LISTE DES FIGURES

LISTE DES TABLES

Table des matières

INTRODUCTION

Durant ces dernières années, nous observons une croissance exponentielle du déploiement des réseaux de transmission sans fil et mobiles. Dans cette classe de réseaux les réseaux mobiles ad hoc (MANETs) occupent une place de choix et sont l'objet d'un grand intérêt tant auprès des scientifiques des professionnels que des utilisateurs. Ce sont des réseaux composés uniquement de systèmes informatiques appelés noeuds qui peuvent communiquer de manière autonome grâce à des ondes radios. Les caractéristiques de base de ces réseaux sont le fait qu'ils ne disposent d'aucune infrastructure préexistante (aucune station de base ou serveur) et se forment de façon spontanée (n'importe où et n'importe quand). Afin de maintenir la connectivité dans un réseau ad hoc mobile, tous les noeuds participants doivent être en mesure d'intervenir dans le processus de routage du trafic. Cette coopération des noeuds ne doit pas être sous le contrôle d'une autorité centralisée d'administration puisqu' il n'en existe pas.

Les réseaux mobiles ad hoc offrent des avantages et une adaptabilité très utiles pour certaines applications dans le domaine militaire, celui du transport et dans d'autres applications tactiques comme les opérations de secours en l'occurrence les incendies, les tremblements de terre etc. Cependant ce type de réseaux est plus vulnérable aux attaques qu'un réseau filaire, en raison de l'utilisation des ondes électromagnétiques comme support de transmission. En conséquence les questions de sécurité se posent de manière plus aiguë dans les MANETs.

Les problèmes de sécurisation des transmissions se posent tant au niveau du routage qu'à celui du trafic de production, qui est la charge utile du réseau. Il convient donc de distinguer deux aspects dans la sécurité des réseaux mobiles Ad hoc: la sécurité du routage et celle des données. Ces deux aspects comportent certaines vulnérabilités et sont exposés à plusieurs types d'attaques. Malheureusement, la majeure partie des protocoles de routage ad hoc n'ont aucune considération de sécurité.

Le présent travail entre dans le cadre de l'étude du problème de sécurisation du routage dans les réseaux mobiles ad hoc (MANETs). Son objectif principal est de proposer un mécanisme de sécurisation d'un protocole de routage et d'étudier l'impact de la solution sur

certains éléments relatifs aux performances dudit protocole. Pour atteindre cet objectif général le travail est divisé en trois parties.

La première partie est consacrée à l'étude générale des réseaux mobiles ad hoc, et présente les différents protocoles de routage existants avec leurs hypothèses et exigences. Dans la deuxième partie est présentée une étude des problèmes de sécurité dans les MANETs et des différentes propositions visant à leur apporter des solutions. Enfin la troisième partie est consacrée à l'étude d'une proposition de solution à l'une des attaques les plus fréquentes dans les MANETs, et contre laquelle peu de protocoles sont capables de lutter. Une simulation des performances de l'algorithme proposé complète cette partie.

PROBLEMATIQUE

Un réseau mobile ad hoc est un système dans lequel les systèmes informatiques qui le composent peuvent communiquer de manière autonome par ondes radio. Les systèmes informatiques, ou noeuds, s'échangent des informations directement ou, si le nœud qu'ils cherchent à atteindre est hors de portée, par le biais de nœuds intermédiaires. L'architecture d'un réseau mobile ad hoc est caractérisée par une absence d'infrastructure fixe préexistante, à l'inverse des réseaux de télécommunication classiques. Un réseau mobile ad hoc doit s'organiser automatiquement de façon à être déployé rapidement et pouvoir s'adapter aux conditions de propagation et aux différents mouvements pouvant intervenir au sein des unités mobiles. Le fait que la taille d'un réseau ad hoc peut être énorme, souligne que la gestion de l'environnement de routage doit être complètement différente des approches utilisées dans le routage classique.

Le problème qui se pose dans le contexte des réseaux mobiles ad hoc est l'adaptation de la méthode de routage utilisée avec le grand nombre d'unités existant dans un environnement, caractérisé par de modestes capacités de calcul et de sauvegarde. La fonction de routage dans ces systèmes est primordiale car il appartient à chaque noeud de se faire sa propre image de la topologie du réseau. Le problème de la sécurité de ces réseaux ad hoc est un sujet d'actualité. De nombreux travaux sont en cours et aucune solution comblant de bout en bout la sécurité d'un réseau ad hoc n'a été présentée à ce jour. L'objectif à atteindre est en effet complexe.

Il s'avère donc nécessaire de proposer des solutions permettant de garantir la sécurité au niveau du routage dans un réseau mobile ad hoc, et prenant en compte les contraintes et caractéristiques des MANETs. Plusieurs solutions existent, mais aucune d'entre elles n'offre aux utilisateurs la protection efficace contre les différentes attaques possibles dans le routage. Il se pose donc le problème crucial qui consiste à chercher à doter les protocoles de routage de mécanismes leur permettant de résister au plus grand nombre d'attaques possible.

Parmi les attaques les plus virulentes, et contre lesquelles peu de protocoles arrivent à lutter, nous trouvons le ‘’wormhole’’, qui rend les protocoles de routage très vulnérables. Parmi les protocoles de routage sécurisé ARIADNE est l'un des plus performants. Cependant lui non plus ne résiste pas au wormhole. Ainsi, une amélioration de la politique de sécurité mise en œuvre par ARADNE s'impose. Nous nous proposons de résoudre ce problème en utilisant la notion de Packet Leashes. Il s'agit d'un mécanisme, qui peut être mis en œuvre par les protocoles de routage pour lutter contre l'attaque wormhole.

I- Généralités et protocoles de routage dans les réseaux mobiles ad hoc

Un réseau mobile ad hoc, appelé généralement MANET (Mobile Ad hoc Network) [24], est un réseau composé d'unités mobiles qui se déplacent dans un territoire quelconque et dont le seul moyen de communication est l'utilisation des interfaces sans fil, sans l'aide d'une infrastructure préexistante ou d'une administration centralisée.

En fait, les réseaux sans fil, les plus couramment déployés aujourd'hui s'appuient sur des infrastructures fixes : sites accueillant des stations de base, dans le cas des réseaux cellulaires ou câbles pour les infrastructures filaires [30]. La connectivité entre les différents éléments du réseau y est organisée et centralisée. Par contre, les réseaux ad hoc sont des réseaux sans fil formés par des appareils, appelés **noeuds**, et ont la capacité de jouer en même temps le rôle de routeur en relayant les paquets entre deux entités qui ne sont pas à portée de communication. Les appareils en question peuvent être aussi variés que des ordinateurs, des PDAs, des téléphones mobiles, etc.

Chaque noeud du réseau est équipé d'une interface radio qui peut être différente d'un noeud à l'autre et reste libre d'intégrer ou de quitter le réseau. En effet, le réseau s'adapte spontanément, pour répondre à un besoin, d'où la terminologie ad hoc (en latin : *pour cela*) et se configure de façon complètement autonome et dynamique en fonction des possibilités de connexions existantes. Ainsi lorsque les noeuds des réseaux ad hoc sont mobiles, on parle de MANET (Mobile Ad hoc Network).

I-1 Caractéristiques des réseaux mobiles ad hoc

Les réseaux mobiles ad hoc ont les caractéristiques suivantes :

- *Une topologie dynamique* : Les unités mobiles du réseau se déplacent de façon libre et arbitraire. Par conséquent la topologie du réseau peut changer à des instants imprévisibles, de manière rapide et aléatoire ;
- *Une bande passante limitée* : Une des caractéristiques primordiales des réseaux basés sur la communication sans fil est l'utilisation d'un médium de communication partagé. Ce partage fait que la bande passante réservée à un hôte soit modeste ;
- *Des contraintes d'énergie* : Les hôtes mobiles sont alimentés par des sources d'énergie autonomes comme les batteries ou autres sources consommables. Le paramètre d'énergie doit être pris en considération dans tout contrôle fait par le système ;

- *Une sécurité physique limitée* : Les réseaux mobiles ad hoc sont plus touchés par le paramètre de sécurité que les réseaux filaires classiques. Cela se justifie par les contraintes et limitations physiques qui font que le contrôle des données transférées doit être minimisé ;
- *L'absence d'infrastructure* : Les réseaux mobiles ad hoc se distinguent des autres réseaux mobiles par la propriété d'absence d'infrastructures. Les hôtes mobiles sont responsables d'établir et de maintenir la connectivité du réseau de façon continue.

I-2 **Le routage dans les réseaux mobiles ad hoc**

I-2-1 **Définition du routage**

Le routage est une méthode d'acheminement des informations à la bonne destination à travers un réseau de connexion donné. Le problème de routage consiste à déterminer un chemin optimal des paquets à travers le réseau au sens d'un certain nombre de critères de performance.

I-2-2 **Le routage de paquets**

Afin de comprendre les attaques sur les protocoles de routage, il est nécessaire de comprendre leur fonctionnement global [21]. Lorsqu'un nœud dans un réseau, veut émettre un message vers un autre noeud, il regarde dans sa table de routage si une route existe vers ce noeud. Si elle n'existe pas, il initie une découverte de route, ***route discovery***, en diffusant sur le réseau, dans les airs pour les accès sans fil, un message de type ***route request***. Le message de ***route request*** contient l'adresse du noeud émetteur, l'adresse du noeud destinataire, un marqueur permettant d'identifier la découverte de route et une liste initialement vide à remplir par les noeuds intermédiaires. Lorsqu'un noeud intermédiaire reçoit ce paquet, s'il n'en est pas le destinataire et si sa table de routage n'indique pas de chemin pour le noeud recherché, il diffuse à son tour le paquet de type ***route request*** en rajoutant son adresse à la liste de noeuds intermédiaires. Dans le cas où le noeud intermédiaire possède dans sa table de routage un chemin pour le noeud destinataire, la majorité des protocoles prévoit que le noeud intermédiaire renvoie directement un message de type ***route reply*** à l'émetteur en indiquant ce chemin. Lorsqu'un paquet de requête atteint son destinataire, ce dernier émet un paquet de réponse du type ***route reply***. Ce paquet transite par les noeuds intermédiaires de la liste. La figure n° 1 suivante schématise l'évolution des messages lors de la découverte de route.

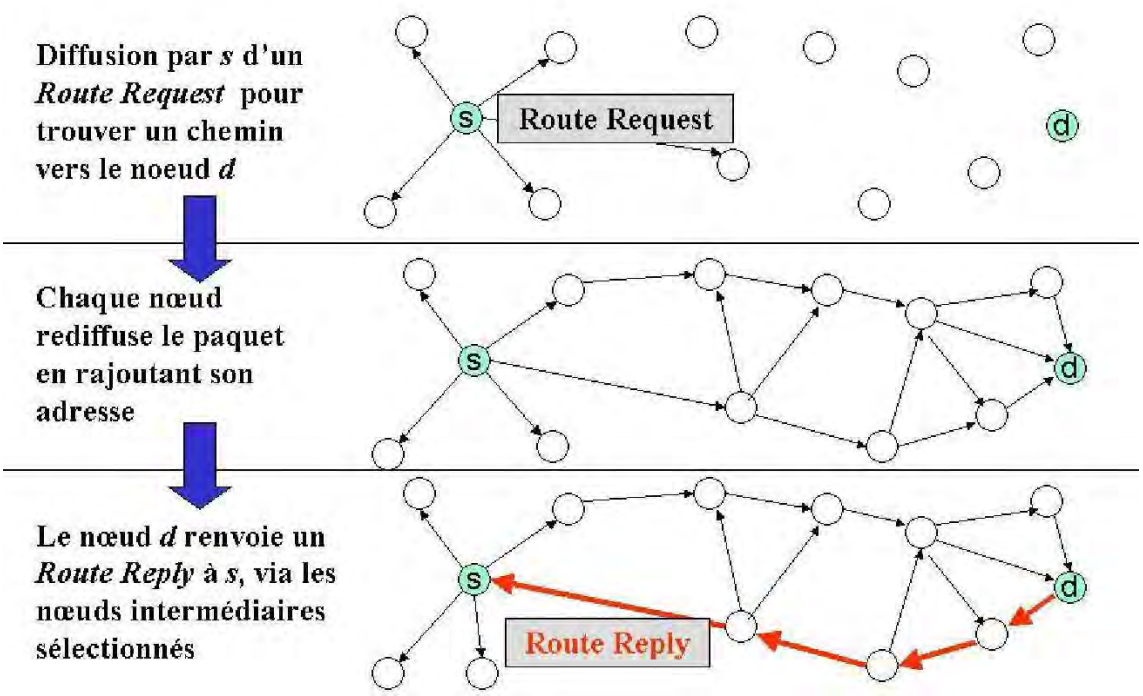


Figure 1 : Découverte de route initiée par le protocole de routage.

Lorsque la réponse atteint l'initiateur de la découverte de route, ce dernier met à jour sa table de routage avec cette nouvelle route, qui consiste en la liste des noeuds intermédiaires avec un coût associé. Le coût sert aux noeuds à effectuer un choix entre plusieurs routes menant à la même destination. Il peut être basé sur le nombre de noeuds intermédiaires traversés ou sur des critères plus complexes comme le débit, la fiabilité des liaisons ou la taille des paquets. Si l'initiateur reçoit ultérieurement une indication comme quoi cette destination peut être jointe avec un coût plus faible par un autre chemin, la table sera mise à jour avec la route ayant le coût le plus faible. Une fois une route établie, un protocole de routage doit aussi mettre en oeuvre un mécanisme de maintenance des routes pour gérer les événements comme la coupure d'un lien entre deux noeuds par lesquels transitent des messages. Lorsqu'un noeud reçoit un paquet de données pour une destination vers laquelle il ne peut plus émettre, il renvoie un message d'erreur de type ***route error*** vers la source du paquet de données. La route doit alors être supprimée de la table de routage. Des optimisations existent permettant à un noeud d'écouter les routes échangées par les autres noeuds et de mettre à jour sa table de routage en conséquence.

I-2-3 **Les protocoles de routage ad hoc**

La stratégie (ou le protocole) de routage est utilisée dans le but de découvrir les chemins qui existent entre les noeuds. Le but principal d'une telle stratégie est l'établissement de routes qui soient correctes et efficaces entre une paire quelconque d'unités, ce qui assure l'échange des messages d'une manière continue**.**

Les protocoles de routage peuvent être classés en différentes familles selon le moment auquel ils initient la découverte de route, selon la manière dont les noeuds d'un réseau se partagent le travail de routage et selon la manière dont les informations de routage sont échangées.

I-2-4 **Classification de protocoles de routage**

Si un protocole initie la découverte de route lorsque le besoin s'en fait ressentir, c'est à dire lorsqu'un paquet doit être transmis vers une destination dont la route n'est pas connue dans la table de routage, il sera considéré comme faisant partie de la famille des protocoles ***réactifs***. Si le protocole initie des découvertes de route régulièrement sans attendre qu'il y ait un paquet à transmettre, il sera dit ***proactif***. Certains protocoles combinent ces deux manières d'initier des découvertes de routes, à la demande et en avance, et sont donc considérés comme ***hybrides***. Certains protocoles de routage n'utilisent pas tous les noeuds d'un réseau pour faire transiter les messages, au contraire ils en sélectionnent certains, en fonction du voisinage ou pour former des cellules. Ces protocoles sont dits ***non uniformes***. Ceux qui utilisent tous les noeuds du réseau capables de router sont appelés ***uniformes***.

Les algorithmes permettant de maintenir la table de routage sont de deux types : les algorithmes basés sur le ***distance vector*** et ceux basés sur le ***link state***. Les protocoles de type ***distance vector*** n'ont qu'une vision partielle du réseau. Les protocoles de type ***link state*** maintiennent leur table de routage à jour grâce à des annonces faites régulièrement par les différents noeuds reflétant l'état de l'ensemble des liaisons du réseau. Ces protocoles ont une vision totale du réseau.

I-2-5 **Les protocoles réactifs ou « à la demande »**

Les protocoles de routage appartenant à cette catégorie [5], créent et maintiennent les routes selon les besoins. Lorsque le noeud a besoin d'une route, une procédure de découverte globale de routes est lancée, et cela dans le but d'obtenir une information spécifiée, inconnue au préalable. Dans ce qui suit, nous allons décrire les protocoles les plus importants de cette classe.

I-2-5-1 **Le protocole de routage« DSR »**

Le protocole (DSR**)** "Routage à Source Dynamique" [3, 16, 24] est basé sur l'utilisation de la technique "routage source". Dans cette technique la source des données détermine la séquence complète des noeuds à travers lesquels les paquets de données seront envoyés.

Un nœud initiateur de l'opération de « découverte de routes » diffuse la demande de route ***route request***. Si l'opération de découverte est réussie, l'initiateur reçoit un paquet réponse de route qui liste la séquence des noeuds à travers lesquels la destination peut être atteinte. Le paquet requête de route contient donc un champ enregistrement de route, dans lequel sera accumulée la séquence des noeuds visités durant la propagation de la requête dans le réseau.

