partent par défaut du principe que tous les noeuds sont dignes de confiance. Sans protocole sécurisé, Si un nœud est compromis, il n'y a pas un moyen de le détecter, ce nœud peut facilement perturber le fonctionnement normal du protocole de routage, et l'acheminement de donnée ne sera pas assuré ni sécurisé.

Un protocole de routage sécurisé doit être capable d'établir une communication entre deux noeuds même en présence de noeuds malicieux. Il doit aussi avoir la capacité de détecter les noeuds malicieux et d'avertir les autres nœuds légitimes du comportement de ces noeuds. Les noeuds légitimes vont donc écarter les chemins contenant des noeuds malicieux.

Dans la littérature, plusieurs protocoles de routage sécurisés sont proposés : SRP, SAODV, ARAN, Ariadne, SEAD. Dans ces protocoles sécurisés les mécanismes cryptographiques sont largement utilisés pour protéger les messages de contrôle de routage, l'intégrité et l'authentification de ces messages sont assurées par l'utilisation de signature numérique ou de MACs (Message Autehtication Code). Dans la suite de cette section, nous présentons quelques protocoles de routage sécurisés proposés dans la littérature, sans entrer dans les détails de conception et de fonctionnement.

SRP (Secure Routing Protocol) [28], proposé par Papadimitratos et Haas, est un protocole de routage sécurisé conçu comme une extension qui peut être appliqué à une multitude de protocoles réactifs existants en particulier DSR. Il nécessite une SA (Security Association) entre chaque paire de nœuds communiquant, dans laquelle une clé secrète est partagée. Lors de l'initialisation de la découverte de route, une RREQ est envoyée dont l'entête SRP contient : le numéro de séquence, un nonce, et MAC (hachage calculé sur l'entête IP, le numéro de séquence et la clé secrète). Le MAC permet au destinataire de vérifier l'intégrité et l'authentification de la requête en calculant un hachage sur les champs de l'entête et le comparant avec le MAC contenu dans la requête.

SAODV est la version sécurisée du protocole AODV proposé par Zapata *et al* [29,30], les messages RREQ et RREP sont signés par le nœud expéditeur, et la

signature est vérifiée par les nœuds intermédiaires avant l'acheminement du message. Ce protocole nécessite un serveur CA pour gérer une PKI, avec laquelle les messages de routage sont signés par leurs créateurs. Il garantie l'intégrité et l'authentification des messages de contrôle du protocole AODV.

SEAD (*Secure Efficient Ad hoc Distance vector* routing protocol) [33] est basé dans sa conception sur le protocole de routage DSDV, pour supporter l'utilisation de SEAD avec des nœuds limités en terme de puissance de calcule et pour lutter contre les attaques DoS qui essaient de causer d'autres nœuds à consommer trop de bande passante réseau et du temps de calcul, ce protocole utilise des fonctions de hachage non réversibles et n'emploie pas la cryptographie asymétrique. Les nœuds s'authentifient les uns avec les autres en utilisant des chaînes de hachage (*one-way hash chain*), pour calculer un *one-way hash chain,*un nœud choisit un nombre aléatoire x , et calcule $h0, h1, h2, h3, \ldots, h_n$ où $h_0 = x$, et $h_i = H(h_{i-1})$, pour $i \leq n$, pour authentifier par exemple h_{i-3} étant donné h_i on calcule $H(H(H(h_{i-3})))$ et on vérifie que le résultat donne h_i, de cette manière, dans SEAD, la valeur h_n est d'abord distribuée, ensuite le nœud inclut ces valeurs dans ses messages, une dans chaque message dans l'ordre inverse de h_{n-1} à h_0, les nœuds récepteurs connaissant h_i et authentifient la valeur h_{i-1} en vérifiant que $H(h_{i-1}) = h_i$

ARAN (Authenticated Routing for Ad hoc Networks) proposé par Dahill *et al* [32], introduit l'authentification, l'intégrité de messages et la non répudiation à un environnement ad hoc en se basant sur l'utilisation de signatures numériques.
Le protocole ARAN impose l'utilisation d'un tiers de confiance ou serveur de certificat CA et supposent que la clé publique de ce serveur est connue par tous les nœuds. Avant de rejoindre le réseau chaque nœud A doit obtenir son certificat de la part de CA :

$$CA \rightarrow A : Cert_A = [IP_A, K_A, t, e]_{K^{-1}_{CA}}$$

Dans la phase de la découverte de route, l'initiateur S diffuse un message de *route discovery* (RDP) contenant son certificat et un nonce N en plus d'autres informations, le tout est signé par la clé privé de S, le premier nœud intermédiaire I_1 recevant le message RDP, ajoute sa signature et son certificat à ce message et le fait suivre. Chaque nœud intermédiaire suivant qui reçoive le message vérifie d'abord la signature de I_{i-1} , la supprime en la replaçant par la sienne ensuite rediffuse le message.

Le destinataire D prend le premier message RDP reçu (pas nécessairement celui de plus court chemin), le signe par sa clé privée et répond avec un message reply REP, le REP est envoyé de la même façon que RDP mais par le chemin inverse, en recevant le REP par l'initiateur S, celui-ci vérifie que son noce N et la signature de D sont valides, cette procédure est schématisée dans la figure 2.2.

ARAN fournie une authentification et intégrité aux messages de contrôle de routage au prix des opérations cryptographiques lourdes et un serveur CA.

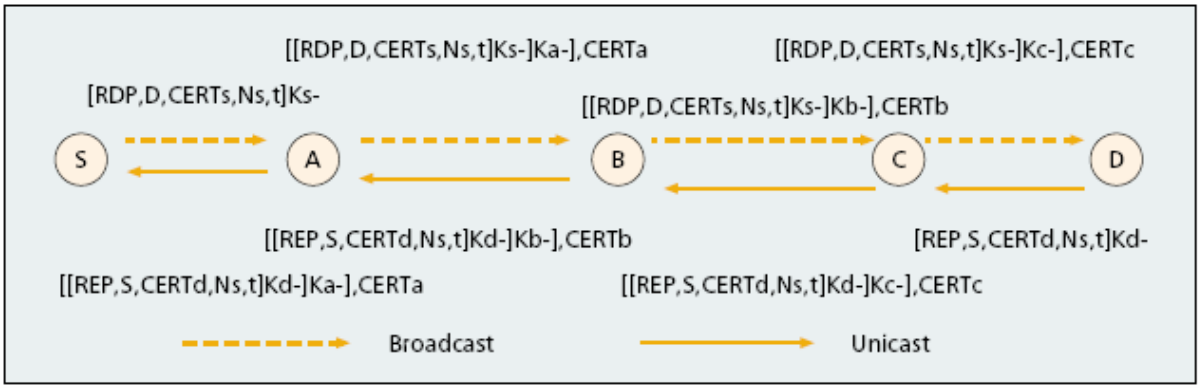


Fig.2.2 Établissement sécurisé de route avec ARAN

Ariadne, ce protocole développé par Hu *et al* [31] est basé sur le protocole DSR et implémente des mécanismes d'authentification des messages de routage en mettant en œuvre TESLA.
Chaque paire de nœuds partage une clé secrète et possède une chaîne de clés TESLA. Dans le processus de la découverte de route, la source (nœud initiateur) diffuse un message de *Route Request* contenant un *time interval* T en plus d'autres informations et protégé par un MAC (Message Authentication Code). Chaque nœud intermédiaire recevant ce massage, vérifie la validité du *time interval* pour s'assurer que la clé n'est pas révélée. Ensuite il ajoute à ce message un MAC qui est calculé à partir de tout le request et de sa clé TESLA, puis il fait suivre le paquet. Le nœud destinataire effectue les mêmes tests, si le paquet *route request* est considéré valide alors le destinataire renvoie un *route reply* à l'initiateur de la route à travers le chemin inverse de celui indiqué dans *route request* Avant de faire suivre le *paquet route reply*, chaque noeud intermédiaire attend jusqu'à ce que sa clé soit révélée selon le *time interval,* il ajoute alors sa clé TESLA avec laquelle il a calculé le MAC dans le paquet *route request*. Cette clé sera utilisée par l'initiateur de la Route Request pour authentifier ce noeud intermédiaire.

Ariadne est capable de fournir des routes authentifiées. Cependant, le protocole est moins adapté aux MANETs à large échelle, cela est parce que, premièrement, la taille de l'entête augmente rapidement avec la longueur de la route, deuxièmement, il est difficile d'estimer à l'avance le *time interval* T lorsque la dimension du réseau est importante.

D'autres protocoles de routage sécurisés non présentés ici, proposés dans la littérature comme par exemple SAR [36], SPAAR [37], EndairA [35] et ADVSIG [34].

On peut constater que l'introduction de mécanismes de sécurité dans un protocole de routage passe nécessairement par une diminution des performances de ce protocole. En effet, les entêtes des paquets de routage voient leur taille augmenter avec l'authentification des messages. De plus, une baisse de réactivité due au temps d'attente pour authentifier un message apparaît. Un compromis entre le niveau de sécurité et performance est donc clairement nécessaire. Il faut rester prudent avec les mécanismes d'authentification de messages. En effet, un attaquant peut générer une multitude de paquets sur le réseau avec une fausse authentification. Les nœuds sont alors submergés par un nombre trop important de paquets non valides à authentifier, ce qui ralentit le fonctionnement du réseau jusqu'à provoquer un déni de service.

2.4.3. Mécanismes pour la gestion de clés (*Key Management*)

L'authentification des parties apparaît comme la pierre angulaire d'un réseau ad hoc sécurisé pour assurer une quelconque confidentialité et intégrité des messages échangés. La solution jusqu'ici est les messages cryptographiquement signés. L'hypothèse générale est que des noeuds en possession d'une clé secrète valide doivent être dignes de confiance. En conséquence, un schéma de *key Management*, efficace et sécurisé est crucial. Toute solution de gestion de cles doit repondre aux questions suivantes :

- **Le modèle de confiance**. Il est indispensable de déterminer comment les différents éléments du réseau peuvent faire confiance les uns aux autres. L'environnement et le domaine d'application peuvent avoir un impact important sur le modèle de confiance. Par conséquent, les relations de confiance entre les nœuds du réseau affectent la manière selon laquelle le système de gestion de clés devrait être conçu.
- **Le cryptosystème**. Selon le contexte du réseau ad hoc quel sera le schéma cryptographique (symétrique ou asymétrique) adéquat quand un niveau semblable de sécurité est nécessaire. A d'autres contextes, il existe également des cryptosystèmes basés sur la courbe elliptique.
- **La génération de clés**. Il est indispensable de déterminer quelles sont les parties autorisées à générer des clés à elles-mêmes ou à d'autres parties et quels types de clés.
- **Stockage des clés**. Dans un réseau ad hoc on peut pas avoir un stockage centralisé des clés ni une réplication de ce stockage. Dans les réseaux ad hoc, un nœud doit stocker localement sa propre clé et les clés des autres éléments dans certains cas. De plus, dans certaines propositions, le partage de secret [1] est appliqué pour distribuer des parts de clés à plusieurs nœuds. Dans une telle approche, la compromission d'un seul nœud n'affecte pas la confidentialité d'une clé secrète.
- **Distribution des clés**. Un service de gestion de clés doit assurer que les clés générées seront distribuées d'une manière sécurisée à leurs possesseurs. Dans

le cas d'un schéma de clés symétrique, toutes parties impliquées doivent recevoir la clé secrète d'une manière hautement sécurisée. Pour les schémas asymétriques, le mécanisme de distribution doit assurer que les clés privées sont délivrées uniquement aux parties autorisées. Pour les clés publiques la confidentialité n'est pas nécessaire mais l'intégrité et l'authentification des clés doivent être préservées.

Un schéma de clé dans un réseaux ad hoc peut être asymétrique ou symétrique, dans le premier cas, ça nécessite le déploiement d'une infrastructure à clé publique PKI (Public Key Infrastructure) où chaque nœud dispose d'une paire de clés publique / privée, dans le second cas, il gère principalement des clés symétriques, chacune étant partagée par tous les nœuds du réseau.

Le choix entre l'utilisation d'un schéma de clé asymétrique ou symétrique dans les réseaux ad hoc, dépend souvent du scénario du réseau (type du réseau, application…etc.), les schémas symétriques sont mieux adaptés aux petits réseaux avec des nœuds limités en terme de capacité de calcul comme les réseaux de capteurs, alors que les schémas asymétriques sont adaptés aux réseaux avec une grande densité de nœuds, ou hautement dynamiques.

Toutefois, les approches de gestion de clés classiques utilisés dans les réseaux filaires ne sont pas bien appropriées aux réseaux ad hoc, vu de leurs caractéristiques dans ce contexte, de nouveaux schémas ont été développées pour faire face aux problèmes rencontrés dans les réseaux ad hoc, ces schémas seront abordés en détail dans le prochain chapitre dans lequel une classification de ces schémas sera donc dressée.

2.4.4. Mécanismes de sécurité complémentaires

2.4.4.1 Les IDS (Intrusion Detection System)

Un mécanisme additionnel appelé IDS (Intrusion Detection System) est souvent utilisé pour protéger les réseaux. Un IDS collecte et analyse les données du trafic afin de déterminer si des utilisateurs non Autorisés sont connectés ou si certains noeuds ont des comportements anormaux. L'utilisation d'un IDS au sein d'un réseau ad hoc peut être très utile pour renforcer la sécurité de celui-ci, puisque les MANETs n'ont pas de frontière de défense, un nœud malicieux peut facilement y pénétrer. Plusieurs architectures d'IDS adaptées aux réseaux ad hoc ont été proposées, dans [38] par exemple, Zhang *et al* proposent une architecture appelée « Distributed and Cooperative Intrusion Detection », dans ce modèle, chaque nœud est responsable de détecter localement et indépendamment les éventuelles intrusions, mais les nœuds voisins peuvent collaborer dans une détection globale. Les agents IDS individuels s'exécutent sur chaque nœud mobile. Chaque agent IDS

collecte les données locales et surveille les activités locales, il détecte les intrusions à partir des traces locales et initie des rapports d'intrusions. Alors que la détection coopérative d'intrusion globale peut être déclanchée quand un agent IDS rapporte une anomalie. Ces agents IDS ensemble forment un IDS pour protéger le réseau ad hoc.

2.4.4.2 Détection des comportements malveillants des noeuds

Pour contrer les attaques sur les mécanismes de routage comme par exemple *black hole*, où un nœud malicieux prétend être un relais pour un autre nœud mais ne transmet pas les messages de données, les techniques de détection des mauvais comportements sont très utiles pour détecter les intrusions dans le réseau et d'identifier les nœuds qui se comportent mal et qui essayent de perturber le réseau, ces nœuds peuvent être des nœuds légitimes qui ont été compromis ou des nœuds malicieux. Ces nœuds malicieux doivent être détectés et signalés aux autres nœuds légitimes et ensuite la mise en place d'une contre-mesure pour exclure ces nœuds du réseau. A noter que des comportements malveillants peuvent ne pas être due à la malice c'est-à-dire dans le cas d'un mal fonctionnement de l'équipement ou épuisement de la batterie.

Marti *et al*. [39] ont développé deux méthodes appelées *watchdog* et *pathrater* pour détecter et éliminer les nœuds malicieux du réseau. Le *watchdog* permet de vérifier si le noeud suivant a réellement fait suivre le paquet, en écoutant tous les paquets qu'il émet. Le pathrater est une technique permettant au protocole de routage d'éviter les nœuds malicieux inscrits dans une liste noire (*blacklist)*. Cette technique se base sur l'écoute passive des transmissions pour déceler si les paquets sont correctement relayés. Il est supposé une connectivité symétrique bidirectionnelle. Si A peut écouter B alors B peut également écouter A. si A transmis un paquet à B et connaît que le prochain saut de B est C, alors A écoute le canal B-C, et si pendant un timeout il constate qu'il n'y ait pas de transmission alors B est considéré malicieux. Il faut rester prudent quant à l'utilisation de ces mécanismes car ils peuvent être détournés par un attaquant. En effet, un nœud malicieux peut aussi faire en sorte qu'un nœud valide soit ajouté à la liste noire, l'isolant ainsi du réseau.

La référence [41] suit le même principe mais fonctionne avec les protocoles à vecteur de distance comme AODV. Il ajoute un champ *next_hop* au paquets AODV de tel sorte qu'un nœud soit conscient du vrai prochain saut dans l'ensemble de ses voisins, chaque détection indépendante est signée et inondée. Les multiples détections des différents nœuds peuvent collectivement révoquer un nœud malicieux de son certificat et ainsi l'exclure du réseau. Toutefois les messages d'alerte signés évitent que l'outil de détection soit abusé par le noeud malicieux.

2.4.4.3 Établissement de la confiance, évaluation des réputations

Dans une étude faite par Jinshan Liu *et* Valérie Issarny [41], un système de recommandation et de réputation a été proposé. Il s'agit de fournir à chaque nœud différents paramètres pour évaluer la qualité des autres nœuds. Parmi ces paramètres, on trouve SRep, qui est la réputation d'un nœud déduite par toutes les données disponibles et SExp qui est la réputation obtenue par les seules expériences entre ce nœud et soi. Chaque nœud possède également des informations sur la qualité de la recommandation des autres nœuds (RRep), ce qui indique la valeur à associer aux recommandations faites par ses pairs. Il est à noter que seule la recommandation faite par un nœud au sujet d'un autre (Rec) est distribuée aux autres nœuds.

Un autre mécanisme proposé pour faire face au comportement égoïste des noeuds d'un réseau ad hoc. Ce mécanisme, nommé CORE [41], se base sur la notion de réputation. Les auteurs montre que les performances d'un réseau MANET se dégradent sévèrement en présence d'un simple comportement illégitime des noeuds. Indépendamment des cas spéciaux comme pour les réseaux militaires pour lesquels une confiance à priori existe entre tous les noeuds, les noeuds d'un réseau ad hoc ne peuvent pas être considérés fiables pour l'exécution correcte des fonctions critiques du réseau.

Des opérations essentielles peuvent être fortement compromises par les nœuds qui n'exécutent pas correctement leur part des opérations comme le routage, l'expédition de paquets, etc... Un mauvais comportement des noeuds qui affecte ces opérations peut s'étendre de l'égoïsme ou du manque simple de collaboration dû au besoin d'économie de batterie ou aux attaques actives comme le déni de service. Une condition de base pour maintenir le réseau opérationnel consiste à imposer la contribution des noeuds aux opérations du réseau en dépit de la tendance contradictoire de chaque noeud vers l'égoïsme, motivée par la pénurie des ressources énergétiques.

Le mécanisme CORE [41], utilisé pour imposer la coopération entre les nœuds se base sur une technique de surveillance distribuée. Ce mécanisme de coopération n'empêche pas un noeud de nier la coopération ou de dévier d'un comportement légitime mais assure que les entités se conduisant mal soient punies en leur refusant graduellement les services de communication. CORE est suggéré comme mécanisme générique qui peut être intégré avec n'importe quelle fonction de réseau comme l'expédition de paquets, découverte de routes, gestion de réseau, et gestion de la localisation. Dans CORE, chaque entité réseau encourage la collaboration d'autres entités en utilisant une métrique de coopération appelée réputation. La métrique de réputation est calculée sur la base des données recueillies localement par chaque noeuds et peut se baser optionnellement sur l'information fournie par d'autres

noeuds du réseau impliqués dans des échanges de messages avec les noeuds surveillés.

2.5. Conclusion

Un réseau ad hoc est un environnement hostile qui apporte plusieurs défis de sécurité. Plusieurs solutions ont été proposées pour sécuriser les réseaux Ad Hoc. Ces solutions diffèrent selon le besoin en sécurité et les moyens possibles (autorité de certification, centre de distribution de clés, algorithmes cryptographiques, …). Il n'y a pas une solution complète satisfaisant toutes les caractéristiques et les contraintes des réseaux ad hoc. Toutefois les solutions proposées sont destinées à contrer certains types d'attaques ou adaptées à un type particulier d'application et de protocole. Actuellement, les principaux travaux de recherche portent sur l'authentification, la gestion de clés et la sécurité des protocoles de routage. Les solutions existantes pour sécuriser le routage ont pour but d'assurer l'authentification et l'intégrité des messages de routage. Il apparaît clairement que les mécanismes de gestion de clés constituent un point sensible pour la sécurité des réseaux ad hoc. L'authentification et une gestion efficace de clés, constituent donc le point de départ incontournable pour la sécurisation des réseaux ad hoc. Cet axe apparaît comme une direction de recherche importante auxquelles plusieurs études sont consacrées. Dans le chapitre suivant, nous présentons une étude des schémas actuels de la gestion de clés dans les réseaux ad hoc, proposés dans la littérature.

Chapitre 3

La gestion de clés dans les réseaux Ad Hoc : Etat de l'art

La recherche sur la sécurité des réseaux ad hoc a été initialement concentrée sur des protocoles de routage puisque ceux-ci ont été considérés comme la partie la plus critique d'un réseau ad hoc. Cependant, une analyse fine des travaux de sécurité du routage indique que la majorité des exigences adressées par les solutions et les mécanismes suggérés ne sont pas nouveaux ou spécifique aux MANET, mis à part d'un problème fondamental qui a été laissé de coté qui est la gestion de clés sans établissement d'une confiance a priori et en absence d'infrastructure. Ainsi, il est nécessaire d'étudier les solutions et les schémas de gestion de clés proposés pour les réseaux ad hoc ce qui fait l'objet de ce chapitre.

3.1. Introduction

Les services de sécurité sont un facteur essentiel au sein d'un réseau ad hoc pour assurer la confidentialité, l'intégrité et l'authentification des communications. La plupart des primitives de sécurité ne peuvent être réalisées que si un système efficace, robuste et sécurisé, de génération et de gestion de clés, soit développé. La gestion de clé est l'un des défis majeurs dans le développement de solutions de sécurité dans les réseaux ad hoc vu de l'absence d'infrastructure et d'autres caractéristiques liées à cette catégorie de réseaux

La gestion de clés inclut différentes techniques cryptographiques, il y a principalement deux catégories de systèmes, symétrique et asymétrique. Les systèmes cryptographiques symétriques sont difficiles à incorporer dans un environnement ad hoc décentralisé et pose le problème de distribution des clés. Les systèmes cryptographiques asymétriques utilisant une infrastructure à clé publique (PKI) ne peuvent pas être directement appliqués à un réseau ad hoc, dont les nœuds sont indépendants et mobiles et pourraient ne pas avoir la possibilité de se connecter

à la CA en permanence. En outre, la présence d'une entité centralisée constitue une vulnérabilité qui pourrait être exploitée par un adversaire pour porter des attaques DoS. Il s'agit d'un sérieux problème qui existe aussi dans les réseaux filaires.

Pour faire face à ces défis, plusieurs approches adaptées aux caractéristiques des réseaux ad hoc ont été proposées. Ce chapitre aborde les différents schémas et solutions en terme de gestion de clés, proposés dans la littérature, et discute leurs avantages et leurs limites.

3.2. La gestion de clés (concepts généraux et approches classiques)

3.2.1 Notions de base

La gestion de clés (*Key Management*) est la fonctionnalité fondamentale de toute architecture de sécurisation pour garantir la confidentialité, l'intégrité et l'authentification des communications. Lorsque on applique des schémas cryptographiques tel que les signatures digitales pour protéger le trafic de données et de contrôle, un service de gestion de clés est indispensable.

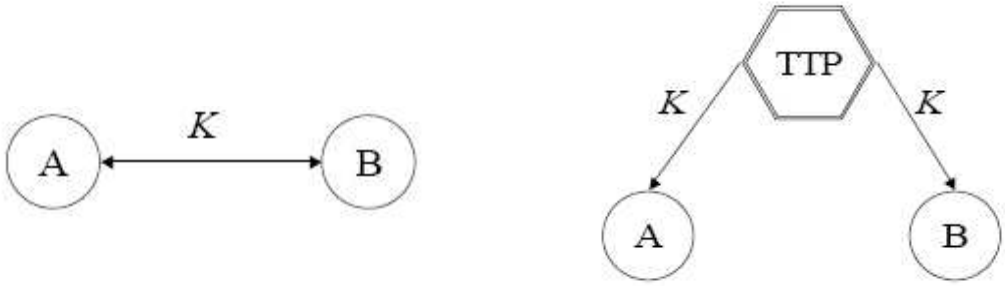
La gestion de clés est le processus par lequel ces clés sont établies et distribuées sur le réseau et mis à jour si nécessaire. Le processus de gestion de clés implémente les mécanismes suivants :

1- Initialisation des utilisateurs du système dans un domaine de sécurité (en registrant les identités des utilisateurs dans un serveur de sécurité)
2- Génération, distribution et installation des clés (faire parvenir la clé symétrique ou la paire de clés publique/privé et le certificat correspondant à l'utilisateur)
3- Organiser et Contrôler la façon d'utilisation des clés (l'utilisation cryptographique prévue d'une clé)
4- Renouvellement, révocation et destruction d'une clé.
5- Stockage et archivage des clés.

Il existe deux modèles dans le processus de distribution de clés :

1- ***Directe*** : Les deux parties communicantes établissent et distribuent la clé en communicant directement comme c'est montré dans la figure 3.1 (a).

2- ***Centralisé*** : Un tiers de confiance (TTP, Trusted Third Party) génère et distribue la clé pour les deux parties (Fig.3.1 (b)).



(a) (b)

Fig.3.1. Les deux modes de distribution de clés

Selon le deuxième modèle, le TTP peut être un centre de distribution de clés (KDC) ou une autorité de certification (CA). Le KDC est utilisé dans le cas des systèmes cryptographiques symétriques, pour générer et distribuer les clés symétriques partagées par les différentes parties communicantes. Tandis que la CA est utilisée dans le cas des systèmes cryptographiques asymétriques, ce type de système nécessite le déploiement d'une infrastructure à clé publique (PKI).

3.2.2. Modèles de confiance pour la gestion de clés

3.2.2.1 PKI

PKI (*Public Key Infrastructure*) est une architecture de sécurité qui fournie la confiance pour l'échange d'informations via les voies de communication insécurisées. Dans une PKI chaque nœud dispose d'une paire de clés publique / privée, on utilise les certificats pour faire correspondre une clé publique à son propriétaire, l'élément de base d'une PKI est l'autorité de certification (CA), celle-ci permet alors d'établir, de distribuer et de révoquer les certificats. Le certificat signé par la CA permet de garantir qu'une clé publique appartient bien à son propriétaire et non à un usurpateur. L'opération de vérification de certificat ne se limite pas à contrôler la signature de la CA. Il est aussi nécessaire de s'assurer que le certificat est toujours en cours validité et qu'il n'a pas été révoqué. En effet, une révocation du certificat est nécessaire si la clé privée du propriétaire est volée ou divulguée.

3.2.2.2 X.509

Le X.509 [46] est l'un des schémas de certificats à clé publique largement utilisé dans les réseaux traditionnels. Mais dans ce schéma une CA est indispensable pour

l'authenticité de la clé publique de tout nœud, la disponibilité de cette CA peut devenir un problème dans des systèmes distribués à large échelle. En effet, si la CA se trouve compromise, ça peut paralyser les fonctions de certification et ainsi la CA devient un point de vulnérabilité de la sécurité. L'implémentation distribuée d'une CA parait alors la solution adéquate afin d'améliorer la disponibilité des fonctions de certification, cela permet d'augmenter également la robustesse de l'autorité contre un certain nombre d'échecs de serveur par l'utilisation de la cryptographie à seuil. Cependant, ces avantages sont obtenus au prix d'une complexité additionnelle de protocole.

3.2.2.3 Kerberos

Kerberos [72] est un protocole d'authentification à tierce partie de confiance. Un service *Kerberos*, résidant dans le réseau, agit comme un arbitre de confiance. Il est basé sur l'utilisation de la *cryptographie symétrique. Kerberos f*ournit un moyen pour vérifier les identités des entités dans un réseau ouvert. Le processus d'authentification commence par une requête d'authentification d'un client à un serveur d'authentification, celui-ci renvoie au client un ticket et une clé secrète appelé clé de session. Cette clé est partagée uniquement entre le client et le serveur. Elle peut être utilisée pour authentifier le client ainsi que le serveur. *Kerberos* constitue le standard des systèmes de distribution des clés symétriques (gardiens des clés) et est très utilisé dans Internet.

