UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH
FACULTÉ DES SCIENCES ET TECHNIQUES FÈS
DÉPARTEMENT D'INFORMATIQUE

PROJET DE FIN D'ETUDES

MASTER SCIENCES ET TECHNIQUES
SYSTÈMES INTELLIGENTS & RÉSEAUX

LA SÉCURITÉ DANS LES RÉSEAUX VANET

ETUDE DE CAS DU PROTOCOLE CSS-OLSR

RÉALISÉ PAR : AZDAD NABILA

SOUTENU LE : 14 JUIN 2016

ENCADRÉ PAR :

MR : RACHID BEN ABBOU
MR : AZEDDINE ZAHI

DEVANT LE JURY COMPOSÉ DE :

PR. RACHID BEN ABBOU
PR. AZEDDINE ZAHI
PR. ABDERRAHIM BENABBOU
PR. MED CHAOUKI ABOUNAIMA

ANNÉE UNIVERSITAIRE 2015-2016

بِسْمِ اللّهِ الرّحْمَنِ الرّحِيمِ

رَّبَّنَا عَلَيْكَ تَوَكَّلْنَا وَإِلَيْكَ أَنَبْنَا وَإِلَيْكَ الْمَصِيرُ

رَبِّ اشْرَحْ لِي صَدْرِى وَيَسِّرْ لِي أَمْرِى وَاحْلُلْ عُقْدَةً مِّن لِّسَانِى يَفْقَهُوا قَوْلِى

Remerciements

Je souhaite avant tout remercier le Tout Miséricordieux, le Très Miséricordieux. Je me prosterne devant ta Grandeur pour te remercier de m'avoir comblée de ta grâce et de m'avoir assistée tout au long de ce voyage dans le jardin du savoir. Merci **Allah** pour le souffle de vie et la santé que tu as accordé à toute ma famille alors que j'étais si loin.

Par la suite, Je tiens à exprimer toute ma reconnaissance à mes Directeurs de mémoire le professeur **Rachid BEN ABBOU**, et le professeur **Azeddine ZAHI** pour les différentes réunions organisées afin de discuter le sujet avec l'équipe constituée de Monsieur Abdelali Boushaba et Monsieur Adil Benabbou, et après lesquelles je me jetais sur mon clavier, emballée par des nouvelles idées, ainsi que pour leurs conseils, remarques et corrections durant la rédaction de ce mémoire.

Je tiens à remercier chaleureusement Monsieur **Rachid BEN ABBOU** pour le temps qu'il a consacré à m'apporter les outils méthodologiques indispensables à la conduite de cette recherche, sa disponibilité, ses précieux conseils, ses réflexions et ses remarques qui m'ont grandement stimulé ainsi que pour ses encouragements tout au long de la rédaction de ce mémoire.

Je remercie grandement Monsieur **Abdelali BOUSHABA** qui, m'a guidé vers les bonnes références, m'a transmis des références d'études sur certaines de mes recherches et qui m'a aidé à avancer et achever ce travail. Je vous remercie du fond du cœur de m'avoir co-encadré, conseillé, et orienté dans toutes les étapes de simulation, ainsi que pour votre disponibilité à tout moment.

Mes remerciements s'adressent également à Monsieur **Adil BENABBOU**, et à tous les enseignants du département informatique de la faculté des Sciences et techniques de Fès pour les grands efforts fournis afin de mener au bon port cette formation de master.

Un immense merci à mes parents, Merci **Maman** de m'avoir donné tant d'amour et de tendresse, et merci **Papa** de m'avoir toujours poussé dans mes intérêts. Vos prières, vos conseils nuit et jour, votre amour, votre discrétion, et tous les sacrifices consentis m'ont guidé chaque jour de ma vie. Merci pour ce que vous avez fait et tout ce que vous ferez encore pour moi. Je suis si fière de vous. Qu'Allah vous préserve bonne santé et longue vie et qu'il m'aide à accomplir pleinement mes devoirs envers vous.

Je remercie également Mon frère Issam et ma sœur Amina qui n'ont cessé d'être pour moi des exemples de persévérance, et de courage. Merci d'être toujours là pour moi et de m'encourager dans tout ce que j'entreprends. Merci pour tous les bons moments passés et à venir, pour notre complicité, pour votre amour et pour toute la joie que vous m'apportez… je vous aime.

Un grand merci à ma chère amie Safae Smiri, je souhaite que l'amitié qui nous a réuni persiste pour toujours et que nous arrivons à réaliser nos rêves.

A tous mes collègues de master SIR, je tiens à vous remercier pour l'esprit d'équipe durant ces deux années de formation passées ensemble ainsi que pour votre amitié qui m'a beaucoup encouragé durant mes travaux.

A ma mère et mon père

A ma sœur et mon frère

A …

Résumé

Dans les prochaines années à venir, les réseaux véhiculaires seront capables de réduire significativement le nombre d'accidents via les messages d'alerte échangés entre les véhicules. Comme le routage est un élément fondamental dans le système de communication véhiculaire, il constituera une cible idéale pour les attaques qui visent à empêcher les messages d'alerte d'arriver à leurs destinations, ce qui pourrait conduire à des situations catastrophiques et mettre ainsi en danger les vies humaines.

Dans ce travail de recherche, nous nous sommes focalisés sur deux protocoles de routage MANET à savoir : AODV (Ad hoc On demand Distance Vector Routing) et OLSR (Optimized Link State Routing) qui appartiennent chacun à une famille, le premier, réactif, il utilise un mécanisme de diffusion dans le réseau pour découvrir les routes valides. Le second est proactif, et utilise un mécanisme qui permet de designer un sous-ensemble de son voisinage responsable de la dissémination des messages de contrôle de topologie TC (Topology Control) dans le réseau à moindre coût. Ces deux protocoles sont susceptibles d'être adaptés pour les réseaux VANETs.
Nous avons cherché les vulnérabilités et les attaques spécifiques à chacun de ces deux protocoles et étudié certaines de leurs extensions sécurisées proposées dans la littérature.

Après avoir comparé entre les extensions sécurisées du protocole OLSR au niveau de protections offertes, nous avons décidé de travaillé sur une variante nommée CSS-OLSR qui ajoute aux éléments du protocole OLSR un message appelé Complete Path Message (CPM). Ce dernier est utilisé pour transmettre le chemin parcouru par un message TC à travers le réseau et permet de détecter les nœuds malicieux en comparant les informations qu'il retourne avec celles des messages injectés par les nœuds du réseau.
Et pour finaliser ce mémoire, nous avons fait des simulations sous ns2 afin d'étudier l'impact du taux des messages CPM sur les performances des paramètres de la qualité de service (QoS) dans un environnement VANET.

Abstract

In the next few years, vehicular networks will be able to reduce significantly the number of accidents by the warning messages exchanged between vehicles. Since routing is a fundamental element in the vehicular communication system, it will be an ideal target for attacks that aim to prevent alert messages from reaching their destinations, thus leading to catastrophic situations and endangering human lives.

In this research, we focused on two MANET routing protocols: AODV (Ad hoc On demand Distance Vector Routing) and OLSR (Optimized Link State Routing) which each belong to a family, the first reactive, it uses a diffusion mechanism in the network to discover the valid routes. The second is proactive, and uses a mechanism to designate a subset of the neighborhood responsible for the spread of topology control messages TC (Topology Control) in the network at lower cost. Both protocols may be adapted for VANETs networks.
We looked for vulnerabilities and attacks specific to each of these two protocols and studied some of their secure extensions proposed in the literature.

After comparing between secured extensions of OLSR in terms of protections offered, we decided to work on a variant named CSS-OLSR that adds to the elements of OLSR a message called Complete Path Message (CPM). This message is used to convey the path traversed by a message TC through the network and allows us to detect misbehaving nodes by comparing its informations with those of messages injected by the network nodes.
And to finalize this research, we made simulations to study the impact of the CPM rate on the performance of quality of service parameters in a VANET environment.

Table des matières

Introduction générale 14
Chapitre 1 : Etat de l'art sur les réseaux VANET 16
1.1. Réseaux sans fil 16
1.1.1. Techniques de communication sans fil 16
1.1.2. Avantages et contraintes de la communication sans fil 17
1.2. Réseaux mobiles 17
1.2.1. Classification des réseaux mobiles 18
1.3. Réseaux ad hoc 18
1.4. Réseaux Véhiculaires Ad hoc 19
1.4.1. Définition d'un réseau VANET 19
1.4.2. Messagerie et architectures des réseaux sans fil véhiculaires 20
1.4.2.1. Entités communicantes 20
1.4.2.2. Types de messages 21
1.4.2.3. Architectures de communication 21
1.4.3. Caractéristiques des réseaux VANET 23
1.4.4. Applications des réseaux VANET 24
1.4.4.1. Les applications liées aux STI et à la gestion du trafic routier 25
1.4.4.2. Les applications liées au confort du conducteur et des passagers 25
1.4.4.3. Les applications liées à la sécurité du trafic routier 26
1.4.5. Technologies d'accès 28
1.4.5.1. Systèmes de communication intra-véhiculaires 28
1.4.5.2. Systèmes de communication extra-véhiculaires 29
1.4.6. Protocoles de routage Ad hoc véhiculaire 32
1.4.6.1. Classification des protocoles de routage 32
1.4.6.2. Le protocole OLSR (Optimized Link State Routing) 34
1.4.6.3. Le protocole AODV (Ad hoc On-demand Distance Vector) 41
Chapitre 2 : Menaces et problèmes de sécurité dans les VANETs 43
2.1. Attaques dans les réseaux sans fil véhiculaires 43
2.1.1. Modèles d'attaquant 43
2.1.2. Menaces au niveau applicatif 44
2.1.3. Attaques contre les protocoles de routage 49
2.1.4. Vulnérabilités et types d'attaques spécifiques au protocole OLSR 49

2.1.4.1. Génération incorrecte du trafic 49
2.1.4.2. Relayage incorrect du trafic 51
2.1.5. Vulnérabilités et types d'attaques spécifiques au protocole AODV 53
2.1.5.1. Classifications des attaques spécifiques au protocole AODV 54
2.1.5.1.1. Attaques élémentaires 54
2.1.5.1.2. Attaques composées 57
Chapitre 3 : Mécanismes et techniques de sécurité dans les VANETs 61
3.1. Services de sécurité et mécanismes 61
3.1.1. Confidentialité 61
3.1.2. Authenticité 62
3.1.3. Intégrité 63
3.1.4. Non-répudiation 64
3.1.5. Disponibilité 64
3.1.6. Contrôle d'accès 65
3.2. La sécurité de routage dans les réseaux VANET 65
3.2.1. Mécanismes de sécurité pour le protocole OLSR 66
3.2.1.1. Secure OLSR 66
3.2.1.2. Architecture de sécurité pour OLSR 67
3.2.1.3. ADVSIG (An Advanced Signature System for OLSR) 69
3.2.1.4. GPS-OLSR (OLSR with GPS information) 72
3.2.1.5. TOLSR (Trust system for OLSR) 74
3.2.1.6. CSS-OLSR (Cooperative Security Scheme for OLSR) 76
3.2.2. Comparaison entre les extensions sécurisées du protocole OLSR 79
3.2.2.1. Les mesures de sécurité standards implémentées pour chaque variante 80
3.2.1.1. Les protections offertes par chaque variante 80
3.2.3. Mécanismes de sécurité pour le protocole AODV 81
3.2.3.1. SAODV (Secure Ad hoc On-Demand Distance Vector) 81
3.2.3.2. ARAN (Authenticated Routing for Ad hoc Networks) 81
3.2.3.3. SEAR (Secure Efficient Ad hoc on demand Routing) 82
3.2.4. Comparaison entre les extensions sécurisées du protocole AODV 83
Chapitre 4 : Simulations et résultats 84
4.1. Contexte et objectif 84
4.2. Processus de simulation dans les VANETs 84
4.2.1. Génération d'une Mappe de simulation 84
4.2.2. Simulation de modèle de mobilité et la génération du trafic 85
4.2.3. Simulation de modèle réseau 85

4.2.4. Traitement de données 85
4.3. Environnement de simulation 85
4.3.1. NS-2 85
4.3.2. VanetMobiSim 87
4.4. Paramètres d'évaluation 87
4.4.1. Taux de paquets délivrés : PDR (Packet Delivery Ratio) 87
4.5. Scénario de simulation 88
4.5.1. Le modèle d'attaque 88
4.5.2. La sécurité avec CSS-OLSR 89
4.5.2.1. Procédure de CSS-OLSR 90
4.5.2.2. Détection du nœud malicieux 90
4.6. Etapes de simulation 92
4.6.1. Modèle de mobilité 92
4.6.2. Modèle de trafic 93
4.6.3. Simulation réseau 93
4.7. Résultats et Analyse 94
4.7.1. Courbes de visualisation du PDR 94
4.7.2. Courbes de visualisation de Délai 95
4.7.3. Courbes de visualisation de la gigue 96
4.7.4. Courbes de visualisation du cout de routage 97
4.7.5. Courbes de visualisation de l'Efficacité 98
Conclusion 100
Références 102
Annexe 1 104
Attaques spécifiques au protocole OLSR 104
Annexe 2 105
Fichier de topologie au format XML 105
Spécifications des champs du fichier 106
Annexe 3 109
Script de simulation 109
Annexe 4 113
Fichier AWK 113

Liste des figures

Figure 1.1 : Hiérarchie des réseaux sans fil……19
Figure 1.2 : Véhicule intelligent ……19
Figure 1.3 : Exemple de réseau véhiculaire……..20
Figure 1.4 : Les modes de communication dans les VANETs……23
Figure 1.5 : La classification des applications dans les VANETs……24
Figures 1.6 : Exemples de scénarios d'applications des réseaux VANET……..27
Figure 1.7 : Protocoles de routage pour les VANETs……..32
Figure 1.8 : Format du paquet OLSR……35
Figure 1.9 : Format d'un message OLSR – HELLO……36
Figure 1.10 : Format d'un message OLSR – TC……38
Figure 1.11 : Sélection des MPRs……39
Figure 2.1 : Classification des attaques contre les réseaux VANET……..44
Figure 2.2 : Attaque de révélation de position géographique d'un véhicule……45
Figure 2.3 : Attaques par l'envoi de messages falsifiés…….. .46
Figure 2.4 : Véhicule caché……..47
Figure 2.5 : Attaque de Tunnel……..47
Figure 2.6 : Attaque déni de service……..48
Figure 2.7 : Usurpation d'identité du nœud *C* par *M* (HELLO) ……49
Figure 2.8 : Attaque sur la sélection des MPR……50
Figure 2.9 : Usurpation d'identité du nœud v par M (TC) ……51
Figure 2.10 : Attaque *wormhole* créée par le nœud *M*……52
Figure 2.11 : Collaboration pour créer un wormhole……52
Figure 2.12 : Invasion de route (route déjà existante) ……58
Figure 2.13 : Invasion de route (lors de l'établissement du chemin) ……59
Figure 2.14 : Création d'une boucle de routage dans une route déjà existante……60
Figure 2.15 : Attaque de tunnel……60
Figure 3.1 : Les mécanismes de sécurité dans les réseaux VANET……..61
Figure 3.2 : Le message de signature élémentaire (Secure OLSR) ……..66
Figure 3.3 : Format de signature associé à chaque message du paquet……..68

Figure 3.4 : Le message ADVSIG……..71
Figure 3.5 : Format de message SIGLOC……..73
Figure 3.6 : Format du message d'accusation……..75
Figure 3.7 : Algorithme du traitement de message CPM……..78
Figure 4.1 : Le processus de simulation dans les réseaux VANET……..84
Figure 4.2 : Synoptique de l'interpréteur de script Otcl avec les librairies C++ de simulation de réseau……..86
Figure 4.3 : Insertion de faux message HELLO……..88
Figure 4.4 : Insertion de faux message TC……89
Figure 4.5 : Détection de faux message HELLO……..90
Figure 4.6 : Détection de faux message TC……..91
Figure 4.7 : Visualisation de la mobilité des nœuds par VanetMobisim……..92
Figures 4.8 : Taux de paquets délivrés……..94
Figures 4.9 : Le délai de bout en bout……95
Figures 4.10 : la gigue. ……..96
Figures 4.11 : Le cout de routage……97
Figures 4.12 : L'Efficacité……98

Liste des tableaux

Tableau 1.1 : Champs Link Code.....37
Tableau 1.2 : Valeurs possible pour le champ Link Code.....37
Tableau 2.1 : Classification des actions malveillantes sur les messages de routage en fonction de l'objectif.....54
Tableau 2.2 : Modifications possibles sur les champs des RREQ.....55
Tableau 2.3 : Modifications possibles sur les champs des RREP.....56
Tableau 2.4 : Modifications possibles sur les champs des RERR.....57
Tableau 3.1 : Les mauvaises conduites.....75
Tableau 3.2 : Les mesures de sécurité standards implémentées par les variantes sécurisées du protocole OLSR.....80
Tableau 3.3 : Les protections offertes par les variantes sécurisées du protocole OLSR.....80
Tableau 3.4 : Les mesures de sécurité standards implémentées par les variantes sécurisées du protocole AODV.....83
Tableau 4.1 : Paramètres utilisés pour le modèle de mobilité.....92
Tableau 4.2 : Paramètres utilisés pour la couche physique.....92
Tableau 4.3 : Paramètres utilisés pour le modèle de trafic.....93
Tableau 4.4 : Paramètres de CSS-OLSR.....93

Liste des abréviations

ADVSIG	An Advanced Signature System for OLSR
AODV	Ad hoc On-demand Distance Vector
ARAN	Authenticated Routing for Ad hoc Networks
CA	Cooperative Awareness
CBR	Constant Bit Rate
CDA	Cooperative Driver Assistance
CDS	Cooperative driving system
ComS	Communities Services
CoNa	Cooperative Navigation
CPM	Complete Path Message
CSM	Cooperative Speed Management
CSS-OLSR	Cooperative Security Scheme for OLSR
DoS	Denial of Service
GPS	Global Position System
IEEE	Institute of Electrical and Electronics Engineers
LBS	Location Based Services
LCA	Lane Change Assistance
LCM	Life Cycle Management
MANET	Mobile Ad-Hoc Network
MPR	Multipoint Relays
NAM	Network AniMator
NS-2	Network Simulator version 2
OBU	On-Board Unit
OLSR	Optimized Link State Routing
OTCL	Object Tools Command Language
PDR	Packet Delivery Ratio

QoS	Quality of Service
RERR	Route ERRor
RHCW	Road Hazard and Collision Warning
RREP	Route Reply
RREQ	Route REQuest
RSU	Road-Side Unit
SAODV	Secure Ad hoc On-Demand Distance Vector
SEAR	Secure Efficient Ad hoc on demand Routing
SIGLOC	SIGnature and LOCalization
SOLSR	Secure OLSR
STI	systèmes de transport intelligents
TC	Topology Control
TCL	Tools Command Language
TOLSR	Trust system for OLSR
VANET	Vehicular Ad-Hoc Network
V2I	Vehicle to Infrastructure
V2V	Vehicle-to-Vehicle
XML	eXtensible Markup Language

Introduction générale

Au cours de la dernière décennie, L'évolution massive des technologies a permis l'avènement des réseaux sans fil qui se sont développés à un rythme accéléré et qui ont touché aussi le secteur automobile en permettant aux véhicules d'échanger des informations très importantes d'un point de vue sécurité routière ou contrôle de trafic (e.g. détection d'un accident, présence d'un embouteillage, approche d'un véhicule à une intersection, etc). Ces informations assisteraient les conducteurs et les aideraient dans la prise des décisions ce qui pourrait réduire significativement le nombre des accidents de la route dans le monde.