I-2-5-2 **Le protocole de routage« AODV »**

Le protocole AODV (Ad hoc On demand Distance Vector) [24] représente essentiellement une amélioration de l'algorithme DSDV. Il réduit le nombre de diffusions de messages, et cela en ne créant les routes qu’en cas de besoin, contrairement au DSDV qui maintient la totalité des routes. L'AODV utilise le principe des numéros de séquence afin de maintenir la consistance des informations de routage.

A cause de la mobilité des noeuds dans les réseaux ad hoc, les routes changent fréquemment ce qui fait que les routes maintenues par certains nœuds deviennent invalides. Les numéros de séquence permettent d'utiliser les routes les plus nouvelles (fresh routes).

I-2-5-3 **Le protocole de routage « TORA »**

Ce protocole TORA (*Temporary-Ordered Routing Algorithm*) [3] a été conçu principalement pour minimiser l'effet des changements de la topologie qui sont fréquents dans les réseaux ad hoc.

Afin de s'adapter à la mobilité des réseaux ad hoc, le protocole stocke plusieurs chemins vers une même destination, ce qui fait que beaucoup de changements de topologie n'auront pas d'effets sur le routage des données, à moins que tous les chemins qui mènent vers la destination ne soient perdus (rompus). Le protocole TORA détermine les chemins entre une source et une destination en créant un graphe acyclique dirigé (*directed acyclic graph*)[1] entre la source et la destination. Cet arbre est créé en utilisant un algorithme de création de route basé sur une pondération des noeuds. Plus précisément, le poids de chaque noeud est égal à la distance séparant le noeud source du noeud destination, la destination ayant un poids nul. Cet arbre est créé par un noeud source.

I-2-5-4 **Le protocole de routage « ABR »**

Le protocole ABR (*Associativity-Based long-lived Routing protocol*) [24, 3] est adapté à un réseau composé de noeuds peu mobiles et de noeuds fortement mobiles. Ce protocole tire partie de la présence de noeuds peu mobiles dans le réseau en les utilisant pour acheminer les messages. Les liens entre les noeuds peu mobiles sont supposés être plus stables que ceux des noeuds fortement mobiles, puisque ces liens se brisent moins souvent que les autres. Chaque noeud du réseau conserve ainsi une table d'associativité définissant sa stabilité d'associativité (ou de connectivité) avec chacun de ses voisins directs. Cette table est mise à jour suite à la diffusion périodique (sur un saut) de messages *hello* par chaque noeud.

Pour découvrir une route vers une destination, un noeud diffuse une requête de localisation. Chaque noeud intermédiaire, lorsqu'il reçoit le message, vérifie qu'il n'est pas la destination et rediffuse ensuite le message en y ajoutant son adresse et sa table d'associativité (c'est-à-dire sa stabilité d'associativité avec chaque voisin direct). Un noeud intermédiaire recevant ce message supprime de la table d'associativité, la stabilité d'associativité

[1] De tels graphes sont pratiquement des arbres et imposent une direction. Souvent ils servent à représenter un circuit combinatoire ou une chaîne de production.

précédemment ajoutée. Il ajoute la stabilité d'associativité entre lui et le nœud qui lui a expédié le message. Finalement, quand la requête de localisation atteint la destination, cette dernière sera capable de choisir la meilleure route parmi les différentes routes produites, c'est-à-dire, celle offrant la meilleure stabilité de connectivité.

I-2-5-5 **Le protocole de routage « CBRP »**

CBRP (*Cluster Based Routing Protocol*) [11, 16] est un protocole particulièrement adapté aux réseaux ad hoc caractérisés par une faible mobilité. En effet, ce protocole est basé sur une gestion de groupes difficiles à maintenir si les noeuds sont fortement mobiles. Plus précisément, ce protocole décompose le réseau en groupes de rayon égal à un saut. Chaque groupe est composé d'un unique coordinateur ayant une complète connaissance du groupe (c'est-à-dire des membres du groupe et des liens entre lui-même et les membres du groupe).

Chaque noeud du réseau maintient deux tables : une table de ses voisins directs, et une table des groupes adjacents, composée de la liste des groupes adjacents et de leur coordinateur respectif. Ces deux tables sont maintenues à jour grâce aux messages hello diffusés périodiquement par chaque noeud sur deux sauts. Ce message contient l'état du noeud (du point de vue de son appartenance à un groupe), l'identifiant du coordinateur du (ou des) groupe(s) au(x)quel(s) appartient le noeud, une liste des noeuds 1-voisins et une liste des groupes adjacents incluant l'identifiant de leur coordinateur respectif. Lorsqu'un noeud reçoit un message hello, il rafraîchit ses deux tables. Plus précisément, si l'expéditeur du message hello ne fait pas partie de son groupe, il définit l'expéditeur comme étant une passerelle vers un groupe adjacent, et ajoute cette information à sa table de groupes adjacents.

Par ailleurs, ces messages hello sont utilisés pour élire un coordinateur. En effet, un nœud n'ayant aucun coordinateur, spécifie dans le message hello qu'il cherche un coordinateur, c'est-à-dire que le noeud se trouve dans l'état indécis. Après expiration d'un délai, si ce dernier ne reçoit pas en réponse un message hello de la part d'un coordinateur, alors, il se définit comme coordinateur. Dans le cas contraire, il se définit comme membre d'un groupe. L'utilisation de groupes permet de limiter le trafic généré lors de la localisation d'un noeud. Plus précisément, lorsqu'un noeud souhaite localiser une destination, il diffuse (sur un saut) sa requête contenant : le coordinateur du (ou des) groupe(s) au(x) quel(s) il appartient, la liste des coordinateurs adjacents et des passerelles correspondantes. Les noeuds

passerelles (spécifiés dans la requête) font suivre l'information aux coordinateurs adjacents. Ces derniers répondent à la requête si la destination appartient au groupe qu'ils coordonnent. Dans le cas contraire, ils font suivre la requête aux coordinateurs des groupes adjacents.

I-2-5-6 **Le protocole de routage « RDMAR »**

Le protocole de Routage basé sur la Micro découverte des Distances Relatives [16] RDMAR a été conçu principalement pour minimiser la charge induite par les changements rapides des réseaux ad hoc.

Le protocole utilise un nouveau mécanisme de découverte de routes, appelé la Micro découverte de Distance Relative ou RDM (Relative Distance Micro-discovery). L'idée de base du RDM est que la diffusion des requêtes peut se faire en se basant sur une distance relative (RD) [1] entre les paires des unités mobiles.

Un algorithme itératif est utilisé pour estimer la RD qui sépare les deux noeuds, et cela en utilisant les informations concernant la mobilité des noeuds, le temps écoulé depuis la dernière communication et l'ancienne valeur de la distance RD. Sur la base de la nouvelle distance calculée, la diffusion de requête est limitée à une certaine région du réseau dans laquelle la destination peut être trouvée. Cette limitation de diffusion, peut minimiser énormément le contrôle du routage, ce qui améliore les performances de la communication.
Dans le protocole RDMAR, la décision du choix de chemin est prise au niveau du noeud destination. Seulement le meilleur chemin choisi sera valide, les autres chemins restent passifs. Dans le cas où un noeud détecte la défaillance d'un lien, il exécute une phase d'avertissement de défaillance afin d'avertir la source de l'invalidité du lien.

I-2-5-7 **Le protocole de routage « SSR »**

Le protocole SSR "Routage basé sur la Stabilité du Signal" (Signal Stability-based Routing) [13, 20] est un protocole de routage réactif dont le choix des routes est basé sur la puissance du signal entre les nœuds, en plus de leur stabilité de localisation. Ce critère de sélection de routes fait que les chemins utilisés durant le routage des données ont une forte interconnexion.

Le protocole SSR inclut deux protocoles qui coopèrent entre eux : le Protocole de Routage Dynamique appelé DRP (Dynamique Routing Protocol), et le Protocole de Routage Statique appelé SRP (Static Routing Protocol). Le premier protocole, le DRP utilise deux tables : une table de stabilité de signal SST (Signal Stability Table), et une table de routage RT. La table SST sauvegarde les puissances des signaux des nœuds voisins, obtenues par l'échange périodique des messages avec la couche de liaison de chaque voisin.

Toutes les transmissions sont reçues et traitées par le DRP. Après la mise à jour de l'entrée appropriée de la table, le protocole DRP fait passer le paquet traité au protocole SSR. Le SSR consulte sa table de routage RT pour la destination spécifiée, et envoie le paquet reçu au voisin suivant. Si aucune entrée (dans la RT) associée au nœud destination n'est disponible, le SSR initie un processus de recherche de routes en diffusant un paquet requête de route. Le paquet requête de route est envoyé une seule fois (pour éviter le bouclage), et uniquement aux voisins vers lesquels existe un lien de forte puissance. Le nœud destination choisit le premier paquet requête de route qui arrive, car il y a une grande probabilité pour que ce paquet ait traversé le meilleur chemin (le plus court, le moins chargé, etc.) existant entre la source et la destination.

Le DRP du nœud destination inverse le chemin choisi, et envoie un message de réponse de route au nœud source. Lors de la réception de cette réponse, le DRP d'un nœud intermédiaire met à jour la table de routage locale suivant le chemin inclus dans le paquet reçu.

Les paquets de recherche de routes qui arrivent à destination prennent nécessairement le chemin de forte stabilité de signal car les nœuds de transit n'envoient pas de paquets à travers les liens de faible puissance de signal. Si la source expire son timeout sans la réception de réponse, elle relance de nouveau un processus de recherche de routes en indiquant cette fois-ci que les canaux de faible puissance peuvent être utilisés.

I-2-5-8 **Le protocole de routage « LAR »**

Le protocole appelé "Routage aidé par la localisation" ou LAR (Location-Aided Routing) [11, 16] est un protocole de routage réactif basé sur l'utilisation des localisations. Ce protocole procède d'une manière très similaire au protocole DSR, la principale différence

entre les deux protocoles réside dans le fait que le LAR utilise les informations de localisation fournies par le système de positionnement global appelé GPS[2] (Global Positioning System) dans le but de limiter l'inondation des paquets de requête de route. Afin d'assurer cela, deux approches peuvent être utilisées.

Dans la première approche, le nœud source définit une région circulaire dans laquelle la destination peut être localisée. La position et la taille de la région sont estimées en se basant sur :

- la position de la destination, telle qu'elle est connue par la source ;
- l'instant qui correspond à cette position ;
- la vitesse moyenne du mouvement de la destination.

Le plus petit rectangle couvrant la région circulaire et le nœud source est appelé la zone de requête. L'information calculée est rattachée au paquet de requête de route. Cela est fait uniquement par le nœud source et les nœuds qui appartiennent à la zone de requête.
Dans la deuxième approche, le nœud source calcule la distance qui le sépare de la destination, et l'inclut dans le paquet de requête de route. Ce dernier est envoyé par la suite aux nœuds voisins. Quand un nœud reçoit le paquet de requête, il calcule la distance qui le sépare de la destination et la compare avec la distance contenue dans le paquet reçu. Dans le cas où la distance calculée est inférieure ou égale à la distance reçue, le nœud envoie le paquet reçu. Lors de l'envoi, le nœud met à jour le champ de distance avec sa propre distance qui le sépare du nœud destination.

Dans les deux méthodes, si aucune réponse de route n'est reçue en dépassant une certaine période (le timeout), le nœud source rediffuse une nouvelle requête de route en utilisant une diffusion pure (sans limitation).

I-2-6 **Les protocoles proactifs, ou « par table de routage »**

Les protocoles de routage proactifs [5] pour les réseaux mobiles ad hoc, sont basés sur la même philosophie que les protocoles de routage utilisés dans les réseaux filaires conventionnels. Ils utilisent deux principales méthodes que sont :

[2] Le GPS (Global Positioning System) est un système de localisation par satellite mis en place par le département américain de la défense. Il permet de déterminer les coordonnées géographiques d'un élément.

- La méthode Etat de Lien ("Link State") ;
- La méthode du Vecteur de Distance ("Distance Vector").

Les deux méthodes exigent une mise à jour périodique des données de routage qui doit être diffusée par les différents noeuds de routage du réseau.
Nous allons décrire dans ce qui suit, les protocoles les plus importants de cette classe :

I-2-6-1 **Le protocole de routage « DSDV »**

DSDV [14] est basé sur l'idée classique de l'algorithme distribué de Bellman-Ford[3] en rajoutant quelques améliorations. Chaque station mobile maintient une table de routage qui contient :

- Toutes les destinations possibles ;
- Le nombre de noeuds (ou de sauts) nécessaires pour atteindre la destination ;
- Le numéro de séquence (SN : sequence number), qui correspond à un nœud destination.

Le SN est utilisé pour faire la distinction entre les anciennes et les nouvelles routes, ce qui évite la formation des boucles de routage.
La mise à jour dépend donc de deux paramètres : le temps, c'est à dire la période de transmission, et les événements.

Un paquet de mise à jour contient :

- Le nouveau numéro de séquence incrémenté, du noeud émetteur. Et pour chaque nouvelle route :
- L'adresse de la destination ;
- Le nombre de noeuds (ou de sauts) séparant le noeud de la destination ;
- Le numéro de séquence (des données reçues de la destination).

Dans ce protocole, une unité mobile doit attendre jusqu'à ce qu'elle reçoive la prochaine mise à jour initiée par la destination, afin de mettre à jour l'entrée associée à cette destination, dans la table de distance. Ce qui fait que le DSDV est lent.

[3] L'algorithme de Bellman-Ford (Richard Bellman et Lester Ford) est un algorithme de programmation dynamique qui permet de trouver les plus courts chemins, depuis un sommet source donné, dans un graphe orienté.

I-2-6-2 **Le protocole de routage « OLSR »**

OLSR (Optimized Link State Routing) [7, 10, 16, 24] est un protocole proactif à état de lien, qui utilise un mécanisme d'inondation[4] optimisé pour diffuser à tous les noeuds du réseau des informations partielles sur les liens. Le trafic de contrôle dans OLSR se compose de deux types de messages: HELLO et TC. Les HELLOs sont envoyés périodiquement par un noeud pour signaler ses liens (symétriques, asymétriques ou MPR) avec les noeuds voisins, et ne sont pas relayés. L'échange de messages HELLO permet à chaque noeud de mémoriser des informations sur son voisinage à deux sauts, informations qui seront par la suite utilisées pour la sélection des MPRs (Relais Multipoints) [10]. Les TCs sont émis périodiquement par un noeud si celui-ci a été sélectionné comme MPR, et contiennent une liste de voisins symétriques du noeud; ces messages sont diffusés dans le réseau entier.

Deux autres types de messages, MID et HNA, sont émis par un noeud ayant des interfaces multiples respectivement OLSR et non OLSR, pour annoncer la configuration de ses interfaces au réseau. Ces messages de contrôle sont encapsulés dans un paquet OLSR. En effet, le système d'inondation est basée sur un sous-groupe de noeuds appelés *Relais Multipoint (MPR)*. Chaque noeud sélectionne ses MPRs parmi ses voisins symétriques de telle façon qu'un message envoyé par le noeud et répété par ses MPRs (son *MPR set*) sera reçu par tous les voisins à deux sauts du noeud en question. Chaque noeud mémorise aussi un *MPR selector set*, qui contient l'adresse de ses voisins qui l'ont sélectionné comme MPR. Les messages de contrôle sont relayés seulement par les MPRs.

I-2-6-3 **Protocole de routage « FSR »**

Le protocole FSR (Fisheye State Routing) [11] est basé sur l'utilisation de la technique "oeil de poisson" (fisheye) qui est utilisé dans le but de réduire le volume d'information nécessaire pour représenter les données graphiques. Dans la pratique, l'œil de poisson capture avec précision les points proches du point focal. La précision diminue quand la distance séparant le point vu et le point focal augmente. Cette technique de l'œil du poisson permet la réduction du volume d'informations nécessaire pour les données graphiques. Elle sous-entend

[4] C'est la technique la plus rudimentaire de diffusion. Elle consiste à répéter un message dans tout le réseau : chaque noeud qui reçoit le message pour la première fois répète le message. Ainsi, de proche en proche, le message inonde le réseau

une diminution du détail et de la précision plus la distance augmente. Pour revenir à notre contexte du routage, on définira la portée ou le champ de vision du poisson en nombre de sauts, plus un nœud est proche plus les données maintenues envers celui-ci seront plus précises. La réduction du volume des données de mise à jour est obtenue en utilisant des périodes d'échanges différentes pour les différentes entrées en fonction de leur distance. Les entrées qui correspondent aux nœuds les plus proches sont envoyées aux voisins avec une fréquence élevée (donc avec une période d'échange relativement petite). Ainsi un grand nombre de données de routage est évité, ce qui réduit le volume des messages qui circule sur le réseau.