3.2.2.4 PGP (Pretty Good Privacy)

PGP [18], une alternative au PKI qui fournit une sécurité pratique pour protéger des communications à basse valeur, telles que des email. Le PGP est basé sur la certification de référence, qui permet à de multiples utilisateurs de « recommander » un certain utilisateur en signant des certificats de sa clé publique, cependant cette approche n'est pas parfaitement sécurisée car un utilisateur malhonnête peut établir de faux certificats.

3.2.2.5 La cryptographie à seuil

Cette technique permet de distribuer la confiance dans un système de gestion de clés. Un schéma de cryptographie à seuil (n, t) permet à n entités de partager la capacité d'effectuer une opération cryptographique (e.g, génération d'une signature digitale), tel que n'importe quel ensemble de t entités peuvent effectuer cette opération conjointement. Tandis qu'il est irréalisable de le faire avec au plus t-1 entités.

Cette technique peut être appliquée à un service de certification dans un réseau sans infrastructure comme les MANET. Dans ce cas, *n* nœuds partage la clé privée de la CA tel que *t* signature partielles sont nécessaires pour reconstruire la signature de la CA.

D'autres protocoles basés sur les clés publique ont été proposés pour les réseaux traditionnels tels que IKE [47] et JFK [48].

3.2.3. Les critères requis dans un système de gestion de clés pour un réseau ad hoc

Toute solution de gestion de clés efficace et sécurisée doit satisfaire les critères suivants :

a) Sécurité

L'authentification et la confidentialité sont un souci primaire pour s'assurer qu'aucun noeud non autorisé ne reçoit la clé qui peut plus tard être employée pour prouver le statut en tant que membre légitime du réseau. Aucun nœud ne devrait fournir des clés privées ou des certificats pour d'autres à moins que les autres aient été authentifiés. Un autre souci de sécurité est la gestion de confiance. Les relations de confiance doivent changer durant la vie du réseau. Le système doit être en mesure de détecter et d'exclure les éventuels nœuds compromis.

b) Robustesse

La robustesse est une condition nécessaire pour tout système de sécurité. Un système de gestion de clés robuste doit assurer une tolérance aux intrusions, cela signifie qu'un système de sécurité ne devrait pas succomber à un simple, ou à quelques noeuds compromis. Le système de gestion de clés devrait survivre en dépit des attaques de déni de service et des noeuds indisponibles. Les opérations de gestion de clés devraient pouvoir être accomplies en présence des noeuds compromis et des noeuds montrant le comportement malveillant, c'est-à-dire, noeuds qui dévient délibérément de leur protocole.

c) Passage à l'échelle

Les opérations de gestion de clé devraient finir d'une façon opportune en dépit d'un nombre variable de noeuds et de densités de noeuds. La fraction de la bande passante disponible occupée par le trafic de gestion de réseau devrait être maintenue aussi réduite que possible. N'importe quelle augmentation du trafic de gestion réduit la bande passante disponible pour des données de charge utile. Par conséquent, le passage à l'échelle des protocoles de gestion de clés est crucial.

d) Simplicité

La simplicité est un facteur crucial pour le succès d'un schéma de gestion de clés, en particulier, dans un réseau ad hoc avec des capacités limités des nœuds sans fil. Le service de gestion de clé idéal pour un réseau ad hoc doit être simple, formé d'une façon spontanée, ne distribue jamais des clés aux nœuds non autorisés, assure que le système de sécurité ne succombe pas sur un petit ensemble de nœuds compromis, facilement permet le renouvellement des clés.
Étant donné que ces conditions sont remplies, nous croyons que la simplicité est en premier lieu une question d'implémentation. Ainsi, l'implémentation d'un protocole de gestion de clé et les algorithmes sous jacents, doit prendre en considération la limite des nœuds et des ressources.

3.3. Les approches de gestion de clé dans les réseaux ad hoc

Plusieurs approches et schémas de gestion de clés dans les réseaux ad hoc ont été proposés dans la littérature. Toutefois ces schémas peuvent être classifiés de plusieurs façons, selon la technique et l'approche employée par le schéma pour la génération et la distribution des clés. Les solutions et les schémas de gestion de clés dans les réseaux ad hoc peuvent être classés en deux catégories : schémas symétriques et schémas à clés publiques (voir la figure 3.2).

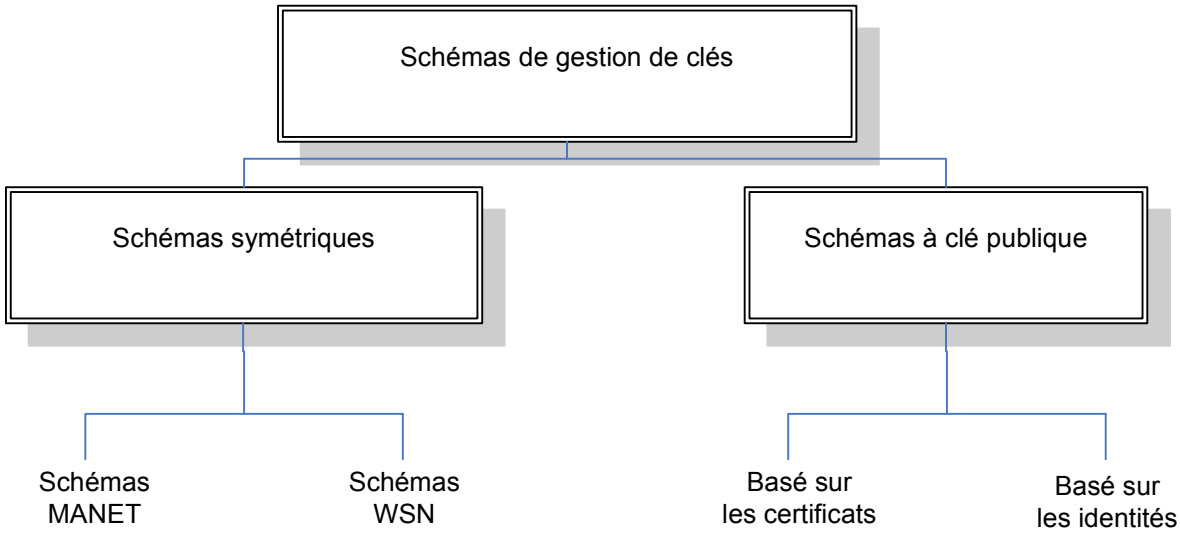


Fig.3.2. Classification des schémas de gestion de clés dans les réseaux ad hoc

La catégorie des schémas à clé publique est définie pour regrouper les schémas où la clé est originaire d'un seul nœud. Les nœuds peuvent bien coopérer

pondant la distribution de clés et n'importe quel nœud possède une paire de clés privée/publique, les schémas à clé publique sont basés sur les certificats traditionnels de clés publiques ou sur les identités. Tandis que, les schémas symétriques sont classés en schémas MANETs et schémas réseaux de capteurs (wireless sensor networks WSN), ce dernier est une nouvelle classe de réseaux ad hoc avec plus de contraintes sur les nœuds que les MANETs traditionnels, ce type de schémas ne sont pas abordés dans le cadre de notre travail. Dans le reste de cette section, nous discutons les différents schémas des deux catégories (symétriques et à clés publiques).

3.3.1. Les schémas symétriques

Nous présentons dans cette section les différents schémas de gestion de clés des réseaux ad hoc suggérées dans la littérature. Ces schémas sont récapitulés dans le tableau 1.

3.3.1.1 L'accord de clé (key agreement)

Dans les schémas de l'accord de clé (*Key agreement*), la clé résulte de la collaboration de plusieurs nœuds où chaque nœud apporte sa contribution à la clé finale. Ces schémas sont caractérisés par l'absence d'un tiers de confiance ou entité centralisée pour la génération et la distribution de clés cryptographiques, au lieu de ça, toutes les parties communicantes coopèrent pour établir une clé secrète commune. Le nombre de participants s'étend de deux parties (clé par paire) à plusieurs parties (établir une clé de groupe). Bien que ces approches ne sont pas nécessairement conçues avec les réseaux ad hoc, intuitivement ces approches semblent bien adaptées à la nature des réseaux ad hoc. Les recherches en matière de l'accord de clé dans les réseaux ad hoc se focalisent sur la manière d'établir une clé commune entre plusieurs participants qui ne se connaissent pas à priori. Les principaux schéma de l'accord de clés sont étudiés dans cette sous section.

Password Authenticated Key Agreement

Asokan *et al* proposent dans [26] un schéma de *Key agreement* pour le scénario suivant : un petit groupe de participants au cours d'une réunion ad hoc sont présents ensemble dans une salle de conférence et font confiance les uns aux autres. Les participants essaient de créer un réseau sans fil entre leurs ordinateurs portables et souhaitent assurer une communication sécurisée entre eux pondant la durée de la réunion, une façon simple consiste à distribuer un mot de passe faible inscrit par exemple sur un morceau de papier qui fait le tour de la salle, à partir de ce mot de passe les participants essaient de créer une clé symétrique utilisée pour sécuriser leurs échange. La difficulté de ce mode de fonctionnement est de trouver un canal sûr pour la distribution de la clé. Une autre manière d'établir une clé symétrique

peut se faire de manière distribuée où tous les nœuds du réseau ad hoc apportent leur contribution à la clé finale.

Les auteurs dans [26] présentent un protocole générique d'authentification EKE (Enkrypted Key Exchange) dans lequel deux nœuds A et B partagent un mot de passe et qui veulent établir une clé de session K dans un réseau ad hoc, de façon à ce que un adversaire ne puisse connaître ni la clé K ni le mot de passe faible de départ P. Le déroulement du protocole EKE est décrit la figure 3.3 où :

- (E_A, D_A) = une paire de clé publique et privée de A
- P = mot de passe faible partagée par A et B
- $Challenge_{A\ ou\ B}$ = valeur aléatoire générée par A, respectivement B
- $S_{A\ ou\ B}$ = valeur aléatoire générée par A, respectivement B
- R = une autre valeur aléatoire générée par B
- h = une fonction publique à sens unique

$$
\begin{array}{l}
(1)\ \mathrm{A} \rightarrow \mathrm{B} : \mathrm{A}, \mathrm{P}(E_A) \\
(2)\ \mathrm{B} \rightarrow \mathrm{A} : \mathrm{P}(E_A(\mathrm{R})) \\
(3)\ \mathrm{A} \rightarrow \mathrm{B} : \mathrm{R}(Challenge_A,\ S_A) \\
(4)\ \mathrm{B} \rightarrow \mathrm{A} : \mathrm{R}(h(Challenge_A),\ Challenge_B,\ S_B) \\
(5)\ \mathrm{A} \rightarrow \mathrm{B} : \mathrm{R}(h(Challenge_B))
\end{array}
$$

Fig.3.3. Le protocole EKE

Ensuite une extension du protocole est proposée dans [26] pour un groupe de participants (version multi-parties) comme présenté dans la figure 3.4, dans ce cas l'ensemble de participants est représenté par{*Mi, i = 1,…,n*} et *Mn* étant le leader. S_i est une valeur aléatoire générée par chaque participant *Mi* qui représente sa contribution à la clé de session finale K, ainsi cette clé est calculée de la manière suivante : *K=f(S1, S2,…,Sn)* où *f()* est une fonction à sens unique. Une contrainte dans ce protocole est qu'un leader doit déclancher les opérations d'authentification et d'échange de messages, ce-ci constitue un point de vulnérabilité du réseau ad hoc.

$$
\begin{array}{l}
(1)\ M_n \rightarrow \mathrm{ALL}\text{: } M_n,\ P(E) \\
(2)\ M_i \rightarrow M_n\text{: } M_i,\ P(E(R_i, S_i)),\ i = 1, \ldots n-1 \\
(3)\ M_n \rightarrow M_i\text{: } R_i(\{S_j, j = 1, \ldots, n\}),\ i = 1, \ldots, n-1 \\
(4)\ M_i \rightarrow M_n\text{: } M_i,\ K(S_i, H(S_1, S_2, \ldots, S_n)),\ \text{for some } i
\end{array}
$$

Fig.3.4. Le protocole EKE multi-parties [26]

Diffie-Hellman multi parties

Le protocole de Diffie-Hellman [27] tel qu'il est présenté permet l'échange de clé entre deux parties A et B, ce protocole est généralisé à de multiples participants pour être adapté pour assurer un secret fort partagé entre les nœuds d'un réseau ad hoc connaissant un mot de passe faible P, dans cette version de protocole il y a *n* participants (*M1, M2,…, Mn*), chaque participant *Mi* génère un secret *Si* et la clé résultante sera $K = g^{S1S2\ldots Sn}$. Pour *n* participants la mise en pratique du protocole est décrite comme suit :

(1) $M_i \rightarrow M_{i+1} : g^{S1S2\ldots Si}, i = 1\ldots n-2$, en séquence. Cette étape nécessite *n-2* envois de messages, à la fin M_{n-1} aura reçu *S1,S2…Sn-1* et pourra calculer $\pi = g^{S1S2\ldots Sn-1}$.
(2) $M_{n-1} \rightarrow Tous : \pi = g^{S1S2\ldots Sn-1}$, en diffusion
(3) $M_i \rightarrow M_n : P(Ci), i = 1\ldots n-1$, en parallèle, avec $Ci = \pi^{\hat{S}i/Si}$ et Ŝi est un facteur d'aveuglement choisi aléatoirement par M_i. Chaque M_i enlève de π sa contribution *Si* et ajoute un facteur d'aveuglement *Ŝi*
(4) $M_n \rightarrow M_i : (Ci)^{Sn}, i = 1,\ldots,n-1$, en parallèle
M_n décrypte (Ci), lui ajoute son *Sn* et l'envoie au M_i correspondant. A ce stage tous les participants pourront calculer $K = g^{S1S2\ldots Sn}$
(5) $M_i \rightarrow tous : M_i, K(M_i, H(M_1,M_2,\ldots,M_n))$, pour *i* donnée en diffusion, ce message est envoyé pour simple vérification, $H(M_1, M_2,\ldots, M_n)$ permet aux participants de vérifier s'ils ont la même vue de la composition du groupe.

Cette application de Diffie-Hellman à de multiples participants a pour inconvénient de nécessiter d'établir au préalable une relation d'ordre entre les nœuds du réseau. De plus, il faut que le pénultième et l'antépénultième éléments aient la possibilité de communiquer avec tous les autres nœuds, ce qui est une contrainte beaucoup trop forte pour les réseaux ad hoc.

Diffie-Hellman dans un hyper cube

Une autre possibilité est étudiée dans [26], en arrangeant les participants dans un hyper cube comme le présente la figure 3.5, avec quatre participants A, B, C, D voulant se mettre d'accord sur une clé de session. Chaque participant a une adresse de 2 bits et génère une contribution Si, dans une première étape, les nœuds A et B exécutent Diffie-Hellman pour deux participants, ils parviennent ainsi à calculer $S_{AB} = g^{SASB}$; en même temps, C et D calculent $S_{CD} = g^{SCSD}$.

La deuxième étape consiste à exécuter Diffie-Hellman entre A et C et B et D, tout en utilisant comme contributions les clés calculées à l'étape 1. Ainsi, à la fin de la deuxième étape les quatre participants détiennent la même clé de session $S_{ABCD} = g^{SABSCD}$. Si le nombre de participants est évalué à $n = 2^d$ participants, chaque participant se voit attribué un sommet dans un hyper cube de dimension *d*. le

protocole procède en d étapes d'échange de clés avec le même principe présenté précédemment. Après les d étapes, tous les participants auront la même clé de session.

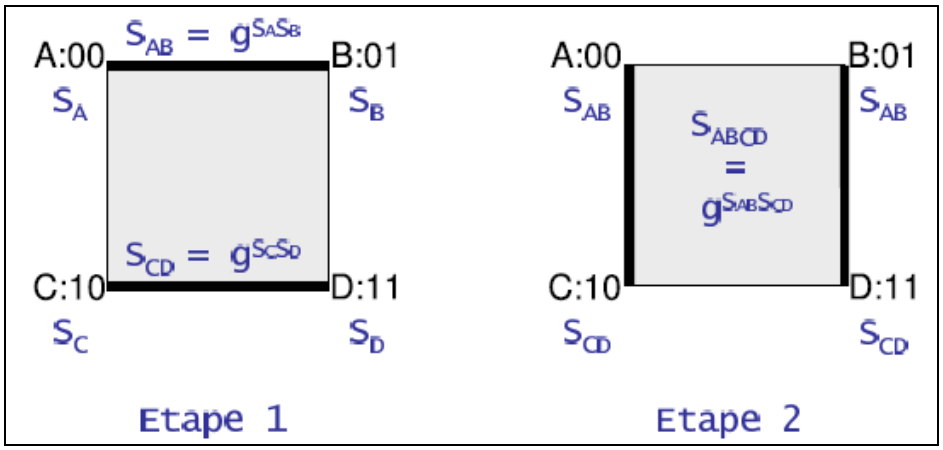


Fig.3.5. Protocole de Diffie-Hellman dans un hyper cube

3.3.1.2 Prédistribution aléatoire de clés

Le schéma basique de prédistribution de clés a été introduit dans [43] ce schéma qui est basé sur une approche probabiliste est constitué de trois phases : prédistribution de clés, découverte de clés partagées et établissement du chemin de la clé. Dans ce schéma un tiers de confiance ou un serveur est indispensable pour la distribution préalable de clés. Dans la première phase avant d'initialiser le réseau, le serveur de prédistribution de clé génère un groupe de clés symétriques. Ensuite il en sélectionne de façon déterministe un ensemble de clés et le délivre à chaque nœud qui veut participer au réseau ad hoc. La sélection des ensembles de clés est déterministe et basé sur l'identifiant du nœud, ce qui permet à chaque nœud de déterminer les clés communes avec les autres nœuds du réseau. Les ensembles de clés sont affectés aux nœuds de telle façon que chaque deux nœuds du réseau peuvent trouver au moins une clé secrète commune dans leurs ensembles de clés. Dans la deuxième phase, après que le réseau est initialisé, chaque nœud découvre les nœuds adjacents et détermine les clés qu'il partage avec ces nœuds, ainsi, chaque paire de nœuds voisins peut utiliser n'importe quelle clé commune pour initialiser une communication sécurisée entre eux. Il est possible qu'il n'existe pas une clé commune entre quelques paires de nœuds, dans ce cas, dans la troisième phase, si un chemin de nœuds qui partagent des clés deux à deux, existent entre une paire de nœud n'ayant pas une clé commune, alors cette paire de nœud peut utiliser ce chemin pour échanger une clé dans le but d'établir un lien direct. Un autre schéma similaire est proposé dans [44], il garantie q ($q \leq 1$) clés commune entre chaque paire de nœuds, ainsi les entité communicantes peuvent choisir aléatoirement une de ces clés ou plusieurs à la fois. Par conséquent, il est très difficile pour des attaquants à

compromettre les communications puisque il n'est pas facile pour eux de connaître les bonnes clés utilisées dans les communications.

3.3.1.3 Le protocole SKiMPy

Puzar *et al* [54] présente un protocole de gestion de clé basé sur la cryptographie symétrique conçu pour les réseaux ad hoc hautement dynamique. A l'initialisation du réseau ad hoc, tous les nœuds génèrent aléatoirement une clé symétrique et un nombre *ID* aléatoire, ensuite, chaque nœud envoie sa clé à ses voisins à travers des messages HELLO. La tache du protocole *SKiMPy* est de garantir que tous les nœuds s'accordent sur une clé parmi l'ensemble généré et qui sera utilisée comme la clé symétrique de tout le réseau. Pour sélectionner la clé symétrique ce protocole introduit les relations ">" "<" entre les clés générées par les nœuds. La meilleure clé (i.e. celle avec le nombre *ID* le plus bas ou le *timestamp* le plus recent) sera choisie comme la clé de groupe locale. La meilleure clé est transférée avec les mauvaises clés aux nœuds du réseau via un canal sécurisé établi à l'aide des certificats prédistribués. La procédure est répétée jusqu'à ce que la meilleure clé soit partagée par tous les nœuds du réseau ad hoc (voir figure 3.6). *SKiMPy* propose de renouveler périodiquement la clé symétrique afin de se protéger contre la révélation de la clé en utilisant la cryptanalyse, les nouvelles clés sont dérivées des clés initiales.

SKiMPy est efficace en terme de préservation de bande passante dans le sens où les nœuds s'accordent localement sur la clé symétrique puisque les informations des clés sont échangées uniquement entre les voisins.

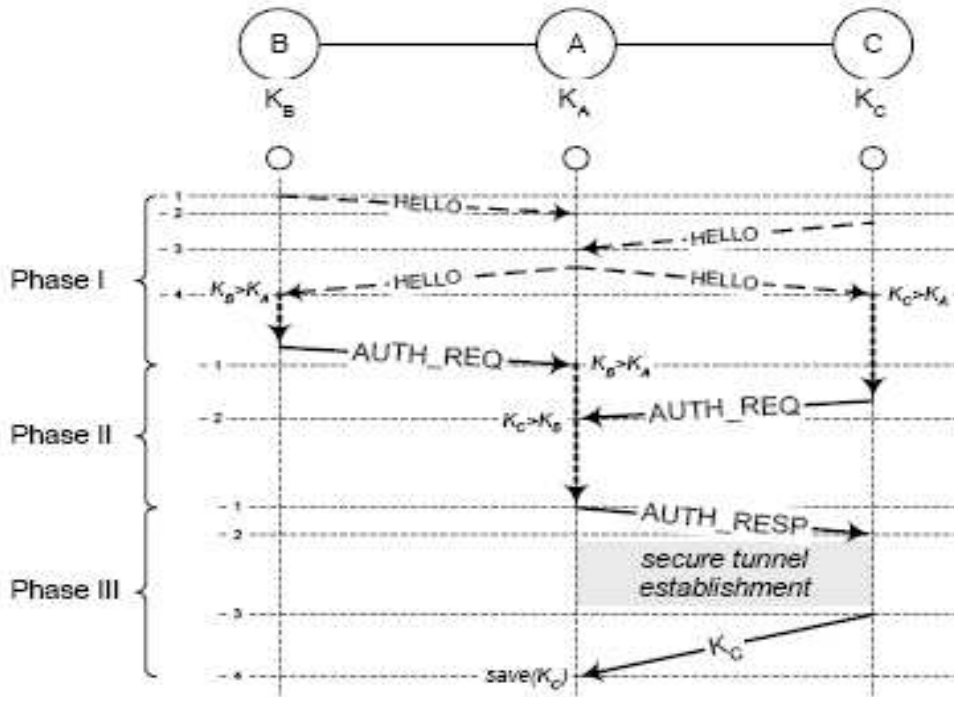


Fig.3.6. Le protocole SKiMPy

Mécanismes	***key agreement***	***Prédistribution aléatoire***	*SKiMPy*
Serveurs	Non	Oui	Non
Principe	Coopération pour générer une clé secrète commune	*Chaque paire de nœuds partage* q ($q \leq 1$) clés commune	Sélectionner la meilleure clé
Avantages	- pas besoin d'un tiers de confiance	- robuste - résiste contre la compromission	- Préserve la bande passante
Inconvénients	- Problème de distribution des clés	- approche probabiliste	- révocation on-line n'est pas possible

Tab.1. Mécanismes de sécurité dans les réseaux ad hoc basés sur la cryptographie symétrique

3.3.2. Schémas à clés publiques

La cryptographie asymétrique semble être le modèle efficace pour réaliser l'authentification sous beaucoup d'aspects. Néanmoins, appliquer ce modèle à un réseau décentralisé et auto organisé tels que les réseaux ad hoc, pose plusieurs questions et difficultés. En effet, la plupart de ces modèles supposent l'utilisation des certificats pour lier une clé publique à son propriétaire, cela nécessite le déploiement d'une infrastructure à clé publique (PKI) qui inclut la création d'un système de distribution de clé, lequel est souvent incarné sous la forme d'une autorité de certification centralisée (CA). Cependant, il est problématique d'établir un service de gestion de clés (*Key Management*) utilisant une seule CA dans les réseaux ad hoc. En effet, la dépendance d'une seule CA pour sécuriser le réseau tout entier présente un point de vulnérabilité ; si la CA est indisponible, les nœuds ne peuvent plus obtenir les clés publiques correctes des autres pour s'authentifier, et ainsi ils ne peuvent plus assurer une communication sécurisée entre eux, si la CA est compromise, elle peut révéler sa clé privée à un adversaire lequel s'en servit pour signer de faux certificats. En conséquence, l'absence d'infrastructure et une autorité centrale dans les réseaux Ad Hoc empêchent l'utilisation d'une PKI traditionnelle et une autorité de certification (CA) pour l'authentification des noeuds. Pour remédier à ce problème, plusieurs solutions sont envisagées, l'une consiste à la distribution des fonctionnalités de la CA (génération de certificats, révocation, renouvellent) aux noeuds du réseau en se basant sur la technique de la cryptographie à seuil, l'autre utilise une architecture de confiance similaire à PGP, et d'autres proposent de remplacer les certificats par une fonction de hachage. Ces approches sont présentées

dans les sous sections suivantes. Les mécanismes basés sur la cryptographie asymétrique sont résumés dans le tableau 3.

3.3.2.1 La technique de Cryptographie à seuil (threshold cryptography)

Pour palier au problème de l'utilisation d'une PKI dans un réseau ad hoc, cette technique utilise les mécanismes de cryptographie à seuil "threshold cryptography" [11][12][13] pour partager un secret entre les membres du réseau et pour distribuer la tâche de signer un certificat à plusieurs nœuds. Plusieurs travaux employant la cryptographie à seuil sont proposés. Le tableau 2 résume les solutions à base de cryptographie à seuil présentées dans cette section.

Autorité de Certification partiellement distribuée

Zhou et Haas proposent dans [1] un service distribué de gestion de clé publique pour les réseaux ad hoc, qui s'appuie sur un schéma de cryptographie à seuil. Le service dans son ensemble possède une paire de clé publique / privée (K / k), qui est utilisée pour vérifier / signer les certificats pour les clés publiques des nœuds du réseau. Il est supposé que tous les nœuds connaissent la clé publique K du service et font confiance à tous les certificats signés par la clé privée correspondante k, cette clé n'est connue par aucun nœud du réseau. Les nœuds clients peuvent envoyer des requêtes pour récupérer les clés publiques des autres nœuds clients ou pour demander une mise à jour de leurs propres clés publiques.

Le service de *Key Management* proposé avec la configuration $(n, t + 1)$ pour $(n \geq 3t + 1)$ est constitué de n nœuds spéciaux appelés serveurs, présents dans le réseau ad hoc, chaque serveur possède également sa propre paire de clé publique / privée et enregistre la clé publique de tous les nœuds du réseau y compris celles des autres serveurs.

Le schéma $(n, t + 1)$ de cryptographie à seuil permet à n parties de partager la capacité d'effectuer une opération cryptographique (e.g. générer une signature digitale). Dans ce cas les n serveurs du service de Key Management partage la capacité de signer les certificats, la clé privée de tout le système est divisée en n secrets partagés $(s1, s2, \ldots, sn)$, comme c'est illustré dans la figure 3.7. Pour créer un certificat, chaque serveur génère une signature partielle pour le certificat et l'envoie au combinateur, qui doit rassembler au moins $t + 1$ signatures partielles correctes pour pouvoir générer la signature valide pour le certificat.

L'application de la cryptographie à seuil garantit que le système peut tolérer un certain nombre $t < n$ de serveurs compromis dans le sens que $t + 1$ de signatures partielles correctes sont nécessaires pour générer une signature valide, avec au plus de t serveurs compromis le combinateur peut encore générer des certificats valides. Le combinateur peut vérifier la validité de la signature partielle des serveurs, si l'un

d'eux envoie une signature partielle incorrecte, le combinateur la rejette et continue à rassembler les $t + 1$ signatures. La figure 3.8 illustre comment les serveurs génèrent un certificat avec la configuration (3, 2), le serveur 2 est compromis et n'a pas pu envoyer sa signature partielle, et le combinateur est capable de générer un certificat valide.