Vu l'importance des informations échangées entre les véhicules et l'ouverture de l'environnement VANET (Vehicular Ad-hoc NETwork), Un attaquant peut émettre des messages d'alerte dont le contenu est falsifié ou empêcher l'acheminement d'un message légitime en visant la disponibilité du réseau aux niveaux des différentes couches de la pile protocolaire. Comme le routage est un service fondamental dans tout système de communication, il peut être une cible idéale pour plusieurs types d'attaques.

Dans ce travail, nous nous sommes surtout intéressés à deux protocoles de routage : OLSR (Optimized Link State Routing Protocol) et AODV (Ad hoc On demand Distance Vector Routing) et en particulier aux vulnérabilités et aux attaques spécifiques à ces protocoles ainsi qu'aux certaines extensions sécurisées proposées pour chacun d'eux.

Dans la suite, nous nous sommes focalisés sur le protocole OLSR. Ce dernier utilise une diffusion optimisée des messages grâce à des nœuds appelés MPR (Relais multipoints), en effet chaque nœud du réseau sélectionne parmi ses voisins à un saut un ensemble minimal des nœuds qui vont lui permettre d'atteindre tous les voisins à 2 sauts, les nœuds constituant cet ensemble s'appellent « Relais multipoints » et ce sont les seuls qui peuvent retransmettre les messages de contrôle de topologie (TC) reçus. A cause de ce rôle important des nœuds MPR, un nœud malicieux tente dans la plupart du temps d'obliger ses voisins à le choisir comme relai multipoint pour avoir une position privilégiée dans le réseau et pouvoir par la suite altérer ou rejeter les messages reçus. Notre objectif est donc, de comparer entre les protections offertes par chacune des extensions sécurisées du protocole OLSR trouvées.
Après avoir fait cette comparaison, on a décidé de travailler sur une variante nommée CSS-OLSR qui permet de détecter le nœud malicieux et définir le type d'attaque effectuée.

Ce mémoire est composé de quatre chapitres :

Le premier chapitre est consacré aux généralités sur les réseaux ad hoc et particulièrement les réseaux VANET. En effet, on a introduit le concept du réseau véhiculaire. On a décrit les entités communicantes, les architectures et les technologies de communication sans fil pour ce type de réseau. On a décrit aussi ses caractéristiques, ses applications, et on a détaillé deux protocoles de routage utilisés dans ces réseaux à savoir : OLSR et AODV.

Le second chapitre présente les menaces et les problèmes de sécurité dans les VANETs au niveau applicatif ainsi que les attaques spécifiques aux protocoles de routage déjà présentés dans le premier chapitre.

Le chapitre 3 présente les services de sécurité qui doivent être déployés dans les VANETs pour prévenir une attaque de sécurité, et détaille quelques variantes sécurisées des protocoles OLSR et AODV.

Le quatrième chapitre contient les explications de simulations effectuées et les analyses des résultats obtenus.

Enfin, une synthèse générale de ce travail de recherche est présentée dans le chapitre de conclusion.

Chapitre 1 : Etat de l'art sur les réseaux VANET

Avec l'adoption des technologies de communication sans fil, les réseaux ont connu ces dernières années un essor spectaculaire et s'imposent aujourd'hui de façon indéniable.
Un tel succès est dû principalement à la vulgarisation des équipements mobiles offrant plus de souplesse, plus de rapidité et moins de frais.

Les réseaux mobiles sans fil peuvent être classés en deux catégories : les réseaux avec infrastructure ou cellulaires qui nécessitent généralement l'installation des stations de base et les réseaux sans infrastructure ou Ad Hoc caractérisés par leur dynamisme, facilité et rapidité de déploiement. Ces caractéristiques les rendent utilisés dans plusieurs applications à savoir la téléphonie, les applications militaires, les applications commerciales et la sécurité routière.

Dans ce chapitre nous allons présenter les réseaux sans fil, leurs catégories, leurs caractéristiques ainsi que leurs contraintes. Les réseaux mobiles, leurs classifications et leurs caractéristiques. Subséquemment, nous aborderons les réseaux mobiles Ad Hoc (MANET) pour enfin parler des réseaux Ad Hoc véhiculaires VANET.

1.1. Réseaux sans fil

Un réseau sans fil (Wireless network) est un réseau dans lequel les machines participantes peuvent communiquer sans liaison filaire. Il est basé sur des liaisons utilisant des ondes radioélectriques (radio ou infrarouge) au lieu des câbles habituels (coaxial, paire-torsadée ou fibre optique).
Dans ce type de réseau, les utilisateurs ont la possibilité de se déplacer dans un certain périmètre de couverture géographique sans perdre le signal.

1.1.1. Techniques de communication sans fil

- ✓ **Liaisons radios**

Le principe est d'émettre des ondes électromagnétiques qui constituent la porteuse du signal à transmettre. Ces ondes sont donc propagées dans toutes les directions et peuvent être captées par plusieurs antennes. Le medium radio est découpé en bandes de fréquences divisées en canaux.

- ✓ **Liaisons infrarouges**

Les liaisons infrarouges sont utilisées dans les communications courtes et en vue, elles sont simples et peu coûteuses. Elles conviennent aux réseaux à faible portée. Les émetteurs et récepteurs à infrarouge sont capables de fournir des débits élevés à des coûts relativement faibles. Les bandes passantes disponibles sont très larges, les liaisons infrarouges pénètrent à travers le

verre, mais pas à travers les murs ou tout autre obstacle opaque, ce qui implique que les communications se font dans la même pièce, ce qui augmente la sécurité.

1.1.2. Avantages et contraintes de la communication sans fil

✓ **Avantage des réseaux sans fil**

La mobilité : La mise en place d'un réseau sans fil entre des éléments portables permet d'éviter les fils de connexion au sein du réseau et les fils d'alimentation, afin de permettre le mouvement libre des utilisateurs avec leurs terminaux portables.

Faibles coûts : Contrairement au réseau filaire où le câblage représente un coût supplémentaire, le réseau sans fil s'affranchit de ce coût. Néanmoins, les protocoles de routage et de configuration doivent être repensés pour permettre la bonne gestion et l'acheminement des données dans le réseau.

✓ **Contraintes des réseaux sans fil**

Dégradation de la qualité du signal : Cette contrainte est causée par l'affaiblissement de la puissance du signal avec la distance et les conditions atmosphériques. De plus, le bruit dû à d'autres signaux parasites cause une altération du signal. D'autres paramètres tels que l'absorption atmosphérique du signal par la vapeur d'eau et l'oxygène, la propagation multi trajet causée par les obstacles entre l'émetteur et le récepteur, font que le signal se dégrade davantage.

***Sécurité* :** La confidentialité des données circulantes sur les réseaux sans fil doit être assurée, car les transmissions radioélectriques sont sensibles aux interférences et sujettes à l'écoute par un utilisateur mal intentionné. Cet utilisateur peut se placer dans le périmètre des équipements du réseau afin de récupérer les informations qui lui permettront d'avoir accès au réseau. Ceci représente le plus grand problème des réseaux sans fil.

***Débit* :** Le simple fait d'avoir un trop grand nombre d'utilisateurs dans un réseau sans fil peut entraîner une diminution importante de débit. Cette diminution de débit peut même conduire à une perte de connectivité ce qui est très contraignant.

1.2. Réseaux mobiles

Un réseau mobile est un système composé de nœuds reliés les uns aux autres par des liaisons de communication sans fil. Ces nœuds sont libres de se déplacer sans perte de leurs connexions au réseau. Un réseau mobile peut contenir des sites fixes pour permettre l'accès à d'autres types de réseaux (filaires).

1.2.1. Classification des réseaux mobiles

Les réseaux mobiles peuvent être divisés en deux classes : les réseaux mobiles basés sur une infrastructure et les réseaux mobiles sans infrastructure.

- ✓ **Réseaux mobiles avec infrastructure (cellulaires)**

Dans ce mode, chaque nœud se connecte à un point d'accès (site fixe ou station de base SB) via une liaison sans fil. L'ensemble formé par le point d'accès et les unités mobiles situées dans sa zone de couverture, est appelé ensemble de services de base (Basic Service Set, noté BSS) et constitue une cellule.

- ✓ **Réseau mobile sans infrastructure**

Dans le mode sans infrastructure, la notion de site fixe ou point d'accès n'existe pas. Toutes les stations du réseau se connectent les unes aux autres afin de construire un réseau point à point (P2P pour peer to peer). Ainsi, chaque machine joue en même temps le rôle de client et le rôle de point d'accès.

1.3. Réseaux ad hoc

Les réseaux ad hoc sont des réseaux sans-fil capables de s'organiser spontanément et de manière autonome dans l'environnement dans lequel ils sont déployés sans infrastructure définie préalablement. La tâche de la gestion du réseau est répartie sur l'ensemble d'entités communicantes par liaison sans-fil, ces entités sont souvent appelées «nœuds». Dans ces réseaux, les entités envisagées sont des terminaux légers et de taille réduite qui fonctionnent sur batterie, donc elles ont des capacités de traitement et de mémoire limitées.

Les réseaux ad hoc, dans leur configuration mobile, sont connus sous le nom de MANET (pour Mobile Ad-hoc NETworks).

Un réseau mobile ad hoc (MANET), est un réseau sans fil qui consiste en une grande population, relativement dense, d'unités mobiles qui se déplacent dans un territoire quelconque et dont le seul moyen de communication est l'utilisation des interfaces sans fil généralement le medium radio, sans l'aide d'une infrastructure préexistante ou administration centralisée, ces unités mobiles jouent à la fois le rôle de terminaux et de routeurs pour permettre le passage de l'information entre elles. Il permet donc à deux nœuds qui sont chacun à portée des ondes l'un de l'autre (condition appropriée de propagation d'ondes radio) de rentrer en communication directement.

Un réseau ad hoc doit être facilement déployé, les nœuds peuvent joindre ou quitter le réseau de manière totalement dynamique sans informer le réseau, et si possible sans effet de bord sur les communications des autres membres.

1.4. Réseaux Véhiculaires Ad hoc

1.4.1. Définition d'un réseau VANET:

Un réseau VANET est une particularité des réseaux MANET où les nœuds mobiles sont des véhicules (intelligents) équipés de calculateurs, de cartes réseau et de capteurs.
Comme tout autre réseau Ad hoc, les véhicules peuvent communiquer entre eux (pour échanger les informations sur le trafic par exemple) ou avec des stations de base placées tout au long des routes (pour demander des informations ou accéder à internet…).

La figure 1.1 représente la hiérarchie des réseaux sans fil où elle schématise l'inclusion des réseaux véhiculaires Ad Hoc VANET dans les réseaux mobile Ad Hoc MANET, les MANET dans les réseaux Mobiles ainsi que les réseaux mobiles dans les réseaux sans fil.

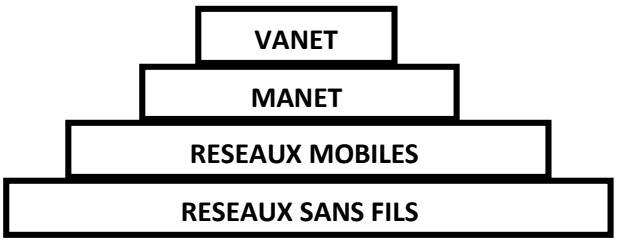


Figure 1.1 : Hiérarchie des réseaux sans fil

- ✓ **Le Nœud du réseau VANET**

Un nœud d'un réseau VANET est un véhicule équipé de terminaux tels que les calculateurs, les interfaces réseaux ainsi que des capteurs capables de collecter les informations et de les traiter. On parle de la notion de « véhicule intelligent ».

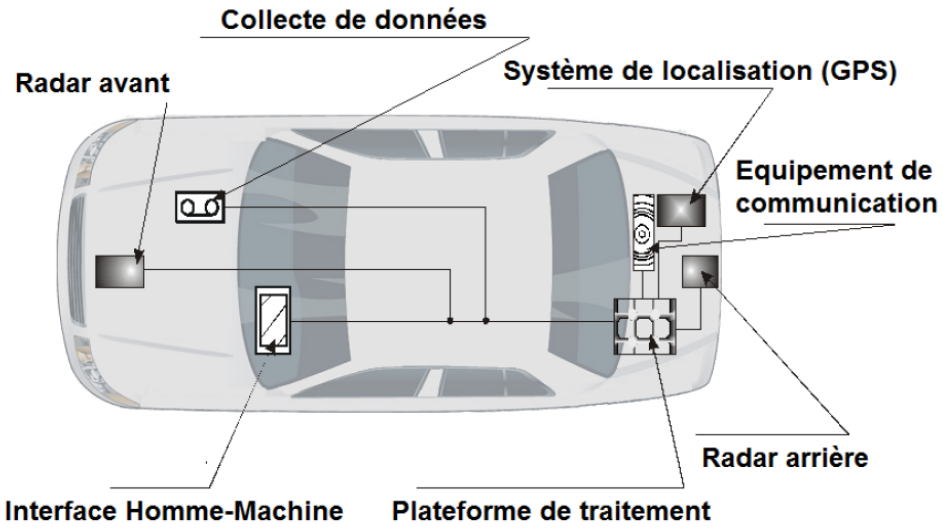


Figure 1.2 : Véhicule intelligent

1.4.2. Messagerie et architectures des réseaux sans fil véhiculaires

Un réseau sans fil véhiculaire est un ensemble d'entités communicantes organisées selon une architecture de communication. Ces entités embarquées peuvent rencontrer différents environnements (urbain, péri-urbain, autoroutier), ayant leurs propres contraintes.