I-2-6-4 Protocole de routage « HSR »

La notion de partitionnement et de groupes est très répandue en pratique dans les réseaux mobiles ad hoc. La notion de groupe peut améliorer les performances des réseaux. Par exemple, les interférences des signaux peuvent être réduites en utilisant différents codes étendus à l'aide des groupes. En plus de cela, le partitionnement peut être exploité dans les réseaux de grande taille afin de réaliser un routage hiérarchique, ce qui réduit le contrôle des données de routage. Le problème principal du routage hiérarchique dans les réseaux sans fil, est la mobilité et la gestion de la localisation. Dans le but de résoudre ce problème, le protocole HSR (Hierarchical State Routing) [11] a été proposé. Il combine les notions de groupes dynamiques, niveaux hiérarchiques avec une gestion efficace de localisation. Dans le HSR, l'image de la topologie du réseau, est sauvegardée sous forme hiérarchique. Le réseau est partitionné en un ensemble de groupes. Dans un groupe, un noeud doit être élu pour représenter le reste des membres. Les représentants des groupes dans un niveau i, deviennent des membres au niveau i + 1. Ces nouveaux membres, s'organisent en un ensemble de groupes de la même manière du niveau bas, et ainsi de suite pour le reste des niveaux. L'adresse hiérarchique suffit pour délivrer les paquets de données à une destination, indépendamment de la localisation de la source, et cela en utilisant la table HSR.

I-2-6-5 Protocole de routage « DREAM »

Ce protocole DREAM [16] est basé sur les informations de localisation des unités mobiles. Le protocole diffuse les données destinées à une certaine destination en effectuant une inondation (propagation) partielle et en utilisant les données de localisation. Cela permet de minimiser la charge du réseau. Chaque noeud du réseau mobile ad hoc échange

périodiquement des messages de contrôle afin d'informer tous les autres noeuds de sa localisation. Lors de l'envoi des données, si la source possède des informations récentes sur la localisation du noeud destination, elle choisit un ensemble de noeuds voisins, qui sont localisés dans la direction source/destination. Si un tel ensemble n'existe pas, les données sont inondées dans le réseau entier. Dans le cas où de tels noeuds existeraient, une liste qui contient leurs identificateurs est insérée à la tête du paquet de données avant la transmission. Seulement les noeuds qui sont spécifiés dans la liste de tête traitent le paquet.

I-2-6-6 **Le protocole de routage « TBRPF »**

TBRPF (Topology broadcast Based on Reverse-Path Forwarding) [17] est un protocole de routage proactif destiné aux réseaux ad hoc mobiles. Ce protocole à l'air moins mature que AODV ou OLSR mais comporte quelques avantages sur eux : la détermination des tables de routage est beaucoup moins gourmande : seules les modifications sont échangées. TBRPF utilise l'algorithme de Dijkstra pour déterminer le chemin le plus court. TBRPF garantit le routage « hop by hop[5]» le long des routes les plus courtes vers chaque destination. Chaque noeud utilisant TBRPF crée un arbre de source (fournissant des chemins vers tous les noeuds accessibles) basé sur l'information partielle de topologie stockée dans sa table de topologie.

Pour réduire au minimum l'occupation de la bande passante, chaque noeud envoie seulement une partie de son arbre de source aux voisins. TBRPF emploie une combinaison des mises à jour périodiques et différentielles pour tenir tous les voisins au courant de la partie rapportée de son arbre source. Chaque noeud a également la possibilité d'envoyer des informations additionnelles de topologie (jusqu'à la topologie complète), pour fournir une fiabilité améliorée dans les réseaux fortement mobiles. TBRPF effectue la découverte du voisinage en utilisant des messages « HELLO » qui rapportent seulement des changements dans le statut des voisins. Ceci a certains avantages, car les messages « HELLO » sont beaucoup plus petits que ceux d'autres protocoles de routage à état de lien.

[5] L'option "Sauts après sauts" est utilisée pour transporter des informations optionnelles qui doivent être examinées par chaque noeud le long du chemin emprunté par le paquet.

I-2-6-7 **Le protocole de routage « GSR»**

Le protocole appelé "Routage à Etat Global" ou GSR (Global State Routing) [16] est similaire au protocole DSDV décrit précédemment. Ce protocole utilise les idées du routage basé sur l'état des liens (LS), et les améliore en évitant le mécanisme inefficace de l'inondation des messages de routage. Le GSR utilise une vue globale de la topologie du réseau, comme c'est le cas dans les protocoles LS.

Le protocole utilise aussi une méthode appelée méthode de dissémination, qui a l'avantage de l'absence d'inondation. Dans ce protocole, chaque nœud i maintient une liste de voisins Ai, dans trois tables différentes telles que la table de topologie TTi, la table des nœuds suivants NEXTi (Next Hop), et la table de distance Di. La table de la topologie TTi contient, pour chaque destination, l'information de l'état de lien telle qu'elle a été envoyée par la destination et une estampille de l'information. Pour chaque nœud de destination j, la table NEXTi contient le nœud vers lequel les paquets destinés à j seront envoyés. La table de distance contient la plus courte distance pour chaque nœud destination.

Les messages de routage sont générés suivant les changements d'états des liens. Lors de la réception d'un message de routage, le nœud met à jour sa table de topologie et cela dans le cas où le numéro de séquence du message reçu serait supérieur à la valeur du numéro de séquence sauvegardée dans la table (exactement comme le fait le protocole DSDV). Par la suite, le nœud reconstruit sa table de routage et diffuse les mises à jour à ses voisins. Le calcul des chemins peut se faire avec n'importe quel algorithme de recherche des plus courts chemins. Par exemple, l'algorithme du GSR utilise l'algorithme de Dijkstra[6] modifié de telle façon qu'il puisse construire la table des nœuds suivants (NEXT) et la table de distance (D) en parallèle avec la construction de l'arbre des plus courts chemins.

I-2-6-8 **Le protocole de routage « ZHLS »**

Le protocole "Routage à Etat de Liens Hiérarchique basé sur les Zones", appelé ZHLS (Zone-Based Hierarchical Link State Routing) [20] est basé sur la décomposition du réseau en un ensemble de zones. Dans ce protocole, les membres d'une zone n'élisent pas de représentants, contrairement à ce qui se fait dans les autres protocoles hiérarchiques. Avec

[6] L'algorithme de Dijkstra résout le problème du plus court chemin pour un graphe G=(S, A) connexe dont le poids lié aux arêtes est *positif* ou nul.

cette décomposition, on a deux niveaux de topologies : le niveau nœud, et le niveau zone. La topologie basée sur le premier niveau donne la façon selon laquelle les nœuds d'une zone donnée sont connectés physiquement.

Dans ce protocole, les paquets qui contiennent les états des liens ou les LSPs (Link State Packet)[7] peuvent être divisés en deux classes : les LSPs orientés nœud, et les LSPs orienté zone. Pour un nœud donné, un paquet LSP orienté nœud contient l'information d'un nœud voisin, tandis qu'un paquet LSP orienté zone contient l'information de la zone. De cette façon, chaque nœud du réseau possède une connaissance complète concernant les nœuds de sa propre zone, et seulement une connaissance partielle du reste des nœuds.

Cette connaissance partielle est matérialisée par l'état de la connexion des différentes zones du réseau. Par conséquent, l'acheminement des données se fait de deux façons : le routage inter zone, et le routage intra zone.

Pour une destination donnée, les données sont envoyées entre les zones en utilisant les identificateurs des zones, jusqu'à ce que les données atteignent la zone finale de la destination.

I-2-6-9 **Le protocole de routage « CGRS »**

Le protocole appelé CGRS (Clusterhead Gateway Switch Routing) [20] utilise principalement l'algorithme de routage DSDV : l'ensemble des unités mobiles du réseau est décomposé en groupes, et chaque groupe élit un représentant. Les nœuds se trouvant à la portée de communication d'un représentant de groupe appartiennent au groupe représenté par ce dernier.

Un nœud de liaison est un nœud qui appartient à la portée de communication de plus d'un représentant de groupe. Cette forme d'organisation peut fortement dégrader les performances des réseaux ad hoc à cause des changements fréquents de leur topologie.

Dans le protocole CGSR, le routage des informations se fait de la manière suivante : le nœud source transmet ses paquets de données à son représentant de groupe. Celui-ci envoie les paquets au nœud de liaison qui relie ce représentant avec le représentant suivant dans le chemin qui existe vers la destination. Le processus se répète jusqu'à ce que l'on atteigne le représentant du groupe auquel appartient le destinataire, et celui-ci lui transmet alors les paquets reçus.

[7] **LSP** est un paquet d'information généré par un routeur au niveau des protocoles à état de liens qui fournit la liste des routeurs voisins.

I-2-6-10 **Le protocole de routage « WRP »**

Le protocole de routage sans fil WRP (Wireless Routing Protocol) [16] est basé sur l'utilisation de la classe des algorithmes de recherche de chemins PFA[8] (Path-Finding Algorithm). En fait, WRP utilise un algorithme de recherche de chemins qui réduit les situations des boucles temporaires et qui limite les mises à jour lors des changements significatifs des entrées de la table de routage. Dans ce protocole, chaque nœud maintient :

- Une table de distance ;
- Une table de routage ;
- Une table de coûts des liens ;
- Une liste de retransmission de messages MRL (Message Retransmission List).

La liste de retransmission de messages permet à un nœud donné de connaître l'ensemble des voisins qui n'ont pas acquitté leur message de mise à jour, et de retransmettre ce message à cet ensemble de voisins. Un nœud envoie un message de mise à jour s'il détecte un changement d'état d'un lien voisin ou après la réception des données de mise à jour d'un autre voisin. Les nœuds présents dans la liste de réponse du message de mise à jour doivent acquitter la réception du message. S'il n'y a pas de changement dans la table de routage, par rapport à la dernière mise à jour, le nœud doit envoyer un message "Hello" pour assurer la connexion. Lors de la réception du message de mise à jour, le nœud modifie sa distance et cherche les meilleurs chemins en se basant sur les informations reçues.

Selon le contexte d'utilisation, ces protocoles offrent des performances différentes. Le tableau 1 ci-dessous propose un récapitulatif des protocoles de routage que nous avons présentés, indiquant leurs principales propriétés (type et catégorie), le contexte d'utilisation dans lequel ils sont particulièrement performants, leur état d'avancement dans le processus de normalisation, le niveau d'implication des noeuds dans le routage.

[8] Ces algorithmes utilisent des données concernant la longueur et le noeud prédécesseur du chemin le plus court, correspondant à chaque destination ;

Tableau 1 : Propriétés des protocoles de routage ad hoc.

Protocoles	Type	Catégorie	Implication des noeuds	Caractéristique du réseau	Normalisation IETF
DSDV	proactif	Orienté destination	Uniforme	Faible mobilité Faible densité	--
OLSR	proactif	Orienté topologie	Non Uniforme	Trafic de données important	RFC 3626
AODV	Réactif	Orienté destination	Uniforme	Trafic de données faible	RFC 3561
DSR	Réactif	Orienté topologie	Uniforme	Trafic de données faible	RFC 4728
WRP	Proactif	Orienté topologie	Non Uniforme		
GSR	Proactif	Orienté Topologie	Uniforme	Forte mobilité	
FSR	Proactif	Orienté topologie	Non Uniforme	Forte mobilité Forte densité Trafic de données important	
DREAM	Géographique proactif	Orienté topologie	Uniforme	Forte mobilité faible densité	
CBRP	Réactif	Hiérarchique de groupe	Non Uniforme	Faible mobilité	Draft expiré
TORA	Réactif	Orienté destination	Uniforme	Forte mobilité	Draft expiré
TBRPF	Proactif	Orienté topologie	Non Uniforme	Trafic de données important	RFC 3684
ABR	Réactif	Orienté destination	Uniforme	Forte mobilité	
ZHLS	Géographique Hybride	Hiérarchique	Non Uniforme		
ZRP	Hybride		Non Uniforme		Draft expiré
LAR	Géographique Réactif	Orienté Topologie	Uniforme	Faible mobilité	
HSR	Proactif	Hiérarchique de groupe dynamique	Uniforme	Faible mobilité	
CGSR	Proactif	Orienté destination	Uniforme	Faible mobilité Faible densité	
SSR	Réactif, Géographique	Orienté destination	Non Uniforme	Faible mobilité Faible densité	
RDMAR	Réactif, Géographique	Orienté topologie	Uniforme	Forte mobilité	

Comme nous l’avons déjà vu, l'architecture d'un réseau mobile ad hoc est caractérisée par une absence d'infrastructure fixe préexistante, à l'inverse des réseaux de télécommunication classiques. Un réseau mobile ad hoc doit s'organiser automatiquement de façon à être déployé rapidement et pouvoir s'adapter aux conditions de propagation et aux différents mouvements pouvant intervenir au sein des unités mobiles (la mobilité).

Le fait que la taille d'un réseau ad hoc peut être énorme, souligne que la gestion de l'environnement de routage doit être complètement différente des approches utilisées dans le routage classique. Le problème qui se pose dans le contexte des réseaux ad hoc est l'adaptation de la méthode de routage utilisée avec le grand nombre d'unités existant dans un environnement caractérisé par de modestes capacités de calcul et de sauvegarde.

Le routage dans les réseaux ad hoc mobiles hérite en plus de ces problèmes spécifiques à savoir (topologie dynamique, énergie limitée, etc.), des problèmes des réseaux de câble traditionnels à l'infrastructure fixe. Il y a plusieurs protocoles bien connus dans la littérature qui ont été spécifiquement développés pour faire face aux limitations imposées par les environnements ad hoc de gestion de réseau.

Il semble donc important que toute conception de protocole de routage doive étudier ces problèmes qui restent aussi très remarquables au niveau de la sécurité.

II- **Sécurité au niveau des MANETs**

La sécurité dans les MANETs est un composant essentiel pour les fonctions de base de réseau comme le routage et l'expédition de paquets: l'opération de réseau peut être facilement compromise si des contre-mesures ne sont pas inclues dans les mécanismes de gestion de réseau. À la différence des réseaux utilisant des câbles traditionnels avec infrastructure fixe, dans les réseaux ad hoc, les fonctions de routage et d'expédition de paquets sont effectuées par tous les noeuds disponibles. Cette différence est au centre des problèmes de sécurité qui sont spécifiques aux réseaux ad hoc. Par opposition aux routeurs classiques, les noeuds d'un réseau ad hoc n'ont pas suffisamment confiance entre eux pour l'exécution correcte des fonctions critiques de réseau. Il y a plusieurs facteurs qui rendent particulièrement difficile la maintenance d'un niveau élevé de sécurité. Pour accentuer le problème certaines des difficultés seront présentées.

II-1 **Vulnérabilités des réseaux mobiles Ad hoc**

A cause de leurs spécificités, de nouveaux types de vulnérabilités apparaissent dans les réseaux mobiles ad hoc. Ainsi, les noeuds sont exposés au vol car ils sont mobiles et la capacité de calcul est limitée, ce qui fait que l'utilisation de solutions lourdes comme les PKI[9] (Public Key Infrastructure) n'est pas pratique ici. Aussi, les services dans les réseaux Ad hoc sont provisoires et les batteries ont une limite d'énergie. Cette dernière vulnérabilité fait que les attaques par Déni de Service par consommation d'énergie sont possibles.

Ces vulnérabilités peuvent être classées selon les spécificités des MANETs comme suit :

- **Vulnérabilité des canaux** : un ennemi peut écouter des messages et injecter clandestinement de faux messages dans le réseau sans devoir obtenir l'accès physique au réseau ;
- **La vulnérabilité des nœuds** : en général les noeuds ne sont pas physiquement très protégés, ils peuvent donc facilement tomber sous la contrôle d'un attaquant ;
- **L'absence de l'infrastructure :** le fait que le réseau ad hoc est censé fonctionner sans aucune infrastructure fixe, fournit des problèmes additionnels. Sans structure fixe, il n'est

[9] PKI (Public Key Infrastructure) est un système de gestion des clefs publiques qui permet de gérer des listes importantes de clefs publiques et d'en assurer la fiabilité, pour des entités généralement dans un réseau.

pas possible de mettre en application les solutions de sécurité basées sur des autorités de certification et des serveurs en ligne ;

- **La topologie changeante dynamiquement** : la topologie constamment changeante dans un réseau ad hoc mobile présente plus de vulnérabilités au niveau du routage dans la mesure où, il est particulièrement difficile de distinguer si l'information de routage défectueuse a été produite par un noeud compromis, ou est le résultat d'un quelconque changement de la topologie du réseau.

Ainsi ces vulnérabilités sont à l'origine de beaucoup de menaces qui nécessitent la mise en place de mécanismes de protection. En outre, les différentes applications auront différentes conditions de sécurité à prendre en compte, en raison de cette diversité des menaces.

II-2 **Les exigences de sécurité du routage dans les MANETs**

Pour résoudre la question concernant la sécurité dans un réseau mobile ad hoc un certain nombre de solutions cryptographiques [31] peuvent être utilisées pour empêcher l'impact des attaquants. Ces solutions s'appuient sur les services tels que la disponibilité, la confidentialité, l'intégrité, l'authentification et la non répudiation.

II-2-1 **La disponibilité**

Le réseau doit à tout moment être disponible pour envoyer et recevoir des messages même s'il est soumis aux attaques. Les menaces possibles pour la disponibilité sont sous forme de déni de service. De même le noeud lui-même peut également être un problème à la disponibilité du moment où il décide de ne plus fournir ses services à profit d'autres noeuds afin de sauver ses ressources propres (par exemple puissance de batterie).