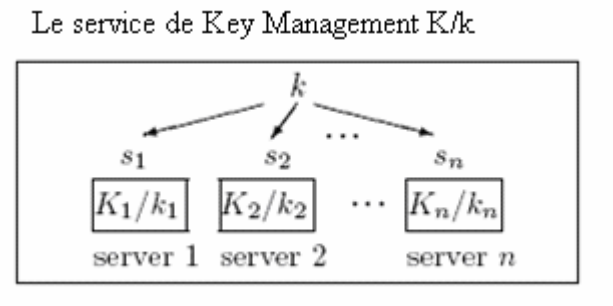


Fig.3.7. La configuration du service de *Key Management* [1]

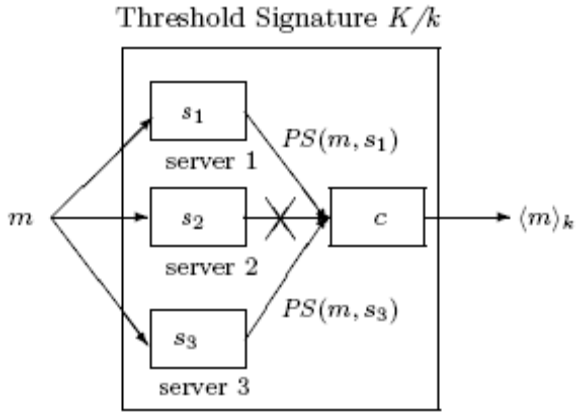


Fig.3.8. Cryptographie à seuil [1]

Cette méthode d'authentification repose donc sur trois éléments avec un réseau de N nœuds :

1. $n < N$, nœuds sont chargés pour jouer le rôle d'autorité de certification CA.
2. l'authentification est basée sur un ensemble de certificats partiels.
3. $t + 1$ certificats partiels sont nécessaires, t étant le seuil de stabilité de l'authentification.

Pour assurer une sécurité proactive de ce service en protégeant le réseau contre un attaquant qui puisse avec le temps compromettre plus de t nœuds l'un après l'autre parmi n nœuds qui partagent les signatures partielles, un système de rafraîchissement permet aux nœuds de renouveler leurs secrets qui sont utilisés pour générer des certificats, et de s'adapter aux changements dans le réseaux.

Cette technique [1] a permis de remédier au problème du déploiement d'une PKI ou une CA pour la gestion des certificats dans les réseaux ad hoc, à cause de l'absence d'infrastructure et une autorité centralisée. Néanmoins, il y a quelques

problèmes avec cette proposition ; les premiers t + 1 noeuds serveurs doivent être initialisés par un tiers de confiance, et cette approche est interactive, cela signifie que les serveurs peuvent ne pas être dans la même porté de transmission que les autres nœuds (beaucoup de messages échangés entre nœuds non voisins). Outre cet inconvénient, il y a deux autres problèmes, premièrement, le nombre t doit être un compromis entre la disponibilité et la robustesse, mais il n'est pas claire comment la valeur de t peut être changée conformément avec le passage à l'échelle du réseau vue de leur topologie dynamique, deuxièmement comme noté dans [7], ce système parait vulnérable à l'attaque Sybil [9] : un attaquant peut prendre autant d'identités que nécessaire pour rassembler assez de secrets partagés et reconstruire la clé privée du système. Le combinateur indispensable pour la reconstruction de la signature, peut à son tour être attaqué et par conséquent devenir un point de vulnérabilité pour le système de sécurité, pour y remédier, Legrand *et al* [6] proposent ainsi une réplication du combinateur.

Autorité de Certification complètement distribuée

Kong *et al* dans [2] et Luo *et al* dans [3][4], proposent une approche similaire à [1] qui consiste à distribuer les fonctions d'une autorité de certification CA (génération de certificats, renouvellement, révocation) sur tout l'ensemble des nœuds du réseau, à la différence de [1] cette approche est complètement distribuée dans le sens que tous les nœuds du réseau partagent le rôle de la CA ($n = N$) ce qui permet de mieux augmenter la disponibilité, n'importe quel ensemble de t + 1 situés sur la même porté radio d'un nœud qui demande un certificat, peut lui générer le certificat demandé. De plus, dans la phase d'initialisation du système, il suffit d'initialiser t + 1 nœuds avec les partages associées, les autres nœuds peuvent être initialisés par les autres nœuds qui sont déjà initialisés, ainsi si un nouveau nœud qui rejoint le réseau et qui n'a pas une part de clé doit contacter au moins t + 1 nœuds parmi l'ensemble initialisé, de cette manière le réseau est progressivement auto initialisé. Cette approche n'utilise pas de combinateur offrant ainsi un meilleur niveau de sécurité, Puisque tous les nœuds participent à la génération des certificats, ils doivent donc tous être protégés contre la compromission. Cependant, la fiabilité de cette technique dépend sur l'hypothèse que chaque nœud devrait avoir au moins $t+1$ nœuds voisins avec des parts de clés valides. Ici, le seuil t est un paramètre critique qui doit être déterminé soigneusement. Si t est plus grand, certains nœuds doivent se délacer pour ramasser les $t+1$ parts de clé. De plus, dans le cas des réseaux à large échelle, la sécurité de ce schéma est diminuée notament quand les nœuds ne sont pas physiquement protégés, car la probabilité que la clé privée de la CA est révélée augmente avec le nombre de nœud qui partage la clé privée de la CA.

Autonomous Key Management (AKM)

Une autre technique a été proposée par Zhu *et al* [49] appelée (AKM) qui fournit un service de CA complètement distribué et auto organisé en se basant sur la cryptographie à seuil. Pour un petit nombre de nœuds le schéma est similaire au précèdent [2], lorsque le nombre de nœuds s'augmente, le schéma introduit une hiérarchisé de secrets partagés clés (clés partielles), pour s'adapter aux MANET avec un grand nombre de nœuds. Ce schéma permet d'établir des certificats avec différents niveaux d'assurance.

La clé privée de la CA est initialement partagée par un groupe de nœuds voisins, ensuite, chaque nœud des N voisins choisit une valeur secrète S_i, et distribue les parts partiels de cette valeur secrète aux autres voisins. La somme des valeurs secrètes individuelles $S = (S1 + S2 + S3 + \ldots + Sn)$ représente la clé privée de la CA et la clé publique correspondante est g^S. Supposant que les nœuds publient leurs valeurs publique, la clé publique de la CA peut être alors dérivée sans révéler sa clé privée, en multipliant ces valeurs $g^S = g^S1 * g^S2 * \ldots * g^Sn$.

Les nœuds N1…N6 et leurs parts secrètes (clés partielles) $f(Ni)$, peuvent être vu comme des niveaux d'une structure arborescente. La probabilité de la compromission augmente avec l'accroissement du nombre des nœuds possédants les clés partielles de la clé privée de la CA. Ainsi, lorsque le nombre de possesseurs de clés partielles atteint un certain niveau, les nœuds sont divisés en petits groupes pour installer une nouvelle valeur secrète de ce groupe. Les certificats signés par un groupe situé au bas niveau de l'arborescente a moins d'assurance que ceux signés par un groupe situé à haut niveau.

Avant de se diviser, les nœuds N1…N6 possèdent les clés partielles $f(N1)…f(N6)$ de la clé privée de la CA, supposant que N1,N2,N3 décident de former un nouveau groupe et N4,N5,N6 un autre, N1 distribue une part de la valeur secrète $f(N1)$ aux autres nœuds dans le même groupe, les autres font la même chose, la nouvelle valeur privée qui sera partagée par les nœuds N1, N2, N3 est la somme de leurs parts $S' = f(N1) + f(N2) + f(N3)$. Lorsque le nombre de nœuds appartenant à la même région (groupe) atteint le seuil spécifié, ce groupe sera encore divisé. Cette technique augmente la robustesse et la tolérance aux intrusions au prix du coût des communications, les nœuds sont supposés être capable de quitter un groupe et appartenir à un autre lorsque il déplace d'une région à l'autre dans le réseau. Implicitement, les nœuds doivent maintenir une vue sur l'hiérarchie de clé et être capables de détecter les frontières de la nouvelle région.

Schéma de clés à seuil dynamique

Raghani *et al* [74] proposent un schéma de CA distribué qui suit le même principe que [2] où un nœud obtient son certificat en communicant avec ses voisins de simple saut. Avec une telle approche, quand le degré d'un noeud dans le réseau diminue au-dessous du seuil, il y a une augmentation substantielle de la latence du service de certification. Pour résoudre ce problème, le schéma proposé offre un support dynamique pour une CA distribuée en lui permettant de modifier dynamiquement la valeur du seuil en cas de besoin ce qui permet de réduire la latence du service de certification. Par conséquent, les auteurs proposent une série de protocoles de surveillance de réseau pour identifier certaines situations où le degré des noeuds change, et donc ajuster dynamiquement la valeur de seuil. Une autre solution similaire est présentée dans [75]. Ce schéma permet facilement et efficacement l'augmentation dynamique du seuil en fonction du niveau de sécurité requis et la disponibilité nécessaire des serveurs.

Self Adjusted Security Architecture

Robin Doss *et al* [50] proposent une nouvelle architecture de sécurité pour les réseaux ad hoc qui combinent les techniques de *cryptographie à seuil* et le *clustering*, l'architecture distribuée d'authentification est proposée pour bien s'adapter à un large réseau avec une topologie dynamique qui change fréquemment, et pour améliorer le processus de distribution de clés publiques aux nœuds du réseau. Dans cette architecture, le réseau total est divisé en groupes géographiquement adjacents appelés *Clusters*. Dans un cluster, un nœud peut jouer différents rôles. Dans chaque cluster on distingue les nœuds suivants :

- *ClusterHead (CH)* : ce nœud est élu pour servir comme un coordinateur locale pour son cluster, il est responsable d'établir et organiser le cluster.
- nœud passerelle ou Getway (GW) : c'est un nœud qui n'est pas nécessairement un CH et qui gère les communication entre Clusters adjacents.
- Cluster membre (CM) : ce sont les autres nœuds légitimes du cluster.

Après l'étape d'initialisation du réseau, le CH diffuse périodiquement dans son cluster des balises contenant des informations d'administration (comme la liste des nœuds CM et GW dans le même cluster). Les nœuds GW transmettent également des balises pour informer leurs clusters à propos des clusters adjacents. Cette architecture se base sur la cryptographie à clé publique pour assurer l'authentification, l'intégrité et la confidentialité. Chaque nœud possède une paire de clés publique / privée, et les clés publiques sont établies et distribuées par un service de CA distribué qui est formé par les CH. Ce concept permet de tirer profit des

avantages de la cryptographie à seuil pour augmenter la disponibilité et la tolérance aux intrusions. Les CH forment un réseau virtuel appelé le réseau de clé, et la clé privée de la CA est partagée entre les CH, en effet, chaque CH détient une partie de la clé privée de la CA. Les membres appartenant au même cluster partage une clé secrète symétrique qui est générée par leur CH. Cela est très utile pour le trafic interne du cluster et pour cacher certaines informations à ceux n'appartenant pas à leur cluster.

La composition du réseau de clé formé par les CH change au fur et à mesure que les CH quittent et rejoignent les clusters. Les clés partielles détenues par les CH doivent aussi être renouvelées pour s'adapter au nombre des CH, cela permet de se protéger contre un attaquant mobile qui puisse compromettre (k) nœud avec le temps.

Pour rejoindre le réseau, un nouveau nœud A doit d'abord trouver un cluster. Afin de s'authentifier, le nouveau nœud ne peut demander la signature du CH que s'il possède le nombre de certificats de garantie exigés par le CH, un certificat de garantie (WarrantCerts) est généré et signé par un membre légal du réseau S, ce nœud est désigné par le CH et considéré comme nœud de confiance, et on va l'appeler Warrant. Le certificat de garantie permet de garantir l'authenticité du nouveau nœud A :

WarrantCert (*A*):= Node (*A*), PubKey (*A*), Validity (*t*), Sign (*S*)

Le nouveau nœud A doit aussi avoir les certificats d'autorisation des nœuds Warrant signés par le CH, ces certificats permettent au CH de vérifier que les nœuds Warrant qui ont signé des WarrantCerts au noeud A, sont autorisés par le CH :

WarrantAutCert(S):= Node(S), PubKey(S), Sign (CHNetwork)

Lorsque le nouveau noeud obtient le nombre de certificats de garanties nécessaire ainsi que les certificats d'autorisation correspondants, il peut les présenter au CH. Le CH vérifie la validité de ces certificats, et pour signer le certificat de la clé publique de A, il collecte les clés partielles des autres CH et génère le certificat demandé à A :

IdCert (A):= Node (A), PubKey (A), Validity (t), Sign (CH-Network)

Avoir sa clé signé, A devient un membre légitime du cluster, le CH lui envoie ensuite la clé symétrique du cluster pour la communication intra cluster (voir figure 3.9). La technique des clusters est importante dans un réseau à large échelle et hautement dynamique. D'autres solutions similaires basées sur le clustering sont proposées dans [51][52][53]

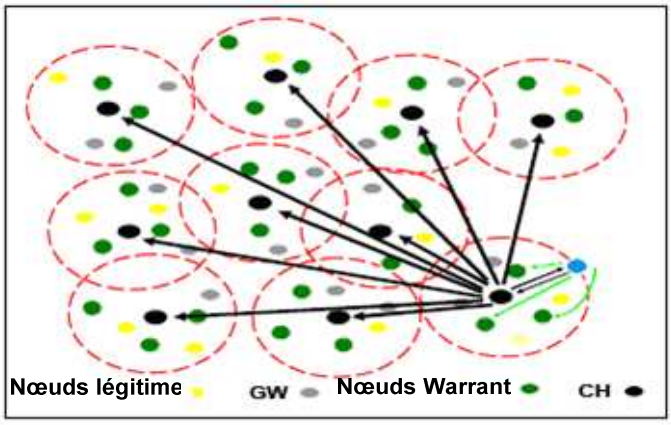


Fig.3.9. Gestion de clés basée sur la cryptographie à seuil et le clustering

Proposition	Distribution	Spécification	Interactive	Renouvellement et rafraîchissement	Révocation	Initialisation Des serveurs
CA P.Distribuée	Partiellement	$n \geq 3t + 1$ $N > n$	Oui	Oui	Oui	Autorité de confiance
CA C.Distribuée	complètement	$N = n$	Non	Oui	Oui	Semi Auto-initialisation
AKM	complètement	$N=n$	Non	Oui	Oui	Semi Auto-initialisation
Seuil dynamique	complètement	$N=n$ *t dynamique*	Non	Oui	Oui	Un tiers de confiance
Self Adjusted S.A	Partiellement	*Basé sur Clusters*	Oui	Oui	Oui	Un tiers de confiance

Tab.2. Solutions à base de cryptographie à seuil

3.3.2.2 L'infrastructure à clé publique auto-organisée (Self-organized PKI)

Une nouvelle approche pour la distribution de clés publiques a été développée par Hubaux *et al* [5], les auteurs proposent une PKI auto-organisée et auto-gérée convenable pour les réseaux ad hoc complètement auto-organisés (i.e. pas d'infrastructure, pas d'autorité centrale, pas de tiers de confiance, pas de serveur central), cette technique est similaire à PGP (pour Pretty Good Privacy) [18] dans le sens que les utilisateurs (nœuds participants au réseau) établissent les certificats de clés publiques les uns pour les autres en se basant sur leurs connaissances personnelles (chaque nœud établit des certificats aux nœuds en qui il a confiance) et sans l'aide d'une CA. Dans PGP la distribution de clés se base sur des répertoires de certificats publiquement accessibles stockés sur des serveurs centraux, ce qui n'est

pas bien appropriés pour les réseaux ad hoc, en opposition, la proposition de Hubaux *et al* n'est pas basée sur ce principe, les certificats de clés publiques sont stockés et distribués par les nœuds utilisateurs eux-mêmes.

Chaque nœud maintient son propre entrepôt local (*local repository*) qui contient un nombre limité de certificats de clés publiques. Il se base sur une notion de confiance transitive : si A croit B et B croit C, alors A croit C (mais C ne croit pas forcément A, la notion n'étant pas symétrique). Cette technique permet de former des chaînes de certifications, pour avoir au final un graphe de certificats. Ainsi, lorsque deux nœuds du réseau veulent vérifier les clés publiques l'un de l'autre, ils échangent leurs entrepôts de certificats et essaient de trouver une chaîne de certificats entre eux dans les listes de certificats échangées. L'efficacité de cette approche dépend du mécanisme de construction des entrepôts locaux de certificats, et des caractéristiques du graphe de confiance. Ce graphe de confiance est un graphe orienté *G (V, E)* qui permet de représenter les relations de confiance entre les nœuds du réseau, V est l'ensemble des sommets qui représente les nœuds (utilisateurs) et E est l'ensemble des arrêtes qui représentent les certificats de clés publiques, plus précisément, il y a une arrête du sommet *u* vers le sommet *v*, si le nœud *u* a généré un certificat pour le nœud *v*. le processus d'établissement d'une chaîne de confiance entre deux nœuds *u* et *v* est schématisé dans la figure 3.10. Une chaîne de confiance entre deux nœuds *u* et *v* est représentée par un chemin du sommet *u* vers le sommet *v*, les arcs en rouge représentent la liste des certificats dans l'entrepôt local de *v*, celle de *u* est représentée par les arcs en vert, ainsi en combinant les deux entrepôts on obtient alors la chaîne de confiance entre *u* et *v*.

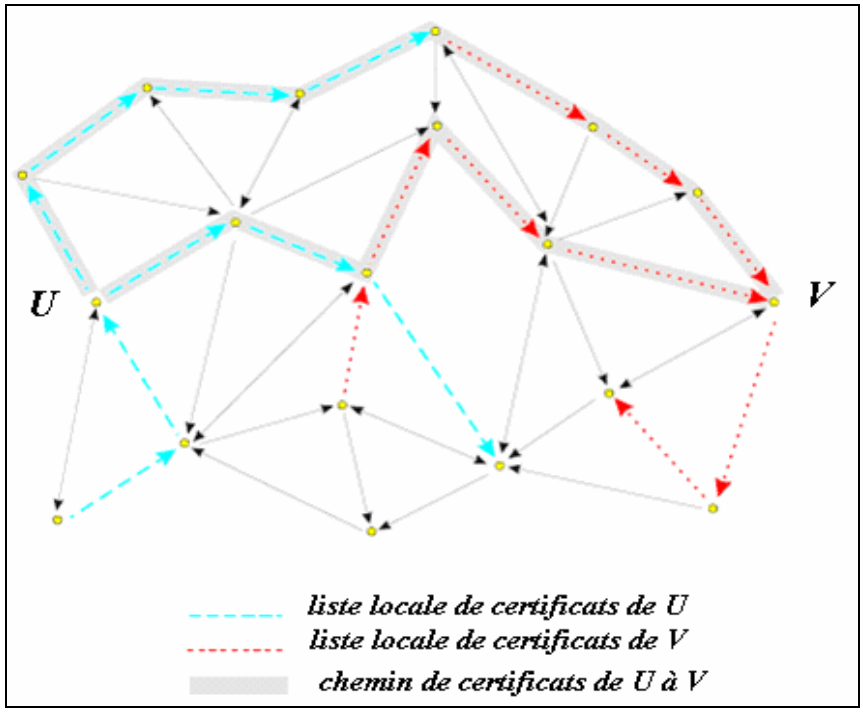


Fig.3.10. Graphe de confiance entre les nœuds dans un réseau ad hoc [5]

Dans ce même article les auteurs proposent certains algorithmes de construction des entrepôts locaux de certificats et étudient leurs performances. Les algorithmes proposés prennent en considération les caractéristiques des graphes de confiance dans le sens que le choix des certificats stockés par chaque nœud dépend de la connectivité du nœud et ses voisins dans le graphe de confiance, plus précisément chaque utilisateur stocke dans son entrepôt local plusieurs chemins de certificats directs et mutuellement disjoints, chaque chemin commence par l'utilisateur lui-même, et chaque certificat sur le chemin est choisi parmi l'ensemble des certificats qui sont connectés au dernier utilisateur sélectionné sur le chemin de telle sorte que le certificat choisi conduit à un utilisateur qui a le plus grand nombre de certificat connectés à lui, cette algorithme est appelé *Maximum degree Algorithm*, le critère de la construction de l'entrepôt local est le degré des sommets dans le graphe de confiance. Dans le deuxième algorithme, les certificats sont sélectionnés dans les entrepôts locaux en se basant sur le nombre des certificats de raccourci (*shortcut certificates*) connectés aux utilisateurs, ici un certificat de raccourci est défini comme étant un certificat que lorsque il est supprimé du graphe, le plus court chemin entre les deux utilisateurs connectés précédemment par ce certificat devient strictement plus grand que deux. Cet algorithme est appelé *Shortcut Hunter Algoritm*. L'analyse de performance de ces algorithmes montre que même un algorithme de construction simple peut atteindre une haute performance dans le sens que tout utilisateur u peut trouver au moins une chaîne de certificats à un autre utilisateur v dans leurs entrepôts échangés avec une grande probabilité même si la taille de l'entrepôt local est plus petite (dans l'ordre de $\sqrt{n}$) en comparant au nombre total n des utilisateurs dans le réseau. Les deux algorithmes sont simulés sur le graphe de confiance PGP, les résultats montrent que même avec un entrepôt de certificats inférieur à $\sqrt{n}$, les deux utilisateurs ont une grande chance (90%) de trouver une chaîne de confiance entre eux.

Néanmoins, cette technique d'authentification reste à garantie probabiliste, du moment que l'existence d'un chemin de certificats entre deux nœuds voulant communiquer n'est pas toujours assurée. De plus le stockage distribué de certificats engendre un surcoût important, qui met en question l'applicabilité à large échelle de l'approche. En outre, des membres malicieux peuvent générer de faux certificats et les intégrer dans le graphe de confiance, pour y remédier, les auteurs dans [5] proposent l'utilisation de métriques d'authentification pour évaluer l'authenticité d'un certificat et la chaîne de confiance à laquelle il appartient.

Cette approche est plus appropriée aux petits réseaux ad hoc, puisque la convergence de longues chaînes de certificats peut être considérable. De plus, si un certificat est révoqué, toutes les chaînes qui le contiennent deviennent invalides et de nouveaux calculs doivent être effectués. Il faut aussi noter que initialement nous avons besoin d'une certaine confiance entre les nœuds pour établir des chaînes de certificats.

Jusqu'à maintenant on a présenté des solutions qui sont basées sur l'établissement des certificats digitales pour faire correspondre une clé publique à l'identité (ID) de son propriétaire. Des solutions alternatives existent, qui font le lien entre un ID et la clé publique associée sans l'aide ou l'utilisation d'aucun certificat digitale. Ces solutions sont proposées dans les sous section suivantes.

3.3.2.3 ID-Based Cryptography (IBC)

Le concept de *ID-based crypto*graphy (IBC) [14] [15] est une forme de la cryptographie à clé publique dans laquelle la clé publique de tout participant est directement dérivée de son identifiant, de cette manière on a pas besoin d'une CA pour la distribution de clés publiques.

Le but de cette technique est d'éliminer les certificats proposés dans les méthodes précédentes, afin d'alléger la surcharge globale. La suppression des certificats se base sur le fait que chaque nœud possède un identifiant, par exemple l'adresse MAC ou IP. En utilisant une technique de hachage (pour avoir des clés publiques similaires), on peut obtenir une clé publique, donc trouvable par tous. Le premier schéma basé sur IBC a été proposé par Shamir [16]. Le mécanisme IBC permet non seulement de fournir un service d'authentification mais également de préserver la bande passante et réduire la charge de calcul des nœuds participants.

Dans un système IBC chaque nœud obtient sa clé privée en consultant le serveur PKG (*Private Key Generator*) via un canal sécurisé. À noter que la sécurité repose sur le fait que seul le PKG peut faire la conversion public vers privé. Pour améliorer la fiabilité de la technique et être aussi plus résistant aux attaques du type DoS. Le PKG peut également être distribué en combinant les deux concepts IBC et la cryptographie à seuil comme c'est proposé dans [17][8][19][22].

3.3.2.4. Cryptography-Based Address (CBA)

Cette approche se distingue de celle de IBC présentée précédemment par le fait qu'au lieu de générer sa paire de clés publique/privée en se basant sur l'ID, un nœud génère une paire de clés publique / privée, ensuite il calcule son ID qui peut être aussi l'adresse dans ce cas, en faisant un hachage sur la clé publique sans avoir besoin d'un serveur ni d'une CA.

Pour être résistant aux attaques de type spoofing et pour être statistiquement unique, la longueur de l'ID (adresse) calculé par la fonction de hachage ne doit pas être trop petite. En effet, avec un algorithme de hachage *l*-bit, un attaquant a besoin en moyenne 2^{l-1} opérations de hachage afin de pouvoir découvrir la clé publique correspondante, et en moyenne $2^{l/2}$ nœuds peuvent créer une collision d'adresse [23]. Par conséquent, les adresses IPv4 ne sont pas adaptées avec ce schéma.

En se basant sur le principe de CBA, Montenegro *et al* [24] propose un mécanisme appelé SUCV (Statistically Unique and Cryptographically Verifiable) dans lequel les ID sont statistiquement unique et cryptographiquement vérifiables, il

y a une très petite probabilité qu'il y a deux nœuds ayant le même ID. L'ID noté CBID (carypto based identifier) est généré comme suit:

$$CBID = hmac\_sha1\_128(sha1(imprint), sha1(PK))$$

Où :

- *PK* est la clé publique
- *imprint* est une valeur aléatoire de 64 bits
- hmac et shal sont des fonctions de hachage

Par exemple si le nœud A veut s'authentifier auprès de B, celui-ci doit vérifier le ID de A comme suit :

$$A \rightarrow B : Public\_Key_A, imprint, \{CBID\}_{private\_key\ A}$$

A envoie à B un message qui contient la clé publique de A et la valeur aléatoire *imprint* ainsi que le CBID de A chiffré par sa clé privée, B effectue un hachage sur la clé publique et *imprint* pour calculer le CBID, alors B peut authentifier A s'il constate que les deux CBID, celui calculé et celui reçu se conforment, en s'assurant que la clé publique est bien celle de A.

Mécanismes	**Cryptographie à seuil**	**Self-organized PKI**	**ID-based cryptography**	**Cryptography-based Adress**	**Smock**
Certificats	Oui	Oui	Non	Non	Non
Serveurs	Oui	Non	Oui	Non	Off-line
CA	Oui	Non	Non	Non	Non
Principe	CA distribuée	Confiance transitive (PGP)	PK=hash(ID)	ID=hash(PK)	Plusieurs clés privées par nœud
Avantages	- Distribution et coopération. - Flexibilité.	- Pas besoin de CA - Autogéré	- Pas besoin de certificats - Préserve la bande passante	- Pas besoin de certificats - Préserve la bande passante - Introduit les adresses IP	- Réduction du nombre de clés générées et stockés par les nœuds - Résister à l'attaque Sybil
Inconvénients	- Nécessite un tiers de confiance pour l'initialisation - Attaques Sybil	- Authentification probablement garantie - Surcoût du stockage et échange de certificats - Confiance initial est nécessaire	- IP-spoofing - Attaque brute force - Approche conceptuelle (ID peut être n'importe quoi)	- Pas de contrôle des nœuds malicieux qui génèrent de fausses adresses	- le problème distribution de clés est non traité - problème de l'authenticité des clés publiques.

Tab.3. Mécanismes de gestion de clés dans les réseaux ad hoc basés sur la cryptographie asymétrique

Les auteurs dans [24] proposent aussi de dériver l'adresse IP du nœud à partir de sa clé publique, ils utilisent les adresses IPv6 de 128 bits pour être plus solides contre les attaques dites brute force. L'adresse IP est obtenue en combinant le préfixe du réseau (64 bits) et la valeur résultante de l'hachage de la clé publique (64 bits), de cette manière il est facile de vérifier si une clé publique appartient bien à un nœud étant donnée l'adresse IP. Cependant, un problème se pose lorsque la clé privée du nœud est compromise ou divulguée, rendant ainsi la clé publique correspondante inutilisable, et par conséquent une paire de clé publique / privée doit être crée, et dans ce cas l'adresse IP du nœud devra aussi changer.

3.3.2.5. Le schéma de clés publiques Smock

Un nouveau schéma de clé publique appelé *smock* est proposé dans [55], ce schéma est conçu pour prendre en considération les capacités limitées des nœuds d'un réseau ad hoc, en particulier leur capacité de stockage. Pour cela, *smock* propose de réduire considérablement le nombre de clés générées et stockées par les nœuds, par rapport à un ensemble de *n* nœuds sur le réseau, ce qui permet à chaque nœud de mémoriser peu de clés, ce qui est bien adapté à ce type de réseau. En outre, ce schéma est capable de résister contre l'attaque *sibyl* et offre un service hautement disponible avec zéro communication pour l'authentification et ainsi d'alléger la surcharge globale et la complexité quant au processus d'authentification. Contrairement aux schémas classiques, *Smock* suppose qu'un nœud peut posséder plus d'une clé privée en même temps. Avant le déploiement du réseau, un ensemble de paires de clés publique / privées est généré $k = \left\{ \left(k_{pub}^{i}, k_{priv}^{i} \right) \middle| \forall i \le a \right\}$, où la i ème paire de clés public/privée est définie comme $\left(k_{pub}^{i}, k_{priv}^{i} \right)$ et $a = |K|$ représente le nombre de paires distinctes générées.
Ensuite, pour chaque noeud est attribué un sous ensemble de clés privées, soit k_{priv}^{i} le sous ensemble de clés privées attribué au nœud *i*. En plus de cet ensemble, chaque nœud doit aussi mémoriser toutes les clés publiques de l'ensemble *K*.