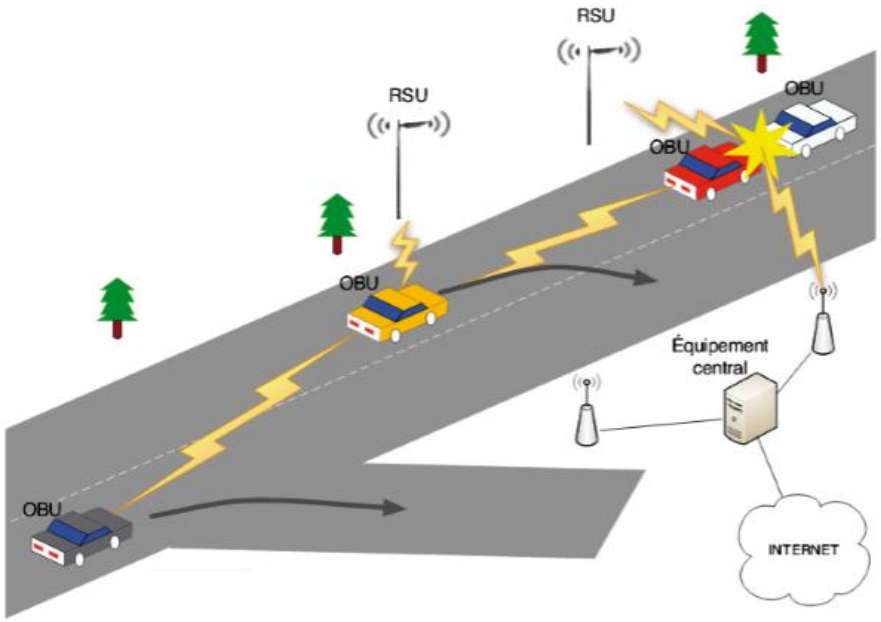


Figure 1.3 : Exemple de réseau véhiculaire

1.4.2.1. Entités communicantes

Dans un réseau sans fil véhiculaire, il existe quatre entités communicantes : l'équipement personnel, le véhicule, l'équipement de bord de route et l'équipement central. La figure au-dessus illustre un exemple de réseau véhiculaire faisant intervenir les différentes entités lors d'un accident de la route.

- *Les équipements personnels* sont les équipements qui peuvent être apportés par l'utilisateur à l'intérieur de son véhicule. Cela peut être un téléphone portable, un ordinateur portable ou encore un GPS autonome. Ces équipements peuvent interagir avec le véhicule. De nos jours, en activant l'interface Bluetooth du téléphone portable, on peut utiliser son téléphone portable par commande vocale (en utilisant les microphones intégrés au véhicule) ou par le biais de l'interface Homme-Machine (IHM) du véhicule.

- *Les véhicules modernes* sont équipés d'un ensemble de processeurs connectés à une plateforme centrale de calcul qui dispose d'interfaces filaires et sans fil. Les véhicules intelligents sont des véhicules équipés d'une unité nommée On-Board Unit (OBU). Cette unité peut enregistrer, calculer, localiser et envoyer des messages sur une interface réseau.

- Les entités de bord de route sont appelées *Road-Side Unit*(RSU). Ces unités peuvent informer les véhicules à proximité en diffusant les conditions de trafic, météorologiques ou spécifiques à la route (vitesse maximale, autorisation de dépassement, etc.). Les RSU peuvent aussi jouer le rôle de station de base en relayant l'information envoyée par un véhicule.

- *L'équipement central* se situe du côté « serveur ». Il est transparent pour l'utilisateur. Cet équipement central pourra être un serveur de stockage, un point d'entrée à un réseau filaire (Internet) ou un serveur de transaction (télépéage par exemple).

1.4.2.2. Types de messages

Les entités formant un réseau sans fil véhiculaire vont générer et s'échanger des messages. En fonction de l'application et du contexte environnemental, un véhicule peut envoyer (ou recevoir) un message de contrôle, d'alerte ou « autre ».

✓ **Message de contrôle**

Le message de contrôle est généré à intervalle régulier. Conventionnellement, chaque véhicule émet un message de contrôle toutes les 100 ms. Ce message, appelé aussi «beacon», contient la position, la vitesse, la direction et l'itinéraire du véhicule émetteur. Grâce aux messages de contrôle, chaque véhicule se crée une vue locale de son voisinage. Le véhicule peut aussi prédire et anticiper des situations accidentogènes ou de congestion. Le message de contrôle est l'équivalent du message HELLO des protocoles de routage. Chaque véhicule se fait donc connaître de son voisinage direct. Bien entendu, les messages de contrôle ne sont pas transférés et utilisent une diffusion à un saut.

✓ **Message d'alerte**

Le message d'alerte est généré lorsqu'un événement est détecté. Cela peut être la détection d'un accident, d'un obstacle ou la réception d'un autre message d'alerte. Les messages d'alerte contiennent en particulier les coordonnées du lieu de l'accident et les paramètres de la zone de retransmission, et ils doivent être de taille réduite pour être transmis le plus rapidement possible.

✓ **Autres messages**

Ce type de message contient tous les messages qui ne sont pas des messages d'alerte ou de contrôle. Ces messages ne sont généralement pas répétés à intervalle régulier. En effet, cela peut être par exemple un message de transaction financière ou l'envoi de courrier électronique.
Tous les messages reçus seront stockés dans un « cache des messages récemment reçus ». Chaque message se verra associé une durée de vie dans le cache.

1.4.2.3. Architectures de communication

Dans les réseaux de véhicules, on peut distinguer deux modes de communication, les communications Véhicule-Infrastructure et les communications Véhicule-à-Véhicule.
Les véhicules peuvent utiliser un de ces deux modes ou bien les combiner s'ils ne peuvent pas communiquer directement avec les infrastructures.

- ✓ **Mode de communication de Véhicule-Infrastructure**

L'architecture Véhicule-vers-Infrastructure (V2I) est composée de RSU, auxquels les véhicules accèdent pour les applications de sécurité, de gestion et de confort. Les RSU sont administrés par un ou plusieurs organismes publics ou bien par des opérateurs autoroutiers. Un véhicule qui informe le service de voirie au sujet d'un obstacle est un exemple de communication V2I. Dans cet exemple, la communication est unidirectionnelle, du OBU vers le RSU.

Nous parlons de I2V dans le cas de communication Infrastructure-vers-Véhicule. Un panneau de signalisation équipé d'un RSU qui envoie une information aux véhicules passant à proximité est un exemple de communication I2V. Dans la suite, par V2I, nous englobons toutes les communications Véhicule-Infrastructure, quelle que soit la direction du trafic de données.

L'inconvénient majeur de cette approche est que l'installation des stations le long des routes est une tâche coûteuse et prend beaucoup de temps, sans oublier les coûts relatifs à la maintenance des stations.

- ✓ **Mode de communication Véhicule à Véhicule**

L'architecture de communication inter-véhicules (V2V ou IVC pour Inter Vehicle Communication) est composée uniquement d'OBU (véhicules légers, poids lourds, véhicules de secours, etc.). Ils forment alors un réseau mobile sans avoir besoin d'un élément de coordination centralisé. Cette situation est essentielle si certains équipements RSU deviennent indisponibles (en panne ou hors de portée). Dans ce cas, le réseau doit continuer de fonctionner. Les véhicules doivent alors collaborer pour assurer la disponibilité du service. Ce mode de fonctionnement est communément appelé « ad hoc » et est utilisé par les VANETs. L'architecture V2V en mode ad hoc peut aussi être utilisée dans les scénarios de diffusion d'alerte (freinage d'urgence, collision, ralentissement, etc.) ou pour la conduite coopérative. En effet, dans le cadre d'applications de sécurité routière, les réseaux à infrastructure montrent leurs limites, surtout en terme de délai. Prenons l'exemple d'un véhicule en difficulté sur la chaussée qui diffuse un message d'alerte. Il semble plus rapide d'envoyer l'information directement aux autres véhicules plutôt que de la faire transiter par une station de base.

- ✓ **Communication Hybride**

La combinaison des communications véhicules à véhicules avec les communications de véhicules avec utilisation d'infrastructures, permet d'obtenir une communication hybride très intéressante. En effet, les portées des infrastructures (stations de bases) étant limitées, l'utilisation des véhicules comme relais permet d'étendre cette distance. Dans un but économique et afin d'éviter la multiplication des stations de bases à chaque coin de rue, l'utilisation des sauts par véhicules intermédiaires prend tout son importance.

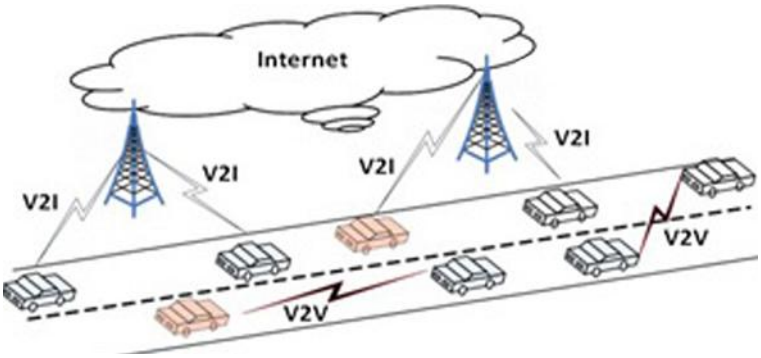


Figure 1.4 : Les modes de communication dans les VANETs

1.4.3. Caractéristiques des réseaux VANET

Les réseaux véhiculaires ont des caractéristiques spécifiques qui les distinguent des réseaux Ad Hoc, à savoir :

- ✓ **La Collecte des informations et la perception de l'environnement proche**

La collecte des informations se fait en utilisant différents capteurs de toutes catégories (caméras, capteurs de pollution, capteurs de pluies, capteurs de l'état de la route et de voiture, etc…) qui permettent au conducteur à bord de son véhicule de disposer d'un certain nombre d'informations et d'une meilleure visibilité pour pouvoir réagir d'une manière adéquate aux changements de son environnement proche.

- ✓ **Capacité de traitement, d'énergie et de communication**

Contrairement au contexte des réseaux Ad Hoc où la contrainte d'énergie à titre d'exemple représente une des problématiques traitées, les éléments du réseau VANET n'ont pas de limite en terme d'énergie et disposent d'une grande capacité de traitement et peuvent avoir plusieurs interfaces de communication (WIFI, Bluetooth et autres). Grâce aux Nouvelles Technologies de l'Information et de la Communication (NTIC) le conducteur peut prendre une décision à l'aide des traitements et des interprétations des informations collectées.

- ✓ **Environnement de déplacement et modèle de mobilité**

Les environnements pris en compte par les réseaux Ad Hoc sont souvent limités à des espaces ouverts ou indoor (comme le cas d'une conférence ou à l'intérieur d'un bâtiment). Les déplacements des véhicules quant à eux sont liés aux structures des routes (intersections, panneaux de signalisation, etc…) et aux stations de base routières (infrastructures) que ce soit dans les autoroutes ou au sein d'une zone métropolitaine. Les contraintes imposées par ce type d'environnement, à savoir les obstacles radio et les effets de la propagation à trajets multiples

(multipath) ou d'évanouissement (fading), affectent considérablement le modèle de mobilité et la qualité des transmissions radio à prendre en compte dans les protocoles de routage. En outre la mobilité est un facteur lié directement au conducteur du véhicule.

- ✓ **Forte mobilité, topologie du réseau et connectivité**

A la différence des réseaux Ad Hoc, les réseaux VANET sont caractérisés par la forte mobilité des nœuds (véhicules), liée à la vitesse des voitures qui est très importante dans les autoroutes. Par conséquent, un nœud peut rejoindre ou quitter le réseau en un temps très court, ce qui rend les changements de topologie très fréquent. De plus, des problèmes peuvent apparaitre quand la majorité des véhicules n'est pas équipée d'un système IVC (Inter Vehicule Communication).

- ✓ **Type de l'information transportée et diffusée**

Un des objectifs des réseaux VANET étant la sécurité routière. Les types de communications s'axeront sur les diffusions de messages d'une source vers plusieurs destinataires. Néanmoins, les véhicules sont concernés par la diffusion d'informations en fonction de leurs positions géographiques et leurs degrés d'implication dans l'évènement déclenché. Dans de telles situations, les communications sont principalement unidirectionnelles.

1.4.4. Applications des réseaux VANET

Les principales applications[1] des réseaux VANET peuvent être classifiées selon le service offert en trois grandes catégories, chaque catégorie peut avoir diverses classes, et au niveau de chaque classe plusieurs applications peuvent être distinguées.

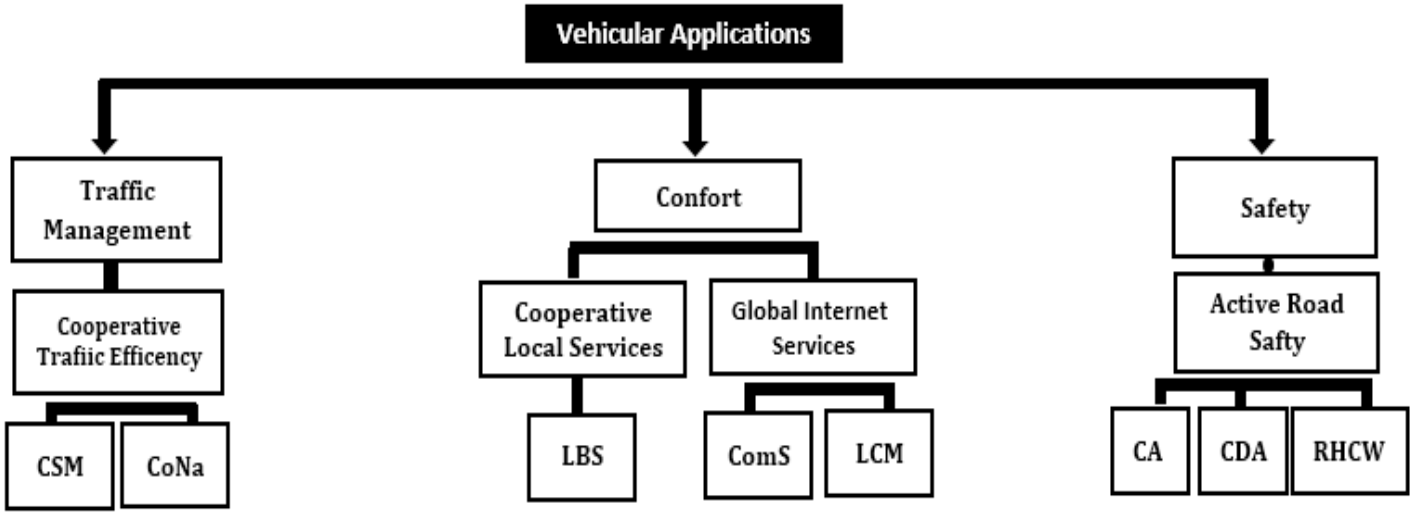


Figure 1.5 : La classification des applications dans les VANET[1]

Ci-dessous, nous identifions un ensemble représentatif des applications des VANETs :

1.4.4.1. Les applications liées aux STI et à la gestion du trafic routier

Les applications liées aux systèmes de transport intelligents (STI) comprennent les messages rappelant les limitations de vitesse ou les distances de sécurité, les systèmes d'aide à la conduite et les véhicules coopératifs : aide aux dépassements de véhicules, prévention des sorties de voies en ligne ou en virage…, etc.

Les applications liées à la gestion de trafic routier visent à optimiser le trafic routier et à prévenir la congestion. Ils consistent à fournir aux conducteurs des informations leur permettant d'adapter leur parcours à la situation du trafic routier, comme exemple : l'ordonnancement des feux de signalisations et la surveillance du trafic. La catégorie de la gestion du trafic est liée à *la classe de l'efficacité du trafic coopératif.*

Depuis que les congestions dépassent la capacité de demande de la circulation, une approche efficace basée sur la gestion du trafic est nécessaire pour réduire la congestion. L'efficacité de trafic coopératif se compose de deux catégories d'applications : les applications liées à la gestion de la vitesse coopérative (CSM) et des applications liées à la navigation coopérative (CoNa).

- ✓ **Les applications de la gestion de vitesse coopérative (CSM) :**

La gestion de la vitesse coopérative (CSM) comprend deux services :

CSM- La notification de la vitesse limite : fournit des notifications concernant la vitesse limite, qui contiennent des limites de vitesse réglementaire actuels et des limites de vitesse recommandé contextuels.

CSM- Vitesse optimale de feu de circulation consultative : Elle est responsable de la vitesse optimale de feu de circulation consultative. Pour ceci, une station sur l'infrastructure fournit des informations au sujet des phases courantes de feu de circulation, le temps restant avant des changements de phases et la durée de chaque phase.

- ✓ **Les applications de la navigation coopérative (CoNa) :**

L'application CoNa propose de nombreux services parmi d'autres :

Ecoulement libre (péage) : Les véhicules sont automatiquement facturés lors de leur passage à travers la zone de péage, en minimisant le retard.

Immatriculation du véhicule, inspection des pouvoirs : l'inspection du véhicule permet de contrôler la légalité des transports de marchandises / personnes.

Les actions d'arrêter les véhicules pour vérifier la validité du permis de conduire ou pour vérifier l'état physique des véhicules avant d'entrer dans une infrastructure routière sont des exemples de contrôles de véhicules. Un réseau de véhicules sans fil permet l'échange de données entre les véhicules et entre les véhicules et les infrastructures routières.

1.4.4.2. Les applications liées au confort du conducteur et des passagers

Cette catégorie d'applications permet d'améliorer le confort des conducteurs et des passagers. Ce confort est illustré par l'accès à internet, la messagerie, le chat inter-véhicule, etc. Les passagers dans la voiture peuvent jouer en réseaux, télécharger des fichiers MP3, envoyer des cartes à des amis…, etc.

Nous citons comme exemple d'application *la gestion des espaces libre dans un parking*, cette application permet de rassembler des informations sur la disponibilité de l'espace libre de stationnement dans les parkings et de renvoyer au conducteur la place libre la plus proche.

L'objectif général de cette catégorie est d'améliorer le confort des passagers. Elle est liée à deux classes, la classe des services locaux de coopération et la classe des services d'internet global.