II-2-2 **La confidentialité**

Elle fournit des informations secrètes au nœud sensible du réseau. Si cette information tombait entre les mains d'un nœud malveillant, ce dernier ne pourra pas savoir son contenu, qui est le résultat d'un chiffrement à l'aide de la cryptographie.

II-2-3 **L’intégrité**

Elle assure que des messages envoyés dans le réseau ne soient pas corrompus. Les attaques possibles qui compromettraient l'intégrité sont des attaques malveillantes sur le réseau, sous forme d'échecs du signal radio.

II-2-4 **L’authentification**

Elle garantit l'identité des noeuds dans le réseau. Si A envoie à B, A sait que c'est B qui envoie le message. En outre B sait que c'est A qui le reçoit. Si l'authentification ne fonctionne pas, il sera possible pour un étranger par une mascarade, d’envoyer et de recevoir des messages sans que quiconque le note.

II-2-5 **La non répudiation**

Elle permet à un noeud de réception d’identifier un autre noeud comme origine d'un message. L'expéditeur ne peut pas nier d’avoir envoyé le message et, est donc responsable de son contenu. Il est particulièrement utile pour la détection des noeuds compromis.

Basés sur cette analyse de menaces, nous discutons maintenant de plusieurs attaques spécifiques qui peuvent viser l'opération du protocole de routage dans les réseaux mobiles ad hoc.

II-3 **Les attaques liées aux protocoles de routage ad hoc**

On peut classifier les attaques possibles [5, 18, 19, 21] en deux catégories: les attaques passives et les attaques actives. Dans les attaques passives, l’attaquant n’interrompt pas le protocole de routage mais tente de découvrir des informations valables en captant le trafic de routage. En général, dans un réseau mobile, un attaquant passif n’est pas détectable. En outre, les informations de routage peuvent révéler des relations existantes entre noeuds. Si une route vers un nœud particulier est plus demandée que vers d’autres noeuds, l’attaquant peut prévoir que le noeud est important pour le fonctionnement du réseau, et sa neutralisation peut anéantir le réseau entier. Pour mener une attaque active, l’attaquant doit être capable d’injecter des paquets arbitraires dans le réseau pour le neutraliser ou d’analyser des paquets destinés aux autres noeuds.

Dans cette section, nous énumérons différents types d'attaques qui sont possibles dans les réseaux mobiles ad hoc.

II-3-1 **Location disclosure (capture de la position du noeud)**

C’est une attaque qui vise les conditions d'intimité d'un réseau ad hoc. Par l'utilisation des techniques d'analyse de trafic, ou par sondage. La tentative de surveillance d’un attaquant peut découvrir la position d'un noeud, ou même la structure du réseau entier.

II-3-2 **Black hole (trou noir)**

Dans une attaque de trou noir [5] un noeud malveillant injecte de faux itinéraires aux demandes de route qu'il reçoit en annonçant le chemin le plus court vers une destination. Ces réponses fausses peuvent être fabriquées par le noeud malveillant dans le but de détourner le trafic réseau pour écouter ou attirer simplement tout le trafic vers lui. Ceci lui permet d'exécuter clandestinement une attaque par déni de service en laissant tomber les paquets reçus. La figure 2 ci-dessous illustre cette attaque.

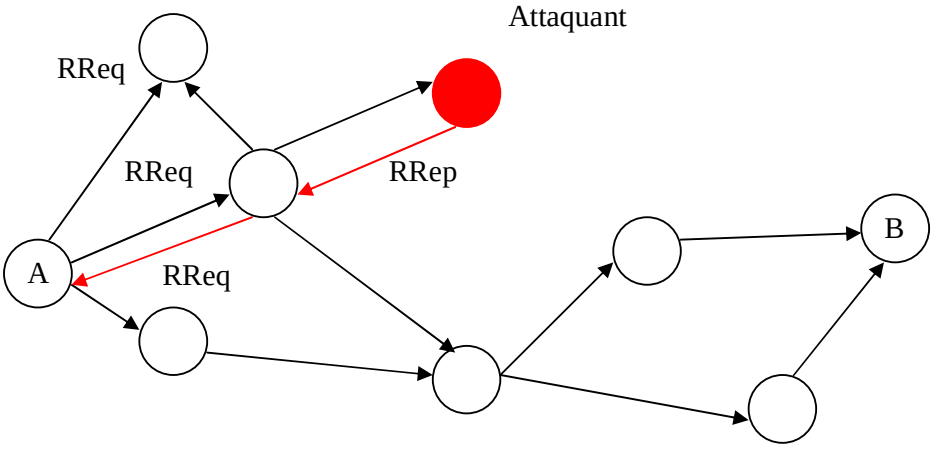


Figure 2 : Attaque du trou noir effectué par un seul attaquant.

II-3-3 **Replay attack (rejeu)**

Un attaquant qui effectue un rejeu injecte dans le trafic les paquets qui ont été capturés précédemment. Cette attaque vise habituellement la fraîcheur des itinéraires, mais peut également être employée pour miner les solutions mal conçues de sécurité.

II-3-4 **Wormhole attack**

L'attaque du trou de ver, wormhole [5, 11, 18] consiste à créer au sein d'un réseau un tunnel entre deux noeuds qui ne sont des voisins directs. Le routage de paquets à travers ce tunnel modifie la topologie logique du réseau. Le tunnel a généralement un débit plus élevé que les liens légitimes. L'attaque de trou de ver est l'une des plus puissantes attaques présentées ici puisqu'elle comporte la coopération entre deux noeuds malveillants qui participent au réseau. En effet, un attaquant A capture des paquets du trafic à un point du réseau et perce un tunnel vers un autre point dans le réseau (noeud B), qui partage une liaison privée avec A. La connectivité des nœuds du réseau établit des itinéraires qui sont sous le contrôle total des deux attaquants (voir la figure 3 ci-dessous).

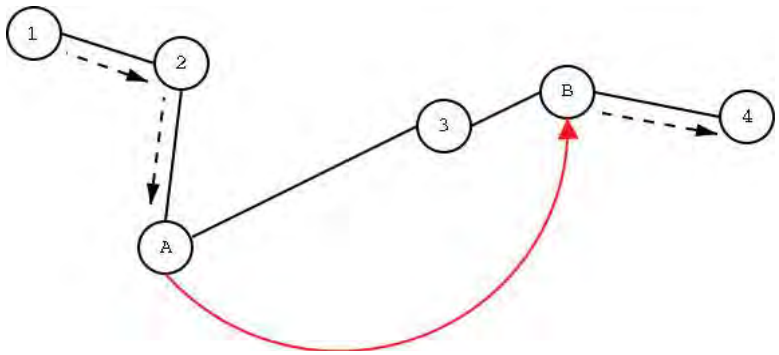


Figure 3 : Attaque du trou de ver (wormhole) par collaboration des deux attaquants A et B.

Il n'existe pas à notre connaissance un protocole qui résiste à ce type d'attaque. Cependant un mécanisme de protection contre cette attaque a été proposé par Hu et al dans [22].

II-3-5 **Blackmail (chantage)**

Cette attaque vise les protocoles de routage qui emploient des mécanismes pour l'identification des noeuds malveillants en propageant les messages qui essayent de rendre les nœuds valables du réseau en nœuds illégitimes. Un attaquant peut fabriquer de tels messages, essayer d'isoler des noeuds légitimes du réseau. La propriété de sécurité, la non-répudiation peut s'avérer utile dans ces cas-ci puisqu'elle lie un noeud aux messages qu'il a produits.

II-3-6 **Déni de service**

Les attaques de type déni de service [5] visent la rupture complète de la fonction de routage du réseau mobile ad hoc. Les exemples spécifiques d'attaques de type déni de service incluent le débordement de table de routage, de la privation de sommeil, etc. Dans une attaque de débordement de table de routage, le noeud malveillant inonde le réseau avec de faux paquets de création de route afin de consommer les ressources des noeuds participants et perturber l'établissement des routes légitimes. La privation de sommeil vise la consommation des batteries d'un noeud spécifique en le maintenant constamment occupé par des décisions de routage.

II-3-7 **Routing table poisoning (Empoisonnement de table de routage)**

Les protocoles de routage maintiennent les tables qui détiennent l'information concernant des itinéraires du réseau. Dans des attaques d'empoisonnement [5] les noeuds malveillants produisent et envoient du trafic de signalisation fabriqué ou alors modifient les messages légitimes d'autres noeuds, afin de créer les entrées fausses dans les tables des noeuds participants. Par exemple, un attaquant peut envoyer les mises à jour de la table de routage qui ne correspondent pas aux changements réels de la topologie du réseau ad hoc. Les attaques d'empoisonnement de table de routage peuvent avoir comme conséquence le choix des itinéraires non optimaux, création des boucles de routage, goulots d'étranglement et également diviser certaines parties du réseau.

Il devient évident, qu'il est extrêmement difficile de lutter contre toutes les attaques relatées ci-dessus ; cependant des tentatives ont été faites en vue de réduire au moins l'impact de certaines attaques. Nous allons à présent énumérer les différents protocoles sécurisés les

plus utilisés et étudier les mécanismes qu'ils ont mis en application pour parer les attaques mentionnées ci-dessus.

II-4 **Les protocoles de routage sécurisés au niveau des MANETs**

Dans la littérature, il existe plusieurs protocoles de routage sécurisés par l'ajout d'une signature ou d'un digest dans les paquets de contrôle. Nous pouvons citer comme exemple, SRP, SLSP, SAODV, ARAN, ARIADNE, SEAD, SAR etc. Dans ce qui suit, nous allons présenter ces protocoles.

II-4-1 **Le protocole SRP (Secure Routing Protocol)**

Le protocole SRP [5, 18, 19, 21, 32] (Secure Routing Protocol) est proposé comme une extension au protocole de routage ad hoc réactif, en particulier le protocole DSR. Le protocole SRP sécurise la phase de découverte de route. Un hôte initialisant une session est alors capable d'identifier des réponses de routes malveillantes. La technique utilisée nécessite des clés privées partagées par les hôtes sources S et destinataires T. En effet, SRP compte sur la disponibilité d'une Association de sécurité SA[10] (Security Association) entre le noeud source (S) et le nœud destinataire (T). Un exemple de SA est une clef symétrique secrète dérivée en utilisant les clefs publiques de S et T. Ainsi S et T peuvent authentifier les messages de l'un et de l'autre par l'intermédiaire de cette clef symétrique partagée, en utilisant un MAC.

Les noeuds intermédiaires qui transmettent par relais la requête RREQ vers la destination ajoutent leur adresse IP au RREQ [21, 23] avant de le réexpédier, mais n'ajoutent aucune information d'authentification. A la réception d'un RREQ, le noeud destinataire vérifie l'intégrité et l'authenticité du RREQ en calculant le MAC des champs de requête, et en les comparant à la valeur du MAC contenu dans l'en-tête SRP. Si le RREQ est valide, le destinataire lance une réponse RREP où il met le MAC du chemin traversé. Il est nécessaire que les RREPs prennent l'itinéraire inverse pour revenir à la source.

Le protocole SRP souffre de l'attaque du trou noir : n'importe quel noeud intermédiaire qui expédie le RREQ pourrait ajouter beaucoup de faux au message. Ceci rend la probabilité de cet itinéraire d'être choisi comme itinéraire vers T par S très basse. C'est

[10] Une association de sécurité est une relation unidirectionnelle entre un émetteur et un récepteur.

parce qu'il n'y a aucune authentification des messages point à point. SRP n'est pas non plus immunisé contre l'attaque du trou de ver (Wormhole).

II-4-2 **Le protocole ARIADNE**

ARIADNE [5, 19, 32] est un protocole de routage ad hoc réactif sécurisé basé sur DSR. Il garantit que l'initiateur peut authentifier chaque noeud intermédiaire sur le chemin vers la destination, dans le message de RREP et qu'aucun noeud intermédiaire ne peut enlever un noeud précédent dans la liste des noeuds dans les messages de RREQ ou de RREP.

ARIADNE permet d'authentifier les messages de routage suivant trois schémas. Le premier utilise des secrets partagés entre chaque paire de noeuds ; le deuxième une combinaison de secret partagé entre les noeuds communiquant et de diffusions authentifiées (TESLA) ; le troisième utilise des signatures numériques.

TESLA qui est un protocole d'authentification « broadcast » utilisé par ARIADNE, dépend de la synchronisation des horloges qui est condition essentielle à son fonctionnement, et un mécanisme permettant de distribuer des clés privées et une clé TESLA (publique) pour chaque nœud. Un noeud partage deux clefs avec chaque autre noeud pour l'authentification des messages. Une signature du paquet de découverte d'itinéraire est produite et apposée avec le paquet original. Le noeud de réception recalcule le HMAC en utilisant la clef envoyée au destinataire et vérifie qu'aucun noeud ne change le paquet RReq et vérifie également l'authenticité du noeud. L'authentification avec TESLA est basé sur l'hypothèse à savoir le noeud source communique avec un noeud destinaire et le noeud destinaire n'est pas malicieux. Si le noeud destinaire est malveillant l'authentification échoue.

Ainsi ARIADNE empêche des attaques de modifications de message mais à un coût certain. La taille des messages de routage augmente linéairement avec la longueur de l'itinéraire au fur et à mesure que tous les MAC sont apposés au message.

II-4-3 **Le protocole ARAN**

ARAN [5, 32] (Authenticated Routing for Ad hoc Networks) est un protocole proactif de routage ad hoc sécurisé. ARAN utilise un serveur de certificats reconnu par tous les nœuds. Avant de joindre le réseau ad hoc, chaque hôte doit obtenir un certificat signé par le serveur. Dans ARAN chaque nœud signe les paquets de découverte et réponse de route avant de les retransmettre. Le protocole ARAN fournit l'authentification point à point des messages de

routage, en utilisant des signatures numériques. Ainsi chaque noeud qui expédie le message RREQ ou le message RREP vérifie la signature du noeud précédent, l'enlève si elle est valide, et puis ajoute sa propre signature.

Le protocole ARAN empêche les attaques de type modification, fabrication, usurpation d'identité. Cependant l'utilisation de la cryptographie asymétrique fait de lui, un protocole très coûteux en terme de CPU et d'usage de la batterie. Malheureusement, ARAN ne résiste pas au Wormhole attaque.

II-4-5 **Le protocole SOADV**

SAODV [5, 32] (Secure Ad hoc On-demand Distance Vector) a été proposé pour sécuriser le protocole AODV. En effet, le protocole SAODV emploie des signatures numériques pour authentifier les champs non mutables et des fonctions de hachage[11] pour authentifier les champs mutables des messages de routage. Ainsi quand un nœud transmet un message RREQ ou un message RREP, il met le champ max hop count identifiable dans la figure 4 ci-dessous, égale au champ TTL et génère un nombre aléatoire α qu'il affecte au champ hash. Ensuite, il applique le hachage $h^{MaxHopCount}(\alpha)$ au paquet de RREQ ou RREP.

N'importe quel noeud intermédiaire relayant le paquet augmente le hop count par 1 et remplace la valeur précédente hash par h (hash). Pour vérifier l'authenticité du hop count courant k, n'importe quel noeud peut vérifier si $h^{MaxHopCount}(\alpha)$ est égal à $h^{MaxHopCount-k}(s)$. Le protocole empêche les noeuds de diminuer le max hop count, mais pas de le garder inchangé, ou de l'augmenter, ce qui va aussi avoir comme conséquence l'impuissance face à des attaques comme le wormhole.

[11] Une fonction de hachage permet de réduire (hacher) toute donnée binaire (un fichier ou un message) en un mot binaire d'une taille fixée

Type	Length	Hash Function	Max Hop Count
Top Hash[12]			
Signature			
Hash			

Figure 4. Entête du protocole SAODV après hachage et ajout de signature.

Cependant, nous pouvons remarquer que le principal problème pour sécuriser les protocoles de routage à la demande tel que AODV est le fait que ces derniers autorisent les nœuds intermédiaires à répondre à la demande de route, sans avoir à signer la réponse pour le compte du destinataire. C'est ainsi que certaines solutions proposent une autorisation de signature de la réponse par les nœuds intermédiaires.

II-4-6 **Le protocole SEAD**

SEAD [5, 32] (Secure Efficient Ad hoc Distance Vector Routing) est un protocole proactif de routage ad hoc sécurisé, basé sur DSDV. SEAD permet d'authentifier l'émetteur d'une information de routage, et autres informations fournies telles que le nombre de nœuds intermédiaires et les numéros de séquence. Afin d'éviter les opérations coûteuses dues aux signatures, SEAD utilise des chaînes de hachage. Ainsi SEAD fait l'hypothèse d'un mécanisme permettant à un nœud de distribuer un élément authentique de la chaîne de hachage. L'idée de base de SEAD est d'authentifier le numéro de séquence et la métrique des messages de mise à jour des tables de routage utilisant des chaînes de hachage.

Cependant, la source de chaque message de mise à jour de la table de routage doit être authentifiée. Dans la mesure où, un attaquant peut créer une boucle de routage à travers une usurpation d'identité. Les auteurs ont proposé deux approches différentes, pour fournir une authentification des nœuds. Le premier se base sur un mécanisme de « broadcast authentification » tel que TELSA. Le second se base sur l'utilisation de MAC (Message

[12] Le champ Top Hash est le résultat de l'application de la fonction de hachage $h^{MaxHopCount}$ (hash)

Authentification Code), en admettant une clé secrète partagée entre chaque couple de nœuds dans le réseau. SEAD ne résiste pas à l'attaque du trou de ver (Wormhole attaque).