Soit k_{Alice}^{priv} dénote le sous ensemble de clés privées détenu par *Alice* et k_{Alice}^{pub} représente le sous ensemble de clés publiques correspondantes, si *Bob* veut envoyer un message secret à *Alice*, il a besoin de connaître k_{Alice}^{pub}, *Bob* est capable de passer le message secret à *Alice* en utilisant les clés publiques k_{Alice}^{pub} pour encrypter le message. Seul, *Alice* peut décrypter ce message si $k_{Alice}^{priv} \not\subset k_{anybody\_else}^{priv}$.

Smock dans sa conception visent trois objectifs :

1- minimiser le nombre de clés stockées par chaque nœud vu de leur capacité limitée, pour cela, dans un réseau de *n* nœuds il faut trouver l'ensemble de paires de clés *K* et une allocation de clés *KA* pour réaliser :

$$\begin{cases} min & |\mathcal{K}| + \max_{i \in V} |\mathcal{K}_i^{priv}| \\ s.t. & \mathcal{K}_i \not\subseteq \mathcal{K}_j \text{ and } \mathcal{K}_i \not\supseteq \mathcal{K}_j \quad \forall i \neq j \end{cases}$$

2- réduire la complexité de calcule ; pour simplifier les opérations de sécurité, chaque nœud souhaite utiliser un petit nombre de clés publiques pour encrypter les message envoyés, et un petit nombre de clés privées pour décrypter les messages arrivées. Pour cela, il faut réaliser l'objectif suivant :

$$\begin{cases} min & \max_{i \in V} |\mathcal{K}_i^{priv}| \\ s.t. & \mathcal{K}_i \not\subseteq \mathcal{K}_j, \mathcal{K}_i \not\supseteq \mathcal{K}_j \ (\forall i \neq j) \text{ and } |\mathcal{K}| \leq M \end{cases}$$

Où *M* est la taille de mémoire pour le stockage de clés dans chaque nœud

3- Condition de résilience : il est claire que si un nœud est compromis toutes ses clés sont compromises. Par conséquent, en moyenne $C(k_c(x), b)$ ensemble distinct de clés sont compromis lorsque un adversaire pénètre *x* nœuds et remonte jusqu'à $C(k_v(x), b)$ ensemble de clés compromis dans le pire cas. Où $K_v(x)$ dénote le nombre max de clés compromis lorsque *x* nœuds sont compromis. Le modèle *smock* définit une métrique de vulnérabilité $V_x(a,b)$ qui est le pourcentage de communications compromises lorsque *x* nœuds sont compromis

$$V_x(a,\ b) = \frac{C(k_c(x), b)}{C(a,b)}$$

Où $a = |\mathcal{K}| \quad b = |\mathcal{K}_i|$

Pour réaliser ces objectifs, les auteurs dans [55] proposent deux algorithmes pour déterminer *a* et *b* et ainsi trouver l'ensemble de clés *K*, mais aussi trouver l'allocation de clés adéquate pour vérifier les objectifs discutés ci-dessus.
Smock permet de garantir une communication sécurisée, et propose un schéma de clé publique efficace et bien adapté aux réseaux avec contraintes et capacités limités des ressources comme les réseaux ad hoc, *smock* peut aussi résister à l'attaque *sybil*. Cependant, cette solution n'aborde pas le problème d'authenticité des clés ni leurs distribution, en effet, il n'y a pas un moyen pour prouver l'appartenance d'une clé à une entité donnée comme le cas d'utiliser les certificats par exemple. Sachant que *smock* suppose que les clés sont générées et attribuées aux nœuds avant déploiement du réseau par un serveur de confiance off-line, et lorsque de nouveaux nœuds rejoignent le réseau, ceux-ci doivent obtenir leurs ensemble de clés en se connectant au serveur off-line ce qui n'est pas bien approprié à un réseau purement ad hoc auto organisé. Et comment les clés publiques des nouveaux nœuds seront distribuées aux

autres nœuds du réseau. De plus, si une clé est révélée elle peut affecter les autres sous ensembles de clés du moment qu'une clé peut être détenue par plusieurs nœuds.

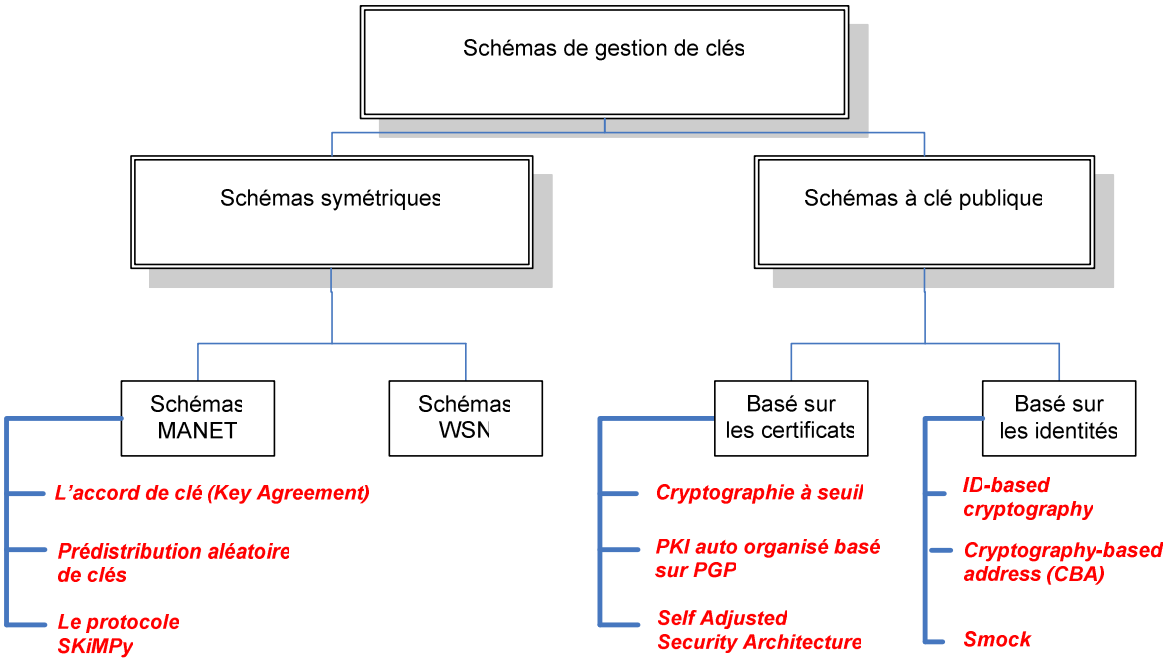


Fig.3.11. Schémas de gestion de clés étudiés

3.4. Discussion

Comme dans n'importe quel système distribué, dans les réseaux ad hoc la sécurité est basée sur l'utilisation d'un système de gestion de clés approprié. Pour être en mesure de protéger les noeuds, par exemple contre l'écoute clandestine en utilisant le cryptage, les nœuds doivent avoir un accord mutuel sur un secret partagé ou échangé des clés publiques. Pour les réseaux ad hoc fortement dynamiques l'échange des clés cryptographiques peut devoir être traité à la demande, donc sans hypothèses sur la négociation préalable des clés secrètes.

Si la cryptographie à clé publique est appliquée, tout le mécanisme de protection repose sur la confidentialité de la clé privée. Par conséquent, comme la sécurité physique des noeuds peut être pauvre, les clés privées doivent être stockés dans les nœuds de manière confidentielle. Pour des réseaux ad hoc dynamique la confidentialité de la clé privée du service doit être garantie, une solution consiste à distribuer des parts de la clé privée à plusieurs nœuds en utilisant la cryptographie à seuil. Le défis majeur des schémas à seuil est qu'ils tolèrent jusqu'à un nombre fixe de corruptions quelque soit la dimension du réseau. En effet, les nœuds d'un réseau

ad hoc présentent généralement des vulnérabilités physiques et sont exposés à plusieurs attaques en particulier l'attaque de l'adversaire mobile. Cette attaque a pour objectif de compromettre temporairement un serveur et se déplacer au prochain serveur victime. Un adversaire mobile pourrait compromettre suffisamment de serveurs sur une longue période et ainsi de compromettre la confidentialité de la clé privée. Par conséquent, les schémas à seuil doivent garantir non seulement la gestion et la distribution de clés mais surtout les deux facteurs importants qui sont la sécurité et la robustesse du mécanisme lui-même.

3.5. Conclusion

Un service efficace de gestion de clés est un élément primordial pour toute architecture de sécurité. Néanmoins, l'absence d'une infrastructure fixe dans les réseaux ad hoc est un défi majeur pour le déploiement d'un système de gestion de clés adapté à la nature de ce type de réseau. Nous avons présenté dans ce chapitre les principales solutions proposées dans la littérature pour palier au problème de la gestion de clés dans les réseaux ad hoc. Ces solutions introduisent différentes approches d'où on peut en tirer trois approches principales : la technique de cryptographie à seuil, le modèle de confiance transitives (PGP) et la cryptographie basée sur les identités. Le reste des schémas se basent sur ces approches. Selon la manière de l'établissement des clés, les schémas présentés dans ce chapitre sont répartis en deux catégories, schémas à clés publiques et schémas symétriques comme c'est illustrée dans la figure 3.11.

Une observation générale est que les schémas de gestion de clés proposés ne sont pas vraiment efficaces pour tous les scénarios des MANETs. La cryptographie à seuil semble une solution efficace pour traiter le problème de déploiement d'une PKI au sein d'un MANET. Néanmoins, la plupart des modèles proposés n'abordent pas le problème des nœuds compromis qui peuvent affecter l'aspect sécurité et robustesse du schéma appliqué. Le passage à l'échelle du réseau est un autre facteur pour lequel il n'y a pas de solution efficace. On peut constater, donc, que les réseaux ad hoc représentent un environnement hostile qui apporte plusieurs défis de sécurité notamment la gestion de clés pour laquelle plusieurs travaux de recherche sont ouverts.

Chapitre 4

Schéma robuste et sécurisé pour la gestion de clés publiques dans les réseaux ad hoc

Dans ce chapitre, nous proposons un nouveau schéma complètement distribué pour la gestion de clés publiques dans les réseaux ad hoc à large échelle, en se basant sur la technique de cryptographie à seuil et le partage de secret. Ce schéma dénommé SRKM (pour Secure and Robust threshold Key Management) fournit une solution efficace pour la gestion décentralisée de certification au sein d'un environnement ad hoc, et tient compte des vulnérabilités de tels environnements. En plus de sa flexibilité et adaptabilité au passage l'échelle de la taille du réseau ad hoc, SRKM garantit une meilleure robustesse et sécurité au service de gestion de clés. Il assure que la clé privée du service (utilisée pour signer les certificats) ne soit pas révélée à un adversaire même lorsqu'un certain nombre de nœuds partageant la clé, deviennent compromis suite à une attaque comme par exemple l'attaque de l'adversaire mobile. Nous présentons également dans ce chapitre, les détails des protocoles et algorithmes relatifs au schéma proposé pour assurer les différentes opérations du service de gestion de clés (partage et de secrets, génération et distributions des clés, attribution de certificats, rafraîchissement de parts de clé, etc.).

4.1. Introduction

Les modèles classiques de PKI ne peuvent pas s'appliquer aux réseaux mobiles ad hoc (MANET) dû à l'absence d'une partie centrale de confiance et à d'autres caractéristiques qui les distinguent des réseaux basés infrastructure. En effet, dans les architectures centralisées de PKI, le service de certification est fourni par une autorité centrale de certification (CA) pour tout le réseau. Cependant, cette approche

n'est pas souhaitable pour les MANET car elle peut mener à un état de vulnérabilité et de faiblesse. La disponibilité de la CA ne peut être garantie suite à une défaillance, compromission ou une attaque DoS, et donc, la sécurisation des communications entre les membres du réseau ne sera pas assurée.

Les schémas de CA distribuée en se basant sur la technique de cryptographie à seuil semblent être une bonne solution au problème de déploiement d'une PKI dans les MANETs, ces modèles permettent de partager la confiance d'une CA entre un ensemble de nœuds et les opérations de la CA (comme la signature d'un certificat digital) sont assurées par collaboration de ces nœuds. Selon ce principe, la clé privée distribuée de la CA est partagée entre les nœuds du réseau en utilisant un schéma de partage de secret. Cependant, le fait de fournir des informations sur la clé privée du service aux nœuds ad hoc peut conduire à un état de vulnérabilité du moment que ces informations (parts de clé) peuvent être utilisées pour révéler la clé privée partagée, ouvrant de ce fait la porte à l'usurpation d'identités et la révocation malveillante des certificats valides. Du fait que la sécurité physique des nœuds ad hoc est relativement pauvre, cela expose les nœuds à plusieurs attaques actives en particulier le cas de l'adversaire mobile, qui attaque, compromet et prend le contrôle d'un nœud pour une période limitée avant de passer à une autre victime. Sur une longue période du temps, un adversaire mobile peut compromettre suffisamment de nœuds pour combiner leurs parts et ainsi découvrir la clé privée du service. Le partage de secret seul, ne peut donc pas se défendre contre cette attaque.

Afin de limiter ce type d'attaque, certaines solutions [1][2][3] emploient le protocole proactif de rafraîchissement (PSS) [56] qui permet aux nœuds de rafraîchir périodiquement leurs parts en générant de nouvelles parts à partir des anciennes, de cette façon, un adversaire ne peut pas combiner des parts de différentes versions pour reconstruire le secret. Zhou [57] définit le concept de la fenêtre de vulnérabilité qui exprime le temps durant lequel les nœuds rafraîchissent leurs parts. La fenêtre de vulnérabilité définit donc l'intervalle de temps dans lequel un adversaire mobile ne doit pas compromettre un nombre de nœuds dépassant le seuil de vulnérabilité. Néanmoins, cette hypothèse n'est pas toujours garantie notamment avec des réseaux à large échelle constitués de grand nombre de nœuds, puisque la fenêtre de vulnérabilité peut augmenter avec l'accroissement du nombre de nœuds partageant la clé privée, ou suite aux attaques DoS qui ralentissent les serveurs et/ou augmentent le délai de livraison de messages. Nous constatons donc, que les schémas classiques à seuil proposés pour les MANETs ne tiennent pas compte d'un aspect très important qui est la sécurité et la robustesse lors du passage à l'échelle du réseau.

Dans ce contexte, notre contribution vise à améliorer la robustesse et la sécurité des systèmes de gestion de clés publiques dans les MANETs tout en assurant un service de certification efficace et hautement disponible. Dans cette contribution,

nous avons proposé un schéma de gestion de clés pour les MANETs dénommé SRKM. Ce schéma tient compte de l'absence d'infrastructure dans les MANETs, et garantit une flexibilité au passage à l'échelle du réseau, tout en assurant les services requis de sécurité (la confidentialité, l'authentification, la tolérance aux intrusions et la disponibilité). Pour assurer l'authentification et la distribution sécurisée des clés sans aucune autorité centrale, nous avons choisi d'utiliser la cryptographie à seuil, jouant le rôle d'une architecture PKI dans le contexte des réseaux ad hoc. Ce schéma suit une approche complètement distribuée (tous les nœuds coopèrent aux opérations de certification) et fournit une meilleur robustesse et sécurité aux opérations de gestion de clés, en assurant que la confidentialité de la clé privée est préservée même si le nombre de nœuds compromis dépasse le seuil de vulnérabilité, ce qui n'est pas possible avec les schémas classiques.

4.2. Paramètres de conception

Dans cette section, nous expliquons d'abord certains concepts et notations utilisées dans ce travail, ensuite nous discutons nos hypothèses sur l'environnement réseau et le modèle d'intrusion. Le tableau 4.1 décrit les principales notations utilisées dans notre travail.

4.2.1 Définitions et notation

Un nœud serveur : est un nœud membre du réseau, l'ensemble de ces nœuds forment une CA distribuée. Chaque nœud serveur possède une part de la clé privée distribuée de la CA (clé partielle) et participe au processus de génération à seuil de signature.

Un partage : un partage est la division d'un secret S entre n participants selon un schéma de partage de secret, le résultat est un ensemble de n parts. Nous notons

$(S1, S2, \ldots, Sn)$ le partage du secret S.

Une part de clé : c'est une clé partielle détenue par un nœud serveur, cette clé partielle est obtenue en divisant la clé privée de la CA selon un schème de partage de secret. Les clés partielles peuvent être combinées pour reconstruire la clé initiale.

Une sous part de clé : c'est une part d'une clé partielle.

Signature partielle ou certificat partiel : chaque nœud serveur utilise sa part de clé pour générer sa propre signature partielle sur le certificat. Les certificats partiels sont ainsi combinés afin de reconstruire le certificat final signé par la clé privée du service du CA distribuée.

Symboles	Description
SK / PK	Paire de clé privée/publique du service de certification, utilisée pour signer / vérifier les certificats à clé publique
K_i / K^{-1}_i	Paire de clé publique/privée du nœud i
SK_i / PK_i	La part privée associée au nœud i et la part publique correspondante
SK_{ij}	Sous part de la part i, associé au sous serveur j
$CERT_i$	Un certificat du nœud i signé par la clé privée SK
$cert_{ij}$	Un certificat partiel généré par un serveur j et destiné au nœud i
$cert_{ij, l}$	Un sous certificat partiel généré par un sous serveur l appartenant à la classe Cj, le certificat est destiné au nœud i
R	Ensemble de nœuds serveurs réels (r_i) tel que $\|R\| = n_1$
V	Ensemble de nœuds serveurs virtuels (v_i) tel que $\|V\| = n_2$
C_i	Classe de sous serveurs dont le représentant est le serveur virtuel v_i, les nœuds de cette classe partagent la part SK_i
h (_)	Une fonction de hachage non réversible.
$Vb(x)$	Une métrique de vulnérabilité où x nœuds sont compromis
$A \rightarrow B : M$	L'entité A envoie à l'entité B le massage M

Tab.4.1. Symboles et notation utilisée.

Fenêtre de vulnérabilité : ce concept a été défini par Zhou [57] afin de désigner les intervalles de temps dans lesquelles un adversaire doit compromettre autant de serveurs qu'il faut pour rompre le système gestion de clés. Cet intervalle n'est pas nécessairement exprimé par un temps notamment dans le cas des systèmes asynchrones comme les réseaux ad hoc. Ils peuvent notamment être exprimés en termes d'événements. La fenêtre de vulnérabilité est exactement l'intervalle entre le début d'un rafraîchissement des parts et là fin de l'opération de rafraîchissement suivante.

***NAC (Nodes Apportionment Coefficient)* :** c'est un paramètre selon lequel on peut déterminer le meilleur partitionnement (nœuds serveurs réels Vs serveurs virtuels) adéquat pour le niveau de sécurité requis. Ce dernier est exprimé par temps requis pour trouver la clé privée du système et au niveau de latence requis (temps pour signer un certificat). Le *NAC* est défini comme le ratio des nœuds serveurs réels à l'ensemble des serveurs virtuels. Pour maximiser la sécurité le *NAC* doit être diminué (*NAC* ≤ seuil de sécurité requis), tandis que pour minimiser la latence le *NAC* doit être maximisé (*NAC* ≥ seuil de latence requis). Le meilleur *NAC* détermine l'optimal des deux paramètres sécurité et latence.

4.2.2. Hypothèses

Dans cette section nous spécifions les modèles réseau et adversaire et les hypothèses auxquelles ils sont soumis.

4.2.2.1. Modèle du réseau

Le schéma de gestion de clés proposé est appliqué à un réseau mobile ad hoc à large échelle et asynchrone, il n'y a pas d'hypothèse sur le délai de traitement et de livraison des messages. Les nœuds du réseau communiquent entre eux via des liens sans fils insécurisés. La communication multi-sauts est assurée par des protocoles de routage ad hoc existants.

Le réseau est constitué de *N* nœuds dont le nombre peut changer dynamiquement lorsque des nœuds partent ou rejoignent le réseau. Chaque nœud possède un identifiant unique *ID* au sein du réseau, il peut être une adresse MAC ou une adresse IP. Nous supposons aussi que chaque nœud est capable de découvrir ses voisins immédiats (de simple saut) en exécutant certain protocole de découverte de voisinage. Cependant, il n'y a pas d'hypothèses sur le nombre de voisins de chaque nœud.

Dans notre architecture nous supposons les points suivants :

1- Chaque nœud *i* possède une paire de clés publique/privée : $[K_i, K^{-1}_i]$
2- Le service de certification est fourni par une autorité de certification distribuée qui possède une paire de clé publique/privée [*PK, SK*], cette paire de clés est utilisée pour vérifier/signer les certificats. Tous les nœuds du réseau connaissent la clé publique de la CA et font confiance à tout certificat signé par sa clé privée.
3- Chaque nœud qui partage la clé privée *SK* en détient une et une seule part (clé partielle).
4- Chaque nœud peut se déplacer, quitter ou rejoindre le réseau arbitrairement.

4.2.2.2. Modèle d'intrusion

Nous discutons brièvement quel genre d'intrusion est autorisé ainsi que les différents modèles d'adversaire pour lesquelles notre proposition apporte une solution positive.

Un adversaire est un nœud malicieux qui essaye par tous les moyens disponibles en sa possession, de rompre (nœuds compromis) et / ou dégrader ou arrêter (attaques DoS) le système de sécurité appliqué. A n'importe quel moment un nœud du réseau est soit correct ou compromis. Un nœud compromis peut cesser de

collaborer, dévier arbitrairement des spécifications de ses protocoles (présenter un comportement Byzantin), et/ou divulguer ou changer les informations stockées localement qu'elles soient privées ou publiques. Un adversaire est supposé être capable de prendre le contrôle total d'un nœud compromis qui exhibe un comportement byzantin, il peut se faire passer pour ce nœud et il a l'accès à toutes les informations stockées dans ce nœud y compris la clé privée du nœud ainsi que sa part de clé utilisée dans le processus de signature de certificats.

De multiples nœuds incorrects peuvent conspirer pour lancer une attaque collaborative, ils peuvent combiner leurs parts pour reconstruire la clé privée du système. Pour simplifier la présentation, nous considérons ce groupe de nœuds compromis comme un seul adversaire. Dans notre travail, nous employons les modèles suivants comme c'est présenté dans [58] et [4] afin de caractériser les adversaires de différentes capacités :

- **Modèle 1 :** pendant toute la durée de vie du réseau, un adversaire est incapable de compromettre *t* nœuds ou plus.

- **Modèle 2 :** il est supposé que le temps est divisé en intervalles, un adversaire est incapable de compromettre *t* nœuds ou plus durant chaque intervalle de temps.

Afin de renforcer la sécurité de notre schéma contre ces deux modèles d'adversaire nous employons la technique de rafraîchissement de parts de clé. Dans ce cas, une fenêtre de vulnérabilité est exprimée par un intervalle (temps ou événements) qui sépare le début d'une exécution de l'opération de rafraîchissement de parts et la fin de l'opération du rafraîchissement suivant (voir figure 4.1).

Pour tester la robustesse de notre schéma, nous définissons deux autres modèles d'adversaire plus puissants :

- **Modèle 3 :** dans chaque fenêtre de vulnérabilité, un adversaire est capable de compromettre jusqu'à *t* nœuds *(t* étant le seuil de vulnérabilité*)*.

- **Modèle 4 :** Il n'y a pas de limite sur le nombre de nœuds compromis durant chaque fenêtre de vulnérabilité. L'adversaire peut alors compromettre un nombre de nœuds dépassant le seuil de vulnérabilité.

Ces quatre scénarios modélisent l'attaque de l'adversaire mobile, ce terme a été proposé par Ostrovsky et Yung [59] pour caractériser les adversaires qui compromettrent temporairement un serveur et se déplacent au prochain serveur victime (par exemple, sous la forme de virus injectés dans un réseau). Un adversaire

mobile pourrait potentiellement compromettre tous les serveurs du réseau sur une longue période.

L'adversaire des modèles 3 et 4 est plus puissant que les deux premiers. Notre schéma améliore la robustesse et la sécurité car il est capable de se protéger contre ces deux adversaires.

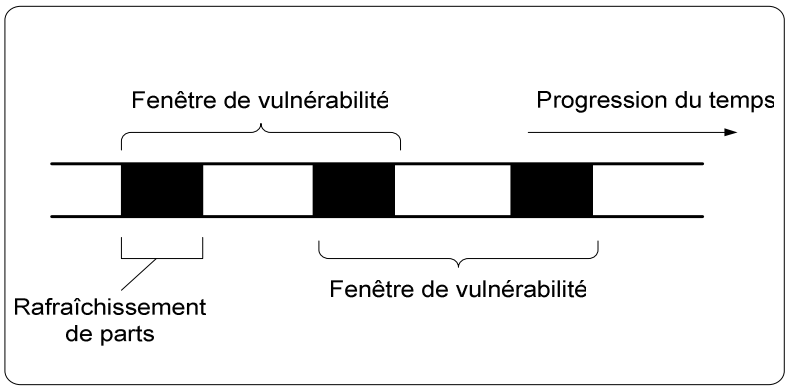


Fig.4.1. Fenêtre de vulnérabilité [57]

4.3. Le schéma (SRKM) pour la gestion de clés basée sur la cryptographie à seuil dans les MANETs

Pour surmonter les défis discutés précédemment, nous proposons SRKM, un schéma de gestion de clés publiques robuste et sécurisé. Le schéma proposé est basé sur la cryptographie à seuil et le partage de secret pour distribuer les clés cryptographiques et fournir un service distribué de certification efficace et flexible.

4.3.1. Critères requis dans la conception du SRKM

Afin de fournir un service de gestion de clés robuste, sécurisé et qui s'adapte au passage à l'échelle du réseau, la conception du SRKM doit satisfaire aux critères suivants :

Confidentialité : La clé privée du service de certification est le secret partagé qui doit rester inconnu aux adversaires pondant la durée de vie du réseau.

Sécurité et robustesse : le système doit pouvoir fonctionner correctement en présence de nœuds malicieux et d'intrusions. Si certains nœuds sont incorrects ou

compromis, cela ne doit pas compromettre tout le système ou révéler sa clé de signature. Le système doit résister aux attaques qui visent à compromettre le système de clés.

Disponibilité : à tout moment pendant la durée de vie du réseau, lorsqu'un client sollicite le service pour un certificat, les nœuds corrects disposent de suffisamment de parts de la clé pour pouvoir les combiner et signer le certificat demandé. Le service doit être disponible à tout moment.

Passage à l'échelle : le réseau ad hoc peut être constitué d'un grand nombre de nœuds, la taille du réseau peut changer fréquemment lorsque des nœuds partent et d'autres rejoignent le réseau. Notre schéma doit donc être flexible et s'adapter au passage à l'échelle du réseau.

4.3.2 Description de l'architecture de SRKM

Nous considérons un réseau mobile ad hoc constitué de N nœuds. Le schéma SRKM est basé sur le modèle de cryptographie à seuil de la configuration (n,t) dans lequel les certificats sont délivrés par une autorité de certification (CA) distribuée formée par un ensemble de n nœuds. Avec ce modèle le service de certification est fourni par coalition d'un nombre seuil de t nœuds. Pondant la durée de vie du réseau, la CA distribuée possède une paire de clés privée / publique (SK/PK) utilisée pour signer / vérifier les certificats de clés publiques. La clé privée SK est partagée entre n nœuds qui forment la CA en se basant sur le schéma polynomial de partage de secret de Shamir [11]. Un certificat est généré en combinant n'importe quel sous-ensemble de t certificats partiels.