- ✓ **La classe des services locaux publics coopératifs**

Cette classe fournit des services basés sur la géolocalisation (LBS) :
Les centres de services et d'intérêt comprennent les stations d'approvisionnement des véhicules en énergie, centres d'entretiens des véhicules, zones de repos, parkings, hôtels/restaurants, lieux touristiques, centres médicaux, postes de polices et postes de péage, etc.
Annonces de services : les entreprises transmettent des données de marketing aux clients potentiels qui passent.
Transmission des vidéos en temps réel : Un véhicule émet et transmet une vidéo en temps réel aux autres véhicules ou bien aux unités aux bords de routes.

- ✓ **La classe de services d'internet global**

Cette classe fournit deux types d'applications : les applications des services aux collectivités (ComS) et les applications de la gestion de cycle de vie des stations(LCM).

1.4.4.3.Les applications liées à la sécurité du trafic routier

La diminution du nombre de personnes blessées ou tuées sur les routes est l'une des principales motivations du développement et de l'étude des communications véhiculaires.
Les applications liées à la sécurité ont suscité une attention considérable car elles sont directement liées à minimiser le nombre d'accidents de la route. Cette catégorie est associée aux applications de la classe «*sécurité routière active*» qui vise à fournir des services de sensibilisation et d'alerte au conducteur à travers trois types d'applications : la sensibilisation coopérative (CA), l'assistance à la conduite coopérative (CDA), et les applications d'alertes de risque de collision (RHCW). En fait, la classe de la sécurité routière active offre des fonctions de sensibilisation qui fournissent des informations au conducteur pendant la conduite normale, avertissent le conducteur des conditions de danger de la route et les accidents probables et aident activement le conducteur à éviter des accidents imminents. En d'autres termes, les applications liées à la sécurité sont responsables de la sensibilisation, la mise en garde et l'assistance.

- ✓ **Les applications de sensibilisation coopérative CA**

Les applications de sensibilisations coopératives (CA) : consistent à sensibiliser les conducteurs des autres véhicules et fournir des informations sur l'environnement aux alentours du véhicule. Plusieurs applications sont offertes dans cette catégorie. Parmi ces applications, nous mentionnons : l'indication d'un véhicule d'urgence, indication de l'approche d'une moto et signalisation d'un véhicule lent. Pour ces derniers exemples d'applications, le véhicule diffuse des messages d'alertes aux véhicules de son entourage. Les informations diffusées aident les conducteurs à s'adapter aux conditions de la route.

✓ **Les applications d'assistance et d'aide à la conduite coopérative (CDA)**

Ces applications fournissent des services d'assistance au conducteur :

Systèmes de conduite coopérative (CDS) : cette application exploite l'échange de données de capteurs ou d'autres informations d'état entre les voitures. Ces systèmes de conduite aident les conducteurs pour maintenir un temps et une distance de sécurité entre les véhicules pour s'assurer que le freinage d'urgence ne causera pas de collisions entre les voitures.

Assistance au changement de voie (LCA) : Cette application assiste le conducteur dans le choix de l'instant optimal pour changer de voie et influe sur le comportement des conducteurs en vue d'améliorer les performances de conduite.

✓ **Application d'avertissement de collision et risque de la route (RHCW)**

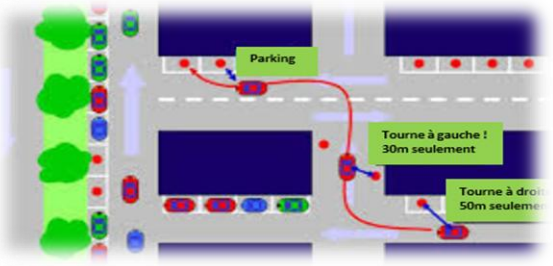
Les applications de la classe (RHCW) fournissent des informations au sujet des collisions imminentes dues à l'état dangereux de la route, obstacles et conducteurs erratiques pour que les conducteurs soient vigilants à la collision imminente. Les systèmes de détection d'accident se fondent sur des radars, des capteurs, ou des caméras afin de détecter une collision imminente. Plusieurs services sont offerts dans cette classe :

Avertissement Coopératif de Collision : Un véhicule surveille activement les messages concernant le statut de la cinématique des véhicules de son voisinage pour avertir les autres véhicules des collisions potentielles.

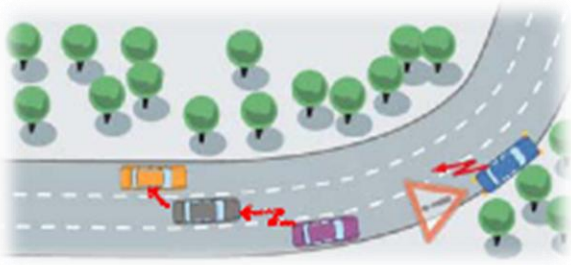
Emergence électronique des feux de stop : un freinage dur d'un véhicule provoque un message d'avertissement qui sera diffusé aux conducteurs mis en danger au sujet de la situation critique avec une latence minimale.

Notification des risques de la route : En détectant un risque de route (par exemple brouillard, fluide, glace, et vent), les véhicules se trouvant dans le même périmètre seront notifiés.

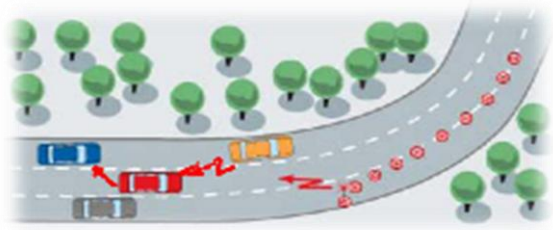
Notification des caractéristiques d'une route : Un véhicule détectant une caractéristique de route (par exemple descente, virage courbe) informe les véhicules voisins.



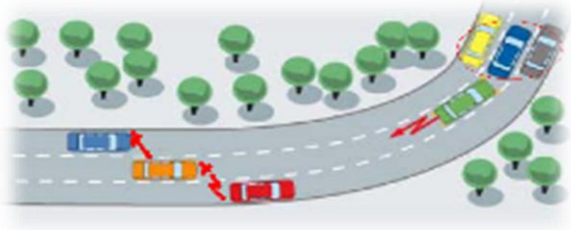
(a) Parking intelligent



(b) Véhicule en panne



(c) Travaux sur les routes



(d) Risque de collision

Figures 1.6 : Exemples de scénarios d'applications des réseaux VANET

1.4.5. Technologies d'accès

Afin de déployer les applications décrites au-dessus, nous faisons un tour d'horizon des technologies de communication sans fil existantes. Ce tour d'horizon permet de présenter les caractéristiques des technologies envisagées pour les VANETs. Il existe deux types de systèmes possibles :

- Les systèmes intra-véhiculaires composés de capteurs internes au véhicule et ne visant pas à diffuser de l'information vers l'extérieur du véhicule.
- Les systèmes extra-véhiculaires visant l'échange d'informations entre une entité et son environnement.

Les systèmes extra-véhiculaires sont divisés en trois sous-systèmes selon leur utilisation :

- Les systèmes de télécommunications, qui sont dominants dans le domaine des communications mobiles, mais qui nécessitent une infrastructure. Ils sont particulièrement utilisés pour les applications de confort de l'usager (Internet à bord, vidéoconférence, autres services payants).
- Les systèmes de radio diffusion (éventuellement numériques), qui proposent de l'information de manière unidirectionnelle. Ils sont particulièrement utilisés pour les applications de gestion du trafic routier.
- Les réseaux informatiques extra-véhiculaires, qui proposent des échanges directs d'informations entre les entités. Ils sont particulièrement utilisés pour les communications V2V et les applications de sécurité routière.

1.4.5.1. Systèmes de communication intra-véhiculaires

Les systèmes intra-véhiculaires ne visent pas la diffusion d'information à l'extérieur du véhicule. Ils sont composés de capteurs, d'une plateforme de calcul et de réseaux filaires ou sans fil. Ces systèmes ont été les premiers développés par les industriels. Chaque constructeur pouvait définir son propre système sans devoir assurer l'interopérabilité avec les véhicules de marque concurrente. Ces systèmes sont connus sous le nom de « systèmes avancés d'aide à la conduite » (ADAS).

Dans la première phase d'acquisition de connaissance sur l'environnement de conduite, les systèmes actuels d'aide à la conduite utilisent deux sortes de capteur ou source d'informations :

Les capteurs proprioceptifs : Ce genre de capteurs fournit des informations internes au véhicule. Ces capteurs se limitent donc à renvoyer des informations sur le comportement et sur les paramètres du véhicule lui-même sans se préoccuper directement de l'environnement de conduite. Toutefois, ces capteurs fournissent des informations précieuses en termes de définition et de détermination du risque. Citons à titre d'exemple les informations sur la vitesse du véhicule acquises grâce à l'odométrie, sur les accélérations (par gyromètre), sur l'état du moteur du véhicule, sur l'état des freins, sur l'adhérence à la route, etc. Ces informations forment une source d'informations indispensable pour connaître, dans un premier temps, l'état et les capacités du véhicule lui-même pour mieux définir le risque encouru et pouvoir proposer, dans un second temps, des solutions pour réduire ce risque.

Les capteurs extéroceptifs **:** contrairement à la première catégorie, ces capteurs embarqués sur le véhicule auront pour mission de percevoir l'environnement de navigation du véhicule. Ils fournissent des informations sur le véhicule lui-même et sur les objets qui l'entourent à partir de leur perception de l'environnement. Citons à titre d'exemple la vision monoculaire ou stéréoscopique, la télémétrie laser ou radar, les ultrasons, etc. Plus communément, il existe le régulateur de vitesses adaptatives ou le parcage automatique. Ces capteurs acquièrent des informations sur les objets dans l'environnement de conduite. Ce genre de capteurs est plutôt utilisé dans la classe des ADAS autonomes puisqu'ils n'exigent aucune interaction physique avec l'environnement et se contentent de percevoir passivement.

Ce type de système n'apporte qu'une connaissance locale et à courte portée de l'environnement du véhicule. Il est donc intéressant de coupler ce système à un système de communication sans fil extra-véhiculaire.

1.4.5.2. Systèmes de communication extra-véhiculaires

Systèmes de télécommunications :

Les systèmes de télécommunications sont également connus sous le nom de réseaux cellulaires mobiles. Cette section traite des standards de télécommunications: GSM et son extension GPRS, et UMTS (4G). L'architecture réseau d'un système de télécommunications contient une station de base qui contrôle l'accès au support et gère le processus d'itinérance (handover).

- ✓ **GSM/GPRS**

Le Global System for Mobile communication(GSM) est la deuxième génération de téléphonie mobile orientée vers la communication de la voix. Avec l'avènement de l'Internet mobile, le General Packet Radio Service(GPRS) a été développé pour permettre la communication des paquets de données. Le GPRS est la génération 2,5 de téléphonie mobile basée sur la commutation de paquets et son débit théorique maximal est de 171,2 kbit/s. Néanmoins, la voix conserve une priorité supérieure dans la plupart des réseaux basés sur GSM. Le GSM/GPRS est un système radio à délai modéré, à faible débit, entre une station de base et un véhicule. Le trafic de données, plus particulièrement quand il transporte des informations de sécurité routière, a des besoins différents. En effet, il exige une communication en temps réel ayant un faible délai, et une fiabilité de données élevée. Cette technologie n'est donc pas adaptée au transport de paquets pour les applications de sécurité du trafic routier. Par contre, le GSM/GPRS fournit une connexion internet (minimale en terme de débit) utilisée par le service SOS de certains constructeurs automobiles par exemple.

- ✓ **UMTS**

L'Universal Mobile Telecommunication System (UMTS) est la norme de la troisième génération de téléphonie mobile. La transmission de données peut atteindre théoriquement des débits de transfert de 1,92 Mbit/s, et de 128 kbit/s pour les équipements mobiles à grande vitesse. Comme chaque utilisateur est lié à un opérateur téléphonique qui gère la facturation, l'UMTS est employé pour l'accès aux services payants tels que l'Internet à bord, la vidéo à la demande ou les jeux en réseau.

Grâce à ses caractéristiques techniques, l'UMTS est plus adapté aux applications de sécurité du trafic routier que le GSM/GPRS. En effet, le débit est constamment augmenté, et comme les applications de sécurité du trafic routier génèrent un volume important de données, l'UMTS répond à ce besoin. Mais des manques perdurent notamment en terme de garantie de délai.
Les systèmes de télécommunications sont une solution peu onéreuse et déjà existante. Mais dans notre contexte de communication véhiculaire sur autoroute ces systèmes n'assurent aucune garantie de délai.
De plus, rien ne dit que les véhicules utiliseront le même opérateur. Il y aura donc un délai supplémentaire afin d'atteindre le(s) réseau(x) opérateur(s) des autres véhicules. Cela explique donc pourquoi les systèmes de télécommunications sont principalement utilisés pour les applications de confort de l'usager.

Systèmes de radiodiffusion numérique

Les systèmes de radiodiffusion numérique proposent de diffuser l'information depuis la station de base jusqu'aux utilisateurs. C'est donc un système unidirectionnel. Leur avantage est que les véhicules reçoivent la même information « au même moment ». Cette section présente trois standards pour la diffusion mobile : RDS/TMC,DAB/DMB, et DVB-T/DVB-H.

- ✓ **RDS/TMC**

Les systèmes de navigation dotés d'un récepteur RDS/TMC (Radio Data System/Traffic Message Channel) leur permettent de calculer les itinéraires en tenant compte des informations délivrées par les opérateurs de service. Le RDS est un système de diffusion de données par la radio permettant d'envoyer des informations, transportées en plus du signal audio normal en modulation de fréquence, grâce à une sous-porteuse de la FM. Le débit de données RDS est de 1,2 kbit/s. Le TMC désigne une norme européenne de diffusion de données numérique sur les systèmes de navigation [ISO 03]. Les données transitent ainsi jusqu'à l'utilisateur sur le canal RDS de la radio FM.

- ✓ **DAB/DMB**

Le DMB (Digital Multimedia Broadcasting) est une évolution du DAB (Digital Audio Broadcasting), développé et normalisé par l'European Telecommunication Standards Institute (ETSI) en 2005. Le DMB utilise un nouveau mode de compression en MPEG-4 qui permet de diffuser de la radio numérique avec des contenus multimédias, mais aussi de la télévision mobile, sur des appareils de petite dimension tels que des téléphones mobiles. Le DMB a été développé afin d'être le remplaçant de la radio FM. Malheureusement, avec un débit de 2,4 Mbit/s, une latence de 100 ms, un délai non borné et une communication unidirectionnelle, ces technologies ne peuvent supporter que les applications d'information de trafic routier.

- ✓ **DVB-T/DVB-H**

Le Digital Video Broadcasting(DVB) est une technologie de diffusion pour la télévision numérique concurrente du DAB/DMB. Le Digital Video Broadcasting-Terrestrial (DVB-T) est un système qui transmet la voix et la vidéo via un flux compressé MPEG. Le DVB-H (Digital Video Broadcasting-Handheld) est une version optimisée de DVB-T, lequel n'avait pas été conçu à l'origine pour un usage nomade. Le DVB-H ajoute au DVB-T une redondance temporelle et une forte protection des flux transmis. Le DVB-H est ainsi adapté pour la réception mobile. Les

différences par rapport au DAB/DMB sont un débit supérieur et une portée réduite. Bien que ces deux technologies soient adaptées pour le transport de vidéo, elles ne répondent pas aux contraintes pour les applications de sécurité du trafic routier. En effet, le DVB-T/DVB-H a une latence de six secondes, ce qui est trop important dans le contexte critique de ces applications. Toutefois, le DVB-T/DVB-H est utilisé pour les applications de confort utilisant les communications I2V uniquement. Par exemple, la diffusion vidéo dans un véhicule roulant à plus de 80 km/h utilise le DVB-H.

Réseaux informatiques extra-véhiculaires

- ✓ **Infrarouge :**

L'infrarouge (IR) est un réseau à visibilité directe. Les émetteurs et récepteurs doivent être proches les uns des autres. Il est adapté pour des communications intervéhiculaires à très courte portée en point à point.
Cette technologie souffre de plusieurs limitations. En plus d'être uniquement en point à point, l'IR est une technologie de « ligne de visée » qui ne peut traverser les murs, et requiert donc que la voie entre les périphériques soit dégagée. Cette technologie souffre aussi des perturbations dues aux interférences lumineuses.

- ✓ **WiMAX :**

Le réseau sans fil métropolitain, WiMAX, basé sur la norme IEEE 802.16, permet d'atteindre des débits de 70 Mbit/s sur un rayon de 50 kilomètres. Avec un débit élevé et un délai modéré, le WiMAX est adapté pour l'accès à Internet. Sa version mobile, Mobile WiMAX (basé sur le standard IEEE 802.16e), offre aussi une connectivité à moyenne et longue portée, mais adaptée pour des véhicules à vitesse modérée.