II-4-7 **Le protocole de routage SLSP**

Le protocole de routage SLSP (Secure Link State Routing Protocol) [5, 32] a été proposé pour sécuriser les protocoles de routages proactifs au niveau des réseaux mobiles ad hoc. Il permet de sécuriser la découverte et la distribution des informations échangées sur l'état des liens entre nœuds du réseau. La principale exigence de SLSP est l'existence d'une paire de clés asymétriques pour chaque interface réseau d'un nœud donné.

Cependant, SLSP se limite seulement à la sécurisation du processus de découverte de topologie; ainsi un nœud malveillant participant à la transmission des données dans le réseau ne peut être détecté.

SLSP peut logiquement être divisé en trois composants à savoir la distribution des clés publiques, la découverte de voisins et les mises à jour des états des liens. Pour éviter, le besoin d'un serveur de gestion des clés, les nœuds envoient leurs certificats en « broadcast» dans les limites de leur zone.

II-4-8 **Le protocole de routage SAR**

SAR (Security-aware Ad hoc Routing) [5, 32] est une approche initiée pour le routage ad hoc introduisant la notion de « métrique de sécurité » au niveau du processus de découverte de route, d'opération de maintenance de route en offrant un routage sécurisé.

SAR est proposé comme étant une extension aux protocoles de routage à la demande tels que AODV ou DSR dans la mesure où il introduit une métrique de sécurité dans les messages de demande de route (RREQ). En effet, l'initiateur envoie en broadcast le message RREQ en ajoutant un champ RQ_SEC_REQUIREMENT qui indique le niveau de sécurité exigé pour la route à découvrir. Ainsi, le nœud voisin qui reçoit le paquet, vérifie s'il peut satisfaire les conditions de sécurité. Si oui, il participe à la découverte de route et fait suivre le message en broadcast à ses voisins en y ajoutant un champ RQ_SEC_GURANTEE pour indiquer le niveau de sécurité maximum qu'il peut fournir. Cependant, si un nœud n'est pas en mesure de garantir la sécurité demandée, il supprime le RREQ. Ainsi, quand le nœud destination reçoit le paquet, il est sûr que la route initiée par la source existe et, est conforme aux mesures de sécurités demandées.

Enfin, la destination envoie un message de route reply (RREP) en ajoutant elle aussi, un champ RQ_SEC_GURANTEE pour indiquer le niveau de sécurité maximum de la route trouvée. Le RREQ prend le chemin inverse, constitué seulement des nœuds qui sont autorisés à participer à la découverte de route.

La métrique de sécurité est définie à partir des attributs reflétant certains mécanismes de sécurité à savoir l'authentification, la non répudiation, l'intégrité etc.

A côté de protocoles ci-dessus, il existe des mécanismes qui peuvent être utilisés par ceux-ci pour renforcer leur niveau de sécurité.

II-5 **Packet leashes**

Dans cette section, nous introduisons la notion de Packet leashes [22] qui n'est pas un protocole complet mais un mécanisme général qui peut être utilisée par un protocole existant pour se protéger contre l'attaque du trou de ver (wormhole). L'idée principale de cette solution est d'ajouter une information supplémentaire à chaque paquet envoyée pour permettre au nœud récepteur de déterminer si un paquet a traversé une distance irréaliste. Les auteurs ont proposé deux approches : temporel (Temporal leashes) et géographique (Geographical leashes).

II-5-1 **L'approche géographique** (Geographical leashes).

La première méthode de construction du Packet leashes est l'utilisation de la localisation géographique de l'information, fournie par des outils tels que le GPS (Global Positioning System) et une synchronisation d'horloge. Une estampille temporelle et l'information de localisation de la source sont ajoutées à chaque paquet envoyé. Le récepteur peut ensuite vérifier la distance traversée par le paquet pendant le dernier saut. Tous les nœuds du réseau ad hoc doivent au préalable s'approprier d'un matériel pour pouvoir suivre leur localisation selon un schéma unifié.

En effet quand un nœud envoie un paquet, il y ajoute sa position locale $\mathbf{p_s}$ et le moment $\mathbf{t_s}$ où il a envoyé le paquet. A la réception du paquet, le récepteur compare les valeurs reçues avec respectivement sa position locale $\mathbf{p_r}$ et $\mathbf{t_r}$ le temps correspondant au moment où il a reçu le paquet. Si les horloges des nœuds récepteur et envoyeur sont synchronisées avec une marge d'erreur égale à $\pm\ \mathbf{\Delta}\ (\mathbf{t_s} - \mathbf{t_r})$, et $\mathbf{v}$ est la limite supérieure de la vitesse de tout nœud, alors

l'hôte récepteur peut évaluer la limite supérieure de la distance entre lui et l'envoyeur $\mathbf{d_{sr}}$. Cette dernière est spécifiquement liée à l'estampille temporelle $\mathbf{t_s}$ attachée au paquet, au temps de réception local $\mathbf{t_r}$, à l'erreur maximale relative à l'information de localisation $\mathbf{\Delta}$, aux positions locales du récepteur et de l'envoyeur $\mathbf{p_s}$ et $\mathbf{p_r}$.
Ainsi $\mathbf{d_{sr}}$ est évaluée par :

$$\mathbf{d_{sr} \leq ||p_s - p_r|| + 2v*(t_r - t_s + \Delta) + \delta.} \quad \mathbf{(1)}$$

Une signature numérique ou toute autre technique d'authentification, doit être employée pour permettre à un récepteur d'authentifier la position et l'estampille temporelle dans le paquet reçu.

La synchronisation d'horloge dans cette méthode n'a pas besoin d'être aussi précise que celle de l'approche temporelle puisque l'information de localisation est aussi utilisée dans le calcul de la distance entre la source et la destination.

II-5-2 **L'approche temporelle** (temporal leashes)

Selon l'approche temporelle, un nœud ajoute une estampille temporelle extrêmement précise à chaque paquet sortant. Le nœud récepteur peut alors authentifier la distance traversée en se basant sur le temps d'expiration $\mathbf{t_e}$ et le fait que la distance est liée à la vitesse de la lumière **c**. Ainsi, il est clair que la solution temporelle requiert la mise en place d'une synchronisation d'horloge très précise dans l'ordre des centaines de nanosecondes entre les nœuds participants.

En effet si nous considérons un expéditeur qui veut envoyer un paquet avec temporal leashes, il doit empêcher le paquet de voyager plus loin que la distance **L**. Ainsi nous avons

$$\mathbf{L > L_{min} = \Delta * c} \quad \mathbf{(2)}$$

Tous les nœuds sont synchronisés avec un seuil maximal d'erreur de synchronisation égale à $\mathbf{\Delta}$, avec **c** égale à la vitesse de propagation de la lumière de notre signal sans fil (c'est à dire la vitesse de la lumière dans l'air, qui est très proche de celle de la lumière dans le vide). Quand l'expéditeur envoie le paquet à un temps local $\mathbf{t_s}$, il doit estimer le temps d'expiration de paquet comme suit :

$$t_e = t_s + (L/c) - \Delta \qquad (3)$$

Quand le récepteur reçoit le paquet au temps local t_R, il vérifie si le temporal leash du paquet n'a pas expiré (c à d $t_R < t_e$), autrement il rejette le paquet. Ceci suppose que les délais d'envoi et de réception sont négligeables. Le récepteur a besoin d'authentifier le temps d'expiration t_e, sinon un attaquant pourrait facilement le changer ce que pourrait faciliter l'attaque du trou de ver (wormhole).

En général, Packet leashes fournit une solution complète au problème de l'attaque Wormhole dans les réseaux mobiles ad hoc. Leur seul besoin est une horloge synchrone précise et la connaissance de la localisation géographique qui manquent dans tous les protocoles sécurisés étudiés.

II-6 **Watchdog et Pathrater**

Les systèmes Watchdog et Pathrater [18, 19, 32] constituent deux extensions du protocole de routage DSR qui tentent de détecter et d'atténuer les effets des nœuds qui refusent de relayer les messages alors qu'ils avaient accepté de le faire.

Le Watchdog a la responsabilité de vérifier, dans un chemin que le nœud suivant a bel et bien relayé le message. Le Pathrater évalue les résultats du Watchdog et choisit ainsi le chemin le plus fiable pour délivrer les paquets.

Chaque nœud participant au réseau mobile ad-hoc emploie les fonctionnalités du Watchdog pour vérifier que les nœuds relayent correctement les paquets reçus. En effet, quand un nœud transmet un paquet à un nœud suivant, il essaie ensuite d'écouter intuitivement si ce dernier va en faire autant. De plus s'il y'a aucun mécanisme cryptographique entre les liens, le nœud écouteur a la capacité de vérifier que le message n'a as été modifié par le relayeur avant la transmission. Le Watchdog d'un nœud maintient les copies des nouveaux paquets relayés et les compare avec les paquets transmis par les nœuds voisins. Si un nœud n'a pas relayé un paquet pendant un intervalle de temps donné (timeout), le Watchdog du nœud qui est supposé l'entendre, incrémente le taux de défaillances du nœud spécifié. Cela signifie effectivement que chaque nœud de réseau ad hoc maintient un taux d'évaluation de la fiabilité de chaque autre nœud qui est en mesure de relayer les paquets. Un nœud est identifié comme malveillant quand son taux de défaillance dépasse un certain seuil de bande passante.

Le Pathrater choisit le chemin pour les paquets à relayer en se basant sur le taux de défaillances fourni par le mécanisme du Watchdog. La mesure du chemin par le Pathrater se fait en calculant la moyenne des taux de défaillances des nœuds faisant partie du chemin ou en utilisant un algorithme de recherche du plus court chemin, si aucun taux de défaillance n'est collecté.

II-7 **TESLA** (Time Efficient Stream Loss-tolerant Authentification)

TESLA [21, 22, 33] a été proposé par Perrig et al. Il 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 **δ**. La valeur **δ** est calculée de manière à ce que l'on soit sûr que le destinataire a reçu le message avant la divulgation de la clé, cette condition garantit l'intégrité du message. Le temps **δ** ne doit pas être trop important pour limiter les latences dans le réseau.

En effet un destinataire doit attendre la divulgation de la clé secrète avant de pouvoir effectivement traiter un message. La figure 5 ci-dessous décrit le fonctionnement de TESLA. En fait, la clé secrète utilisée pour le MAC est issue d'une chaîne de clés. Un élément de la chaîne $\mathbf{k_i}$ est calculé de la manière suivante où **F** est une fonction de hachage.

$$\mathbf{k_{i+1} = F\ (k_i)} \qquad \mathbf{(4)}$$

L'élément initial $\mathbf{k_n}$ est choisi par l'émetteur. Celui-ci va utiliser ces clés par ordre croissant c'est à dire en commençant par k_1. A la réception, le destinataire pourra vérifier la relation (4) où $\mathbf{k_i}$ est la clé dernièrement reçue et $\mathbf{k_{i+1}}$ correspond à la clé précédente.

Cette condition assure que la clé $\mathbf{k_i}$ fait bien partie de la chaîne de clés de l'émetteur (figure 5), ce qui garantit, en plus de l'intégrité, la propriété d'authentification du paquet. Il est à noter que ce processus doit être initialisé par l'authentification du premier paquet émis à l'aide d'une signature numérique.

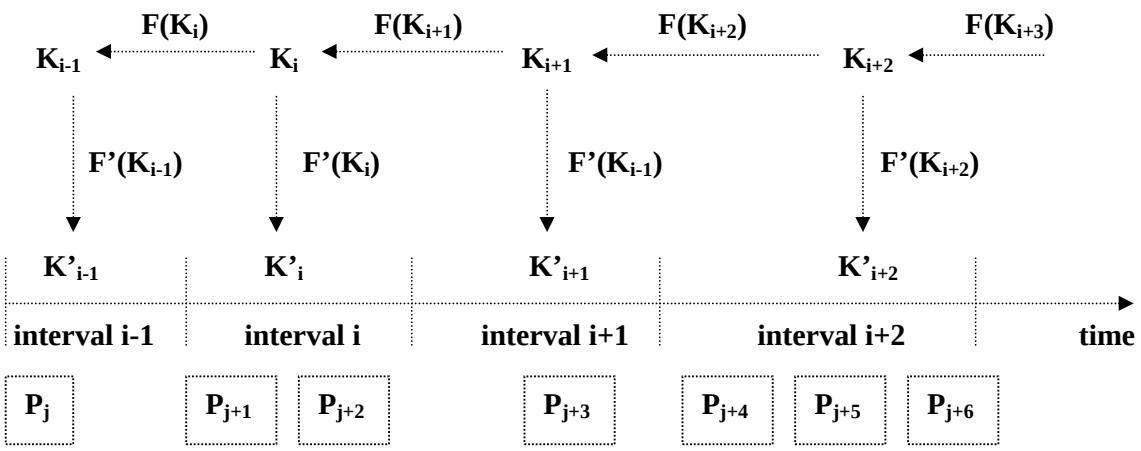


Figure 5 : Mécanisme de génération de clé par TESLA par une chaîne de hachage.

Chaque clé dans la chaîne principale à sens unique correspond à un intervalle de temps. Chaque fois qu'un expéditeur annonce un message, il appose un MAC au message, en utilisant la clé correspondant à l'intervalle de temps courant. Par exemple pour le paquet $\mathbf{P_{j+3}}$, l'émetteur calcule le MAC en utilisant la clé ($\mathbf{K'_{i+1}}$).

Un expéditeur envoie le paquet P_{j+1} dans l'intervalle i. Quand le récepteur reçoit le paquet P_{j+1}, il peut employer la clé authentifiée $\mathbf{K_i}$ annoncée dans le paquet P_{j+1} pour déterminer i. Il vérifie alors le dernier intervalle de temps possible x depuis l'expédition du paquet (basé sur l'horloge lâchement synchronisée). Si (x < i+d) avec d la limite du retard sur la clé, alors le paquet est sauvé. Ainsi l'expéditeur n'a pas encore atteint l'intervalle où il révèle le $\mathbf{K_i}$ principal, la clef qui vérifiera le paquet $\mathbf{P_j}$ avec :

$$\mathbf{P_j = \{ M_j \,||\, MAC(K'_i, M_j) \,||\, K_{i-d} \}.} \qquad \mathbf{(5)}$$

II-7-1 **Synchronisation des horloges avec TESLA**

Nous décrivons maintenant un protocole simple à double sens de la synchronisation[13] qui répond à l'exigence pour TESLA [28], où le récepteur connaît une limite supérieure sur l'horloge de l'expéditeur (figure 6).

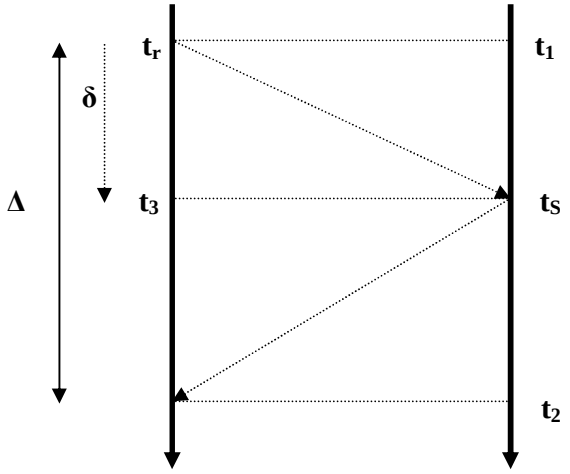


Figure 6 : La synchronisation directe entre une source et une destination.

Le récepteur publie une demande de la synchronisation **(sync, t_r)** au temps local t_r, lorsque l'horloge de l'expéditeur est au t_1 de temps. L'expéditeur répond à la demande en donnant avec précision son temps de réception t_s **(sync, t_r, t_s)**. Avec TESLA, le récepteur est seulement intéressé par une limite supérieure le temps de l'expéditeur. A la réception, avec son temps courant t_R, il évalue la limite supérieure le temps de l'expédition courant comme suit :

$$t_s \leq t_R - t_r + \Delta \qquad (6)$$

Δ l'erreur maximale (réelle) du temps de synchronisation

II-8 **Comparaison entre les propositions sécurisées**

Cette section présente un bref résumé sur les protocoles de routage ad hoc sécurisés précédemment présentés. Chaque protocole a un ensemble de conditions opérationnelles

[13] La procédure de synchronisation revient à s'assurer que deux horloges immobiles dans un repère indiquant le même temps correspondent à des événements simultanés.

différentes et assure la protection contre différentes attaques en utilisant des approches particulières. Par conséquent, une comparaison détaillée peut permettre de déterminer l'adéquation d'un protocole pour un domaine spécifique d'application.

II-8-1 **Exigences et paramètres des protocoles sécurisés de routage ad hoc**

En effet, les protocoles sécurisés et les mécanismes étudiés ici, proposent des solutions au problème de sécurité du routage ad hoc, basées sur certaines hypothèses et des exigences opérationnelles.