A la différence des autres schémas à seuil [1][2], SRKM repose sur le principe suivant :

À l'intérieur du cryptosystème à seuil (n, t), les parts de certains serveurs (nœuds processeurs de clé partielle) sont elles-mêmes partagées selon un schéma de partage (m, k) par d'autres nœuds appelés sous-serveurs. De cette manière, les nœuds serveurs possèdent chacun une part de la clé et le reste des nœuds sont répartis en classes appelée classes de partage dénotés par C_i $(i = 1,\ldots,n_2)$, chaque classe possède un représentant appelé serveur virtuel. Ce dernier partage sa part de clé SKi avec les nœuds de sa classe C_i selon le (m, k) partage $Pi = (SK_{i1},\ldots,SK_{im})$.
Pour deux classes différentes nous avons deux partages différents, si C_i et C_j sont deux classes de sous serveurs, alors les partages associés à ces classes sont respectivement Pi et Pj avec $Pi \neq Pj$, cela signifie qu'une sous part de Pi ne peut pas être combinée avec une sous part de Pj.

Si un client sollicite les serveurs qui forment la CA pour un certificat, chacun va générer une signature partielle (certificat partiel). Pour un serveur virtuel, il devra solliciter l'ensemble des nœuds de sa classe (sous-serveurs). Une coalition de k nœuds de cette classe est nécessaire pour générer un certificat partiel en combinant leurs sous parts. L'architecture de SRKM est illustrée dans la figure 4.2. Dans cette architecture, les nœuds du réseau ad hoc sont répartis en trois catégories, les nœuds serveurs réels, les serveurs virtuels et les sous serveurs, à noter qu'un noeud qu'il soit réel, virtuel ou sous serveur est un participant physique dans le réseau ad hoc (une entité sans fil et mobile) :

- **Nœud serveur réel (serveur réel)** : Un noeud membre du réseau qui détient réellement une part de la clé privée du système CA (part stockée localement). L'ensemble des serveurs réels est dénoté par $R = \{r_1,\ldots,r_{n1}\}$ tel que $|R|=n_1$ étant le nombre de serveurs réels dans le réseau.
- **Nœud serveur virtuel (serveur virtuel)** : Ce nœud est considéré comme un serveur qui participe également au processus de génération de certificat et qui est sollicité comme les autres serveurs réels. A la différence d'un serveur réel, ce nœud ne possède pas une part de clé, et donc lui seul, ne peut pas attribuer un certificat partiel comme les autres serveurs réels. Un serveur virtuel détient uniquement une sous part de la part de clé qui lui est virtuellement associée. Ce nœud a en fait deux rôles ; un serveur qui doit collaborer avec les autres serveurs réels pour la génération à seuil des certificats, d'un autre coté il joue le rôle d'un sous serveur qui collabore avec les autres nœuds sous serveurs de sa classe pour reconstruire le certificat partiel correspondant à leur classe. L'ensemble des serveurs virtuels est dénoté par $V = \{v_1,\ldots,v_{n2}\}$ où $|V|=n_2$ étant le nombre des serveurs virtuels dans le réseau.
- **Nœud sous-serveur** : Un sous serveur qui partage avec les autres sous serveurs de sa classe la part de clé associée à leur classe, chacun en détient une sous part qui sera combinée avec les autres sous parts de la même classe pour reconstruire le certificat partiel correspondant. Ces nœuds sont répartis en classes dénotées par $C_i = \{c_{i1},\ldots,c_{im}\}$ (i = 1,…,n_2) telle que : c_{i1} représente un sous serveur appartenant à la $i^{ème}$ classe et $|C_i| = m$ (la part de clé est partagée entre m nœuds dans cette classe). Chaque classe est constituée d'un groupe de m nœuds dont un nœud serveur virtuel jouant le rôle de chef de sa classe. Quand un nœud serveur virtuel parte ou se compromet, un autre sous serveur de sa classe sera élu pour remplacer l'ex-nœud virtuel. Ainsi le rôle du serveur virtuel peut être joué par n'importe quel sous-serveur de sa classe.

Dans SRKM, une configuration de (n, t) signifie qu'il y a n serveurs dans le réseau qui sont sollicités directement par les clients pour les opérations de

certification. Cependant, il n'y a pas n parts de clé dans cette configuration mais uniquement n_1 parts détenues par les serveurs réels où $0 < n_1 < t$, sachant que t-1 représente le seuil de vulnérabilité (la clé peut être trouvée en ayant t parts), car parmi les n serveurs il y a (n-n_1) serveurs virtuels. De cette façon un adversaire mobile ne pourra jamais trouver la clé privée SK avec t nœuds compromis dans chaque fenêtre de vulnérabilité puisque parmi les t nœuds compromis il y en a qui sont virtuels ou sous serveurs (possèdent une sous part) appartenant à différentes classes et par conséquents leurs sous parts ne peuvent pas être combinées. Par contre aux autres schémas à seuil de (n, t) comme ceux proposés par Zhou [1] et Kong [2], si t nœuds sont compromis alors tout le système est rompu et la clé privée est révélée. Notre schéma permet donc de se protéger efficacement contre l'attaque de l'adversaire mobile et de renforcer ainsi la sécurité et la robustesse du système de *gestion de clés* basé sur la cryptographie à seuil.

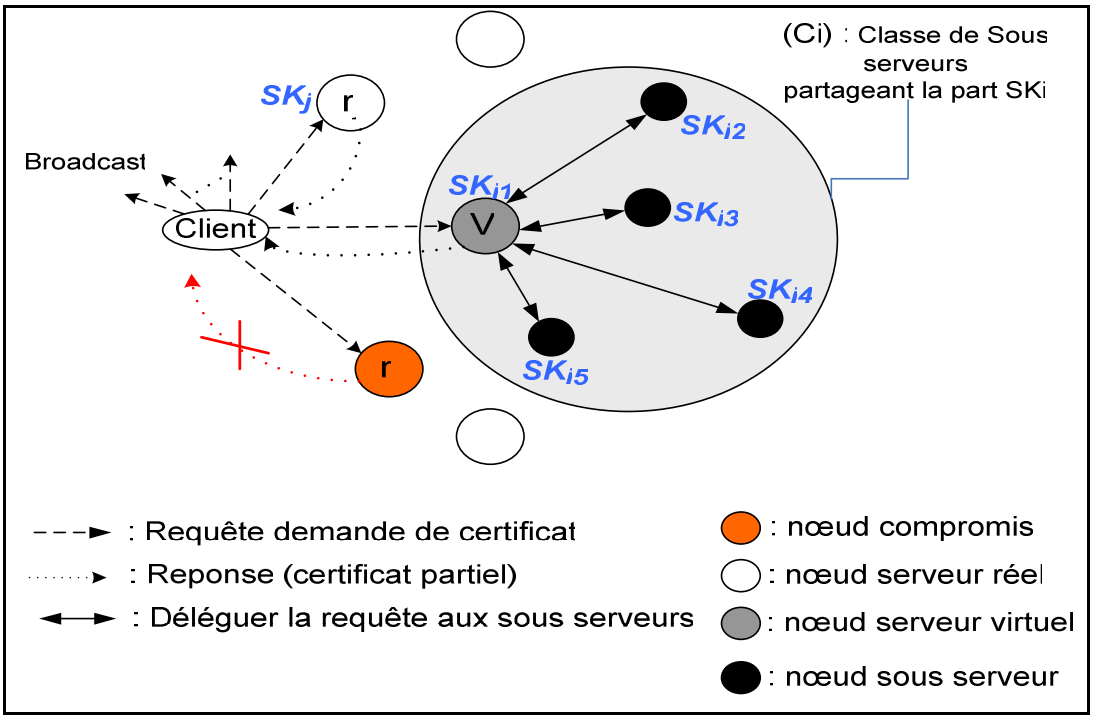


Fig.4. 2. Aperçu de SRKM

4.3.3 Opérations de base et primitives cryptographiques

Nous discutons dans cette section les différentes techniques employées par notre schéma ainsi que les opérations de base.

4.3.3.1. Le partage de secret

Dans SRKM, le protocole de division de la clé privée du service de certification en n parts et le protocole de reconstruction de celle-ci sont basés sur le schéma de partage de secret de Shamir[11]. Ce schéma est appelé aussi partage de secret à seuil (n, t). Le but est de mettre en commun (partager) un secret S entre plusieurs entités (figure 4.3) en utilisant un polynôme aléatoire $f(x)$. Si le degré de $f(x)$ est de $t-1$ alors, à partir, seulement de t entités on peut reconstruire le secret S par interpolation de Lagrange. Cependant, pour $t-1$ ou moins d'entités, il est impossible de reconstruire S. t étant le seuil.

Soit $f(x)$ un polynôme aléatoire de degré $t-1$ telle que $f(x) = a_0 + a_1 x + \ldots + a_{t-1} x^{t-1}$ avec $a_0 = S$. Le secret S sera divisé en n parts ($s_1, s_2, \ldots, s_n$) qui sont obtenues par $S_i = f(i)$ pour $1 \le i \le n$ telle que i est l'identité de l'entité X_i, et on associe chaque part (S_i) à l'entité X_i. ($s1,s2,\ldots,sn$) est alors appelé partage du secret S.
Le secret S peut être calculé à partir de n'importe quel sous-ensemble de t parts. Sans perte de généralité nous marquerons ce sous–ensemble par : $f(1),\ldots, f(t)$. On calcule S à partir de ces parts en utilisant l'interpolation de Lagrange :

$$S = \sum_{i=1}^{t} f(i) \cdot Li(0)$$

Où $L_i(0)$ est le coefficient de Lagrange telle que :

$$L_i(x) = \frac{\prod_{j \neq i} (x - x_j)}{\prod_{j \neq i} (x_i - x_j)}$$

Dans SRKM, les serveurs obtiennent leurs parts de clé à l'initialisation du réseau par une partie de confiance ou par l'auto initialisation pour les serveurs qui viennent de rejoindre le réseau, nous détaillons cette opération dans la section **(6.1).**

4.3.3.2. Le service de Certification

Dans SRKM, lorsqu'un nœud sollicite le service pour un certificat, une coalition de t nœuds serveurs est formée à la volée. Chaque serveur r_i qui reçoit la requête génère une signature partielle sur le certificat en utilisant sa part SK_i. Tandis qu'un serveur virtuels v_i doit faire passer la requête aux sous serveurs de sa classe Ci, il doit combiner sa sous part avec $k-1$ sous parts valides de sa classe pour reconstruire le certificat partiel correspondant. Une fois le nœud client collecte t signatures

partielles sur le certificat, il peut les combiner pour avoir son certificat final signé par la clé privée (*SK*) de la CA distribuée. Ce certificat est vérifié par la clé publique de la CA (*PK*). A noter que la clé privée (*SK*) n'est jamais découverte (reste inconnu à tous les membres) lorsque *t* nœuds coopèrent pour signer un certificat, ce qui garantit le paramètre confidentialité.

4.3.3.3 Renouvellement des parts

Afin de garantir une sécurité proactive pour résister aux adversaires qui compromettent progressivement les serveurs sur une longue période, nous employons la technique de rafraîchissement de parts de clé [60][56] pour renforcer la sécurité de notre schéma. Cette technique permet de rafraîchir périodiquement les parts. Les nœuds rafraîchissent leurs parts par collaboration et sans divulguer la clé privée du système de signature qui est partagée entre eux. Chacun calcule sa nouvelle part à partir de l'ancienne part. Les nouvelles parts constituent un nouveau partage (*n*, *t*) de la clé privée du service *SK*. Après rafraîchissement, les nœuds suppriment leurs anciennes parts et gardent seulement les nouvelles parts. Un adversaire ne peut pas combiner les anciennes parts avec les nouvelles pour calculer la clé privée partagée. Ainsi, l'adversaire est mis au défi de compromettre *t* nœuds entre deux opérations consécutives de rafraîchissement de parts. Cette technique repose sur la propriété d'homomorphisme suivante :
Si (s_1,s_2, . . . , s_n) est le (n,k) partage du secret S et (s'_1, s'_2, . . . , s'_n) est un (n, k) partage d'un autre secret S', alors ($s_1 + s'_1$, $s_2 + s'_2$, . . . ,$s_n + s'_n$) est le (n, k) partage de $S + S'$. Alors, Si $S' = 0$, on obtient un nouveau (n,k) partage de S.

4.3.3.4. Vérification des parts de secret

A l'initialisation du réseau, les premiers serveurs sont initialisé par une partie de confiance qui calcule et distribue à chacun une part, ensuite ces serveurs peuvent collaborer pour initialiser d'autres serveurs qui rejoignent le réseau sans avoir besoin de cette partie de confiance. Dans le processus de rafraîchissement de parts, les nœuds concernés doivent aussi calculer et distribuer entres eux certaines parts. Certains de ces nœuds peuvent être compromis et distribuent de faux parts de clé, comment alors peut-on vérifier la validité des parts reçues. Le schéma de partage de secret de Shamir employé dans notre proposition ne permet pas de faire une telle vérification sur les parts reçues. Pour détecter les parts incorrectes, nous employons les techniques de VSS (Verifiable Secret Sharing) [61][62]. Un schéma de VSS génère des informations supplémentaires publiques pour chaque part en utilisant une fonction de hachage non réversible. A partir de ces informations public chaque nœud peut tester la validité de sa part.

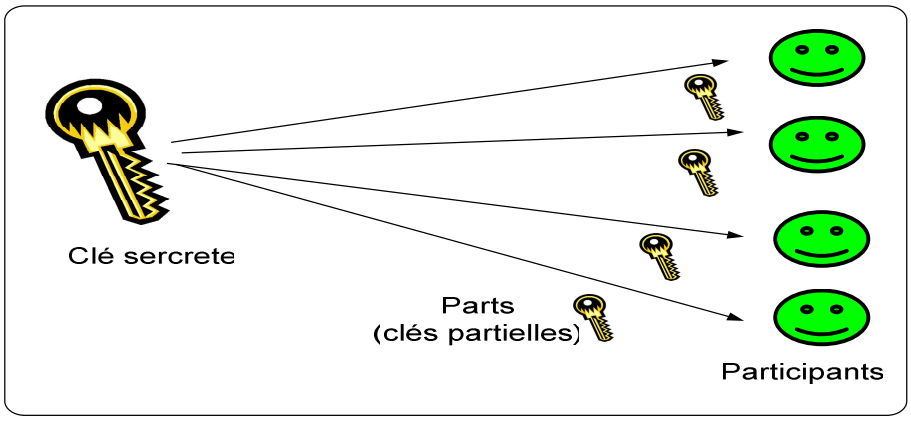


Fig.4. 3. Partage d'une clé secrète entre un ensemble de participants

4.4. Détailles cryptographiques des protocoles du SRKM

4.4.1. Initialisation du cryptosystème à seuil

Nous supposons un réseau ad hoc de N nœuds, le processus d'initialisation permet d'initialiser les n premiers nœuds qui forment le service de certification telle que ($n < N$). Cela consiste à initialiser les serveurs et les sous serveurs en leur distribuant les parts de clé selon le principe discuté dans la section **4.3.3.1**. Ainsi, nous supposons une partie de confiance qui génère un partage ($SK_1, SK_2, \ldots, SK_n$) de la clé privée du service de certification (SK) selon une configuration à seuil (n, t), et Distribue ces parts aux nœuds serveurs sans divulguer la clé SK. Ce processus s'effectue en deux phases : La première phase consiste à initialiser les serveurs en distribuant à chacun une part de la clé (figure 4.4 (a)), les n serveurs initialisés se diviseront en serveurs réels et virtuels. La deuxième phase concerne l'initialisation des sous-serveurs, à ce stade les sous serveurs sont organisés en classes, ensuite chaque sous serveur recevra une sous-part de la part partagée par sa classe (figure 4.4 (b)). Les sous serveurs sont auto-initialisés sans avoir besoin de la partie de confiance.

4.4.1.1. Initialisation des serveurs (Création et distribution des parts de clé)

Dans cette phase, la partie de confiance effectue les opérations suivantes :

- Générer un polynôme aléatoire de degré $t-1$:
  $$f(x) = SK + a_1 x + \ldots + a_{t-1} x^{t-1} \pmod p$$
  p est un grand nombre premier et $a_1, a_2, \ldots, a_{t-1}$ sont des coefficients aléatoires $\in Zp$ et SK étant la clé privée du service de certification.

- Calculer les parts de la clé privée SK :
  $$SK_i = f(i) \pmod p \ / \ i = 1, \ldots, n$$
  Pour simplifier i est un entier qui représente l'identité d'un nœud. SK_i est la part associée au nœud i.

- Distribuer d'une manière sécurisée SK_i au nœud i pour tout $i = 1, \ldots, n$.
- Désigner les ensembles R et V (les serveurs peuvent être choisis sur différents critères tels que la puissance des nœuds ou la connectivité de ceux-ci).

4.4.1.2. Initialisation des sous-serveurs (Organisation des classes et distribution des sous parts)

A partir de ce stade nous n'aurons pas besoin de la partie de confiance. Les nœuds qui n'appartiennent pas à l'ensemble R s'auto-organisent (se grouper) en classes Ci où $|Ci| = m$, chaque classe (Ci) doit comporter exactement un serveur virtuel de (V) qui sera le représentant de la classe et sa part sera partagée entre les membres de cette classe (sous serveurs). Les classes sont formées de telle sorte que chaque sous serveur doit être capable de communiquer avec le serveur virtuel de sa classe (existence d'un chemin entre les deux nœuds). L'algorithme 1 décrit le processus de création des classes de sous serveurs qui est initié par les nœuds serveurs virtuels, selon cet algorithme le processus de création des classes est constitué des deux phases *Request* et *Reply*, un nœud serveur virtuel i initie une classe et diffuse une requête $ClassReq_i$ qui inclut l'identifiant de sa classe (*ClassID*), il attache aussi un nonce Ni et son certificat $CERTi$ pour permettre aux autres nœuds de l'authentifier, toute requête reçue par un nœud (j) qui n'est pas en aucune classe sera mise dans une file d'attente (*ReqQueue*) locale. Ce nœud répondra à une requête dans la file avec $ClassReply_j$ qui inclut son nonce (Nj) et le nonce de la requête (Ni). Il attache aussi un TTL qui sera décrémenté à chaque saut par les nœuds intermédiaires, si le TTL est expiré, alors il répondra à une autre requête dans la file. Quand un serveur virtuel (i) reçoit $ClassReply_j$, il vérifie d'abord que $TTL > 0$ puis authentifie le nœud j, si le Reply est accepté et si $|Ci| < m$, alors le nœud j sera ajouté comme membre de

la classe de ce serveur virtuel ($Ci = Ci \cup j$), ensuite il faut informer la source en lui envoyant une confirmation (*ACK).*

Algorithme 1 : *Création des classes de sous-serveurs*

$TTL = 0$; $ReqQueue = \emptyset$; $Ci = \emptyset$; Ni, Nj : Nonce ; h : hash function
for each node i **do**
  if node $i \in V$ **then**
    $ClassReq_i = [IDi, ClassID, N_i, CERT_i]_{Ki^{-1}}$;
    $i \rightarrow$ ALL : { $ClassReq_i$, $[h(IDi, ClassID, N_i)]_{Ki^{-1}}$, $CERTi$ };
  end if

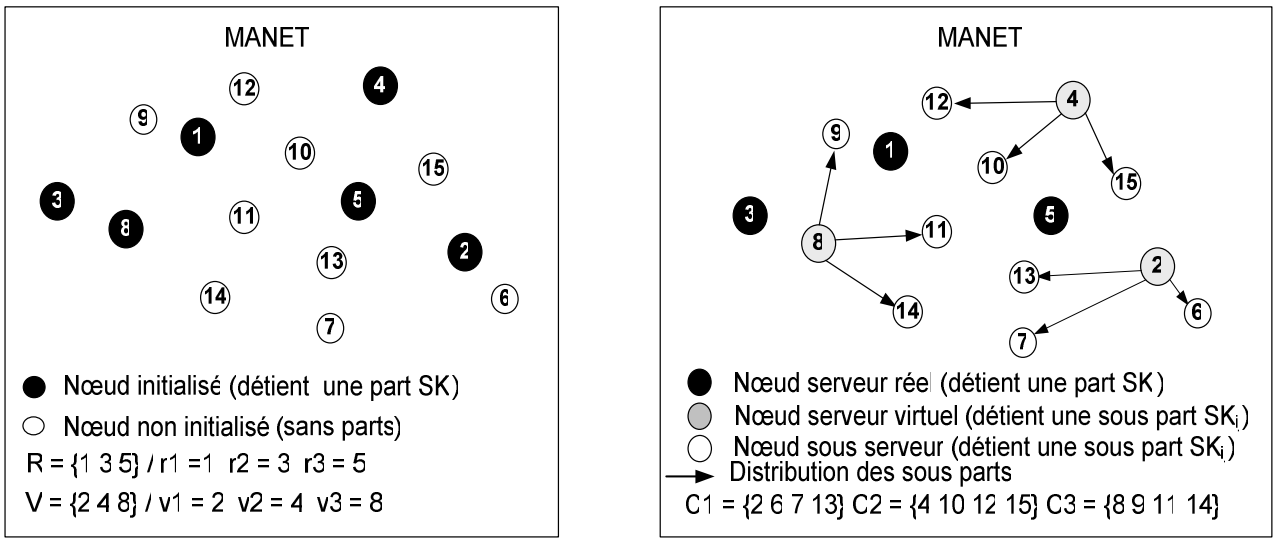
  for each node j ($j \notin V \cup R$, $j \notin C_l$: $l = 1... n_2$) **do**
    if j receive non-duplicate $ClassReq_i$ **then**
      Authenticate node i ;
      $ReqQueue = ReqQueue + ClassReq_i$;
    end if
    While not receive_j ACK **do**
      if $TTL = 0$ (TTL Expired) **then**
        Process $ClassReq_i$ in $ReqQueue$ Heading ;
        initialize (TTL) ;
        $ClassReply_j = [IDj, IDi, ClassID, N_j, N_i, CERT_j]_{Kj^{-1}}$;
        $j \rightarrow i$: { $ClassReply_j$, $[h(IDj, IDi, ClassID, N_j, N_i)]_{Kj^{-1}}$, TTL, $CERTj$ };
      end if
    end while
  end for

for each node ($i \in V$) **do**
  if i receive non-duplicate $ClassReply_j$ **and** $|Ci| < m$ **then**
    Authenticate node j ;
    if $TTL > 0$ **then**
      $Ci = Ci \cup j$;
      $i \rightarrow j$: {$[ACK, IDi, IDj, ClassID, Nj]_{k_i^{-1}}$, $[h(ACK, IDi, IDj, ClassID, Nj)]_{k_i^{-1}}$ $CERTi$} ;
    else
      $i \rightarrow j$: « Error TTL Expired » ;
    end if
  end if
end for

Une fois toutes les classes de sous serveurs sont crées, chaque serveur virtuel $i \in V$ génère un partage de sa part SK_i selon une configuration à seuil (m, k) et envoie d'une manière sécurisée les sous parts crées ($SKij$, $j=1,..,m$) aux nœuds de sa classe. Cette procédure est appelée *Class Sharing* qui est décrite comme suite :

- **Step 1 :** $\forall v_i \in V$, vi génère un polynôme aléatoire spécifique à sa classe Ci
$$g_i(x) = SK_i + bi_1\ x + \ldots + bi_{k-1}\ x^{k-1}\ \ (mod\ p)$$
- **Step 2 :** calculer un (m, k) partage de Ski
$$(Ski1,\ldots,SKim)\ /\ SK_{ij} = g_i(c_{ij})\ (mod\ p)$$
- **Step 3 :** $\forall\ c_{ij} \in Ci$ ($j = 1,\ldots,m$), distribuer SK_{ij} à c_{ij}
- **Step 4 :** v_j garde une sous part et supprime sa part SK_i et toute autre information en relation avec.

En exécutant la procédure de *Class Sharing*, on obtient un arbre de clés à deux niveau comme c'est illustré par la figure 4.5, le premier niveau de l'arbre représente un partage de secret (n, t) de la clé privée du service SK, tandis que le deuxième niveau représente les partages de secret (m, k) des parts associées aux nœuds serveurs virtuels.



(a) **(b)**

Fig.4. 4. Initialisation des nœuds : (a) création d'un partage (n, t) et distribution de parts (b) création des classes et distribution des sous parts

La figure 4.4 présente un exemple d'un réseau ad hoc de 15 nœuds, dans la première phase d'initialisation (*Fig* 4.4 (a)) la partie de confiance sélectionne les serveurs réels et virtuels selon des critères de performance, ensuite, dans la deuxième phase les nœuds sont auto-organisés pour former des classes de sous serveurs afin de partager les parts virtuelles, dans cet exemple ces parts sont *SK2, SK4, SK8* qui seront partagées respectivement par les classes *C1, C2, C3*. à la fin les nœuds du réseaux sont répartis en trois ensembles *R, V et Ci* comme c'est montré dans le tableau 4.2.

R	*V*	*C1*	*C2*	*C3*
1	2	2	4	8
3	4	6	10	9
5	8	7	12	11
		13	15	14

Tab.4. 2. Exemple de Répartition de15 nœuds à l'initialisation

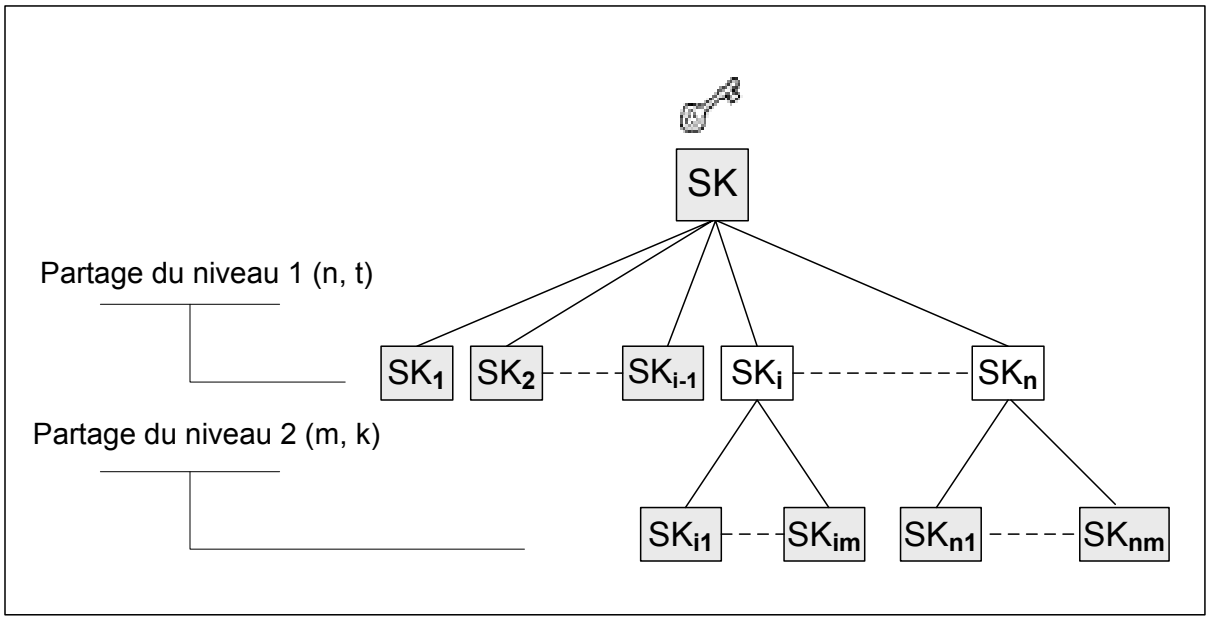


Fig.4. 5. Arbre de partage à deux niveaux

4.4.2. Rafraîchissement des parts de clé

Afin de renforcer la robustesse de notre schéma, nous employons le rafraîchissement de parts de clé en se basant sur les techniques de rafraîchissement de parts discutées dans la section **4.3.3.3**. On cherche à mettre à jour uniquement les parts sans modifier la clé privée du service (*SK*). Dans SRKM, notre protocole de rafraîchissement, permet de rafraîchir non seulement les parts mais aussi les sous parts de la part virtuelle SK_i sans avoir besoin de recréer et repartager à nouveau SK_i, les sous serveurs rafraîchissent localement leurs sous parts sans en révéler aucune information ni au serveur virtuel ni aux autres membres de leur classe.

Après une opération de rafraîchissement de parts, on obtient un nouveau (n, t) partage $(SK_1', SK_2', ..., SK_n')$ de *SK*. Chaque serveur réel $r_l \in R$ ($l = 1 \ldots n_1$) régénère une nouvelle part. Cependant, chaque sous serveur ($\forall Ci, \forall c_{ij} \in Ci$) doit calculer une

nouvelle sous part $SK_{ij}^{'}$ telle que : $SK_i^{'} = \sum_{j=1}^{k} SK_{ij}^{'}.l_{cij}(0)$ où $SK_i^{'}$ est la nouvelle part virtuelle qui correspond au nouveau partage $(SK_1^{'}, SK_2^{'}, ..., SK_i^{'}, ..., SK_n^{'})$.