- ✓ **WiFi :**

Aujourd'hui, la technologie Wireless Fidelity (WiFi) est devenue omniprésente dans les ordinateurs portables, les téléphones portables ou les consoles de jeux. Grâce à cette démocratisation et le faible coût de production, la technologie WiFi est une technologie abordable pour le déploiement de réseaux sans fil véhiculaires. Depuis la fin des années 1990, date d'apparition des premiers équipements utilisant la technologie WiFi sur le marché, trois spécifications de la couche physique pour le standard IEEE 802.11 furent ajoutées afin d'accroitre la vitesse de transmission. La dernière spécification en date est le 802.11n qui propose des débits théoriques de 300 Mbit/s. Malheureusement, en pratique le surcoût du protocole réduit de moitié les débits potentiels de la couche application. Cette dégradation de débit peut être pénalisante, surtout dans les réseaux véhiculaires. À première vue, la couverture radio omnidirectionnelle de 400 mètres semble suffisante pour maintenir une connectivité multisaut dans le milieu autoroutier ou urbain. Mais de nombreux travaux de recherche ont démontrés qu'à cause des caractéristiques uniques des VANETs, cette technologie ne peut pas être appliquée telle quelle. À plus forte raison dans le contexte d'application de sécurité du trafic routier où le IEEE 802.11(g) affiche un taux de perte de paquets élevé à vitesse élevée.

- ✓ **DSRC:**

Dedicated Short Range Communication(DSRC) regroupe un ensemble de technologies dédiées aux communications véhiculaires. À l'origine, la technologie DSRC a été conçue pour

répondre au besoin de transactions financières électroniques (télépéage). C'était un modèle de communication à courte portée (4 à 10 mètres) avec des débits inférieurs à 1 Mbit/s. Ensuite, le standard DSRC a évolué à partir du IEEE 802.11a vers la norme IEEE 802.11p ou WAVE (Wireless Access for Vehicular Environments) afin de répondre aux caractéristiques des VANETs. Le DSRC propose un canal de communication spécialement conçu pour transmettre des messages de très haute priorité à l'instar de certains messages critiques liés à la sécurité routière. Le WAVE présente aussi des caractéristiques beaucoup plus adaptées à la mobilité (comme des temps d'établissement de connexion plus courts) qui permettent l'envoi à la volée d'informations à des véhicules roulants à grande vitesse. Il présente une bonne fiabilité avec un taux d'erreur de 10^{-6} à 160 km/h. La technologie IEEE 802.11p est particulièrement adaptée pour les applications à portée moyenne et sensibles au délai.

1.4.6. Protocoles de routage Ad hoc véhiculaire

Le routage est le mécanisme qui permet de trouver et maintenir un chemin de communication entre une paire de nœuds distants dans un réseau VANET, et qui fonctionne selon deux phases distinctes :

- Une phase de signalisation assurée par des échanges de messages de contrôle afin de permettre la construction et le maintien des chemins,
- Une phase d'acheminement des paquets de données.

En raison des caractéristiques de ces réseaux, les protocoles de routage conçus pour les réseaux filaires ne peuvent être directement utilisés. Pour fonctionner, ces protocoles doivent prendre en considération certains aspects liés à l'environnement dans lequel ils sont déployés tels que les changements de la topologie dus à la mobilité des nœuds, l'absence d'une entité centrale de gestion…, etc.

1.4.6.1. Classification des protocoles de routage

Selon la manière dont les nœuds établissent les chemins nous pouvons distinguer deux grandes classes de protocoles : les protocoles de routages basés sur la topologie (topology-based)[2] et les protocoles de routages géographiques (position-based)[3].

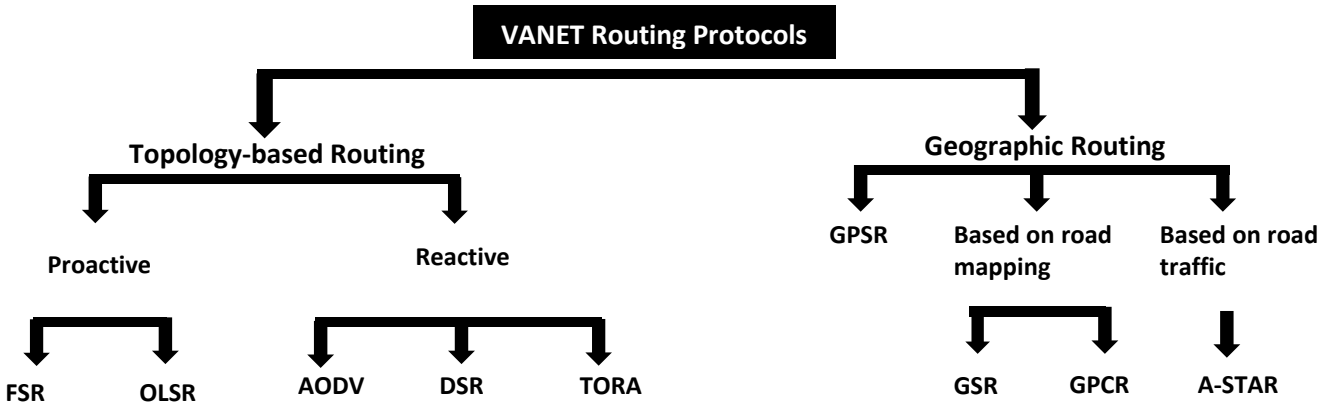


Figure 1.7 : Protocoles de routage pour les VANETs

1.4.6.1.1. Protocole de routage basé sur la topologie

Ces protocoles de routage utilisent les informations des liens qui existent dans le réseau pour l'acheminement des paquets. Ils peuvent être classifiés en trois familles : protocoles proactifs, réactifs et hybrides.

Le principe des *protocoles proactifs* (table-driven) est de maintenir à tout instant une vue globale et cohérente de la topologie du réseau, et de construire des routes entre les nœuds avant qu'elles ne soient demandées. Ces protocoles exigent que chaque nœud maintienne une table de routage indiquant par quel voisin passer pour atteindre un destinataire. Grâce à ces informations, chaque nœud dispose à tout instant d'un chemin vers n'importe quel autre nœud du réseau.

Pour traiter les changements de topologie, les nœuds diffusent des messages de contrôle à travers le réseau. Les protocoles Fisheye State Routing (FSR) et Optimized Link State Routing (OLSR) ont été proposés pour les réseaux ad hoc véhiculaires[2].

Les *protocoles réactifs* (on-demand driven) construisent des chemins uniquement lorsque ces derniers sont requis par un nœud source et ne gardent que les routes en cours d'utilisation par le processus de routage. On dit alors que la topologie du réseau est découverte à la demande.

Ainsi lorsqu'un nœud cherche à communiquer avec une destination pour laquelle il ne connait pas le chemin, il lance un processus de découverte dans le réseau (généralement par inondation). Cette phase de découverte se termine lorsque le chemin est trouvé. Ces chemins formés sont susceptibles d'être rompus à cause de la haute mobilité des véhicules. Les ruptures de liens sur les chemins sont alors traitées au moyen d'un mécanisme de maintenance, dont le but est de les identifier, puis si possible de les corriger.

Les protocoles hybrides combinent les principes des deux catégories précédentes. Ils utilisent les mécanismes des protocoles proactifs pour découvrir les proches voisins. Mais, pour le reste du réseau, cette catégorie agit comme les protocoles réactifs.

En général, les protocoles basés sur la topologie ne supportent pas les réseaux qui dépassent quelques centaines de nœuds[4].

1.4.6.1.2. Protocole de routage géographique

Les protocoles de routage géographique (ou basés sur la position) utilisent des coordonnées géographiques (par exemple, fournies par GPS) afin de trouver un chemin vers la destination[5]. Pour atteindre cet objectif, les coordonnées géographiques des nœuds sont incluses dans les tables de routage.

Concrètement, un nœud inclut l'identifiant et la position de la destination (fournis par le protocole de routage lui-même ou par un protocole de service de localisation[6] indépendant) dans le paquet à envoyer, et par la suite les nœuds intermédiaires utilisent les informations géographiques incluses dans ce paquet et celles disponibles dans leurs tables de routage pour

retransmettre le paquet et répètent le même mécanisme jusqu'à ce que celui-ci atteigne la destination.

L'avantage majeur de ces protocoles par rapport aux protocoles précédents est qu'ils réduisent considérablement la signalisation (les paquets de contrôle), notamment dans les réseaux larges et dynamiques.

1.4.6.2. Le protocole OLSR (Optimized Link State Routing)

Description du protocole :

Le protocole OLSR [7] appartient à la famille des protocoles proactifs et il a été développé pour les réseaux Ad-Hoc.

Le protocole OLSR utilise des échanges périodiques de messages HELLO pour permettre à chaque nœud de connaître ses voisins à 1-saut et à 2-sauts. Par la suite, OLSR utilise une diffusion optimisée des messages de contrôle grâce à l'utilisation des relais multipoints MPR.
Ceci permet à chaque nœud d'avoir une vue de la topologie et ainsi utiliser un algorithme de plus court chemin pour calculer sa table de routage vers toutes les destinations.

L'inondation par relais multipoints réduit considérablement la charge du trafic dans le réseau car juste une partie des nœuds participent au processus d'inondation. Ainsi, le protocole OLSR est très souhaitable pour les réseaux très denses.

Formats des messages OLSR :

Le protocole OLSR utilise un format de paquet unifié pour tout type de données relatives au protocole. Ceci permet de faciliter l'extensibilité du protocole sans casser la compatibilité ascendante.

Chaque paquet encapsule un ou plusieurs messages. Les messages partagent un format d'entête commun, qui permet aux nœuds d'accepter et de transmettre correctement des messages d'un type donné.

En dehors du paquet, OLSR utilise les messages HELLO qui servent pendant les étapes de découverte et de déclaration de voisinage, les messages TC qui servent à échanger les informations de topologie, les messages MID pour déclarer des nœuds avec des interfaces multiples et les messages HNA qui permettent à un nœud de déclarer ses connexions avec des réseaux non OLSR.

i. Format du paquet OLSR :

La disposition de base de n'importe quel paquet dans OLSR est comme suit (en omettant les entêtes IP et UDP) :

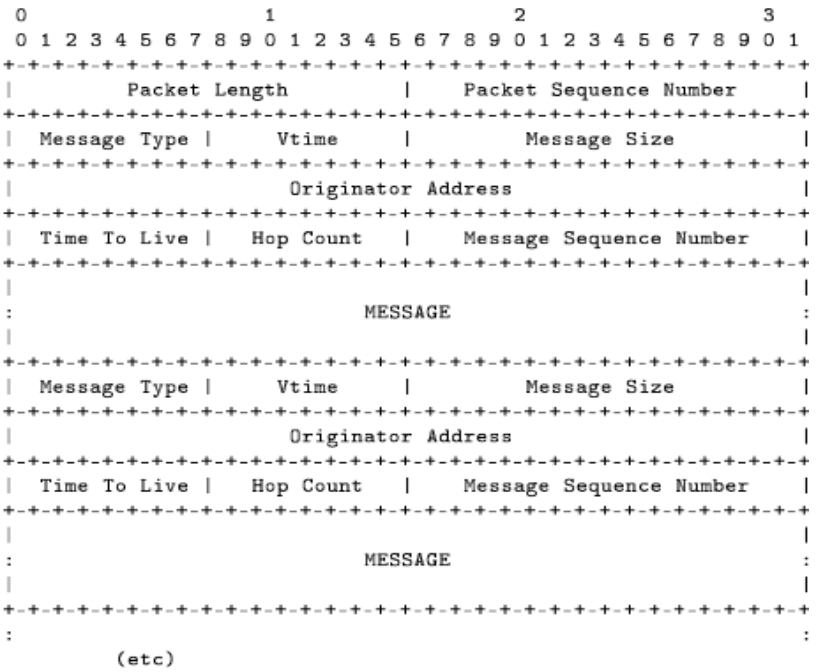


Figure 1.8 : Format du paquet OLSR[7]

- **L'en-tête du paquet**

Packet Length : ce champ indique la longueur du paquet en octet.

Packet Sequence Number : ce champ indique le numéro de séquence du paquet, Il doit être incrémenté par 1 à chaque fois qu'un nouveau paquet OLSR est transmis. Si le paquet ne contient aucun message, il est immédiatement rejeté.

- **L'en-tête du message**

Message Type : Ce champ indique quel type de message doit être trouvé dans la partie «MESSAGE » du paquet.

Vtime : Ce champ indique pour combien de temps après la réception d'un message, un nœud doit considérer l'information contenue dans le message comme valide, à moins qu'une mise à jour plus récente de l'information ne soit reçue.

Message Size : Ce champ donne la taille du message, compté en octet et mesuré à partir du champ « type de message » et jusqu'au commencement du prochain « type de message » (ou - s'il n'y a aucun message suivant - jusqu'à la fin du paquet).

Originator Address : contient l'adresse principale du nœud originaire du message

Time To Live : Ce champ contient le nombre maximum de sauts qu'un message peut passer.

Hop Count : Ce champ contient le nombre de sauts qu'un message a atteint. Avant qu'un message ne soit retransmis, ce champ doit être incrémenté par 1. Au début, il est mis à «0» par le générateur du message.

Message Sequence Number **:** Tout en produisant un message, le nœud « générateur » assignera un numéro d'identification unique à chaque message. Ce nombre est inséré dans le champ numéro de séquence du message. Le numéro de séquence est incrémenté de 1 pour chaque message généré par le nœud. Les numéros de séquence sont utilisés pour s'assurer qu'un message donné n'est pas retransmis plus qu'une fois.

ii. Format du message HELLO :

Pour s'adapter aux opérations de détection de liens, de détection de voisinage et la signalisation de choix des MPR, aussi bien que pour s'adapter pour les futures extensions éventuelles, une approche semblable au format global de paquet est considérée. Ainsi, le format proposé pour un message Hello est comme suit :

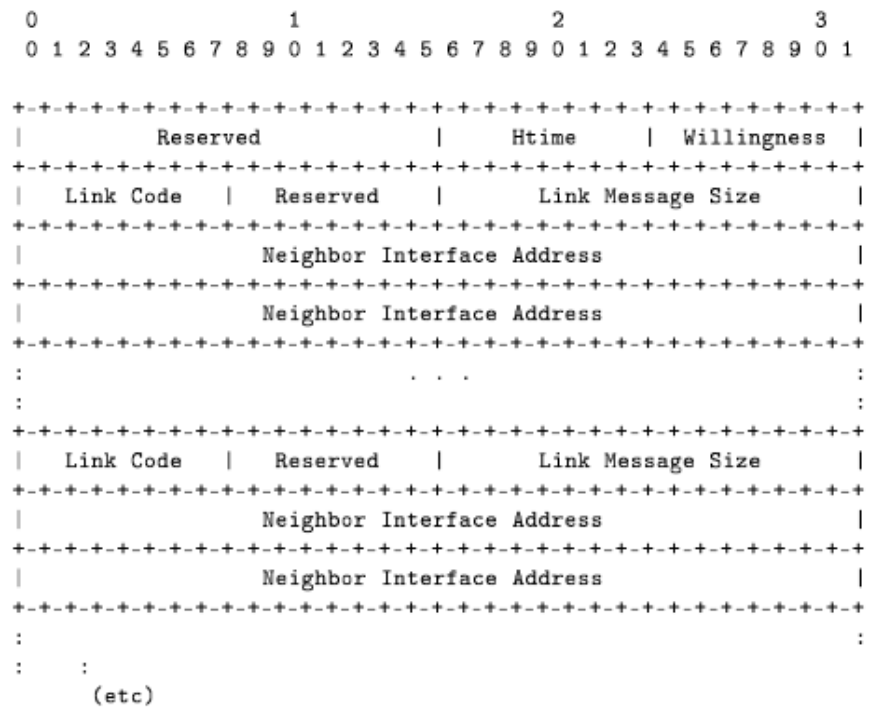


Figure 1.9 : Format d'un message OLSR - HELLO[7]

Ceci est envoyé comme donnée dans la partie « message » du format général du paquet décrit dans la section précédente, avec le « type de message » mis à la valeur *HELLO_MESSAGE*, et le champ TTL à 1 (un).

Reserved : ce champ doit être mis à la valeur « 0000/Hex»

HTime : ce champ indique l'intervalle d'émission hello employé par un nœud sur cette interface particulière, c'est-à-dire le temps avant transmission du prochain hello.

Willingness : ce champ indique la volonté d'un nœud à jouer le rôle de MPR pour d'autres nœuds. Un nœud avec la volonté WILL_NEVER ne doit jamais être choisi comme MPR par d'autres nœuds. Un nœud avec la volonté WILL_ALWAYS doit toujours être choisi comme MPR. Par défaut, un nœud DEVRAIT annoncer une volonté de WILL_DEFAULT.

Link Code : ce champ indique des informations sur le lien entre l'interface de l'expéditeur et la liste suivante des interfaces du voisin. Il indique également des informations sur le statut du voisin. Les codes de liens non reconnus par un nœud, sont discrètement jetés.

Link Message Size : la taille du message de lien, comptée en octet et mesurée à partir du champ « link code » jusqu'au prochain champ « link code ».

Neighbor Interface Address : l'adresse de l'interface d'un nœud voisin.

Le champ *Link Code* de taille 8 bits, contient à la fois les informations concernant les liens vers les nœuds voisins et le type de ces derniers.