Nous résumons dans le tableau 2 suivant, certaines propriétés et les exigences de chacun de ces protocoles précédemment étudiés.

Tableau 2. Paramètres et exigences des protocoles de routage sécurisés dans les MANETs.

Solution proposée	Approche de routage	Métrique du routage	Exigences
SRP	routage A la demande	Distance	Existence d'une association de sécurité entre chaque noeud source et destination. Un nœud malicieux ne doit pas être en mesure de corrompre un message de traitement du routage du protocole.
ARAN	routage A la demande	---	Mise en place d'une autorité de certification (CA) de confiance. Chaque nœud doit connaître à priori la clé publique du CA
ARIADNE	routage A la demande	Distance	Synchronisation de l'horloge et existence d'un secret partagé entre paire de nœuds. En plus d'une clé TESLA authentifiée par chaque nœud du réseau et un principe de circuit authentique de découverte de route par lequel le nœud forwarde le RREQ. La clé TESLA est distribuée aux participants via un serveur de distribution de clé mis en place.
SAODV	routage A la demande	Distance	Mise en place d'un système de gestion de clés pour l'acquisition et vérification des clés publiques.
SEAD	Echange De tables	Distance	Synchronisation d'horloge ou un secret partagé entre paire de nœuds.
SLSP	Echange De tables	Distance	Les clés publiques des nœuds doivent être certifiées. Pas de collusion entre nœuds malicieux.
SAR	routage A la demande	Une métrique de sécurité	Mécanisme de distribution de clé ou de secret partagé
TESLA	NE	----	Mécanisme d'authentification par broadcast et une synchronisation d'horloge
Packet leashes	NA	NA	Synchronisation d'horloge extrêmement précise et information de localisation géographique
Watchdog et Pathrater	A la demande	Fiabilité de route	Pas de collaboration entre les nœuds malicieux Taux de défaillances des nœuds.

Il est évident de la comparaison que la plupart des protocoles exigent l'existence d'un tiers en ligne de confiance, par exemple une autorité de certification, afin de faciliter l'acquisition et la vérification des clefs publiques des noeuds qui participent au réseau ad hoc. Les protocoles qui entrent dans cette catégorie sont ARAN, SAR, SEAD, SAODV. La condition opérationnelle de SRP est semblable puisqu'elle a besoin d'une association de sécurité préétablie entre chaque nœud source et noeud destinataire. Le protocole de SEAD exige l'existence d'un schéma de distribution des clés pour l'authentification entre les noeuds. Ceci peut être réalisé avec un mécanisme d'authentification d'émission tel que TESLA, qui exige des noeuds du réseau d'avoir des horloges synchronisées. ARIADNE exige deux clefs secrètes partagées entre chaque paire de noeuds et la synchronisation afin d'employer TESLA comme méthode pour authentifier des messages à diffusion générale. Quand au fonctionnement réussi du Watchdog et Pathrater il exige qu'il n'y ait jamais de collaboration entre les nœuds malicieux afin d'effectuer des attaques sur le routage.

II-8-2 **Analyse de sécurité**

Dans cette section, nous présentons une analyse des valeurs concernant le comportement des protocoles examinés et de leur applicabilité dans les environnements ad hoc mobiles. Dans le meilleur des cas, un protocole de routage sécurisé ad hoc devrait pouvoir assurer la protection contre toutes les catégories d'attaques précédemment mentionnées dans la section II-3. Le tableau 3 suivant présente les réactions des protocoles sécurisés de routage face aux différentes attaques.

Tableau 3. Défense contre les attaques.

Protocoles	**Attaques**						
	Location disclosure	*Black hole*	*Replay attack*	*Wormhole attack*	*Blackmail*	*Denis de service*	*Routing table poisoning*
ARAN	Non	Non	Oui	Non	NA	Non	Oui
SRP	Non	Non	Oui	Non	NA	Oui	Oui
SEAD	Non	Non	Oui	Non	NA	Oui	Oui
ARIADNE	Non	Non	Oui	Non	NA	Oui	Oui
SAODV	Non	Non	Oui	Non	NA	Non	Oui
SLSP	Non	Non	Oui	Non	NA	Oui	Oui
Watchdog et Pathrater	Non	Oui	Non	Non	Non	Non	Non
SAR	Non	Non	Oui	Non	NA	Non	Oui
Packet leashes	NA	NA	NA	Oui	NA	NA	NA
TESLA	Non	Oui	Oui	Non	NA	Non	Oui

En réalité, étant donné la nature fortement dynamique des réseaux mobiles ad hoc et les différents scénarios de leur application, il est difficile de concevoir une solution générale qui peut assurer la protection contre tous types d'attaques. Ainsi de par ces résultats, nous nous sommes proposés, en guise de contribution dans ce travail, d'effectuer une amélioration d'un des protocoles sécurisés. Alors, nous avons pris l'exemple de ARIADNE, identifié un de ces problèmes qu'est la non-résistance au wormhole attaque et essayé d'améliorer ces performances.

III- **Proposition de solution pour ARIADNE contre l'attaque Wormhole**

Après avoir effectué une étude minutieuse sur la sécurité du routage en l'occurrence les protocoles de routage sécurisés et les différents types d'attaques possibles pour ces protocoles au niveau des MANETs; nous essayons maintenant d'identifier un problème très fréquent et, dans la mesure du possible apporter une solution. Ce qui va entrer dans le cadre de notre contribution dans ce vaste océan de recherche qu'est la sécurisation du routage au niveau de MANETs.

Cependant, après avoir identifié les possibles attaques qu'un nœud malicieux peut mener au niveau des protocoles de routage malgré les mesures de sécurité mises en place, nous sommes arrivés à la conclusion que l'attaque du trou de ver (wormhole) était la plus difficile à déceler. De ce fait, parmi tous ces protocoles sécurisés relatés dans ce document, il n'y a pas un qui peut résister à cette attaque.

De par ce constat, nous nous sommes proposés de prendre un protocole sécurisé en l'occurrence le protocole ARIADNE et de lui ajouter certains mécanismes de sécurité afin qu'il puisse lutter contre le wormhole.

III-1 **Rappel sur les insuffisances de ARIADNE**

Comme nous l'avons remarqué, ARIADNE ne résiste point à l'attaque du trou de ver. Ainsi des études ont montré que la seule solution pour que ce protocole empêche cette attaque, est la mise en place d'une politique de synchronisation très précise des horloges dans TESLA.

Pour rappel, le mécanisme de synchronisation utilisé par TESLA est le suivant :

- Le récepteur envoie **(sync, t_r)** au temps $\mathbf{t_r}$;
- L'expéditeur répond à la demande avec son temps de réception $\mathbf{t_s}$ **(sync, t_r, t_s)** ;
- Le récepteur, avec son temps courant $\mathbf{t_R}$, évalue la limite supérieure le temps de l'expéditeur courant comme suit : $\mathbf{t_s \leq t_R - t_r + \Delta}$

Avec $\mathbf{\Delta}$ l'erreur maximale (réelle) du temps de synchronisation.

IV-2 **Les bases de la solution proposée**

La solution proposée se base sur le concept de packet leashes qui n'est rien d'autre qu'un mécanisme permettant de contrer l'attaque du trou de ver. Ce mécanisme peut être utilisé par tout protocole de routage sécurisé en vue de se protéger contre le wormhole.

Cependant, l'idée s'inspire de l'approche temporelle (temporal leashes) qui se résume en deux phases qui sont :

- Synchronisation des horloges avec une erreur sur le temps de synchronisation égale à Δ ;
- Implémentation du temps d'expiration de paquet $\mathbf{t_e}$ dans la synchronisation avec

$$\mathbf{t_e = t_s + (L/c) - \Delta} \quad \text{et} \quad \mathbf{L > L_{min} = \Delta * c}$$

, où **c** vitesse de la lumière et **L** la distance maximale que peuvent parcourir les paquets.

III-3 **La solution introduite dans la synchronisation au niveau de TESLA**

La synchronisation des horloges dans TESLA à été modifiée grâce au temporal leashes en introduisant la notion de temps d'expiration comme suit :

- Le destinataire annonce la demande de synchronisation en envoyant à la source (**sync, t_r, L**) au temps $\mathbf{t_r}$;
- La source vérifie alors la condition ($\mathbf{L > L_{min} = \Delta * c}$), calcule le temps d'expiration $\mathbf{t_e}$ avec son temps de réception $\mathbf{t_s}$ ($\mathbf{t_e = t_s + L/c - \Delta}$) et envoie au destinataire son accord de se synchroniser (**sync, t_r, L, t_e, t_s**) ;
- Le destinataire teste enfin si ($\mathbf{t_s < t_e}$) et ($\mathbf{t_s \leq t_r - t_R + \Delta}$). Si les deux conditions restent vraies, alors les horloges sont synchronisées, sinon la procédure est annulée et une nouvelle demande est envoyée si possible.

Avec toujours **c** vitesse de la lumière, $\mathbf{\Delta = t_s - t_r}$ l'erreur sur le temps de synchronisation
L la distance maximale que peuvent parcourir les paquets et $\mathbf{t_R}$ le temps courant de l'horloge du destinataire à la réception de (**sync, t_r, L, t_e, t_s**).

Cette synchronisation favorise finalement une mise en place d'une structure solide pour ARIADNE dans ses perspectives de lutter contre l'attaque du trou de ver (wormhole). L'algorithme suivant nous a permis d'implémenter cette solution dans le simulateur NS2.

III-3-1 **Algorithme de la synchronisation**

Début procédure

c = 300000 km/s = 3.10^8 m/s vitesse de la lumière,

Le nœud destinataire **D** définit une distance **L**,

D enregistre son temps local **t_r,**

D envoie la demande de synchronisation (**sync, t_r, L**) à la source **S**,

Tant que **S** reçoit (**sync, t_r, L**) **faire**

Enregistrer son temps local **t_s**,

Calculer **$\Delta = t_s - t_r$,**

Calculer L_{min} = **Δ * c,**

Si (**$L > L_{min}$**) **alors**

Calculer **$t_e = t_s + L/c + \Delta$,**

Envoyer au destinataire (**sync, t_r, L, t_s, t_e**),

Sinon

Demande rejetée,

Fin si

Fin tant que

Tant que **S** reçoit (**sync, t_r, L, t_s, t_e**) **faire**

Enregistrer son temps **t_R** au moment de la réception,

Si (**$t_s < t_e$**) et (**$t_s \leq t_r - t_R$**) **alors**

Synchronisation précise,

Valider la synchronisation,

Sinon

Pas de précision dans la synchronisation,

Envoie une nouvelle demande,

Fin si

Fin tant que

Fin procédure

III-4 **Implémentation de la solution dans NS**

Notre plus grand problème dans ce travail résidait sur comment intégrer les protocoles sécurisés ARIADNE, SEAD, ARAN dans NS2 pour effectuer la simulation. Comme nous le savons bien, les seuls protocoles implémentés dans ns-allinone sont les protocoles qui n'ont aucun mécanisme de sécurité à l'image de DSR, DSDV, TORA, AODV.

Le travail a pu être effectif grâce à une extension découverte dans le site de Rice Monarch Project Software Distributions (http://www.monarch.cs.cmu.edu/software.html). Cette extension permet de simuler les protocoles ARIADNE et SEAD. Après téléchargement du patch nommé **ariadne-sead.tgz** qui est un fichier zippé, il faudra déziper suivant les instructions se trouvant de le fichier README. Pour déziper, la commande **tar zxvf ariadne-sead.tgz** a été utilisée. Ainsi on voit deux répertoires nommés dsr et dsdv. Dans ces répertoires se trouvent respectivement les fichiers hdr_sr.h, dsragent.h, dsragent.cc, teslacache.cc et le fichier dsdv.cc. En fait, à l'exception de fichier teslacache.cc, tous les autres existaient déjà dans les répertoires dsr et dsdv qui se trouvent dans ns-2.29. Mais il faudra les remplacer avec les nouveaux fichiers du patch.

Après avoir mis ces fichiers dans les répertoires dsr et dsdv qui ont pour chemin d'accès (/home/taphakane/ns-allinone-2.29/ns-2.29). Il faut faire une recompilation de ns-allinone-2.29 en faisant à nouveau un (**./install**).

Cette nouvelle opération signalera des erreurs causées par l'intégration de ces nouveaux fichiers. Donc il faudra suivre les messages d'erreurs et essayer d'entrer dans les fichiers et corriger au fur et à mesure. Ceci demande une bonne connaissance du langage C++, car des modifications devront se faire au niveau des codes sources.

De même, l'implémentation de cette variante d'ARIADNE, n'a pas été très facile dans la mesure où les mêmes problèmes que précédemment seront rencontrés. Certains fichiers de configuration de TESLA seront modifiés afin de changer l'ancienne version de la synchronisation pour une adaptation à la solution proposée.

Ensuite nous avons comparé les résultats avec ceux obtenus avec ARIADNE.

III-4-1 **Tests et validation**

Afin de tester et de valider notre travail, nous avons procédé à une simulation de la proposition qui est une variante de ARIADNE. Cette simulation a été effectuée avec ns-allinone-2.29.

Ensuite nous avons effectué une étude de performance pour pouvoir valider ce travail. Les aspects de performance évalués dans le cadre de ce travail sont principalement : la fraction de paquets délivrés, le délai des paquets transmis de bout en bout, et la charge normale de routage.

III-4-2 **Les mesures de performance**

Trois mesures de performance ont été évaluées durant notre étude :

- **La fraction de paquets délivrés** : est le rapport des paquets de données livrés aux destinations par rapport à ceux produits par les sources comme suit :
  fraction de paquets délivrés (pdf %) = (Nbrpreçus / Nbrpenvoyés) *100 ;
- **Délai des paquets transférés de bout en bout** : Ceci donne le temps moyen que doit prendre les paquets de transfert de la source à la destination. Il peut être basé sur les paquets de routage (exécution du protocole) ou le paquet de données (efficacité du protocole de routage) ;
- **Charge normale de routage** : le nombre de paquets de routage transmis par paquet de données livré à la destination comme suit :
  charge normale de routage = (paquets de routage envoyés) / Nbrpenvoyés.

Ces mesures sont importantes pour les trafics de type best-effort[14] car elles permettent l'évaluation de la capacité des protocoles de routage. Cependant, il faut noter que ces trois métriques de performances ne sont complètement indépendantes.

[14] Le service best-effort offre le transfert de paquets mais sans aucune garantie sur le délai qui dans le pire des cas peut être infini, c'est à dire le paquet peut être perdu. Les applications utilisant ce service sont des applications élastiques, c'est à dire qu'elles peuvent s'adapter aux conditions variables de bande passante disponible et aux variations du délai des paquets.

III-4-3 **Modèle de simulation avec ns-allinone-2.29**

Effectuer une simulation dans ns2 est assez difficile et exige beaucoup de connaissances et de travaux. Des difficultés ont été rencontrées au cours de l'installation au niveau de l'inscription du code de simulation. De même obtenir des résultats utiles était plus compliqué. En effet, l'élément clé d'un réseau radio [37] est l'objet "Mobilenode" constitué de composantes comme le Link Layer (LL), Interface Queue (IfQ), MAC layer. Pour compléter le scénario il faut en outre, définir le type d'antenne employée, le type de modèle de propagation voulu, le type de protocole routage ad hoc employé par les terminaux mobiles.

Un réseau radio possède aussi un objet appelé "God" (General Operation Director) [36, 37], unique pour chaque simulation. Il s'agit d'un objet qui mémorise toutes les informations inhérentes de l'état du réseau et des nœuds et est inconnu pour tous les participants de la simulation. Tous ces paramètres ont été pris en compte dans le script de simulation comme illustré dans le tableau 4 suivant :

Tableau 4: Paramètres de simulation pour ARIADNE et la variante de ARIADNE.

Model de propagation radio	Two-ray ground r
Protocole MAC (Medium Access Control)	802.11
Type de trafic	CBR sur du UDP
Nombre de noeuds (nn)	160
Taille du terrain de simulation	(nn+20)*30
Portée de transmission	250m
Nombre de noeuds malicieux	5, 10, 15, 20, 25…..
Taille des paquets	1140 bits
Taux de données	4 paquets / sec
Délai de génération de la signature	8,5 msec
Erreur maximale de synchronisation (**Δ)**	**Δ** = 0,1s
Le nombre de bit pour le hachage	80 bits

III-4-4 **Le script de simulation**

Ecrire un code de simulation pour un réseau mobile ad hoc avec NS2 nécessite une définition de plusieurs paramètres à savoir : les types des composants réseaux, le type des antennes utilisées, le protocole de routage à étudier, le modèle de trafic dans le réseau, le modèle de mouvement utilisé par les nœuds etc. Le code pour ce document est visible au niveau de l'annexe. L'exécution de ce script de simulation génère deux fichiers à savoir le fichier trace et le fichier nam [36] (voir annexe).