Soit $(SK_1, SK_2, ..., SK_n)$ est le (n, t) partage de la clé privée du service (SK), ce partage est calculé sur le polynôme $f(x)$, SKi est une part de SK.

On suppose aussi que (S_1,S_2,…,S_n) est le (n, t) partage d'un secret (S), ce partage est calculé sur un polynôme aléatoire $k(x)$, S_i est une part du secret S.
D'après la propriété d'homomorphisme du partage de secret de Shamir, il en résulte que : $(SK_1 + S_1, SK_2 + S_2, ..., SK_n + S_n)$ est le (n, t) partage de (SK+S). En mettant S = 0 nous obtiendront un nouveau (n, t) partage $(SK_1^{'}, SK_2^{'}, ..., SK_n^{'})$ de la clé privée (SK) du service (**Figure 4.6**), telle que :

$$SK_i^{'} = SK_i + S_i \quad \textbf{(1)}$$

Soit $k(x) = \sum_{i=1}^{n} k_i(x)$ alors :

$$k(i) = \sum_{j=1}^{n} k_j(i) = S_i \quad \textbf{(2)}$$

En suivant ce principe, $SK_l^{'}$ ($l = 1…n_1$) et $SK_{ij}^{'}$ ($i = n_1+1…n$, $j = 1…m$) sont calculées comme suit :

- Chaque nœud $p \in R \cup V$ génère un polynôme aléatoire $k_p(x) = 0 + c_{p1}x + c_{p2}x^2 + ... + c_{pt-1}x^{t-1} = \sum_{i=1}^{t-1} c_{pi}x^i$ où c_{pi} sont des coefficients aléatoires.
- p Distribue d'une manière sécurisée $k_p(r)$ à chaque nœud $r \in R \cup V$ ($r \neq p$)
- Chaque nœud $r \in R \cup V$ collecte $k_p(r)$ ($p = 1…n$ / $p \neq r$) et les vérifie en utilisant le schéma VSS (section **4.3.3.4**). ensuite il calcule $S_r = \sum_{p=1}^{n} k_p(r)$

A partir de ce stade chaque nœud serveur réel $r_l \in R$ ($l = 1…n_1$) peut calculer sa nouvelle part :

$$\boxed{SK_l^{'} = SK_l + \sum_{j=1}^{n} k_j(l)} \quad \textbf{(3)}$$

Maintenant, nous allons montrer comment calculer $SK_{ij}^{'}$ ($i = n1+1 \ldots n$, $j = 1 \ldots m$), nous supposons d'abord que $(S_{i1}, S_{i2}, ..., S_{im})$ étant le (m, k) partage de la part S_i calculé sur le polynôme $\lambda_i(x)$. Donc :

$$S_i = \sum_{r=1}^{k} \lambda_i(r) l_r(0) \quad \textbf{(4)}$$

Si on suppose que $\lambda_i(x) = \sum_{l=1}^{m} \lambda_{il}(x)$ alors : $\lambda_i(r) = \sum_{l=1}^{m} \lambda_{il}(r) = S_{ir}$ **(5)**

Dès lors, les sous serveurs calculent leurs nouvelles sous parts selon les étapes suivantes :

- Tout nœud $v_i \in V$ distribue d'une manière sécurisée S_i (calculé par la formule 2) à l'ensemble des sous serveurs de sa classe Ci.

  $v_i \rightarrow c_{ij} : S_i \quad (j = 1 \ldots m)$
- Chaque nœud $c_{ij} \in C_i$ génère $\lambda_{ij}(x) = S_i + e_{j1}x + e_{j2}x^2 + ... + e_{jk-1}x^{k-1}$ ensuite pour tout nœud $c_{ir} \in C_i$ ($r = 1 \ldots m$ / $r \neq j$), envoyer $\lambda_{ij}(r)$.

Ainsi, chaque nœud $c_{ij} \in C_i$ calcule sa nouvelle sous part selon la formule suivante :

$$\boxed{SK_{ij}^{'} = SK_{ij} + \sum_{r=1}^{m} \lambda_{ir}(j)} \quad \textbf{(6)}$$

Les nouvelles sous parts ($SK_{ij}^{'}$) sont correctes si l'équation suivante est vérifiée :

$$SK_i^{'} = \sum_{j=1}^{k} SK_{ij}^{'} . l_j(0) \quad \textbf{(7)}$$

Preuve :

En substituant (6) dans (7) on a :

$$\sum_{j=1}^{k} \left[SK_{ij} + \sum_{r=1}^{m} \lambda_{ir}(j) \right] . l_j(0)$$

$$= \sum_{j=1}^{k} SK_{ij} l_j(0) + \sum_{j=1}^{k} \sum_{r=1}^{m} \lambda_{ir}(j) . l_j(0)$$

$$= \sum_{j=1}^{k} SK_{ij} l_j(0) + \sum_{j=1}^{k} S_{ij} l_j(0) \quad \textbf{(Selon (5))}$$

$$= \sum_{j=1}^{k} SK_{ij} l_j(0) + S_i \qquad \textbf{(Selon (4))}$$

$$= SK_i + S_i$$

$$= SK_i'$$

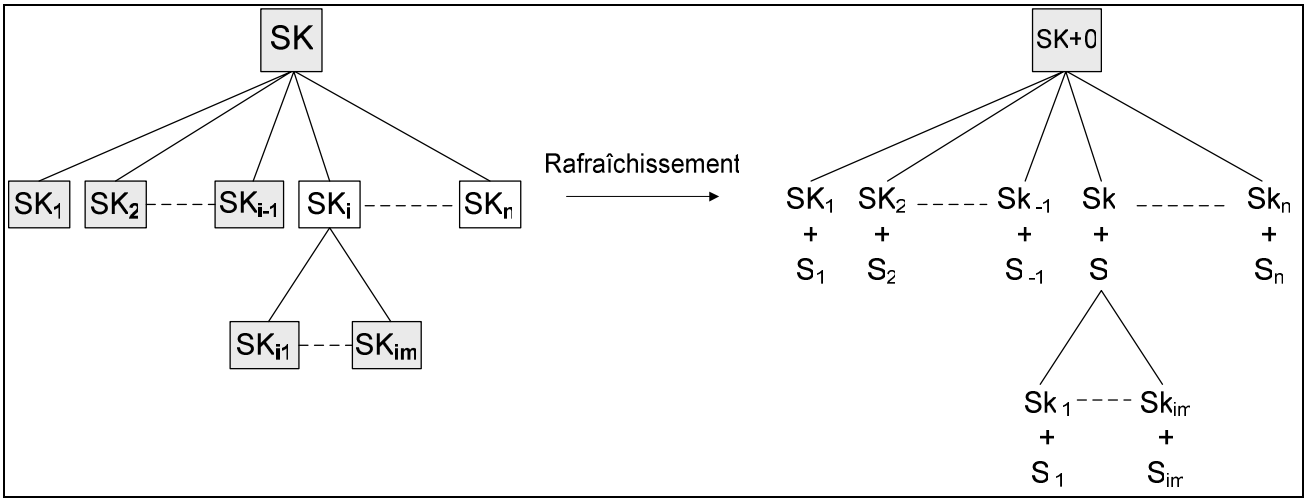


Fig.4. 6. Rafraîchissement de parts et sous parts

4.4.3. Attribution à seuil des certificats

Dans SRKM, un certificat d'un nœud i noté ($CERT_i$) est attribué par une coalition quelconque de t nœuds serveurs, chaque nœud dans cette coalition génère une signature partielle ($cert_{ij}$) sur le certificat. Ces signatures sont ensuite combinées pour avoir une signature valide du service sur le certificat final. Plusieurs algorithmes de signature et multi signatures à seuil ont été proposées dans la littérature. Kong [2] introduit une technique d'attribution à seuil de certificat basée sur RSA, cependant, sa méthode ne permet pas de vérifier si un certificat partiel est correcte. Donc, si dans une collection de t certificats partiels il y'en a un qui est incorrect, il n'y a pas un moyen de le détecter. Par conséquent, un nœud compromis qui attribue de faux certificats partiels ne peut pas être capturé, ce qui affecte le service de certification. Pour éviter ce problème, nous proposons un schéma de signature à seuil pour l'attribution des certificats dans SRKM, notre schéma est basé sur la difficulté du DLP (problème du logarithme discret).

De nombreux schémas basés sur ElGamal [63] et le Standard (DSS) [64] sont proposés mais ça nécessite le calcule de l'inverse des secrets, et une telle opération est onéreuse ce qui ne s'adapte pas aux réseaux MANET. Le schéma de multisignature dans [65] semble être plus efficace. En effet ce schéma n'utilise pas le calcule de l'inverse et permet de se protéger efficacement contre un ensemble de nœuds qui

agissent en collude pour trouver la clé privée du service en utilisant leurs parts. Cependant nous trouvons que ce schéma n'est pas bien approprié à notre modèle SRKM, car il ne s'adapte pas au passage à l'échelle du réseau et au protocole de rafraîchissement. De plus, ça nécessite la présence d'une partie de confiance pour distribuer les parts aux nœuds nouvellement ajoutés au réseau, ce qui n'est pas le cas avec SRKM. Afin de traiter ce problème, nous modifions le schéma proposé dans [65] pour qu'il s'adapte à l'architecture de notre modèle SRKM, mais nous ne présentons pas les détails du schéma proposé dans [65], nous présentons directement la version modifié de ce schéma que nous avons proposé.

Avec notre schéma de signature, il est possible de vérifier chaque certificat partiel afin de détecter les nœuds malicieux qui attribuent de faux certificats. En outre, notre schéma est basé sur DLP qui est plus rapide et moins onéreux que RSA.

Les détails de l'algorithme d'attribution à seuil d'un certificat *CERTi* à un nœud *i* sont montrées comme suit :

Soit *p* et *q* deux grands nombres premiers satisfaisant *q* / *(p-1)*, G_q est un sous groupe de Z_p de l'ordre *q*, *g* est un générateur de G_q. soit *m* le message envoyé qui représente le certificat à signer, ce message inclut l'identité et la clé publique correspondante du nœud client, *h*(.) est une fonction de hachage non réversible, et PK/SK est la paire de clé publique/privée du service de certification telle que $PK = g^{SK}$

Tout d'abord, le nœud *i* choisit un groupe de *t* nœuds serveurs, sans perte de généralité soit $B = \{j \,/\, j = 1 \ldots t\}$ l'ensemble de ces nœuds. Chaque nœud $j \in B$ recevant la requête et décide de la servir, génère une signature partielle. On distingue deux cas pour générer une signature partielle selon que $j \in B$ est un nœud serveur réel ou virtuel. (figure 4.7)

a) Cas des serveurs réels :

Si $j \in R$ (nœud serveur réel) alors :

- *j* choisit un entier aléatoire $ej \in Z_q$ et diffuse $rj = g^{ej}$ dans le groupe *B*
- *j* diffuse dans le groupe *B*, $PKj = g^{SKj \,.\, lj(0)} \ (mod\ p)$ où *SKj* est la part de clé du nœud *j*.

j calcule son certificat partiel pour *i*, spécifique au groupe *B* selon la formule suivante :

$$cert_{ij} = SK_j l_j(0) + e_j E \qquad (\mathrm{mod}\, q) \qquad \textbf{(8)}$$

Où : $E = h(m \| R)$ et $R = \prod_{j=1}^{t} r_j$

j envoie $cert_{ij}$ au nœud *i*, ce dernier peut vérifier la validité de ce certificat partiel par :

$$g^{cert_{ij}} \stackrel{?}{=} PK_j r_j^{E} \pmod p \qquad \textbf{(9)}$$

Si *t* certificats partiels sont valides, le nœud *i* obtient son certificat *CERTi* signé par la clé privée du service (*SK*), en combinant ces certificats partiels comme suit :

$$CERT_i = \sum_{j=1}^{t} cert_{ij} \qquad \textbf{(10)}$$

De cette manière *(R, CERTi)* est considérée comme une signature du service de certification sur le certificat du nœud *i*. Le certificat *CERTi* attribué au nœud *i* est vérifié en utilisant la clé publique du service (*PK*) comme suit :

$$g^{CERT_i} \stackrel{?}{=} PK.R^{E} \pmod p \qquad \textbf{(11)}$$

b) Cas des serveurs virtuels :

Si $j \in V$ (nœud virtuel) alors :

- *j* fait passer la requête du client (nœud *i*) à sa classe de sous serveurs C_j en diffusant une autre requête.
- En recevant cette requête, chaque nœud $c_{jl} \in C_j$ (serveur virtuel *j* compris) choisit un entier aléatoire $e_{jl} \in Z_q$ et envoie $r_{jl} = g^{ejl}$ et $X_l = g^{skjl \,.\, l_{cjl}(0)}$ à *j*
- *j* sélectionne uniquement un groupe de *k-1* sous serveurs avec lesquels il continuera les opérations suivantes, soit $Bj = \{c_{j1} \ldots c_{jk-1}\}$ l'ensemble de ces sous serveurs.
- *j* calcule et diffuse $r_j = \prod_{l=1}^{k} r_{jl}$ dans le groupe *B*
- *j* diffuse $PK_j = \prod_{l=1}^{k} x_l^{\,l_j(0)}$ dans le groupe *B*
- selon la formule **(12)** chaque nœud $c_{jl} \in B_j$ calcule un sous certificat partiel et l'envoie à *j* :

$$cert_{ij,l} = SK_{jl} l_{cjl}(0) l_j(0) + e_{jl} E \pmod q \qquad \textbf{(12)}$$

- si le nœud *j* collecte *k-1* sous certificats partiels, il peut donc les combiner avec le sien pour reconstruire un certificat partiel ($cert_{ij}$) spécifique au groupe *B* comme suit :

$$\boxed{cert_{ij} = \sum_{l=1}^{k} cert_{ij,l}} \quad \textbf{(13)}$$

Un certificat d'un serveur virtuel *j* est valide s'il satisfait à l'équation **(9)**

Preuve :

$$g^{cert_{ij}} = g^{\sum_{l=1}^{k} cert_{ij,l}} \quad \text{Selon (13)}$$

$$= g^{\sum_{l=1}^{k} SK_{jl} l_{cjl}(0) l_j(0) + e_{jl} E} \quad \text{Selon (12)}$$

$$= g^{SK_j l_j(0) + \sum_{l=1}^{k} e_{jl} E}$$

$$= g^{SK_j l_j(0)} . g^{\sum_{l=1}^{k} e_{jl} E}$$

$$= PK_j . \prod_{l=1}^{k} g^{e_{jl} E}$$

$$= PK_j . \prod_{l=1}^{k} r_{jl}^{E}$$

$$= PK_j . r_j^{E}$$

4.4.4. Passage à l'échelle et reconfiguration requise

SRKM est basée sur une approche complètement distribuée, cela signifie que chaque nœud collabore dans le processus de gestion de clé en exécutant un rôle particulier (serveur réel, serveur virtuel ou sous serveur). Ainsi, si un nouveau nœud vient de rejoindre le réseau, il doit se faire attribuer un rôle et participer aux opérations de certification comme tous les autres membres du réseau. Toutefois, un nœud peut cesser de collaborer parce qu'il est compromis ou présente un comportement byzantin ou tout simplement il quitte le réseau, ce qui dégrade relativement les performances du système si plusieurs nœuds ne collaborent pas. Le changement de la topologie dû au mouvement dynamique des nœuds peut également affecter certains facteurs de performance, le cas par exemple où une classe échoue à reconstruire sa part de clé par ce que plusieurs de ses membres ont déplacé

et sont devenus injoignables par le serveur virtuel de cette classe. Afin de préserver les performances du système tout au long de la durée de vie du réseau, SRKM doit être capable de s'adapter au changement dynamique de la topologie et en générale au passage à l'échelle du réseau. Cette partie aborde les reconfigurations requises pour la stabilité des paramètres de performance (Latence, disponibilité, sécurité).

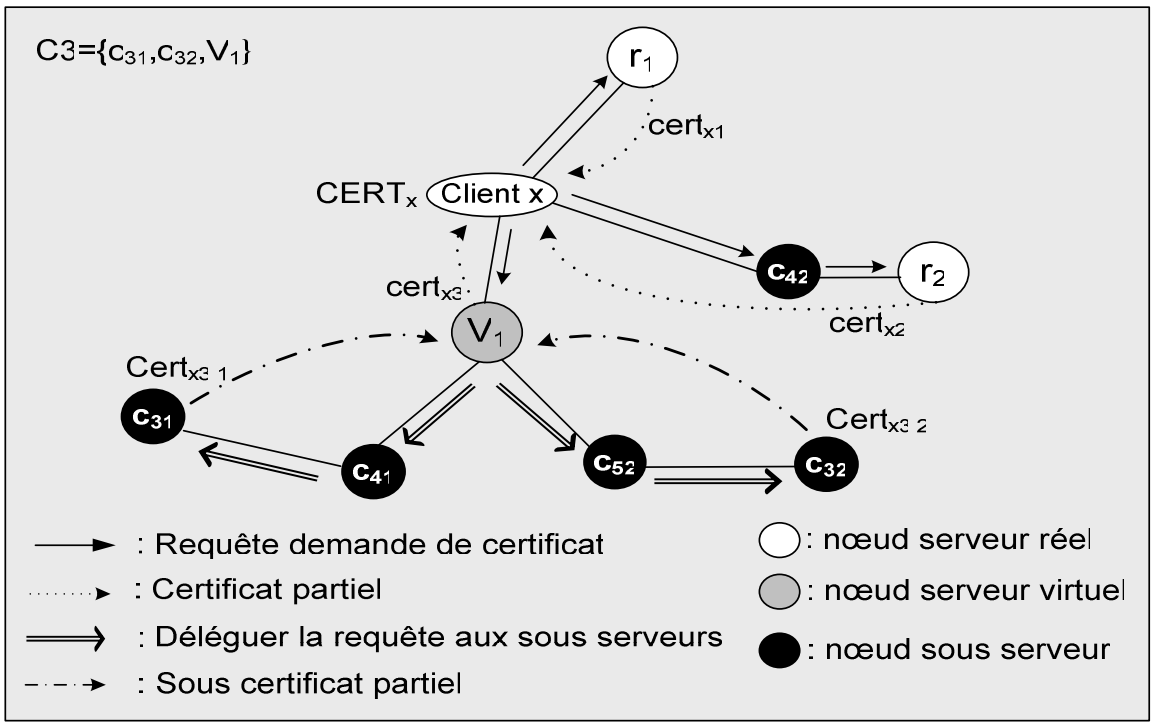


Fig.4.7. Attribution à seuil des certificats

4.4.4.1. Opération Log-On *(nouveaux nœuds dans le réseau)*

Cette opération est le moyen par lequel un nouveau nœud rejoint le réseau et devient un participant correct du service de certification. Le nouveau nœud peut se faire attribuer un rôle selon un mécanisme dynamique, il sera admit comme :

- nœud serveur réel et demander sa part de clé.
- rejoindre une classe particulière de sous serveurs.
- ou initier une nouvelle classe.

Un rôle sera attribué au nouveau nœud selon l'état actuel du système qui est défini par plusieurs critères (la densité des nœuds serveurs réels respectivement virtuels, la taille des classes, etc.) en rapport avec certains paramètres de performance (latence, disponibilité, sécurité, etc.). Par exemple, nous ne pouvons pas rajouter des serveurs réels si le *NAC* dépasse le seuil du niveau de sécurité requis, sachant que le

nombre de nœuds serveurs réels ne doit pas en aucun cas dépasser *n1* pour des raisons de sécurité. L'algorithme 2 (*log-on*) décrit comment attribuer un rôle particulier à un nouveau nœud *i* selon la configuration actuelle du cryptosystème à seuil. Selon cet algorithme, un nouveau nœud *i* vérifie d'abord si le *NAC* ne va pas affecter le niveau de sécurité requis en ajoutant un autre serveur réel, dans ce cas *i* sera admis comme nœud serveur réel et va demander sa part de clé (minimiser la latence), sinon *i* doit vérifier si la disponibilité au niveau de la reconstruction de la part de clé dans chaque classe est satisfaite selon le schéma à seuil (*m, k*). Ainsi, *i* diffuse une requête *ClassStatusReq* pour savoir quelle est la classe qui n'a pas suffisamment de nœuds, en recevant cette requête chaque nœud serveur virtuel envoie au nœud *i* une réponse (*ClassStatusRep*) qui inclut l'identifiant et la taille actuelle de sa classe. Si la classe de plus petite taille est inférieure au seuil *k* le nœud *i* va la rejoindre, sinon si cette classe est encore inférieure à *m* et qu'il y a suffisamment de nœuds serveurs réels et de classes pour attribuer un certificat alors *i* va enrichir cette classe (augmenter la disponibilité). Sinon le nœud va initier une nouvelle classe et devient le nœud serveur virtuel de cette classe (augmenter la sécurité). Ce processus est illustré par l'organigramme dans la figure 4.8.
Après un certain nombre de nœuds rejoignant le réseau, les paramètres du système peuvent changer comme suit :

$(n, t) \rightarrow (n', t) \;/\; n' > n$ (ajout d'un détenteur de part, serveur réel ou virtuel)
$V \rightarrow V' \;/\; |V'| > |V|$ (ajout d'un nœud serveur virtuel)
$C \rightarrow C' \;/\; |C'| > |C|$ et $C = \cup C_i$ (ajout d'une nouvelle classe de partage)

Quand un nouveau nœud est admis comme serveur réel, il doit demander sa part de clé auprès d'un groupe de *t* serveurs, ceux-ci peuvent coopérer pour générer une nouvelle part de clé et l'envoyer au nouveau nœud, et ce, sans l'intervention d'aucune autre partie de confiance, le processus de création d'une nouvelle part de *i* est décrit par la procédure getshare(*i*) comme suit :

Procédure *getshare(i)*

- *i* envoie une requête *getShareReq* à un groupe de *t* nœuds serveurs
  $i \rightarrow j$: (*getShareReq, CERTi, IDi*), $j \in R \cup V$
  sans perte de généralité soit $j = 1,\ldots,t$

- chaque serveur *j* recevant la requête et décide de la servir envoie à *i* ce-ci :
  $P_{ji} = f(j) l_j(i) + \psi_j \pmod q$ où : ψ_j est un facteur d'aveuglement telle que $\sum_{j=1}^{t} \psi_j = 0$.

- *i* calcule sa part de clé comme suit : $SK_i = \sum_{j=1}^{t} P_{ji}$

- si j est un nœud serveur virtuel alors, il fait passer la requête à un ensemble de k sous serveurs de sa classe, sans perte de généralité soit à c_{jl} pour tout $l = 1,\ldots,k$. Ensuite chaque c_{jl} retourne à j ce-ci : $g_j(l).l_{cjl}(0).l_j(i) + \psi_{jl}$ où $\sum_{l=1}^{k} \psi_{jl} = \psi_j$
- alors j envoie à i ce-ci $P_{ji} = \sum_{l=1}^{k} g_j(l).l_{cjl}(0).l_j(i) + \psi_{jl} \pmod q$

Algorithme 2 : *Log-on*

Constant
i : the new node ; ***k*** : class threshold ; ***t*** : global threshold ; ***m*** : number of class shareholders according to the threshold sheme (*m, k*);
var init
min_Size = m ;

for each New Node *i* **do**

```
if NAC < required_security_value then
    getShare (i) ;
    R = R ∪ i ;
else
    Broadcast (ClassStatusReq) ;
    While Waiting_time > 0 do
        if recieve(ClassStatusRep) then
            if ClassStatusRep.Size < min_Size then
                min_Size = ClassStatusRep.Size ;
                ClassID = ClassStatusRep.ClassID ;
            end if
        end if
    end While
    if min_Size < k then
        joinClass (ClassID) ;
    else if min_Size < m and |R ∪ V| ≥ t then
            joinClass(ClassID) ;
    else
        createClass(Ci) ;
        Ci = Ci ∪ i ;
        V = V ∪ i ;
    end if
end if
```

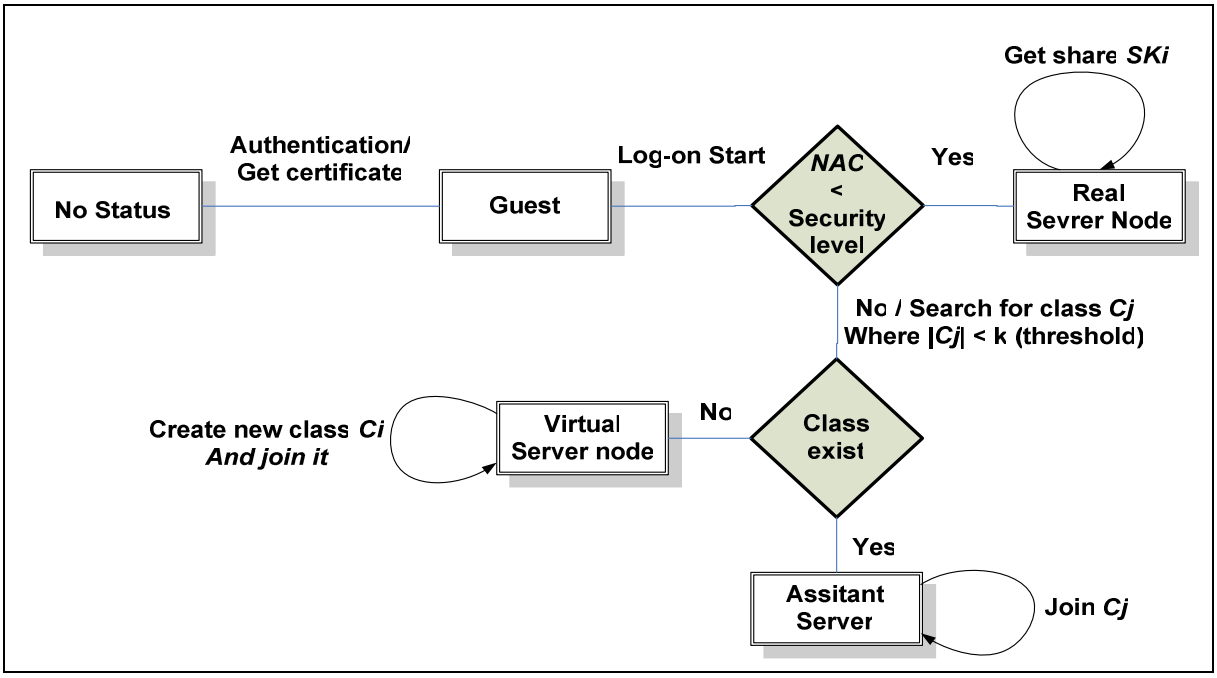


Fig.4.8. États possibles d'un nouveau nœud durant le processus Log-on

4.4.4.2. Opération leave (Quitter le réseau)

Dans SRKM, quand un nœud i quitte le réseau, cela provoque également un changement de la configuration du système, on distingue alors différents cas :

***i* ∈ *R* :**
Si i est un nœud serveur réel, il sera tout simplement enlevé de l'ensemble des serveurs réels $R = R - i$ et la configuration à seuil (n, t) devient (n', t) tel que $n' < n$. chaque nœud réel qui parte diminue le NAC ce qui influe sur le paramètre latence.

***i* ∈ *V* :**
Si i est un nœud serveur virtuel, un autre nœud sous serveur de la même classe que i sera élu pour remplacer le serveur virtuel, le nœud favorisé est celui qui a la plus forte connectivité avec les autres membres de sa classe.

***i* ∈ C_i :**
Si i est un sous serveur, la configuration à seuil de classe (m, k) passe à (m', k) tel que $m' < m$.

4.4.4.3 Changement dynamique de la topologie (Mobilité des nœuds)

Comme nous l'avons déjà mentionné, les réseaux ad hoc sont caractérisés par une topologie dynamique due au mouvement dynamique des nœuds. Ainsi, un nœud qui appartient à une classe particulière peut se déplacer à travers le réseau et s'éloigner de sa classe jusqu'à ce qu'il devient injoignable par le nœud serveur virtuel de cette classe, si plusieurs nœuds de la même classe font la même chose, la classe risque d'échouer à reconstruire sa part de clé quand elle est sollicitée. Afin d'améliorer la disponibilité au niveau des classes avec une mobilité forte des nœuds, nous proposons que si un nœud en déplacement qui détecte qu'il est déconnecté de sa classe doit changer la classe à la quelle il appartient (déplacer d'une classe à l'autre). Ainsi, ce nœud essaye d'abord de trouver une classe joignable, cela est similaire à une opération de Log-on décrite précédemment sauf qu'ici le nœud cherche uniquement à rejoindre une classe et il ne changera pas de rôle (il reste sous serveur), si le nœud réussit à trouver une classe il demande une sous part pour la nouvelle classe et supprime l'ancienne sous part. de cette manière les classes sont équilibrées au fur et à mesure que les nœuds de différentes classes se déplacent.