Le Tableau 1.1 et le Tableau 1.2 présentent la liste des valeurs possibles pour les deux champs de type de lien et type de voisin selon les spécifications du *RFC3626.*

0	1	2	3	4	5	6	7
				Type de voisin		Type de liens	

Tableau 1.1 : Champs Link Code

Type de lien	
UNSPEC_LINK	Pas d'informations
ASYM_LINK	Lien asymétrique
SYM_LINK	Lien symétrique
LOST_LINK	Lien est perdu
Type de voisin	
SYM_NEIGH	Voisin symétrique
MPR_NEIGH	Voisin a été sélectionné comme MPR
NOT_NEIGH	Pas de voisins / Pas encore symétrique

Tableau 1.2 : Valeurs possible pour le champ Link Code

iii. Format du message TC :

Le format proposé pour le message TC est comme suit :

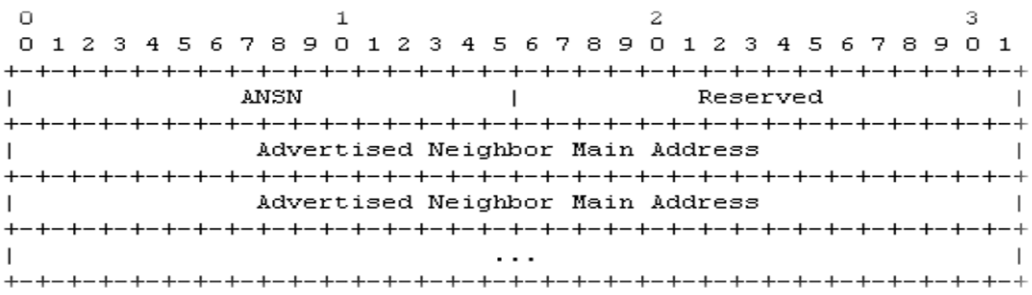


Figure 1.10 : Format d'un message OLSR – TC[7]

Ce format est envoyé comme donnée dans la partie message du paquet avec le "type de message" initialisé à *TC_MESSAGE*. Le TTL doit être initialisé à 255 (valeur maximale) pour diffuser le message dans le réseau entier.

Numéro de séquence des voisins (ANSN). Un numéro de séquence est associé à l'ensemble des voisins du nœud émetteur. Chaque fois qu'un nœud détecte un changement dans l'ensemble de ces voisins, il incrémente ce numéro de séquence. Ce numéro de séquence est envoyé dans ce champ ANSN du message TC pour conserver une trace de l'information la plus récente. Quand un nœud reçoit un message TC, il peut décider sur la base de ce numéro de séquence, si l'information reçue concernant les voisins du nœud générateur est plus récente que celle déjà reçue.

Adresse Principale du Voisin. Ce champ contient l'adresse principale du nœud voisin. Toutes les adresses principales des voisins du nœud générateur sont mises dans le message TC.

Réservé. Ce champ est réservé, et doit être initialisé à "0000/Hex".

Les Relais Multipoints :

L'objectif des relais multi-points est de limiter localement le nombre de retransmissions lors d'une inondation : chaque nœud dispose d'un ensemble de relais multipoints choisis parmi ses voisins et seuls ces relais multi-points peuvent retransmettre les paquets émis par le nœud. Les paquets sont toutefois reçus par tous les voisins. Si l'ensemble des relais multi-points est plus petit que l'ensemble des voisins, il en résulte immédiatement une réduction du trafic retransmis, en outre, plus cet ensemble est petit, plus la réduction du nombre de retransmissions est efficace.

Pour savoir s'il peut retransmettre un paquet reçu, chaque nœud doit donc maintenir la liste des nœuds qui l'ont choisi comme relai multi-points. Ces derniers sont les sélectionneurs multi-points du nœud.

L'ensemble des MPR pour un nœud i, noté MPR(i), est un sous-ensemble minimal choisi parmi ses voisins symétriques à un saut qui satisfont les propriétés suivantes :

- Chaque nœud parmi les voisins à deux sauts de i doit avoir un lien symétrique avec au moins un élément du sous-ensemble MPR(i),
- Plus le sous-ensemble des MPR est petit, plus il est optimal

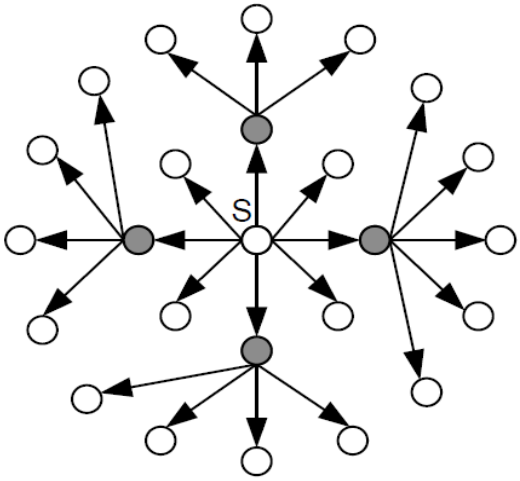


Figure 1.11 : Sélection des MPRs

***Déclaration des Relais multipoints* :**

Afin de fournir les informations sur la topologie nécessaire pour construire les routes et ainsi garantir le routage des paquets, chaque nœud qui a été sélectionné comme étant MPR, diffuse périodiquement des messages de contrôle de la topologie TC (*Topology Control*). Ces messages sont reçus par tous les nœuds mais transmis juste par les MPR.

Chaque nœud *x* qui a été sélectionné comme MPR maintient une liste des voisins qui l'ont sélectionné comme relais multipoint. On définit cet ensemble par :

$$\mathbf{MPRSel(x) = \{y \in N1(x) \mid x \in MPR\ (y)\}}$$

Chaque message *TC,* envoyé par un nœud *x*, contient la liste **MPRSel(x)** ainsi qu'un numéro de séquence (*ANSN*) associé au message. Ces messages permettent à chaque nœud de maintenir à jour sa table d'information sur la topologie et ainsi faciliter le calcul de sa table de routage.

Le fonctionnement du protocole OLSR :

Les réseaux Ad-Hoc sont caractérisés par une topologie dynamique et changeante. Afin de détecter tout changement dans le réseau et générer les informations sur la topologie, le protocole OLSR se base essentiellement sur la détection et la mise à jour de la liste des voisins de chaque nœud.

On peut classifier les liens entre deux nœuds en trois catégories :

***Asymétrique* :** Un lien est dit asymétrique si le premier nœud reçoit les messages de l'autre nœud mais il n'a pas reçu la confirmation que l'autre nœud l'entend.

***Symétrique* :** Un lien est dit symétrique si chaque nœud entend l'autre.

Perdu **:** Un lien est dit perdu si ce lien a été déclaré précédemment étant symétrique ou asymétrique mais à ce moment aucun message n'est reçu du nœud déclaré perdu.

Dans le but de découvrir les nœuds voisins, chaque nœud envoie périodiquement à tous ses voisins des messages **HELLO**. Ces messages contiennent :

- La liste des adresses des voisins pour lesquels le nœud a reçu un paquet HELLO
- La liste des adresses des voisins qui sont accessibles par un lien bidirectionnel
- La liste des adresses des voisins que le nœud a choisis comme relais multipoints.

Ces messages servent à :

- La découverte des identités des nœuds voisins ;
- La découverte de l'état des liens ;
- La signalisation de la sélection MPR.

Lorsqu'un nœud reçoit un message HELLO d'un voisin, alors il place le voisin dans la première liste. Si en outre, son adresse est dans la première liste du message reçu, alors il considère que le lien entre lui et le voisin est bidirectionnel, il place donc le voisin dans la seconde liste. Enfin, si son adresse est dans la troisième liste, cela signifie que le voisin est un de ses sélectionneurs multi-points, le nœud sait ainsi qu'il doit retransmettre les paquets de ce voisin en diffusion.

A l'aide des messages HELLO de ses voisins, un nœud peut calculer la liste de tous ses voisins à deux sauts, il s'agit de l'ensemble des voisins de ses voisins qu'il ne connait pas directement. Il possède alors toutes les informations requises pour sélectionner un ensemble de voisins qui formeront ses relais multi-points. Lors de l'envoi du prochain message HELLO, cette liste sera transmise aux voisins et donc aux relais multi-points qui seront ainsi informés de leur rôle. L'ensemble des relais multi-points doit être recalculé lorsque l'ensemble des voisins change ainsi que lorsque l'ensemble des voisins à deux sauts change.

Le processus de sélection des relais multi-points fait que les liens entre un nœud et ses relais multi-points sont tous bidirectionnels. Cette caractéristique permet de ne pas souffrir de problèmes liés à des liens unidirectionnels dans le routage.

Chaque nœud envoie régulièrement en diffusion un message TOPOLOGY CONTROL (TC) à destination de l'ensemble du réseau afin de déclarer l'ensemble de ses sélectionneurs multi-relais. L'utilisation des relais multi-points pour l'envoi par inondation d'un tel message permet de limiter le nombre de retransmissions inutiles du message, tout en garantissant que tous les nœuds du réseau reçoivent le message.

La réception de ces messages permet à chaque nœud de construire une topologie du réseau basée sur les relais multi-points. L'algorithme de Dijkstra est ensuite mis en œuvre pour trouver une route pour chaque nœud.

Une conséquence du fait que seuls les sélectionneurs multi-points sont transmis dans les messages TOPOLOGY CONTROL est que les routes sont toutes une suite de relais multi-points, de

la source à la destination. Une démonstration formelle montre que ces routes formées uniquement de relais multi-points représentent le plus court chemin.

Calcul des routes :

Chaque nœud maintient une table de routage qui lui permet d'acheminer les paquets vers un destinataire. Ces tables de routage sont calculées grâce à l'algorithme de plus court chemin de Dijkstra[8] en se basant sur les informations conservées par les nœuds et aussi les informations fournies par les messages de contrôle TC. Ces tables de routage sont recalculées à chaque changement survenu dans la topologie et ainsi permettre de mettre à jour les routes vers toutes les destinations dans le réseau.

1.4.6.3. Le protocole AODV (Ad hoc On-demand Distance Vector)

Le protocole AODV [9] est un protocole de routage réactif à vecteur de distance. Il ne construit pas a priori la table de routage mais réagit à la demande et essaie de trouver un chemin avant de router les informations. Tant que la route reste active entre la source et la destination, le protocole de routage n'intervient pas, ce qui diminue le nombre de paquets de routage échangés entre les nœuds constituant le réseau. Lorsqu'un nœud essaie de communiquer avec un autre nœud du réseau, l'échange de messages se fait en plusieurs étapes décrites ci-dessous :

Découverte de route :

Lorsqu'un nœud source a besoin d'une route vers une certaine destination et qu'aucune route n'est disponible (la route peut être non existante, avoir expiré ou être défaillante), la source diffuse en *broadcast* un message de demande de route RREQ (*Route REQuest*). Ce message contient un identifiant (RREQ_ID) associé à l'adresse de la source (@SRC) qui servira à identifier de façon unique une demande de route. Le nœud source enregistre cet identifiant de paquet RREQ ([RREQ_ID, @SRC]) dans son historique (*buffer*) et l'associe à un *timer* qui décomptera sa durée de vie au-delà de laquelle cette entrée sera effacée.

Quand un nœud intermédiaire qui n'a pas de chemin valide vers la destination reçoit le message RREQ, il ajoute ou met à jour le voisin duquel le paquet a été reçu. Il vérifie ensuite qu'il ne l'a pas déjà traité en consultant son historique des messages traités. Si le nœud s'aperçoit que la RREQ est déjà traitée, il l'abandonne et ne la rediffuse pas.
Sinon, il met à jour sa table de routage à l'aide des informations contenues dans la requête afin de pouvoir reconstruire ultérieurement le chemin inverse vers la source. Il incrémente ensuite le nombre de sauts HC (Hop Count) dans la demande de route et la rediffuse.

Il est à noter qu'AODV utilise le principe des numéros de séquence pour pouvoir maintenir la cohérence des informations de routage. Ce numéro, noté SN (Sequence Number), est un champ qui a été introduit pour indiquer la fraicheur de l'information de routage et garantir l'absence de boucles de routages.

À la réception d'un paquet RREQ, la destination ajoute ou met à jour dans sa table de routage un chemin vers le nœud voisin duquel il a reçu le paquet ainsi qu'un chemin vers la source. La destination génère ensuite une réponse de route RREP qu'elle envoie en *unicast* vers le prochain

saut en direction de la source. Notons qu'un nœud intermédiaire peut aussi générer un RREP si la requête l'autorise à le faire (bit *destination_only* de la RREQ mis à 0) et qu'il dispose déjà dans sa table de routage d'un chemin valide vers la destination.

Les nœuds intermédiaires qui reçoivent la RREP vont mettre à jour le chemin qui mène à la destination dans leur table de routage et retransmettre en unicast le message (après avoir incrémenté le nombre de sauts) vers le nœud suivant en direction de la source sachant que cette information a été obtenue lors du passage de la RREQ.
Lorsque la réponse de route atteint la source, un chemin bidirectionnel est établi entre la source et la destination et la transmission de paquets de données peut débuter.

Maintenance des routes :

Afin de maintenir les routes, une transmission de messages HELLO est effectuée. Ces messages sont en fait des réponses de route (RREP) diffusés aux voisins avec un nombre de sauts égal à un. Si au bout d'un certain temps, aucun message n'est reçu d'un nœud voisin, le lien en question est considéré défaillant. Alors, un message d'erreur RERR (*Route ERRor*) se propage vers la source et tous les nœuds intermédiaires vont marquer la route comme invalide et au bout d'un certain temps, l'entrée correspondante est effacée de leur table de routage. Le message d'erreur RERR peut être diffusé ou envoyé en *unicast* en fonction du nombre de nœuds à avertir de la rupture de liaison détectée. Ainsi, s'il y en a un seul, le message est envoyé en *unicast* sinon, il est diffusé.

AODV a l'avantage de réduire le nombre de paquets de routage échangés étant donné que les routes sont créées à la demande et utilise le principe du numéro de séquence pour éviter les boucles de routage et garder la route la plus fraîche. Cependant, l'exécution du processus de création de route occasionne des délais importants avant la transmission de données.

Chapitre 2 : Menaces et problèmes de sécurité dans les VANETs

Comme on a vu dans le premier chapitre, les applications de sécurité du trafic routier utilisent les messages d'alerte pour informer le conducteur de situations potentiellement dangereuses (conditions de route dégradées, freinage d'urgence d'un autre véhicule, obstacle, etc.). Si ces alertes sont envoyées à tort, ou à outrance, alors l'utilisateur n'y prêtera plus d'attention. L'alerte elle-même peut devenir une menace, et provoquer des accidents à cause des réactions (inutiles, inappropriées, inadaptées) des utilisateurs. Ainsi, un attaquant pouvant injecter des messages falsifiés dans le VANET, pourra causer la « désensibilisation » de l'utilisateur ou des accidents, contrairement à l'objectif d'amélioration du trafic routier.

Pour pouvoir sécuriser un VANET, il est nécessaire de connaître les menaces possibles. Ainsi, dans ce chapitre nous allons détailler les modèles d'attaquant, les menaces de sécurité au niveau applicatif ainsi que les vulnérabilités des deux protocoles AODV et OLSR déjà présentés dans le premier chapitre.

2.1. Attaques dans les réseaux sans fil véhiculaires

2.1.1. Modèles d'attaquant

Pour bien identifier les attaques possibles sur un réseau VANET, il est nécessaire de définir les modèles d'attaquant possibles.
Nous définissons les critères de classification d'attaquant suivants :

- ✓ **Actif ou Passif** :

Un attaquant passif ne peut qu'écouter clandestinement le canal de transmission. Cette attaque peut être conduite par un voisinage curieux, mais aussi pour une entreprise qui cherche à créer des profils de conducteurs. Un attaquant actif peut générer, modifier, rejeter ou rejouer des messages afin de disséminer de fausses informations. Le but d'un attaquant actif est de s'octroyer des privilèges afin d'améliorer son environnement de conduite. Ainsi, il peut usurper l'identité d'un véhicule de secours pour faciliter son déplacement.

- ✓ **Interne ou Externe :**

Un attaquant interne est un membre authentifié du réseau qui peut communiquer avec les autres membres du réseau. Comme il fait partie du réseau, il possède déjà quelques informations sur les nœuds constituant le réseau (comme les clés publiques utilisées par les autres véhicules). Un attaquant interne peut causer plus de dommages au réseau que l'attaquant externe qui a un accès limité au système.

- ✓ **Malicieux ou Rationnel :**

Un attaquant malicieux cherche à prouver une capacité ou une réussite personnelle. Pour cela, il cherche à détecter des zones de vulnérabilité et à les exploiter pour perturber le système, ou blesser des membres du réseau. Des attaquants qui causent délibérément des accidents de la route sont considérés comme malicieux. Par conséquent, l'attaquant malicieux est prêt à tout pour arriver à ses fins quels que soient les coûts et les conséquences. Par opposition, l'attaquant rationnel vise l'accomplissement d'une tâche spécifique sur le réseau en défaveur (ou en faveur)

d'une personne identifiée. Les attaques rationnelles sont plus prévisibles que les attaques malicieuses.