Cependant avec ce fichier trace [36, 37], l'évaluation de performance n'a pas été un rude travail dans la mesure où nous avons écrit un programme java (voir annexe) qui permet de parcourir toutes les lignes (voir annexe) depuis le début pour calculer le nombre de paquets envoyés (Nbrpenvoyés), le nombre de paquets reçus (Nbrpreçus), nombre de paquets perdus (Nbrpperdus) et le nombre de paquets de routage envoyés et reçus, les délais etc. Ces données calculées permettront l'évaluation des différents critères de performance à savoir:

III-5 **Résultats et discussion**

L'exécution de notre programme java a permis de récupérer les valeurs inscrites dans les tableaux (voir annexe) qui suivent et de tracer les courbes de variation des critères de performance relatés ci-dessus en fonction du nombre de nœuds malicieux.

III-5-1 **La fraction de paquets délivrés**

Les courbes de variation pour la fraction de paquets délivrés sont visibles sur la figure 7 ci-dessous

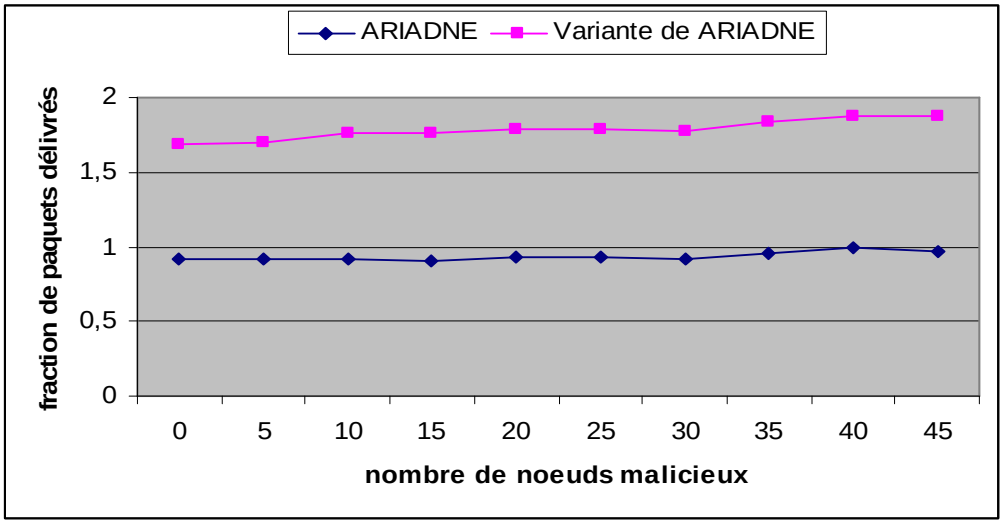


Figure7 : La fraction de paquets délivrés en fonction du nombre de nœuds malicieux.

Le graphe nous montre que ARIADNE délivre moins de paquets que sa variante avec une augmentation de 19.14 % au fur et à mesure que le nombre de nœuds malicieux augmente. Ceci est la conséquence de la synchronisation des horloges dans TESLA qui en plus d'avoir apporté une protection contre l'attaque Wormhole, résiste aussi à plusieurs autres attaques à l'image de blackhole, d'où une diminution considérable des pertes de paquets.

III-5-2 **La charge normale du routage**

Les variations de la charge normale du routage sont les suivantes:

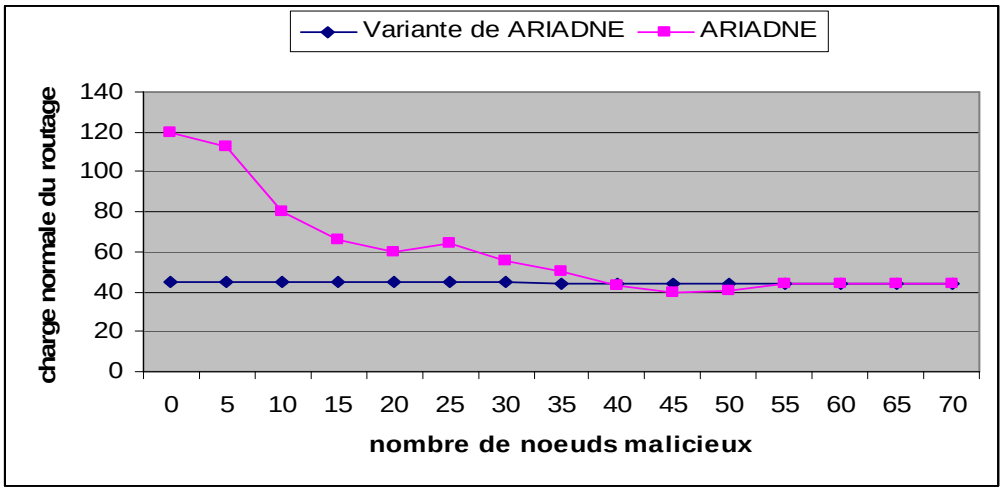


Figure 8 : La charge normale du routage en fonction du nombre de nœuds malicieux.

Les résultats montrent que la charge normale est beaucoup plus grande avec ARIADNE qu'avec la variante. Ce qui était prévisible dans la mesure où l'attaque du trou de ver augmente le nombre de paquets à router avec un nombre de nœuds malicieux considérable. En fait, avec la synchronisation précise des horloges, les nœuds malicieux vont voir leurs activités anéanties. Ainsi les fonctions réseaux vont finalement être identiques à celles avec le protocole ARIADNE.

III-5-3 **Le délai de bout en bout**

Nous avons les courbes de variation du délai de bout en bout illustrées par la figure ci-dessous

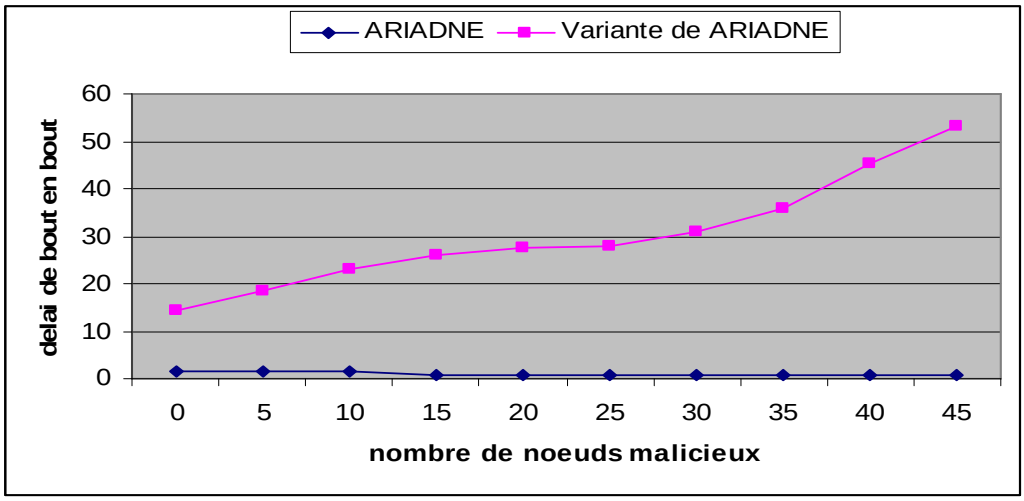


Figure 9 : Le délai de bout en bout en fonction du nombre de nœuds malicieux.

On note ici qu'au niveau de la variante de ARIADNE, le délai est plus considérable du fait que certaines opérations de calcul sont ajoutées, en l'occurrence le calcul du temps d'expiration au niveau de la synchronisation.

Cependant, il apparaît assez clairement que la solution reste gourmande en ressources. L'utilisation d'un mécanisme de synchronisation est coûteuse dans la mesure où elle nécessite plus de 10% de la charge CPU. Bien que conscient que toute sécurité à un prix, qu'elle se fait au détriment par exemple de la qualité de service, il serait intéressant de faire une étude plus approfondie sur cette contribution puis des simulations afin d'évaluer à quel point cette solution perd en performance pour pouvoir ainsi déterminer si en pratique elle est viable.

Conclusion et perspectives

Le présent travail a montré à quel point les réseaux mobiles ad hoc constituent, de par leur nature, un véritable challenge pour la sécurité informatique. Les spécificités de ces réseaux sont principalement la transmission en milieu ouvert, les topologies dynamiques, l'absence d'autorité centrale, la nécessité de bonne coopération des nœuds, l'hétérogénéité des participants avec pour certains des capacités restreintes (la durée de vie des batteries, et des ressources de stockage). Toutes ces contraintes concourent à rendre la sécurité des réseaux mobiles ad hoc difficile et complexe à appréhender. Ce sujet va devenir d'autant plus critique que le développement de tels réseaux va rapidement s'amplifier. En effet, les réseaux mobiles ad hoc sont stimulés par l'évolution rapide des technologies informatiques vers la miniaturisation et l'intégration.

Notre travail a permis de faire ressortir qu'il existe de nombreuses études théoriques mais finalement peu d'applications pratiques qui puissent satisfaire l'ensemble des contraintes inhérentes aux infrastructures ad hoc. Il est aussi apparu clairement que les mécanismes de routage constituent un point sensible pour la sécurité des réseaux mobiles ad hoc. La profusion de propositions diffusées au sein du groupe de travail MANET de l'IETF[15] montre clairement le manque de maturité sur ce sujet. Ainsi les deux axes de recherche que sont les mécanismes de routage sécurisés d'un côté, la détection des intrusions de l'autre, apparaissent comme des directions de travail primordiales. La prise en compte de ces deux problématiques doit se faire rapidement afin d'assurer un déploiement des réseaux mobiles ad hoc fiables et sécurisés.

Considérant le premier aspect nous avons, dans la deuxième partie de ce travail, étudié les attaques les plus fréquentes dans les MANETs ainsi que les protocoles et mécanismes permettant de leur résister. Ceci nous a permis de conclure qu'aucun des protocoles présentés n'offre une résistance à toutes les attaques recensées, et ouvre un grand champ d'investigations qui tendraient à doter un ou plusieurs protocoles de mécanismes permettant de lutter efficacement contre une attaque particulière, et de mieux les sécuriser de cette manière.

[15] L'Internet Engineering Task Force (**IETF)**, est un groupe informel, international, ouvert à tout individu, qui participe à l'élaboration de standards pour Internet.

Partant de ce constat nous avons proposé, dans la troisième partie de notre travail, un mécanisme permettant de doter ARIADNE (un des protocoles étudiés) d'une capacité de résistance à l'attaque de type wormhole, contribuant ainsi à sécuriser d'avantage ce protocole. Le mécanisme proposé a été mis en œuvre et certains éléments de performance le concernant ont été testés dans un environnement de simulation que nous avons mis en place. Les résultats de cette simulation montrent clairement les avantages que notre variante a sur le protocole de référence - ARIADNE - en termes de performance.

Tout au long de notre travail nous avons soulevé des problèmes qui ouvrent bien des perspectives de développement. La première concerne par exemple de la qualité de service. En effet il serait intéressant de faire une étude plus approfondie sur cette contribution afin d'évaluer à quel point cette solution pourrait perdre en performance, dans le but de déterminer si en pratique elle est viable.

Une autre perspective de ce travail serait d'améliorer les stratégies de routage des différents protocoles sécurisés en les dotant de mécanismes de sécurité leur permettant de résister aux différentes attaques qui ont été présentées. Une troisième voie serait l'étude des possibilités d'utilisation de certains protocoles dans les systèmes de localisation géographiques comme les GPS.

Bibliographie

[1] Agglou, G Tafazolli, R AGGELOU “Determining the optimal configuration for the relative distance micro discovery ad hoc routing protocol” Telecommun, Athens .

[2] RFC 3561 C. Perkins, Nokia Research Center

[3] Dr Nadjib BADACHE, Master Degree Dissertation, University of USTHB, September 2000, Le routage dans les réseaux mobiles ad hoc.

[4] RFC 2501, Scott Carson et Joseph Macker.

[5] Farooq Anjum, Petros Mouchtaris “Security for Wireless Ad hoc Networks” Wiley-interscience A John Wiley and Sons, INC., Publication.

[6] Murthy, S Garcia-Luna-Aceves, JJ “A more efficient Path Fading Algorithm” Dept of Comput Eng & Inf Sci, California Univ, Santa Cruz, CA.

[7] RFC 3626, P Jacquet, ED ; Hipercom, INRIA l'octobre 2003.

[8] Thèse présentée par Laurent Viennot à l’université Paris 7. Autour des graphes et du routage, Novembre 2005.

[9] 802.11 Wireless Network suide- The Definitive Guide Par Matthew S Gast**.**

[10] Thèse de doctorat de l’Université de Paris 6 – Pierre et Marie Curie Security Schemes for the OLSR Protocol for Ad Hoc Networks, Daniele Raffo, on September 15, 2005.

[11] Françoise SAILHAN, Localisation des ressources dans les réseaux ad hoc, Thèse Doctorale, Ecole doctorale Informatique, télécommunication, universitaire LIP.

[12] RFC 3417

[13] RFC 4728, D Johnson (Rice University), Y Hu (UIUC), D Maltz (Microsoft Research).

[14] Charles E. Perkins and Pravin Bhagwat “Highly Dynamic Destination Sequenced Distance Vector Routing (DSDV) for Mobile Computer.

[15] RFC 3561, C Perkins (Nokia Research Center), E Belding Royer, University Of California.

[16] Tayeb Lemlouma, Mini projet, Le routage dans les réseaux mobiles ad hoc, Université des Sciences et Technologie de Houari Boumèdiene, Septembre 2006.

[17] RFC 3684, R Ogier (SRI International), F Templin (Nokia), M Lewis (SRI International), Février 2004

[18] P. Papadimitratos and Z J Haas “Secure routing for mobile ad hoc networks” In Communication Networks and Distributed Systems Modelling and Simulation Conference SCS Press, 2002.

[19] Refik Molva and Pietro Michiardi, Security in Ad hoc Networks, Institut Eurecom, 2229 Route Des Crêtes, 06904 Sophia-Antipolis, France1.

[20] Françoise SAILHAN, Localisation des ressources dans les réseaux ad hoc, Thèse Doctorale, de Ecole doctorale Informatique, télécommunication et électronique, composant Universitaire LIP.

[21] Valérie Gayraud, Loutfi Nuaymi, Francis Dupont, Sylvain Gombault, Bruno Tharon 'La Sécurité dans les Réseaux Sans Fil Ad Hoc' Security Lab, 1, Avenue de Belle-Fontaine.

[22] A Perrig, Hu, D.B. Johnson, Wormhole protection in wireless ad hoc networks, Technical Report TR01-384, Department of Computer Science, Rice University, December 2001.

[23] Raghav Bhaskar 'Protocoles cryptographiques pour les réseaux mobiles ad hoc' Thèse Doctorale, Ecole Polytechnique, 26 juin 2006.

[24] RFC 3418

[25] AAFID: http://www.cerias.purdue.edu/about/history/coast/projects/aafid.php

[26] Jean Marc Percher, Bernard Jouga « Détection d'intrusions dans les réseaux ad hoc » SSTIC 2003 – Symposium sur la Sécurité des Technologies de l'information et des communications 10-12 Juin 2003 Rennes- France.

[27] http://seclab.cs.ucdavis.edu/

[28] RFC 4082

[30] Guy Pujolle " Les Réseaux" Groupe Eyrolles, 2006 ISBN : 2-212-11987-9

[31] Recommandation X.800, CCIT

[32] Patroklos G Argyroudis, Donal O'Mahony 'Secure Routing for Mobile Ad hoc Networks' Networks and Telecommunications Research Group Department of Computer Science University of Dublin, Trinity College.

[33] A Perrig, R Canetti, D Song, and J D Tygar "Efficient and secure source authentication for multicast" In Proceedings of the Symposium on Network and Distributed Systems Security (NDSS 2001), pages 35–46. Internet Society, February 2001.