4.5. Discussion

Dans les approches classiques de cryptographie à seuil (n, t) complètement distribuées (tous les nœuds du réseau sont des possesseurs de parts de clé), il suffit pour un adversaire de compromettre n'importe quel ensemble de t nœuds dans une durée T pour qu'il puisse reconstruire la clé privée du service. Cependant, avec notre schéma, il faut, au meilleur cas $n_1+k(t-n_1)$ nœuds pour pouvoir trouver la clé sachant que $n_1+k(t-n_1) > t$. En outre, un adversaire est mis au défis non seulement de compromettre ce nombre de nœuds dans une durée T, mais aussi il faut que chaque k nœuds dans les $n_1+k(t-n_1)$ doivent être dans la même classe du moment que l'adversaire ne peut pas combiner des parts et des sous parts de différentes classes de partages.

Une autre limite que nous pouvons trouver dans les approches classiques de cryptographie à seuil (n, t) complètement distribuées, Si on considère dans un réseau de taille N qu'un adversaire du modèle 2 (section **4.2.2.2**) est incapable de compromettre t nœuds dans chaque intervalle T où T est la fenêtre de vulnérabilité, Néanmoins, il pourra atteindre t nœuds pondant $T' > T$ ce qui ne lui permet pas de trouver la clé parce qu'il ne peut pas combiner les parts de l'époque T avec celles de l'époque T'. Cependant il existe m telle que pour un réseau de taille $N + m$ on aura $T \geq T'$ De cette manière, la fenêtre de vulnérabilité croit avec le nombre de possesseurs de parts de clé ce qui donne plus de chances à un adversaire mobile de compromettre plus de t nœuds.

Nous définissons une métrique de vulnérabilité *Vb(x)* qui est la probabilité que la clé privée (*SK*) du service soit révélée lorsque *x* nœuds sont compromis.

En comparant notre schéma avec le schéma complètement distribué de *Kong* [2] et le schéma partiellement distribué de *Zhou* [1] on trouve que notre schéma peut résister à 100 % contre les attaques de l'adversaire du modèle 2 et 3 et avec une probabilité *Vb(x)* dans le modèle d'adversaire 4 où le nombre de nœuds compromis peut dépasser le seuil *t*. Le tableau 4.3 présente le niveau de vulnérabilité des différents schémas de cryptographie à seuil par rapport à notre proposition, ce tableau montre que les schémas [1] et [2] ne peuvent résister qu'aux modèles d'adversaire 1 et 2, alors que le notre peut résister à tous les modèles d'adversaire présentés précédemment (1, 2, 3 et 4).

Nœuds compromis	**Zhou *Vb(x)***	**Kong *Vb(x)***	**SRKM *Vb(x)***
$x < t$	0	0	0
$x = t$	1	1	0
$t < x < n_1+k(t-n_1)$	1	1	0
$x \geq n_1+k(t-n_1)$	1	1	]0,1]

Tab.4.3. Vulnérabilité des schémas à base de cryptographie à seuil Vs SRKM

4.6. Conclusion

Sécuriser les réseaux ad hoc est notoirement difficile, notamment en raison de l'absence d'une infrastructure en ligne. En particulier, la gestion de clé est un problème qui a été abordée par de nombreux chercheurs, mais avec des résultats limités. Dans ce chapitre, nous avons présenté une solution de gestion de clés pour les réseaux ad hoc. Notre schéma est basé sur la cryptographie à seuil, cette technique parait une bonne solution au problème de déploiement d'une PKI dans un environnement ad hoc. Plusieurs schémas utilisant cette technique ont été proposés pour fournir un service distribué de gestion de clé dans les réseaux ad hoc, la limite de ces schémas est qu'ils tolèrent jusqu'à un certain seuil *t*-1 de nœuds compromis. Notre solution vise à améliorer l'aspect robustesse et sécurité contre les nœuds compromis qui essayent de révéler la clé privée du service. Notre schéma fonctionne en présence de nœuds compromis et la confidentialité de la clé est preservée même si le nombre de nœuds compromis dépasse le seuil de vulnérabilité *t-1* sous une certaine probabilité qui reste très limitée, notre schéma s'adapte au passage à l'échelle de réseau, et il est renforcé par la technique de rafraîchissement de parts.

Chapitre 5

Simulation et mesures de performances

5.1. Introduction

Ce chapitre est consacré à l'étude des performances de notre architecture de gestion de clé (SRKM). Ainsi, nous avons mené une série de simulations en implémentant un prototype de simulation sous l'environnement de développement MATLAB. La simulation réalisée vise un double objectifs : 1) évaluer l'efficacité de notre schéma et démontrer qu'il est bien approprié aux MANETs, 2) Tester si le schéma proposé fournit une sécurité et robustesse effective dans un processus de certification en présence de nœuds compromis.
Les scénarios de mobilité des nœuds sont générés en se basant sur le modèle de mobilité « Random Waypoint ». Dans notre cas nous avons varié la vitesse des nœuds et les temps de pause.
Pondant la simulation nous nous somme intéressé aux métriques suivante : *le taux de succès de certification*, *le délai moyen de certification* et *la vulnérabilité* du système de certification contre les nœuds compromis.
Dans la suite de ce chapitre nous commençons d'abord par une brève description des paramètres de simulation et du modèle de mobilité utilisé, ensuite nous présentons en détail les résultats de notre simulation.

5.2 Paramètres de simulation

Les scénarios de simulation ont été implémentés avec l'environnement de développement MATLAB, la simulation a été réalisée avec des réseaux MANETs de dimensions allant de 100 à 200 noeuds, qui sont dispersés aléatoirement dans une région de 1 km x 1 km et chaque noeud a une portée de transmission noté (σ).

Nous considérons que deux nœuds sont voisins si la distance entre eux ne dépasse pas la portée de transmission. Nous supposons que les noeuds ont les mêmes caractéristiques physiques et capacités de traitement, et sont doté d'une interface de communication sans fil avec un débit de *22 Mbps*, par contre, des restrictions sur la bande passante et les erreurs dans le canal sans fil ne sont pas pris en considération.

Les simulations sont basées sur des scénarios événementiels, et les requêtes de certification arrivent selon une loi de poisson avec un inter arrivé de 10 secondes, et la durée de simulation est de 900 secondes.

5.2.1. Modèle de mobilité

Dans notre simulation, les nœuds du réseau ad hoc se déplacent selon le modèle de mobilité *Random Waypoint* [66]. Le modèle Random Waypoint divise le mouvement d'un nœud en deux phases. Tout d'abord, un nœud attend pour une période de temps aléatoire (temps de pause), Dans la deuxième phase, le participant choisit une destination aléatoire à l'intérieur de la zone de la simulation et se déplace vers cette destination à une vitesse choisie de façon aléatoire.

Si le nœud a atteint cette destination, le processus recommence. Le temps de pause, la destination à l'intérieur de la zone de simulation, et la vitesse des nœuds sont choisis avec une répartition uniforme. La Fig. 6.1 illustre un exemple de la trace d'un nœud mobile qui se déplace selon le modèle Random Waypoint dans un carré de 1000m × 1000m.

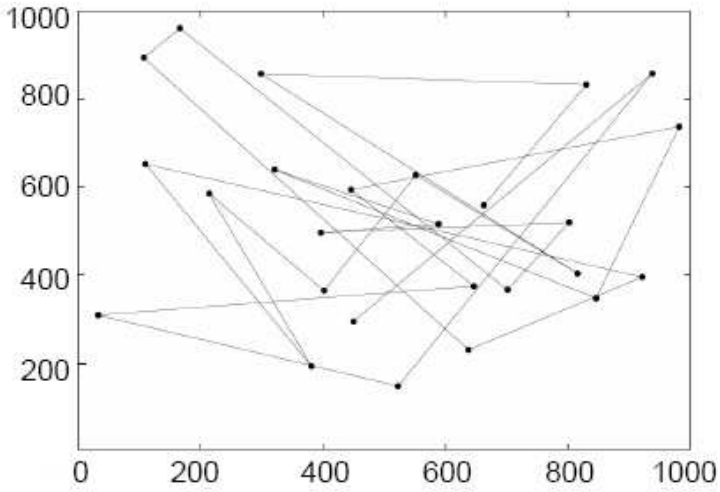


Fig. 6.1. Mouvement d'un nœud selon le modèle de mobilité Random Waypoint

Le modèle waypoint est largement utilisé pour l'évaluation des performances des différents aspects des réseaux ad hoc. Par conséquent, nous considérons ce modèle de mobilité dans notre évaluation, le temps de pause est compris entre 5s et 20s et la vitesse des nœuds est uniformément répartie [0m/s, 20m/s].

5.3. Résultats de simulation

5.3.1. Le taux de succès de certification

Cela mesure le ratio du nombre certificats délivrés avec succès (création ou renouvellement de certificats) au nombre total des requêtes de certification. On note μ, le taux de succès de certification, donc, sa valeur peut être calculée comme suit :

$$\mu = \frac{NCscss}{NCtot}$$

Ici, *NCscss* est le nombre de certificats délivrés avec succès, tandis que *NCtot* est le nombre totale de requêtes de certification. Cette métrique permet d'évaluer la capacité de notre schéma (SRKM) à fournir un service de certification à clé publique plus efficace et hautement disponible en comparant avec les autres travaux.

5.3.1.1. Impact du partitionnement du réseau ad hoc

Nous allons d'abord examiner l'impact de la connexité du réseau sur les performances du service de certification. La connexité du réseau dépend de la portée de transmission (σ) des nœuds qui constituent le réseau ad hoc. En effet, si σ est grand, la connexité du réseau devient alors plus forte.

La connectivité des nœuds est une condition nécessaire dans les schémas de gestion de clé basé sur la cryptographie à seuil, et peut avoir un impact négatif sur la disponibilité du service de certification. En effet, dans certaines situations, le service de certification peut échouer à délivrer un certificat à un nœud *i* pour la simple raison que le service, au moment de l'arrivée d'une requête de certification, ne peut pas former une coalition de *t* serveurs qui soient tous joignable par le nœud *i* pour pouvoir combiner leurs certificats partiels.

Ainsi, nous effectuons des mesures de l'impact de σ sur le taux de succès de certification (μ), cette évaluation est effectuée sur un réseau de taille N = 100 nœuds, la configuration du schéma à seuil est fixée à t = 10, n=20 et NAC = 0.81.
Avec les mêmes paramètres de simulation, nous comparons notre schéma (SRKM) avec le schéma complètement distribué de *Kong* [2]. La figure 5.2 montre que le taux de succès μ dans les deux schémas diminue lorsque la portée de transmission des noeuds se réduit. Cependant, on trouve que dans notre schéma SRKM, la disponibilité du service de certification est relativement stable. En effet, comme c'est montré dans la figure 5.2 nous avons plus que 65% des certificats sont délivrés avec succès pour une portée de transmission σ = 100m ou plus, rappelons que les nœuds ont les mêmes caractéristiques physiques. Tandis que dans le schéma de *Kong*, la disponibilité du service de certification est plus faible lorsque σ est moins de 160m.

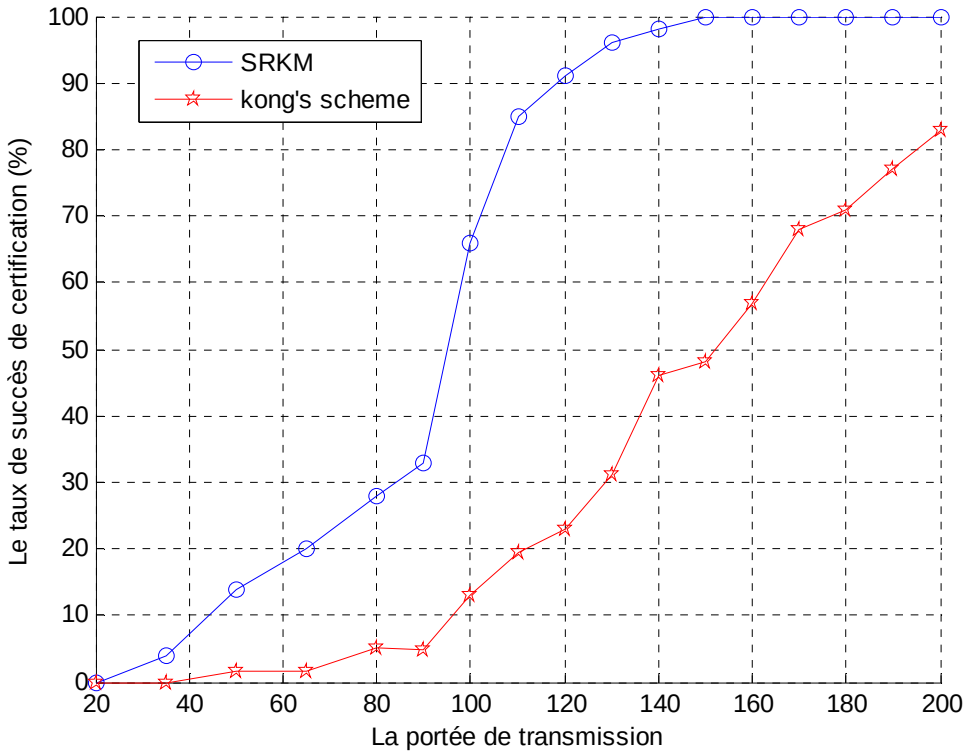


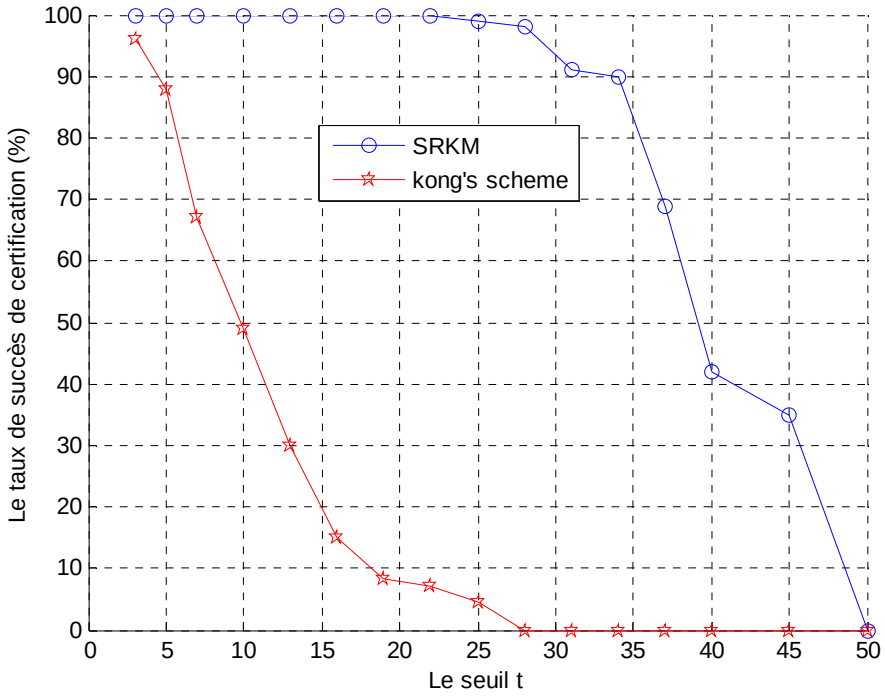
Fig. 5.2. Impact de la portée de transmission sur le taux de succès de certification, N=100, t=10, n=20

5.3.1.2 Impact du seuil (t)

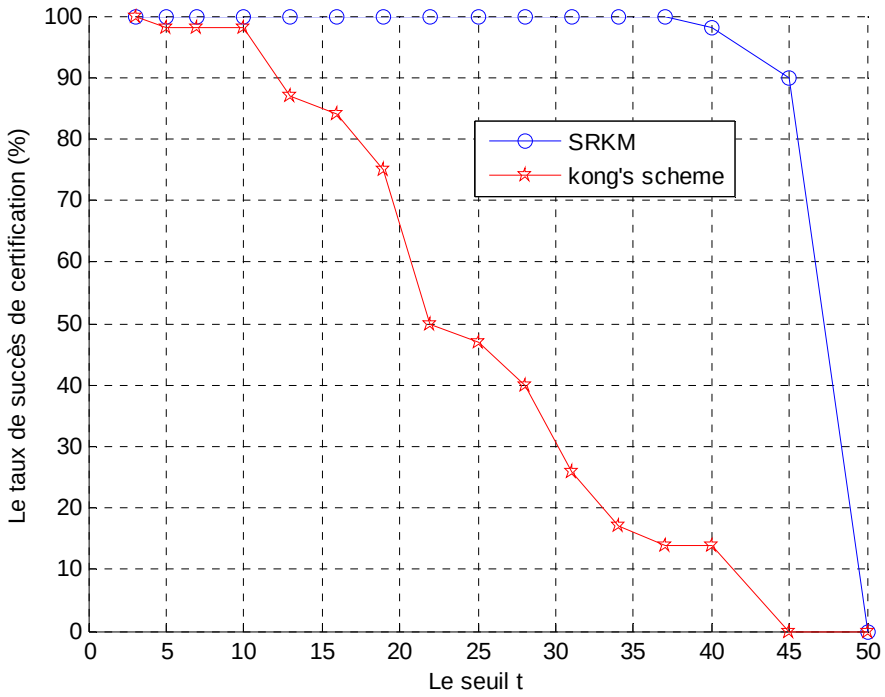
La valeur du seuil *t* peut avoir un grand effet sur les deux aspects les plus importants du service de certification qui sont la sécurité et la disponibilité. La sécurité assure que le service de certification est capable de se protéger contre les nœuds malicieux qui compromettent progressivement les nœuds corrects qui forment le service dans le but de combiner leurs parts de clé et ainsi révéler la clé privée du service. La disponibilité assure qu'il y a toujours suffisamment de parts conformément au seuil *t*, pour satisfaire une requête de certification.

Un seuil très élevé assure une plus grande sécurité, mais la disponibilité et le temps de latence pourraient ne pas être satisfaits, car un nœud qui sollicite le service pour un certificat doit collecter un grand nombre de certificats partiels valides, et si plusieurs serveurs sont compromis ou indisponibles, alors ce nœud peut échouer d'avoir un certificat valide. Si le seuil est abaissé, il devient facile pour un noeud à construire son certificat dans le niveau de QoS requis ou le délai d'authentification spécifié, mais l'aspect sécurité est compromis.

Nous nous sommes intéressé à étudier l'impact du seuil *t* sur le taux de succès de certification, nous comparons notre schéma SRKM et celui de *kong* [2]. Nous avons fait varié la valeur du seuil *t* en observant le taux de succès μ. Comme c'est montré dans la figure 5.3, l'augmentation du seuil n'a pas une grande influence sur μ avec notre schéma SRKM en comparant avec le schéma de *kong* dans lequel μ devient plus faible lorsque *t* dépasse la valeur 10. Le choix du seuil est influencé par divers facteurs tels que la taille du réseau, la vitesse des nœuds, la portée de transmission des nœuds, etc. D'après les figures 5.3 (a) et 5.3 (b), dans SRKM avec t=45, le taux de succès μ s'agrandit de 35% à 90% avec une porté de transmission de 100m et 180m respectivement. Ainsi, dans le schéma SRKM, on peut accroître le seuil pour augmenter la sécurité sans affecter la disponibilité.



(a)



(b)

Fig.5.3. Impact de t sur le taux de succès de certification (a) N = 200, n = 50, σ = 100m (b) N = 200, n = 50, σ =180m

5.3.1.3 Le taux de succès de certification en présence des nœuds compromis

Maintenant, nous nous intéressons à l'évaluation de la robustesse de notre schéma avec la présence des nœuds compromis dans le réseau. Dans le cadre de cette expérience, nous notons qu'un noeud compromis se caractérise par un comportement imprévisible. Ainsi, si les requêtes de certification sont émises à un nœud compromis, ce-ci a la possibilité de : (1) ne pas répondre aux requêtes ou (2) signer de faux certificats ou (3) de répondre correctement à une requête. Nous imitons les nœuds compromis en prenant aléatoirement un certain pourcentage des nœuds du réseau et les programmer pour agir selon (1), (2) ou (3).

La figure 5.4 montre que les performances de SRKM ne se dégradent pas rapidement lorsque le nombre de nœuds compromis dans le réseau augmente et le taux de succès de certification reste approximativement stable jusqu'à 50% de nœuds compromis dans le réseau pour lequel μ=80%. Ainsi, la robustesse est assez élevée lorsque il y a de plus en plus des nœuds compromis dans le réseau, et ainsi la disponibilité du service de certification est améliorée.

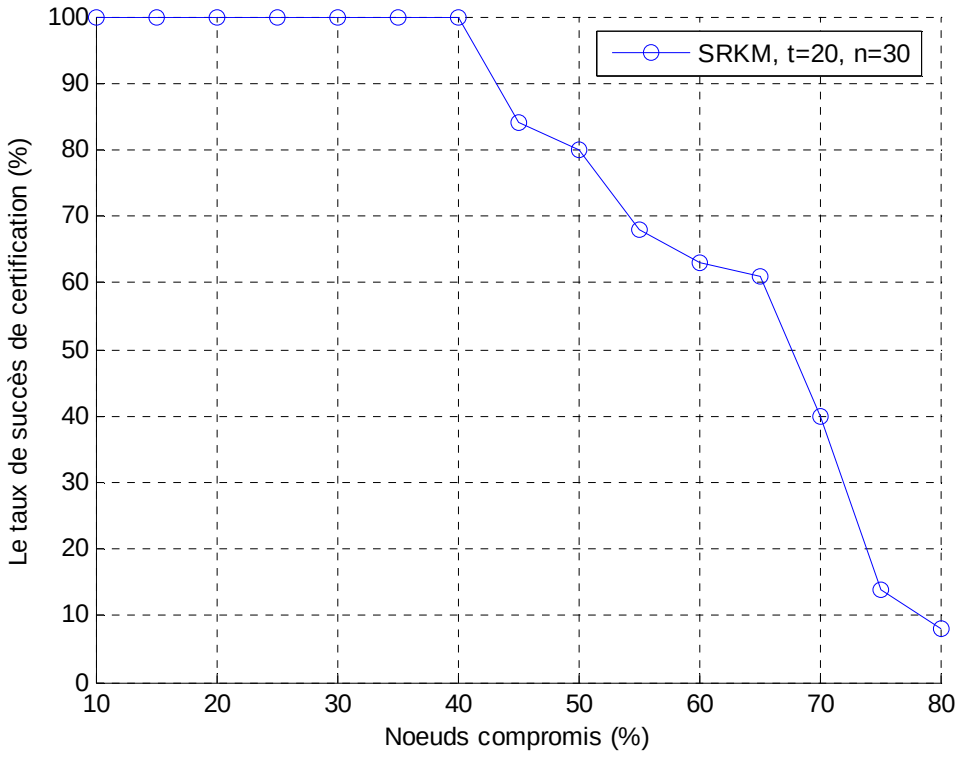


Fig. 5.4. Le taux de succès de certification Vs. La présence des nœuds compromis, N = 150,σ = 150m

5.3.2. Le Délai Moyen de Certification (DMC) Vs. Le NAC

Dans cette expérience, nous avons paramétré le réseau à la configuration n=30, N=150, σ=150m. Nous rappelons que dans le schéma SRKM les n nœuds serveurs sont divisés en deux groupes ; nœuds serveurs réels et nœuds serveurs virtuels, la répartition de ces nœuds (nombre des nœuds réels Vs. Nœuds virtuels) est déterminée par le paramètre *NAC* qui est le ratio des nœuds serveurs réels au nombre des nœuds virtuels par rapport à n. Ainsi, la valeur du paramètre *NAC* est très importante dans les performances de SRKM. Dans les applications qui imposent des contraintes sur le délai d'authentification et la bande passante, la valeur du *NAC* doit être élevée, cela signifie que lorsque il y a un grand nombre de serveurs réels dans le réseau, réduit les délais du service de certification (temps de réponse), car un nœud serveur réel n'aura pas besoin de la collaboration d'autres nœuds pour délivrer un certificat partiel, par contre un serveur virtuel doit collecter *k-1* certificats partiels pour pouvoir construire un certificat partiel valide. Maintenant, si on veut augmenter la sécurité du service de certification, le *NAC* doit être plus petit. Ainsi, si la plupart des serveurs sont virtuels, alors il sera plus difficile pour un adversaire mobile de découvrir la clé privée du service en ayant t ou plus de parts des nœuds compromis.
On constate alors, que pour maximiser la sécurité revient à minimiser le *NAC*, tandis que pour réduire le coût de communication il faut maximiser le *NAC*.

Dans cette expérience, la métrique de vulnérabilité *Vb(x)* (définie dans le chapitre précédent) est utilisée pour évaluer l'aspect sécurité, le nombre de nœuds compromis (x) est fixé à $x=k.t$, dans ce cas la probabilité que la clé privée du service soit révélée lorsque x nœuds sont compromis est calculée ainsi :

$$Vb(x) = \sum_{i=0}^{n_1} \frac{C_{n_1}^{i} C_{m}^{k} (t-i)}{C_{N}^{x}}$$

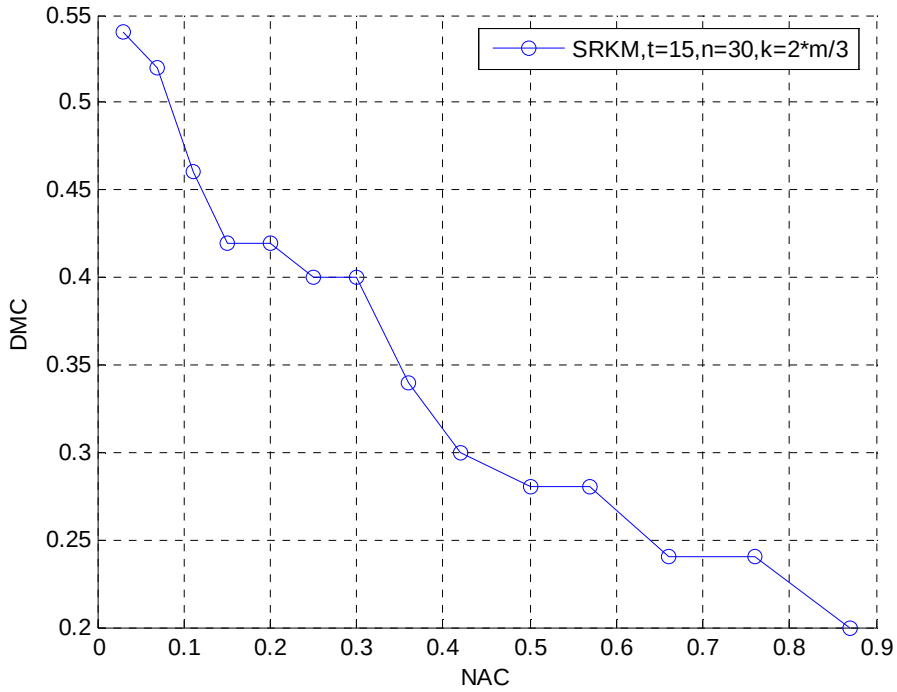


Fig. 5.5. Le Délai Moyen de Certification (DMC) Vs. Le NAC, N=150, σ =150

La figure 5.5 montre que lorsque le *NAC* est de plus en plus grand, le DMC se réduit. Cependant, un grand *NAC* ne convient pas aux conditions de sécurité, comme le montre la figure 5.6 *Vb*(*x*) augmente lorsque le *NAC* augmente. Ainsi, on fait varier le *NAC* en augmentant sa valeur, le DMC diminue mais *Vb*(*x*) augmente, par exemple pour *NAC* = 0.1, un certificat peut être délivré dans 0.5 secondes en moyenne, mais *Vb*(x)=6.19 x 10^{-28} tandis que si on augmente le *NAC* à 0.87, le DMC s'abaisse à 0.2 secondes mais *Vb*(x) s'augmente à 4.6 x 10^{-5} .