✓ **Mal intentionné ou Involontaire :**

Un attaquant est dit mal intentionné s'il vise délibérément à remettre en cause le bon fonctionnement du réseau. Ce type d'attaquant est à distinguer d'un attaquant involontaire qui peut par exemple lancer (sans le vouloir) une attaque à partir d'un capteur défectueux.

✓ **Indépendant ou Collaboratif :**

Les attaquants peuvent agir indépendamment les uns des autres ou bien collaborer. Lorsqu'ils collaborent, les attaquants s'échangent des messages et coopèrent afin de rendre l'attaque plus efficace. Par exemple, des véhicules attaquants collaboratifs annoncent un embouteillage fictif pour convaincre les véhicules honnêtes. Ces derniers vont alors changer de chemin, libérant ainsi la voie pour les attaquants.

✓ **Local ou Étendu :**

Un attaquant peut avoir une portée d'action limitée, même s'il contrôle plusieurs entités (OBU ou RSU). On dit qu'il est local parce que la portée limitée des OBU et des RSU, rend l'attaque limitée. Un attaquant étendu contrôle plusieurs entités qui sont éparpillées sur le réseau, ce qui lui confère une portée étendue.

2.1.2. Menaces au niveau applicatif

L'attaque délibérée ou non d'un VANET repose sur un but précis. Nous dressons une liste des attaques évidentes ou faisables et qui constituent un risque non négligeable en cas de réalisation[10] [11]. En raison de l'impossibilité d'envisager toutes les attaques possibles dans les réseaux véhiculaires, nous nous limitons aux exemples suivants :

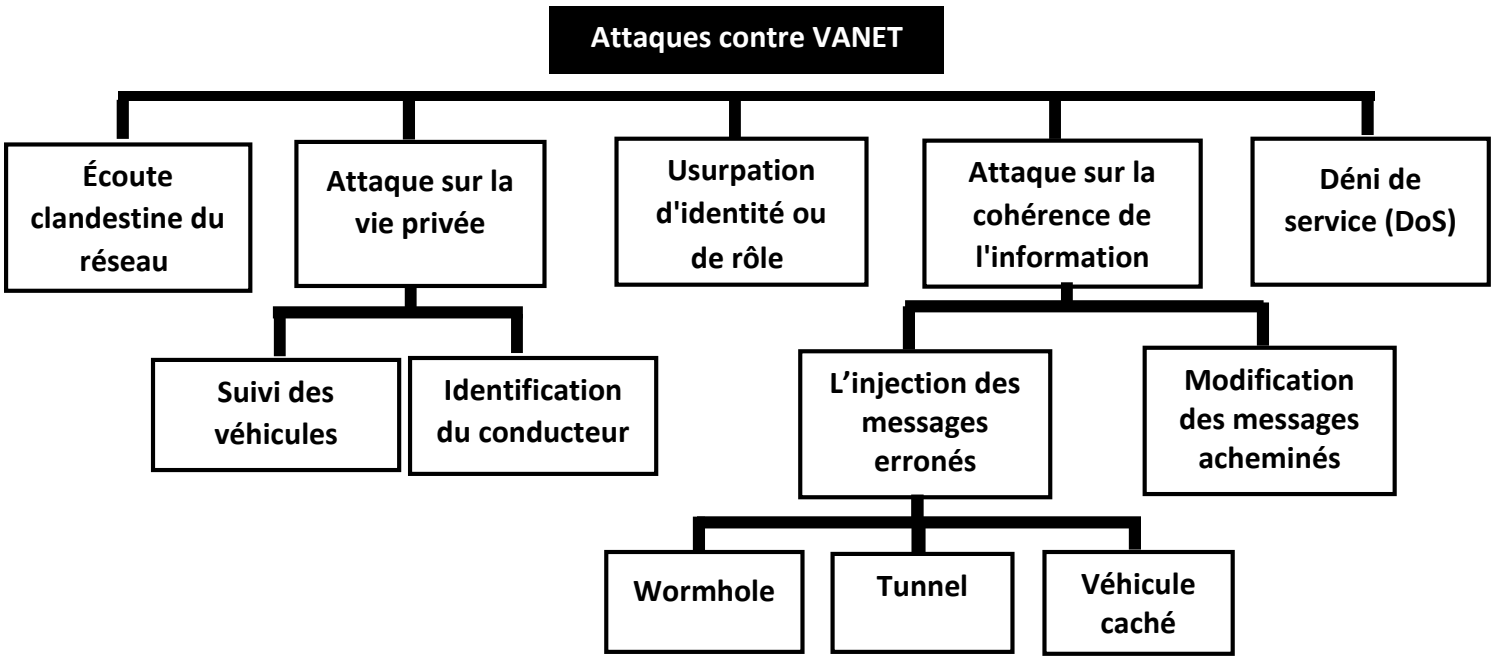


Figure 2.1 : Classification des attaques contre les réseaux VANETs

✓ **Écoute clandestine du réseau :**

▪ Dans cette attaque, l'entité malveillante collecte les données transmises dans le réseau afin d'en extraire une information dont elle pourrait tirer profit.

▪ L'attaquant peut être *Interne* ou *Externe, Mal intentionné, Passif* et *Indépendant*.

▪ Un exemple d'attaque est un attaquant qui espionne une transaction commerciale, typiquement un paiement électronique à un péage, en vue d'en extraire les informations bancaires.

✓ **Attaque sur la vie privée :**

▪ Dans cette attaque, l'entité malveillante essaie d'obtenir l'identité ou des informations personnelles d'un utilisateur du réseau. Cela peut également se traduire par tracer l'activité et les déplacements de cet utilisateur. Pour identifier et tracer une victime, l'attaquant peut utiliser toute chaîne de caractères identificatrice dont la récurrence est constatée dans les échanges de la victime. Cette chaîne de caractères peut être une adresse IP, une adresse MAC, des informations d'identification d'un certificat, etc.

▪ D'après les modèles d'attaquants, l'attaquant peut être *Interne* ou *Externe, Mal intentionné*, *Passif* et *Indépendant.*

▪ L'utilité de cette attaque est diverse et dépend de l'entité collectant les informations :

- Une entreprise de location de voitures qui veut suivre ses propres véhicules de manière illégitime.
- Un attaquant pourra par exemple traquer les déplacements d'un convoyeur de fonds afin de lui tendre un piège.

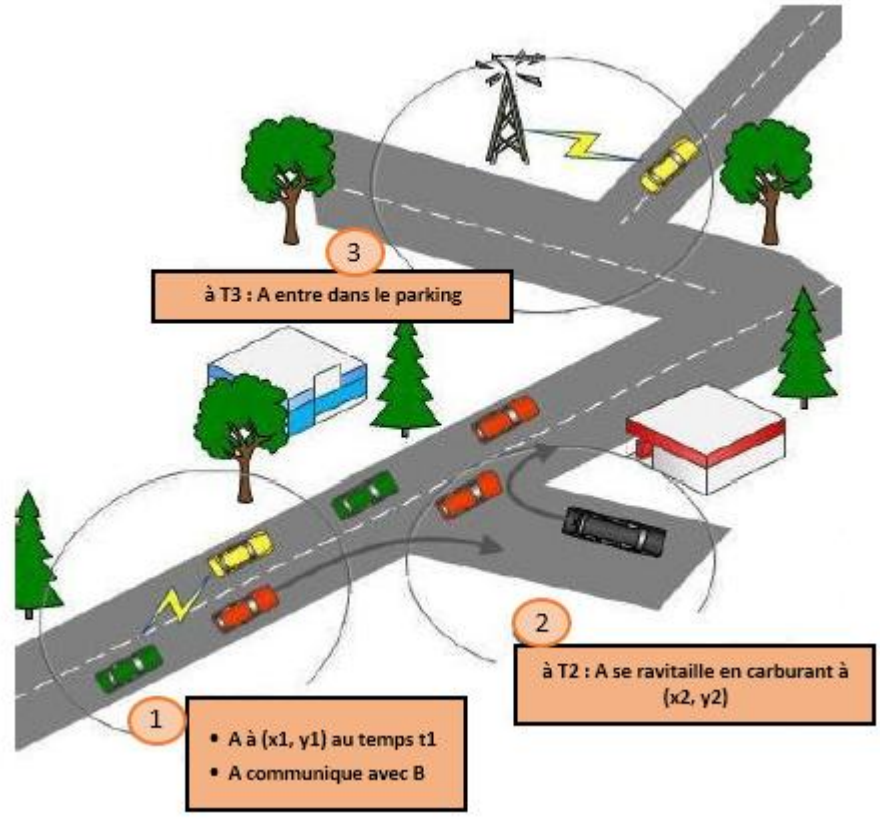


Figure 2.2 : Attaque de révélation de position géographique d'un véhicule [10]

✓ **Usurpation d'identité ou de rôle :**

▪ Dans cette attaque, l'entité malveillante utilise une fausse identité ou de fausses lettres de créance pour se faire passer pour une entité légitime ou pour jouir des privilèges de cette dernière.

▪ L'attaquant peut être Interne ou Externe, Malicieux ou Rationnel, Mal intentionné, Actif et Indépendant.

▪ Par exemple, un conducteur avide peut se comporter comme un véhicule d'urgence pour accélérer sa propre vitesse.

✓ **Attaque sur la cohérence de l'information :**

▪ Dans cette attaque, l'entité malveillante porte atteinte à la cohérence des informations acheminées dans le réseau en les modifiant ou en injectant des informations erronées. L'intention de l'attaquant est d'altérer la perception qu'ont ses victimes des conditions de circulation (position, vitesse, direction).

▪ Dans cette attaque, l'attaquant est Interne ou Externe, Intentionnelle, Actif et Indépendant ou collaboratif.

▪ Par exemple, un message contenant des informations erronées peut être émis pour :

- Causer des accidents.
- Rediriger le trafic routier de manière permettant la libération de la route utilisée.
- Dévier le trafic afin de créer des embouteillages avant d'exploser une bombe par les terroristes.

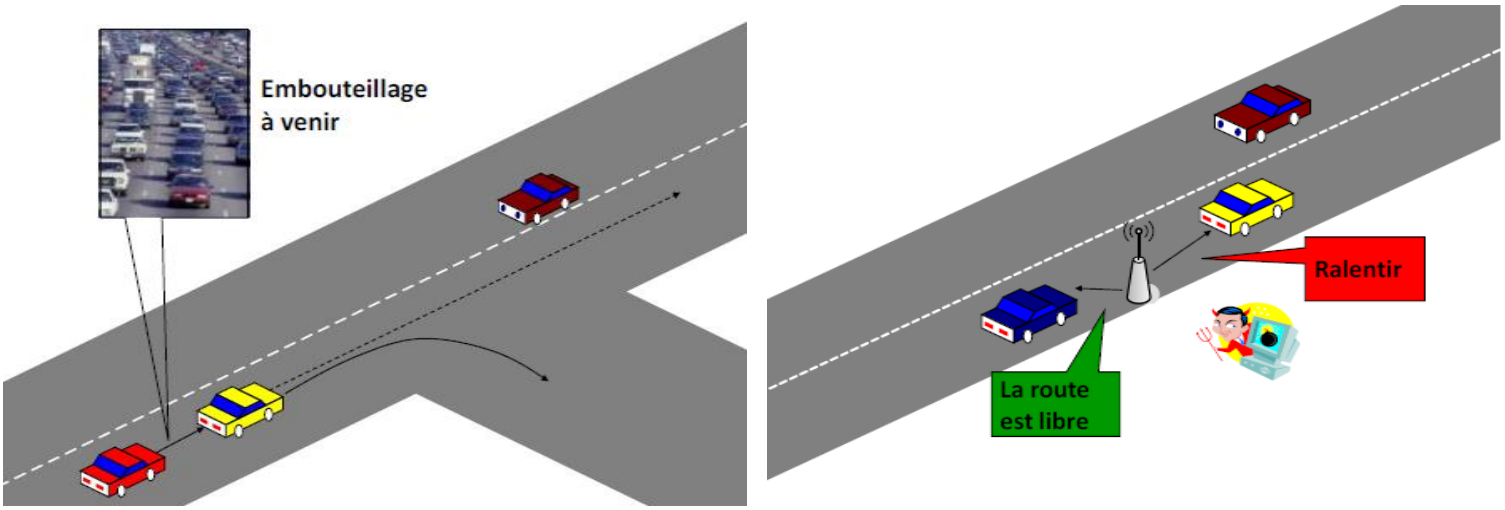


Figure 2.3 : Attaques par l'envoi de messages falsifiés[10]

▪ Parmi les variantes de cette attaque générique, on peut citer :

1. **Véhicule caché** : C'est un exemple de falsification des informations de positionnement. Dans le protocole de distribution des messages d'alerte, si un véhicule diffusant l'alerte détecte un voisin mieux positionné que lui pour diffuser, alors il arrête d'émettre. Ce protocole permet de réduire la congestion du canal radio.

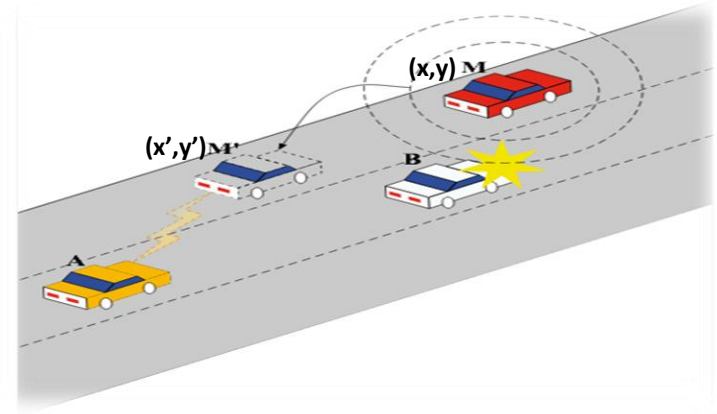


Figure 2.4 : Véhicule caché

L'attaquant M(x,y) fait croire qu'il est en meilleure position M'(x',y') afin d'être le seul à émettre l'alerte. Mais il ne va pas diffuser l'information d'alerte, rendant le véhicule en danger caché des autres véhicules.

2. **Tunnel** : Comme le signal GPS connaît des pertes (dans un tunnel ou dans certaines zones perturbatrices), un attaquant peut exploiter cette perte de positionnement temporaire. En effet, il peut envoyer de fausses données dès la sortie du « tunnel » avant que le véhicule victime ne reçoive une mise à jour de position authentique.

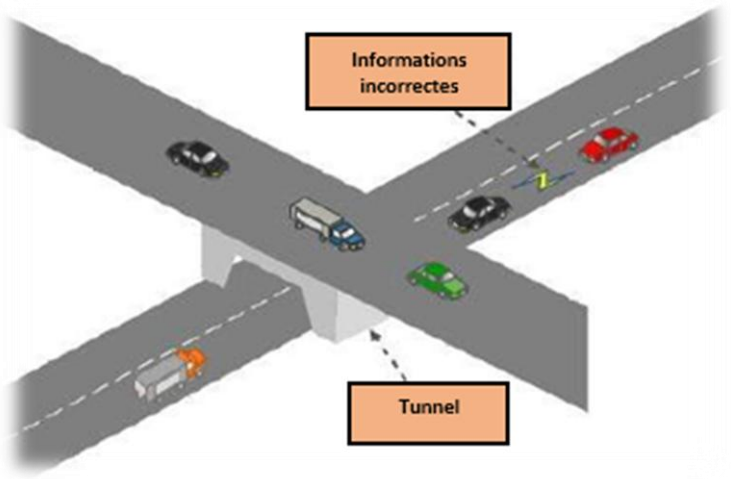


Figure 2.5 : Attaque de Tunnel[10]

3. **Wormhole** : Un attaquant qui contrôle plusieurs entités éloignées, peut établir un tunnel entre ces entités et peut ainsi injecter des données d'un endroit à l'autre. Il diffuse ainsi des informations erronées (mais signées) à divers endroits. C'est un exemple d'attaque Étendue.

✓ **Déni de service (DoS) :**

Dans ce type d'attaque, l'entité malveillante empêche l'accès normal aux services du réseau. Ce type d'attaque peut être généré en brouillant le canal radio, en surchargeant ou en épuisant les ressources du réseau par des requêtes abondantes, en exploitant la vulnérabilité des protocoles, ou en ayant une attitude non coopérative (refus de relayer des paquets par exemple).

L'attaquant peut être *Interne* ou *Externe, Mal intentionné, Actif* et *Indépendant*.