[34] http://seclab.cs.ucdavis.edu/

[35] http://www.ipa.go.jp/STC/IDA/index.html

[36] http://www.isi.edu/nsnam/ns/

[37] NS-2: Principes de conception et d'utilisation par P. Anelli & E. Horlait Version 1.3

Glossaire

ABR - Associativity-Based Routing

AODV - Ad Hoc On-Demand Distance Vector

ARAN - Authenticated Routing for Ad hoc Networks

Ariadne

CGSR - Clusterhead Gateway Switch Routing

CBRP - Cluster Based Routing Protocol

DSDV - Destination-Sequenced Distance-Vector

DSR - Dynamic Source Routing protocol

DREAM - Distance Routing Effect Algorithm for Mobility

FSR - Fisheye State Routing Protocol

GSR - Global State Routing Protocol

GPS - Global Positioning System

God - General Operation Director

HSR - Hierarchial State Routing protocol

IETF - Internet Engineering Task Force

LAR - Location-Aided Routing protocol

MAC - Message Authentication Code

MANET - Mobile Ad Hoc Network

NS2 - Network Simulator 2

OLSR - Optimized Link State Routing Protocol

RFC - Request For Comments

SA - Security Association

SAODV - Secure Ad hoc On-demand Distance Vector

SAR - Security-aware Ad hoc Routing

SEAD - Secure Efficient Ad hoc Distance vector routing protocol

SLSP - Secure Link State Protocol

SRP - Secure Routing Protocol

SSR - Signal Stability Routing

TBRPF - Topology Dissemination Based on Reverse-Path Forwarding

TESLA - Time Efficient Stream Loss-tolerant Authentication

TORA - Temporally Ordered Routing Algorithm

TTL - Time To Live

WRP - Wireless Routing Protocol

ZRP - Zone Routing Protocol

ZHLS - Zone-Based Hierarchial Link State Routing

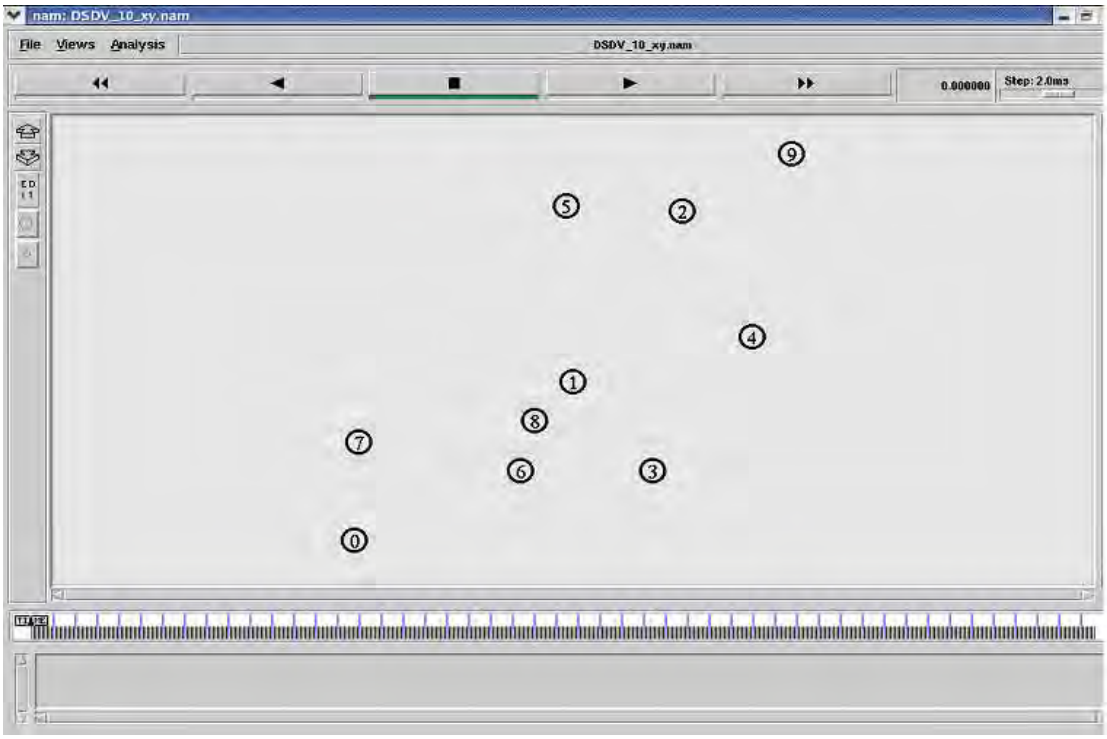
ANNEXE 1 : Données pour la variante de ARIADNE

Nombre de noeuds	Fraction de paquets délivrés	Charge normale de routage	Délai des paquets transférer de bout en bout
0	0,92115	45	13,017
5	0,916802	45	17,008
10	0,91935	44,946	21,557
15	0,90873	44,87	25,09
20	0,933333	44,85	26,640
25	0,929999	44,53	26.987
30	0,92198	44,56	30,00
35	0,9579	44,36	35,11
40	0,98991	44,36	44,823
45	0.97270	44,31	52,7712

ANNEXE 2 : Données Pour ARIADNE

Nombre de noeuds	Fraction de paquets délivrés	Charge normale de routage	Délai des paquets transférer de bout en bout
0	0,7586	120	1.427
5	0,781056	112,87	1,421
10	0,8395	80	1.379
15	0,85603	66,146	0.8604
20	0,85109	60	0,8604
25	0,85131	64,55	0,8573
30	0,85294	55,097	0,776
35	0,87583	50,43	0,75109
40	0,8896	46,88	0,5891
45	0,90307	40	0,5832

ANNEXE 3 : Visualisation d'un réseau ad hoc de dix nœuds grâce au fichier nam.



ANNEXE 4 : Analyse du fichier de simulation trace

Le fichier trace contient toutes les informations concernant l'ensemble du réseau. C'est le fichier qu'il faut exploiter pour évaluer tous les critères de performances d'un réseau donné. Après chaque simulation, des dossiers de trace enregistrent les mouvements du trafic et des nœuds qui sont produits. Ces dossiers doivent être analysés afin d'extraire les mesures de performance du réseau considéré. Le nouveau format de trace ci-dessous a été employé pour l'analyse.

s -t 0.000000000 -Hs 0 -Hd -2 -Ni 0 -Nx 0.00 -Ny 50.00 -Nz 0.00 -Ne -1.000000 -Nl AGT -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.0 -Id 9.0 -It cbr -Il 1140 -If 11 -Ii 0 -Iv 32 -Pn cbr -Pi 0 -Pf 0 -Po 0

r -t 0.000000000 -Hs 0 -Hd -2 -Ni 0 -Nx 0.00 -Ny 50.00 -Nz 0.00 -Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.0 -Id 9.0 -It cbr -Il 1140 -If 11 -Ii 0 -Iv 32 -Pn cbr -Pi 0 -Pf 0 -Po 0

d -t 0.675567223 -Hs 5 -Hd -1 -Ni 5 -Nx 891.19 -Ny 440.82 -Nz 0.00 -Ne -1.000000 -Nl RTR -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 5.255 -Id -1.255 -It message -Il 32 -If 0 -Ii 2 -Iv 32

Ici, nous voyons qu'un paquet a été envoyé (s) au temps (t) 0.267662078 seconde, par le noeud source (Hs) 0, au noeud destinataire (Hd) 1. L'identification de noeud de source (Ni) est 0, il a un x-coordonnée (Nx) de 0.00, un y-coordonnée (Ny) de 50.00, et un z-coordonnée (Nz) de 0.00, le niveau d'énergie du nœud (Ne) est 1.000000, le niveau de trace (Nl) est AGT (agent). L'information de la couche MAC est fournie par la durée (Ma) 0, adresse d'Ethernet de destination (Md) 0, l'adresse d'Ethernet de source (Ms) est 0 et type d'Ethernet (Mt) est 0. L'information de niveau de paquet d'IP a l'identification de paquet (Id), Le type de paquet (It) est cbr, taille de paquet (Il) est 1140 bits, la valeur du Time to Live de paquets (TTL) (Iv) est 32, les informations sur l'application de CBR présentée par les étiquettes (Pn), le Pf pour le nombre de fois où le paquet a été expédié est 0 et le Po pour le nombre optimal d'expédition est 0.

ANNEXE 5 : Code de la simulation

```
#===================================================================
# Define options
#===================================================================
set opt(chan)           Channel/WirelessChannel  ;# channel type
set opt(prop)           Propagation/TwoRayGround ;# radio-propagation model
set opt(netif)          Phy/WirelessPhy          ;# network interface type
set opt(mac)            Mac/802_11               ;# MAC type
set opt(ifq)            Queue/DropTail/PriQueue  ;# interface queue type
set opt(ll)             LL                       ;# link layer type
set opt(ant)            Antenna/OmniAntenna      ;# antenna model
set opt(ifqlen)         50                       ;# max packet in ifq
set opt(nn)             10                       ;# number of mobilenodes
set opt(adhocRouting)   DSR
set opt(cp)             ""                       ;# connection pattern file
set opt(cp)             "/home/taphakane/testns2/ns-allinone-2.29/ns-2.29/tcl/mobility/scene/cbr-50-10-4-
512"
set opt(sc)             ""                       ;# node movement file.
set opt(sc)             "/home/taphakane/testns2/ns-allinone-2.29/ns-2.29/tcl/mobility/scene/scen-
670x670-50-600-20-0"
set opt(x)   [expr ($opt(nn)+20)*30]   ;# x coordinate of topology Mn*30m
set opt(y)           [expr ($opt(nn)+20)*30]    ;# y coordinate of topology
set opt(seed)           0.0                      ;# seed for random number gen.
set opt(stop)           100                      ;# time to stop simulation
set opt(cbr-start)      0.0

#===================================================================
source /home/taphakane/testns2/ns-allinone-2.29/ns-2.29/tcl/mobility/com.tcl

source /home/taphakane/testns2/ns-allinone-2.29/ns-2.29/tcl/lib/ns-bsnode.tcl

ns-random 0.0
set ns_ [new Simulator]
set chan [new $opt(chan)]
set prop [new $opt(prop)]
$ns_ use-newtrace
set tracefd [open $opt(adhocRouting)_$opt(nn)_xy.tr w]
$ns_ trace-all $tracefd
set namtrace [open $opt(adhocRouting)_$opt(nn)_xy.nam w]
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)
#
# create topography object
#
set topo [new Topography]
# define topology
#
$topo load_flatgrid $opt(x) $opt(y)

# create God
#
set god_ [create-god $opt(nn)]
#
# configure mobile nodes
```

```
#
$ns_ node-config -adhocRouting $opt(adhocRouting) \
             -llType $opt(ll) \
             -macType $opt(mac) \
             -ifqType $opt(ifq) \
             -ifqLen $opt(ifqlen) \
             -antType $opt(ant) \
             -propType $opt(prop) \
             -phyType $opt(netif) \
             -channelType $opt(chan) \
             -topoInstance $topo \
             -wiredRouting OFF \
             -agentTrace ON \
             -routerTrace ON \
             -macTrace OFF

for {set i 0} {$i < $opt(nn)} {incr i} {
 set node_($i) [$ns_ node]
 $node_($i) random-motion 0
}
#
# positions des noeuds
#
$node_(0) set X_ 0.0
$node_(0) set Y_ 50.0
$node_(0) set Z_ 0.0
for {set i 1} {$i < ($opt(nn)-1) } {incr i} {
             $node_($i) set X_ [expr rand()*$opt(x)]
             $node_($i) set Y_ [expr rand()*$opt(y)]
             $node_($i) set Z_ 0.0
}
$node_([expr $opt(nn)-1]) set X_ $opt(x)
$node_([expr $opt(nn)-1]) set Y_ $opt(y)
$node_([expr $opt(nn)-1]) set Z_ 0.0

#############################################################################
for {set i 0} {$i < $val(nn)} {incr i} {
          if { $i != $selected_sink} {
#Setup a UDP connection
          set udp_($i) [new Agent/UDP]
          $ns_ attach-agent $node_($i) $udp_($i)
          $ns_ connect $udp_($i) $null
          $udp_($i) set fid_ 2
          puts "set udp_($i) as a CBR source"
 #Setup a CBR over UDP connection
          set cbr_($i) [new Application/Traffic/CBR]
          $cbr_($i) attach-agent $udp_($i)
         $cbr_($i) set packet_size_ 1140
          $cbr_($i) set rate_ 4k
          $cbr_($i) set maxpkts_ 1000000
          $cbr_($i) set random_ false

          $ns_ at $val(start_traffic) "$cbr_($i) start"
         $ns_ at $val(finish_traffic) "$cbr_($i) stop"
     }
```

```
}
#############################################################################

# define initial node position in nam
#
for {set i 0} {$i < $opt(nn)} {incr i} {
  $ns_ initial_node_pos $node_($i) 50
}
#
# tell all nodes when the simulation ends
#
for {set i 0} {$i < $opt(nn) } {incr i} {
  $ns_ at $opt(stop).0 "$node_($i) reset";
}

$ns_ at $opt(stop).0002 "puts \"NS EXITING...\" ; $ns_ halt"
$ns_ at $opt(stop).0001 "stop"
#############################################################################
set grfile [open file.tr w]
set recinterval 0.01
$ns_ at 0.0 "record"

proc record  {} {
     global ns_ grfile recinterval cbr
     set now [$ns_ now]
     puts $grfile "$now [$cbr set window_] [$cbr set cwnd_]"
     $ns_ at [expr $now+$recinterval] "record"
}
#############################################################################
proc stop {} {
  global ns_ tracefd namtrace grfile
  $ns_ flush-trace
  close $tracefd
  close $namtrace
  close $grfile
puts "running nam"
      exec nam DSR.nam &
      exit 0
}
exec awk $awkCode DSDV_10_xy.tr
}
# begin simulation
puts "============================================="
puts "Starting Simulation..."
puts $opt(nn)
puts "awk -f avgStats.awk src=0 dst=[expr $opt(nn)-1] flow=11 pkt=1140
$opt(adhocRouting)_$opt(nn)_xy.tr > $opt(adhocRouting)_$opt(nn)_xy.tr.out && cat
$opt(adhocRouting)_$opt(nn)_xy.tr.out"
puts "============================================="
$ns_ run
```

ANNEXE 6 : Code java pour exploration du fichier trace

```
import java.util.*;
import java.lang.*;
import java.io.*;
public class parsetrace  {
   public static void main (String args[]) {
    String s, thisLine, currLine, thisLine1;
    int j=0;
    FileInputStream  fin, fin1;
    FileOutputStream fout,fout1;
    final int FILES = 45;
    final int MAX_PACKETS = 400000;
     try {
        int i=0, sends=0, receives=0;
        int drops=0,packet_id=0, highest_packet_id = 0;
        int line_count=0,current_line=0, routing_packets=0;
        int count=0;
              float pdfraction, time=0, packet_duration=0, end_to_end_delay=0;
        float avg_end_to_end_delay=0;
        float start_time[] = new float[MAX_PACKETS];
        float end_time[] = new float[MAX_PACKETS];
        float sent_packets[] = new float[MAX_PACKETS];
        float received_packets[] = new float[MAX_PACKETS];
        String tokens[] = new String[100];

        // initialiser le temps de commencement
        for (i=0; i<MAX_PACKETS ; i++)
        start_time[i] = 0;
               fout =new FileOutputStream ("traceoutput.txt");
        DataOutputStream op = new DataOutputStream(fout);
              for (int h=0;h<FILES+1;h++) {
          j=0;
          sends=0; receives=0; routing_packets=0;
          highest_packet_id = 0;
          end_to_end_delay=0;
```

```
for (i=0; i<MAX_PACKETS ; i++)
{ start_time[i] = 0; end_time[i]=0;}

fin =  new FileInputStream ("final_sources"+h+".tr");
DataInputStream br = new DataInputStream(fin);

while ((thisLine = br.readLine()) != null) {
  // scan it line by line
  java.util.StringTokenizer st = new java.util.StringTokenizer(thisLine, " ");
  i=0;
  while(st.hasMoreElements())
  tokens[i++]= st.nextToken();

  if (tokens[0].equals("s") || tokens[0].equals("r")|| tokens[0].equals("f"))
     {
      //  verification du temps
      if (tokens[1].equals("-t")) time = Float.valueOf(tokens[2]).floatValue();

      // analyse du paquet à partir de son identifiant
      if (tokens[39].equals("-Ii")) packet_id = Integer.valueOf(tokens[40]).intValue();

      /// calculer le nombre de paquets envoyés
      if (tokens[0].equals("s")&&tokens[18].equals("AGT")&&tokens[34].equals("cbr"))
      sends++;

      // trouver le nombre de paquets  au cours de la simulation
      if (packet_id >highest_packet_id) highest_packet_id = packet_id;

      if (start_time[packet_id] == 0) start_time[packet_id] = time;

     // calcul du temps auquel le paquet est reçu et du délai de transmission
     if (tokens[0].equals("r") && tokens[18].equals("AGT") && tokens[34].equals("cbr"))
      {
        receives++;
        end_time[packet_id] = time;
      }
      else end_time[packet_id] = -1;
```

```
                // calcul du nombre de paquets routés
                if ((tokens[0].equals("s") || tokens[0].equals("f"))  && tokens[18].equals("RTR")
                &&  (tokens[34].equals("AODV") || tokens[34].equals("DSR")
                || tokens[34].equals("message") ))

                routing_packets++;
               }
          }
         // calcul de la durée de tous les paquets
        for (packet_id = 0; packet_id <= highest_packet_id ; packet_id++) {

          packet_duration = end_time[packet_id] - start_time[packet_id];
          if (packet_duration >0) end_to_end_delay += packet_duration;
        }

       // calcul de la moyenne des délais de transmission de bout en bout des paquets
        avg_end_to_end_delay = end_to_end_delay / (receives );

        // calcul de la fraction de paquets délivrés
        pdfraction = ((float)receives/(float)(sends))*100;

        System.out.println(" \envoyés "+sends);
        System.out.println(" recus "+receives);
        System.out.println(" routing overhead (packets) "+ routing_packets);
        System.out.println(" Normalized routing load "+(float)routing_packets/(float)receives);
        System.out.println(" fraction de paquets délivrés " +pdfraction);
        System.out.println(" delai de transmission de bout en bout " +avg_end_to_end_delay);
        op.writeBytes(" " +sends);
        op.writeBytes(" "+receives);
        op.writeBytes(" "+ routing_packets);
        op.writeBytes(" "+(float)routing_packets/(float)receives);
        op.writeBytes(" " +pdfraction);
        op.writeBytes(" " +avg_end_to_end_delay);
        op.writeChar('\n');

        }
    }
```

```
    catch (Exception e) {
      e.printStackTrace();
      }
  }  }
```

.