Par conséquent, on constate que le choix du *NAC* doit déterminer l'optimale (sécurité, latence)

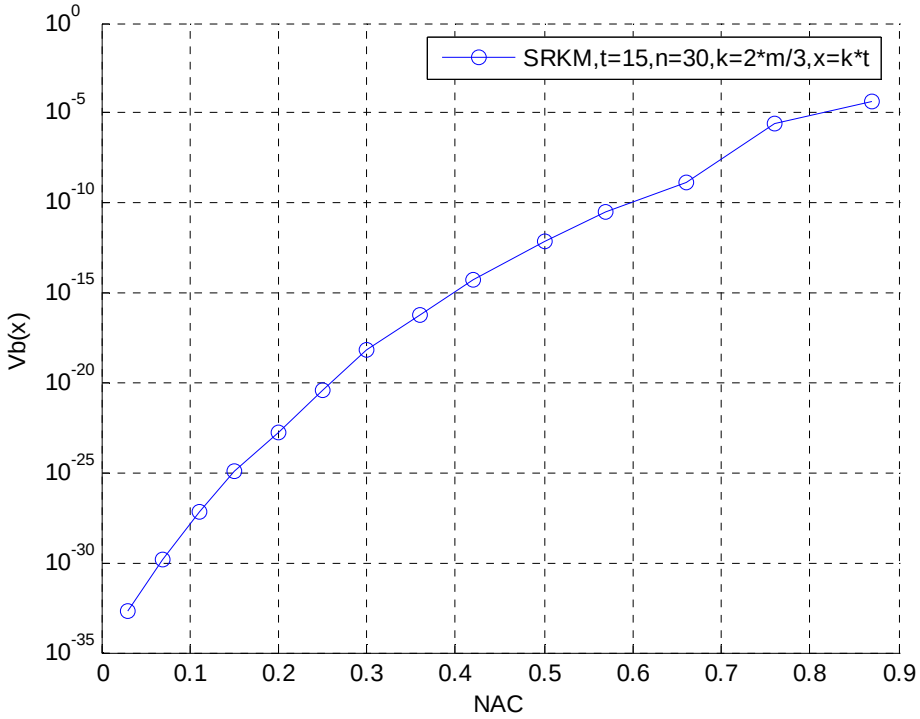


Fig. 5.6. Vb(x) Vs. Le NAC, N=150, σ =150, x=k.t

5.3.3. Impact de la mobilité

Nous nous intéressons dans cette partie de simulation à examiner l'efficacité de notre schéma vis-à-vis du changement de la topologie dû à la mobilité des nœuds et montrer que le service de certification est disponible quelque soit la topologie du réseau et quelque soit la fréquence de son changement. Dans le cadre de cette expérience, on suppose que les nœuds ont les mêmes caractéristiques physiques et qu'ils se déplacent à la même vitesse. Le mouvement des nœuds repose sur le modèle *Random Waypoint* tel que le temps de pause des nœuds est varié entre 5s, 10s, 15s et 20s.

Dans ce contexte, nous avons variée la vitesse des nœuds de 1*m/s* jusqu'à 20*m/s* pour mesurer l'impact de la mobilité sur le taux de succès de certification ainsi que le délai moyen de certification, plus on augmente la vitesse, les nœuds suivront une forte mobilité.

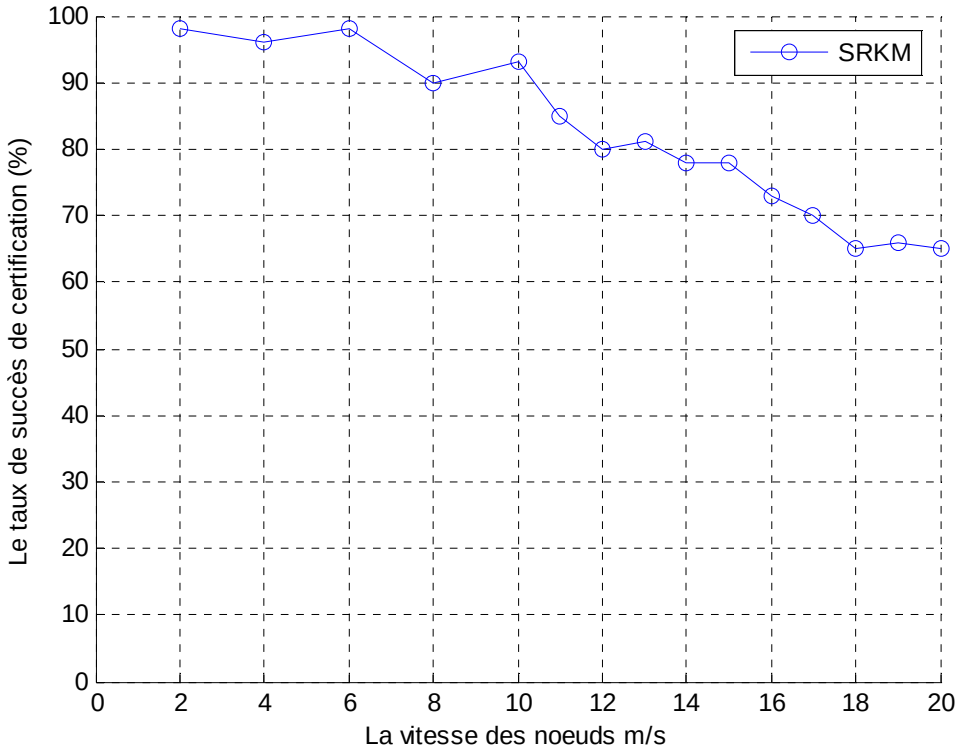


Fig.5.7. Impact de la mobilité sur le taux de succès de certification, *N=100, σ=150, n=30, t=20.*

Les figures 5.7 montre que le taux de succès de certification est relativement stable lorsque on augmente la vitesse des nœuds. C'est la même chose pour le DMC qui varie entre 0.37 s et 0.48 s comme c'est montré dans la figure 5.8. Cela confirme l'efficacité de notre schéma et l'adaptabilité à un réseau ad hoc dynamique.

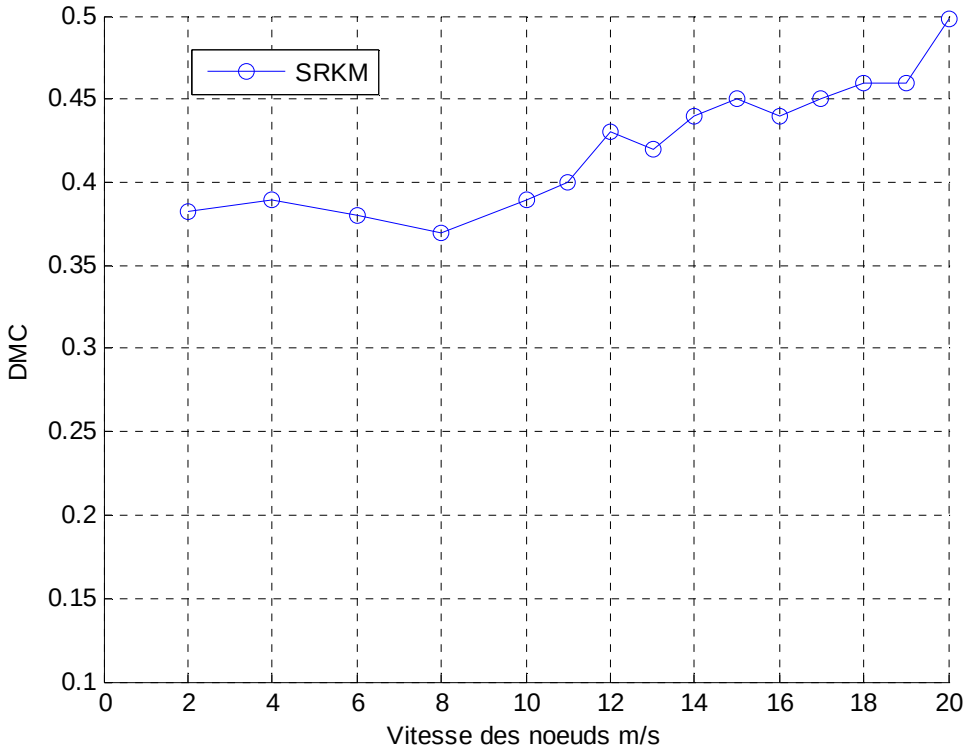


Fig.5.8. Le DMC Vs. Vitesse des nœuds, *N=100, σ =150, n=30, t=20.*

5.4. Conclusion

Dans ce chapitre, nous avons effectué quelques mesures de performances pour évaluer l'efficacité et la sécurité du schéma proposé (SRKM), nous avons réalisé notre simulation en utilisant l'environnement MATLAB, durant la simulation nous sommes basé sur des métriques de base tels que le taux de succès de certification, le délais moyen de certification et la sécurité de la gestion de clés. Les résultas obtenus montrent que le schéma proposé s'adapte bien aux réseaux ad hoc à large échelle en comparant avec les autres schémas à seuil. D'après la simulation, nous constatons que certains paramètres tels que le seuil *t* et le *NAC* doivent être soigneusement choisis et peuvent avoir un impact important sur les performances du schéma de gestion de clés SRKM. Nous avons aussi montré à travers cette simulation la sécurité et la robustesse de notre schéma en présences des nœuds compromis.

Conclusion générale et perspectives

Fournir une communication sécurisée dans les futurs systèmes mobiles en particulier les réseaux ad hoc est un véritable challenge. En effet, un réseau ad hoc est un environnement hostile qui apporte plusieurs défis de sécurité, dus à des caractéristiques et spécificités propre à ce genre de technologie (absence d'administration centralisée, liens sans fils, routage spécifique, etc.). A des contraintes de sécurité générales présentes dans toute infrastructure réseautique, s'ajoutent de nouvelles vulnérabilités spécifiques à ce paradigme de réseau. Dans ce contexte, la gestion de clés représente l'élément primordial pour toute architecture de sécurité pour assurer la confidentialité, l'intégrité et l'authentification des communications. Les infrastructures à clés publiques (PKI) étaient toujours considérées comme une bonne solution pour accomplir cette tache. Cependant, l'absence d'infrastructure fixe et d'une autorité de confiance induit inévitablement de nouvelles vulnérabilités qu'il faut prendre en considération lors de la conception d'une architecture de gestion de clés dans les réseaux ad hoc.

Dans ce travail, nous avons dans un premier temps présenté les réseaux ad hoc et les défis majeurs inhérent à ce paradigme, pour lesquels la sécurité devient un véritable enjeu. Ensuite, nous avons étudié les principales solutions proposées pour sécuriser les réseaux ad hoc. Ces solutions identifient trois directions principales de recherche : Authentification, sécurité des protocoles de routage et la gestion de clés. Nous avons montré que les mécanismes de gestion de clés constituent un point sensible pour la sécurité des réseaux ad hoc, et de nombreux travaux de recherche sont proposés dans littérature afin de fournir des schémas efficaces de gestion de clés adapté aux réseaux ad hoc. Nous avons discuté ces travaux de recherche en les classant selon la technique et l'approche employée par le schéma pour la génération et la distribution des clés. Nous avons montré que la cryptographie à seuil est la solution la plus appropriée pour le déploiement d'une PKI au sein d'un réseau ad hoc. Toutefois, les solutions actuelles de gestions de clés exhibent encore des limites et ne s'adaptent pas au passage à l'échelle du réseau.

Ayant définit les motivations pour la conception d'un schéma de gestion de clés efficace et sécurisé pour un réseau ad hoc, nous avons proposé SRKM à cet effet. Un schéma de gestion de clés flexible et sécurisé afin de fournir un service de certification complètement distribué dans un environnement ad hoc. SRKM est basé

dans sa conception sur la cryptographie à seuil. L'objectif principal de notre proposition est d'améliorer la robustesse et la sécurité dans la gestion de clés contre les nœuds compromis qui peuvent par conspiration révéler la clé de signature du service de certification. SRKM est conçu pour pouvoir accomplir les opérations de certification d'une manière sécurisée en présence de nœuds compromis. Selon ce modèle, la clé de signature est partagée entre *n* nœuds appelés serveurs. La part de clé de certains serveurs est encore partagée par d'autres nœuds appelés sous-serveurs qui constituent des classes de partage. Outre sa flexibilité, ce schéma fournit une robustesse qui s'adapte à l'échelle du réseau. Nous avons montré à travers une simulation l'efficacité du schéma proposé. Les résultats obtenus démontrent que notre schéma s'adapte à la nature des réseaux ad hoc et fournit une sécurité aux opérations de gestion de clés.

Comme perspectives à notre travail, nous prévoyons dans un premier temps de compléter notre schéma en développant un mécanisme de révocation des certificats qui s'adaptent à l'architecture de notre schéma, et un mécanisme qui permet aux nœuds de détecter les nœuds compromis et de les isoler du service de certification. Nous envisageons aussi d'approfondir l'évaluation de performances pour étudier l'impact du schéma de gestion de clés sur la qualité de service QoS fournie par une application donnée.

Nous prévoyons également, d'étendre notre architecture de gestion de clés SRKM en combinant la technique de *cryptographie à seuil* et le *clustering*, dans ce modèle, un réseau est divisé en clusters. Les chefs de clusters ou *ClusterHead* (CH) constituent un réseau virtuel appelé réseau de clés où la clé privée du service de certification est partagée entre les CH qui forment une CA distribuée. La part de clé associée à un CH est partagée entre les membres du cluster. Un certificat est signé en contactant au moins *t* CH. Un CH construit son certificat partiel en contactant *k* membres de son cluster. La technique de *clustering* permet de faciliter l'organisation des nœuds et fournit une flexibilité au processus de certification pour bien s'adapter à un large réseau avec une topologie dynamique.

Nous nous sommes focalisé durant ce travail sur la gestion de clés dans les réseaux ad hoc. Ce thème de recherche se situe dans le cadre d'un domaine plus général qui est la sécurité dans les réseaux spontanés. Un réseau spontané permet à un ensemble de machines hautes d'être connectées rapidement et facilement avec un minimum d'infrastructure préalable, voire sans infrastructure, chaque nœud contribue activement à la vie du réseau. Les réseaux pair à pair (P2P) illustre ce concept. Nous prévoyons dans ce cadre d'adapter notre schéma SRKM aux réseaux pair à pair, en tenant compte de leurs particularités et leurs caractéristiques.

Références

[1] L. Zhou and Z. Haas. “Securing ad hoc networks”. IEEE Network,13(6):24–30, 1999.

[2] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. “Providing robust and ubiquitous security support for mobile ad hoc networks”. In Proceedings of the 9th International Conference on Network Protocols (ICNP), November 2001.

[3] H. Luo, J. Kong, P. Zerfos, S. Lu, and L. Zhang, “URSA: Ubiquitous and Robust Access Control for Mobile Ad Hoc Networks,” IEEE/ACM Transactions on Networking, pp. 1049–1063. December 2004

[4] H. Luo and S. Lu, “Ubiquitous and Robust Authentication Services for Ad Hoc Wireless Networks,” UCLA-CSD-TR-200030.

[5] Jean-Pierre Hubaux, Levente Butty and Srdan Capkun .”The Quest for Security in Mobile Ad Hoc Networks”. Proceedings of the 2nd ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), Long Beach, CA, 2001. pp. 146_155

[6] V.Legrand. Rapport de DEA. “Etablissement de la confiance et réseaux Ad Hoc”. Le Germe de confiance, EDIIS, Laboratoire CITI, INRIA ARES, 2003

[7] Srdjan Capkun, Jean-Pierre Hubaux, and Levente Buttyan. “Mobility Helps Security in Ad Hoc Networks“. MobiHoc’03, June 1–3, 2003, Annapolis, Maryland, USA, ACM 2003.

[8] H. Deng, A. Mukherjee, and D. Agrawal, “Threshold and Identity- Based Key Management and Authentication for Wireless Ad Hoc Networks,” Proc. Int’l Conf. Information Technology: Coding and Computing (ITCC ’04), Apr. 2004.

[9] J. Douceur. “The Sybil attack”. In Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS), 2002.

[10] L. Butty´an and J.-P. Hubaux(Eds). “Report on a Working Session on Security in Wireless Ad Hoc Networks”. Mobile Computing and Communications Review, 6(4), 2002.

[11] Adi Shamir. “How to share a secret”. Communications of the ACM, 22(11):612–613, 1979.

[12] A.J. Menzes , P.C. van Oorschot, and S.A. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, FL, 1997.

[13] Y. Desmedt. “Threshold cryptography”. European Transactions on Telecommunications, 5(4):449 457, July–August 1994.

[14] Dan Boneh, Matthew K.Franklin. “Identity based encryption from the weil pairing”. Advances in Cryptology- crypto 2001, Lecture Notes in Computer Science 2139, pp213-229, 2001

[15] Martin Gagné. “Identity-Based Encryption: a survey”. RSA Laboratories CryptoBytes, 6(1):10–19, 2003.

[16] Adi Shamir. “Identity-based cryptosystems and signature schemes”. In Proceedings of CRYPTO ’84 on Advances in Cryptology, pages 47–53, Santa Barbara, CA, USA, 1984. Springer-Verlag New York, Inc.

[17] A. Khalili, J. Katz, and W. Arbaugh, “Toward Secure Key Distribution in Truly Ad Hoc Networks,” Proc. IEEE Workshop Security and Assurance in Ad Hoc Networks, Jan.2003.

[18] Philip Zimmermann. “The Official PGP User’s Guide”. MIT Press, 1995. http://www.pgpi.org.

[19] Y. Zhang, W. Liu, W. Lou, Y. Fang, and Y. Kwon, “AC-PKI: Anonymous and Certificateless Public-Key Infrastructure for Mobile Ad Hoc Networks,” Proc. IEEE Int’l Conf. Comm, pp. 3515-3519, May 2005

[20] Tyler Moore, Jolyon Clulow, Shishir Nagaraja, and Ross Anderson. “New Strategies for Revocation in Ad-Hoc Networks”. F. Stajano et al. (Eds.): ESAS 2007, LNCS 4572, pp. 232–246, 2007

[21] Claude Crépeau and Carlton R. Davis. « A Certificate Revocation Scheme for Wireless Ad Hoc Networks”. Proceedings of the 1st ACM Workshop Security of Ad Hoc and Sensor Networks Fairfax, Virginia, 2003

[22] Yanchao Zhang, Wei Liu, Wenjing Lou and Yuguang Fang. “Securing Mobile Ad Hoc Networks with Certificateless Public Keys”. IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL.3, NO. 4, pp. 386-399, 2006

[23] H Krawczyk, M.Bellare, R.Canetti. “Hmac: Keyed-hashing for message authentication”. Feb 1997. RFC 2104

[24] Gabriel Montenegro, Claude Castelluccia. “Statically unique and cryptographically verifiable (sucv) identifiers and addresses”. In proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS 2002), California, USA, 2002

[25] Frank Stajano and Ross Anderson. “The Resurrecting Duckling:Security Issues in Ad-Hoc Wireless Networks”. In Proceedings of 3rd AT&T Software Symposium, Middletown, New Jersey, USA, Oct 1999.

[26] N. Asokan, P. Ginzboorg : “Key Agreement in Ad Hoc Networks”, Computer Communications, vol. 23(17), 2000, pp. 1627-1637.

[27] Whitfield Diffie and Martin E. Hellman. ”New Directions in Cryptography”, IEEE Transactions on Information Theory, IT-22, no. 6, pp. 644{654, 1976.

[28] P. Papadimitratos and Z. J. Hass. “Secure routing for mobile ad hoc networks”. In Proceedings of SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS), San Antonio, TX, January 2002

[29] Manel Guerrero Zapata, 0N.Asokan. “Securing ad-hoc routing protocols“. In proceedings of the 2002 ACM Workshop on Wireless Security (WiSe), pages 1-10, September 2002

[30] Manel Guerrero Zapata. “Secure ad hoc on-demand distance vector (SAODV) routing”. First published in the IETF MANET Mailing List, 2002

[31] Yih Chun Hu, Adrian Perrig and David B. Johnson. “Ariadne : A secure on-demand routing protocol for ad hoc networks”. Proceedings of The 8th ACM International Conference on Mobile Computing and Networking, 2002

[32] Kimaya Sanzgiri, Bridget Dahill, Brian Neil Levine, Clay Shields, Elizabeth M. Belding-Royer. “A secure routing protocol for ad hoc networks”. In proceedings of 2002 IEEE International Conference on Network Protocols (ICNP), Paris, 2002.

[33] Y.-C. Hu, D. B. Johnson, and A. Perrig. “Secure efficient distance vector routing in mobile wireless ad hoc networks”. In Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications
(WMCSA), June 2002.

[34] Daniele Raffo, Cedric Adjih, Thomas Clausen, Paul Muhlethaler. “An advanced signature system for olsr”. In proceedings of the 2nd ACM workshop on Security of ad hoc and sonsor networks (WiSe 2004). pp10-16, USA,2004

[35] L. Buttya´n and I. Vajda, “Towards Provable Security for Ad Hoc Routing Protocols,” in Proceedings of the 2nd ACM workshop on Security of Ad hoc and Sensor Networks, 2004, pp.94–105.

[36] Seung Yi, Prasad Naldurg, and Robin Kravets. “Security-aware ad-hoc routing for wireless networks”. In Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001), Long Beach, CA, USA, October 2001

[37] Stephen Carter, Alec Yasinsac. “Secure position aided ad hoc routing protocol”. In proceedings of the IASTED International Conference on Communications and Computer Networks (CCN02), 2002

[38] Y. Zhang and W. Lee. “Intrusion detection in wireless ad-hoc networks”. In Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom), 2000

[39] S. Marti, T.J. Giuli, K. Lai, M. Baker: “Mitigating routing misbehavior in mobile Ad Hoc networks”, Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking, August 2000.

[40] Farooq Anjum and Petros Mouchtaris. “SECURITY FOR WIRELESS AD HOC NETWORKS”. By John Wiley & Sons, Inc. 2007

[41] Jinshan Liu et Valérie Issarny. “Enhanced Reputation Mechanism for Mobile ad hoc Networks“. Second International Conference on Trust Management (iTrust'2004), 2004

[42] Michiardi P. Molva R. “Core : A COllaborative REputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks”. In Proceedings of IFIP Communications and Multimedia Security Conference (CMS), Portoroz, Slovenia. (2002)

[43] L. Eschenauer and V. Gligor, “A Key-management Scheme for Distributed Sensor Networks.” In Proceedings of the 9th ACM Conference on Computer and Communications Security, November 2002, pp. 41–47.

[44] H. Chan, A. Perrig, and D. Song, “Random Key Predistribution Schemes for Sensor Networks,” in IEEE Symposium on Security and Privacy, May 2003, pp. 197–213.

[45] Adrian Perrig, Ran Canetti, J.D. Tygar, Dawn Song. Efficient Authentication and Signing Multicasts Streams over Lossy Channels, IEEE Symposium on Security and Privacy, September 2002.

[46] Public-Key Infrastructure (X. 509), PKIX Working Group, The internet Eng. Task Force (IETF), http://www.ietf.org/html. charters/pkix-charter.html, 2005.

[47] D. Harkins and D. Carrel, “The Internet Key Exchange (IKE),” RFC 2409, November 1998; www.ietf.org/rfc/rfc2409.txt?number ¼ 2409.

[48] W. Aiello, S. M. Bellovin, M. Blaze, R. Canettia, J. Ioannidis, A. D. Keromytis, and O. Reingold, “Efficient, DoS-Resistant, Secure Key Exchange for Internet Protocols,” in Proc. ACM Computer and Communications Security Conference, Washington, DC, USA, 2000, pp. 48–58.

[49] B. Zhu et al., “Efficient and Robust Key Management for Large Mobile Ad Hoc Networks,” Computer Networks, vol. 48, no. 4, July 2005, pp.657–82.

[50] Atef Z. Ghalwash, Aliaa A. A. Youssif, Sherif M. Hashad and Robin Doss. Self Adjusted Security Architecture for Mobile Ad Hoc Networks (MANETs). 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007). 2007

[51] Edith C.H. Ngai and Michael R. Lyu. An Authentication Service Based on Trust and Clustering in Wireless Ad Hoc Networks: Description and Security Evaluation. Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC’06). 2006

[52] A. Shajin Nargunam and M.P Sebastian, A. Shajin Nargunam and M.P Sebastian. Cluster Based Security Scheme for Mobile Ad Hoc Networks.©2006 IEEE

[53] KEUN-HO LEE, SANG-BUM HAN, HEYI-SOOK SUH, CHONG-SUN HWANG AND SANGKEUN LEE. Authentication Protocol Using Threshold Certification in

Hierarchical-cluster-based Ad Hoc Networks. JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 23, 539-567 (2007)

[54] M. Puzar et al., “SKiMPy: A Simple Key Management Protocol for MANETs in Emergency and Rescue Operations,” Proc. ESAS ’05, 2005.

[55] Wenbo He,Ying Huang, Klara Nahrstedt, Whay C. Lee. SMOCK: A Selfcontained Public Key Management Scheme for Mission-critical Wireless Ad Hoc Networks. Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'07).2007

[56] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive Secret Sharing or: How to Cope with Perpetual Leakage. extended abstract, IBM T.J. Watson Research Center, November 1995.

[57] LIDONG ZHOU, FRED B. SCHNEIDER and ROBBERT VAN RENESSE. APSS: Proactive Secret Sharing in Asynchronous Systems. ACM Transactions on Information and System Security, Vol. 8, No. 3, 2005, Pages 259–286.

[58] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, \Proactive secret sharing," Extended abstract, 1995.

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

[60] Y. Frankel, P. Gemmell, P. MacKenzie, and M. Yung. Optimal Resilience Proactive Public-Key Cryptosystems. In FOCS, pages 384–393, 1997.

[61] P. Feldman, A practical scheme for non-interactive verifiable secret sharing, in: Proceedings of the 28th IEEE Symposium on the Foundations of Computer Science,1987, pp. 427–437.

[62] T. Pedersen, Non-interactive and information-theoretic secure verifiable secret sharing, in: Advances in Cryptology-Crypto_91, LNCS 576, 1992, pp. 129–140.

[63] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactionson Information Theory 31 (1985) 469–472.

[64] National Institute for Standards and Technology. Digital signature standard (DSS), 1998. FIPS 186-1.

[65] Chuan-Ming Li, Tzonelih Hwang and Narn-Yih Lee. Threshold Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders. Springer-Verlag. 1998

[66] Johnson DB, Maltz DA. Dynamic source routing in ad hoc wireless networks. Mobile Computing, Kluwer Academic Publishers, 1996; 153–181.

[67] Daniele Raffo, Security Schemes for the OLSR Protocol for Ad Hoc Networks. PhD thesis, University of Paris 6, 2005.

[68] PoWah Yau and Chris J. Mitchell. Security Vulnerabilities in Ad Hoc Networks. Research report, Mobile VCE Research Group Information Security Group Royal Holloway, University of London. 2004

[69] V. Gayraud, L. Nuaymi, F.Dupont, S. Gombault and B. Tharon. La sécurité dans les réseaux sans fil ad hoc. In Symposium sur la Sécurité des technologies de l'information et de la Communication SSTIC, Rennes France, June 2003.

[70] Wenjia Li and Anupam Joshi. Security Issues in Mobile Ad Hoc Networks - A Survey. Research report, Department of Computer Science and Electrical Engineering , University of Maryland, Baltimore County, 2007.

[71] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), San Francisco, CA, USA, April 2003.

[72] J. Kohl and C. Neuman, "The Kerberos Network Authentication Service (V5)", IETF RFC 1510, The Internet Society, September 1993

[73] ANNE MARIE HEGLAND, ELIWINJUM, STIG F. MJØLSNES, CHUNMING RONG, ØIVIND KURE, AND PÅL SPILLING. A SURVEY OF KEY MANAGEMENT IN AD HOC NETWORKS. IEEE Communications Surveys &Tutorials. 2006

[74] Raghani S, Toshniwal D, Joshi R. Dynamic support for distributed certification authority in mobile ad hoc networks. ICHIT '06: Proceedings of the 2006 International Conference on Hybrid Information Technology, IEEE Computer Society: Washington, DC, USA, 2006; 424–432,2006.

[75] Pietro RD, Mancini LV, Zanin G. Efficient and adaptive threshold signatures for ad hoc networks. Electron. Notes Theor. Comput. Sci. 2007;171(1):93–105,