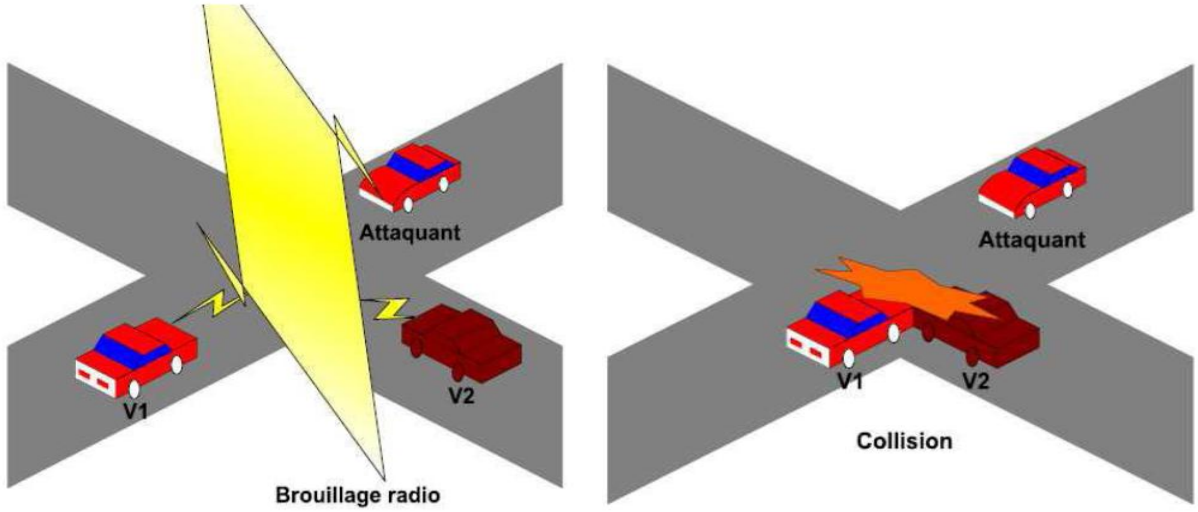


Figure 2.6 : Attaque déni de service

Ainsi, On peut classifier les menaces de sécurité dans les réseaux VANET selon le but des attaquants en quatre catégories :

1. ***Interférence du trafic routier*** : L'application influence le comportement routier et peut être détournée pour entraîner des situations potentiellement accidentogènes.

2. ***Subversion de la responsabilité*** : Lors d'un accident, un attaquant peut vouloir porter de fausses accusations contre un usager de la route ou permettre aux autres attaquants de rester non identifiés.

3. ***Dépréciation de la vie privée*** : À partir des informations de vitesse et de temps contenues dans un message d'alerte, un attaquant peut générer un profil de déplacement. Il pourra alors revendre l'information ou l'utiliser à des fins malveillantes.

4. ***Contrôle à distance de véhicule*** : À long terme, on peut supposer une automatisation de la conduite. C'est pourquoi en exploitant les vulnérabilités existantes, il sera possible de prendre le contrôle à distance d'un véhicule.

2.1.3. Attaques contre les protocoles de routage

Comme le routage est un service fondamental dans tout type de réseau, il constitue une cible idéale pour les attaques dans les VANETs. En effet, si les règles du protocole de routage utilisées n'étaient pas bien conçues, l'entité malveillante peut les manipuler afin d'interrompre l'acheminement d'un message lié à la sécurité, par conséquent, ces réseaux VANETs auront un impact négatif sur la sécurité routière en présence des attaquants.

Dans cette section nous allons présenter les attaques spécifiques aux protocoles de routage OLSR et AODV déjà présentés dans le premier chapitre.

2.1.4. Vulnérabilités et types d'attaques spécifiques au protocole OLSR

Dans un réseau utilisant le protocole de routage OLSR, chaque nœud doit correctement générer et renvoyer les messages HELLO et TC selon les spécifications du protocole. Une faille dans ce processus aurait un effet sur le bon fonctionnement du protocole de routage et ainsi sur le réseau. Or, le protocole OLSR ne fournit aucune spécification de sécurité à prendre en compte ce qui rend ce protocole vulnérable à plusieurs types d'attaques. Dans cette section, on fournira les différentes vulnérabilités du protocole OLSR[12] [13] qui sont classifiées selon deux catégories : Génération incorrecte du trafic et Relayage incorrect du trafic (Voir le résumé dans Annexe1).

2.1.4.1. Génération incorrecte du trafic :

On peut distinguer deux façons de génération du trafic de contrôle incorrect :

1. ***Mystification d'identité (Identity spoofing)*** : Un nœud malveillant peut générer un trafic de contrôle prétendant qu'il est un autre nœud.

2. ***Mystification de lien (Link spoofing)*** : Un nœud malveillant peut signaler une relation de voisinage avec des nœuds inexistants ou qui ne font pas partie de ses voisins. Il peut aussi signaler un ensemble incomplet de voisins.

i. Génération incorrecte des HELLOs

- **Identity spoofing**

Un nœud malveillant peut usurper l'identité d'un autre nœud en utilisant les messages HELLO (*Identity spoofing*). Dans la Figure 2.7, le nœud malveillant *M* peut usurper l'identité du nœud *C* en envoyant des messages HELLO prétendant qu'il est le nœud *C*. Dans ce cas, les nœuds *A* et *B* vont annoncer à ses voisins que le nœud *C* est accessible à travers le nœud *M*. Ceci peut causer des conflits de routes vers le nœud *C* dans tout le réseau.

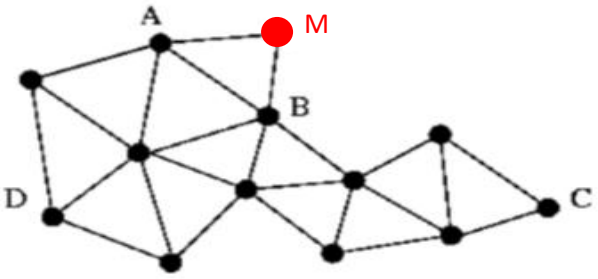


Figure 2.7 : Usurpation d'identité du nœud *C* par *M* (HELLO)

- **Link spoofing**

Lorsqu'un nœud malicieux oblige ses voisins à le choisir comme relais multipoint. On appelle cette vulnérabilité *attaque sur la sélection des MPR par mystification de lien.* Ceci peut survenir lorsqu'un nœud malveillant signale dans ses messages HELLO une fausse relation de voisinage avec des nœuds inexistants ou des nœuds éloignés qui ne font pas partie de ses voisins (*Link spoofing*). Comme il est le seul à avoir un lien avec ces nœuds, la spécification du protocole OLSR, lui permet d'être choisi comme relais multipoint par ses voisins. Ceci entraine une fausse sélection des MPR par tous les voisins de ce nœud malicieux. Dans la Figure 2.8, le nœud malicieux *M* annonce dans ses messages HELLO un lien avec un nœud inexistant *i*. Dans ce contexte, le nœud *a* le choisit comme MPR. Par la suite, tous les messages entre les nœuds *a* et c passent par *M* et ce dernier peut les modifier ou les rejeter. Or, dans l'absence de ce lien inexistant entre *M* et *i,* le nœud *a* peut choisir soit *b* soit *M* comme MPR. Dans le même contexte, un nœud malveillant *M* peut créer des liens virtuels avec tous les nœuds à 2 sauts d'un nœud *a*. Ainsi, le nœud *M* est l'unique MPR de *a*. Le nœud malicieux peut par la suite l'ignorer dans ses messages TC et le nœud *a* se retrouve alors couper du réseau. Ce type d'attaque s'appelle *Node Isolation Attack*[14] .

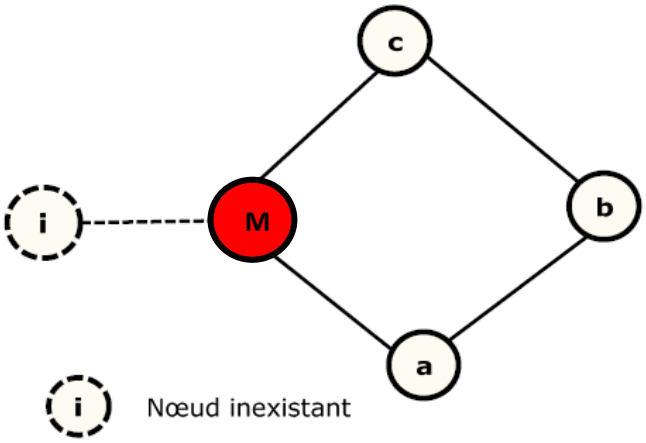


Figure 2.8 : Attaque sur la sélection des MPR

Un nœud malveillant peut aussi déclarer dans ses messages HELLO un ensemble incomplet de ses voisins. Les nœuds ignorés peuvent être coupés du reste du réseau si le nœud malveillant est leur seul lien.

ii. *Génération incorrecte des TCs*

Un nœud malicieux peut envoyer des messages TC et usurper l'identité d'un autre nœud dans le réseau. Dans la Figure 2.9, le nœud *M* génère des messages TC ayant pour origine le nœud *v* et déclarant les nœuds *e* et *g* comme voisins. Lorsqu'un nœud dans le réseau reçoit les messages TC (par exemple le nœud *a*), il conclut que les nœud *e*, *g* et *v* sont voisins. Ceci entraine des conflits des routes dans le réseau et une mauvaise vision de la topologie par les nœuds.
Lorsque le nœud *a* reçoit un message TC de la part du nœud *M* et *v* (pour lui c'est de la même origine), il va rejeter le message avec le plus petit numéro de séquence ANSN (*Advertised Neighbor Sequence Number).* Le nœud malveillant doit alors générer des messages TC avec des ANSN plus grands que ceux envoyés par le nœud *v.* Ce mécanisme peut être considéré en lui-même comme

un type d'attaque à part entière. En effet, il suffit pour un nœud d'écouter les messages TC des autres nœuds légitimes et envoyer par la suite des messages TC avec usurpation d'identité des nœuds avec un numéro de séquence plus grand que celui des messages des victimes.

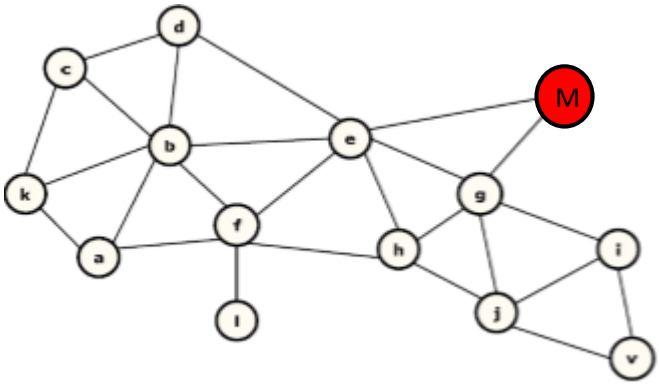


Figure 2.9 : Usurpation d'identité du nœud v par M (TC)

Finalement, un nœud malveillant peut être choisi comme MPR légitime par d'autres nœuds.
Par la suite, il peut refuser de générer les messages TC ou déclarer dans ses TC un ensemble incomplet de voisins qui l'ont choisi comme MPR. Ceci peut entrainer la déconnexion du réseau de l'ensemble des nœuds non déclarés dans les messages TC du MPR malveillant.

iii. Génération incorrecte des MID/HNA

Une autre attaque concerne l'envoi de messages MID/HNA déclarant des interfaces inexistantes, ce qui a des effets délétères envers les nœuds essayant de joindre ces interfaces.

2.1.4.2. Relayage incorrect du trafic :

Les opérations de routage et de communication dans les réseaux Ad-Hoc se basent essentiellement sur le relayage correct du trafic de routage et de données. Un relayage incorrect a des conséquences sur le bon fonctionnement du réseau. On peut distinguer différents types d'attaques dans cette catégorie.

i. Relayage incorrect du trafic de contrôle

Un nœud malveillant peut être choisi comme MPR légitime par d'autres nœuds mais il refuse de relayer les messages TC des autres MPR. Dans le cas où il n'existe pas de route qui ne passe pas par le nœud malicieux, le refus de ce dernier de relayer les messages TC peut avoir comme conséquence une perte de connectivité de certains nœuds dans le réseau.

ii. Attaque par retransmission des messages de contrôle

Un nœud malicieux peut renvoyer à d'autres nœuds des messages de contrôle (TC ou HELLO) déjà envoyés dans le passé par d'autres nœuds qu'il a pu écouter à travers le réseau.
Il faut que les messages renvoyés par l'attaquant aient des numéros de séquence plus élevés sinon ces messages seront rejetés par les nœuds qui ont reçu une copie originale de ces messages. Ce

type d'attaque cause un échange de fausses informations et un conflit dans le calcul de la topologie qui peut entrainer des problèmes de routage.

iii. Attaque wormhole

Ce type d'attaque redirige le trafic entre deux zones géographiquement éloignées pour ainsi avoir une bonne position géographique pour contrôler le trafic qui passe par lui.

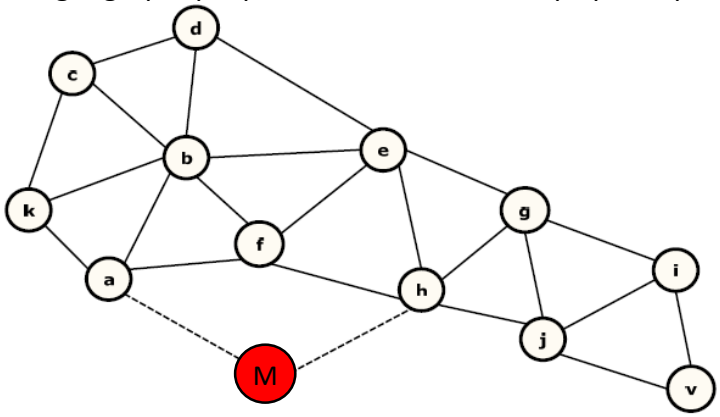


Figure 2.10 : Attaque *wormhole* créée par le nœud *M*

Dans la figure 2.10, le nœud malicieux *M* crée un lien virtuel entre les nœuds *a* et *h* sans être visible par les deux nœuds. Le but est de leur faire croire qu'ils sont des nœuds voisins. En effet, le nœud *M* renvoie les messages HELLO du nœud *h* vers *a* et inversement. Ainsi, chacun de ces nœuds va déclarer par la suite qu'il a un lien symétrique entre eux. La route entre *a* et *h* devient alors une route préférée par les autres nœuds car c'est le plus court chemin. Ce chemin est totalement contrôlé par le nœud malveillant *M* ce qui présente un danger pour l'intégrité et la confidentialité des messages. Deux nœuds malicieux *M1* et *M2* peuvent aussi collaborer pour créer une attaque *wormhole* entre deux zones très éloignées (voir la figure 2.11).

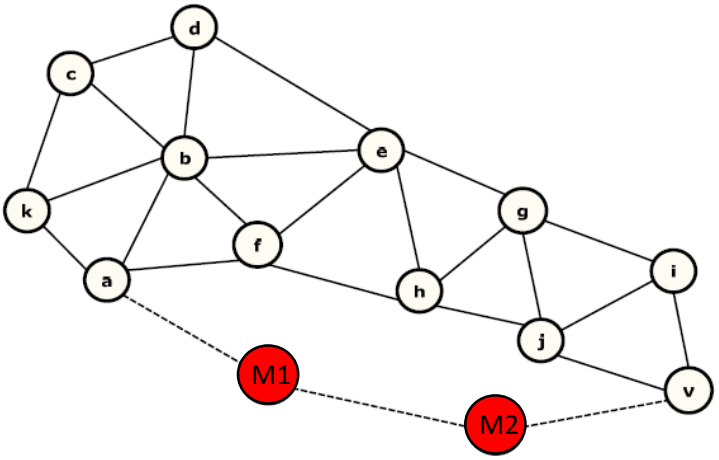


Figure 2.11 : Collaboration pour créer un wormhole

iv. Attaque trou-noir

Un nœud malicieux qui a été choisi par ses voisins comme MPR peut rejeter tous les paquets de données reçus de ses voisins (*Blackhole Attack*). Ce type d'attaque entraine une perte de connectivité et la dégradation de la communication.

2.1.5. Vulnérabilités et types d'attaques spécifiques au protocole AODV

Dans cette section, nous nous intéressons de plus près aux attaques sur le protocole de routage ad hoc AODV. Nous présentons comment les nœuds malhonnêtes opèrent pour aboutir à leur objectif. Ces comportements malveillants se basent essentiellement sur une ou plusieurs actions élémentaires.

Dans [15] , Peng Ning et Kun Sun ont commencé par identifier les objectifs qui peuvent être visés par des attaquants à savoir :

Perturbation de route : Signifie soit briser une route existante ou bien empêcher un nouvel itinéraire d'être établie.

Invasion de route : Signifie qu'un attaquant interne s'insère dans une route entre deux extrémités d'un canal de communication.

Isolation de nœud : L'isolation d'un nœud permet de l'empêcher de communiquer avec les autres nœuds du réseau.

N.B : Perturbation de route vise une route avec deux extrémités données, tandis que l'isolation du nœud vise tous les itinéraires possibles.

Consommation des ressources **:** Se réfère à la consommation de la bande passante de communication dans l'espace de stockage réseau ou au niveau des nœuds individuels. Par exemple, un attaquant interne peut consommer la bande passante du réseau en formant une boucle dans le réseau.

Ning et Sun ont classifié les actions malhonnêtes sur AODV en deux catégories :

i. atomiques, résultant de la manipulation d'un seul message de routage
ii. composées, définies comme étant une collection d'actions atomiques.

Les manipulations effectuées sur un message de routage peuvent être :

- ✓ Effacer un paquet ;
- ✓ Modifier un ou plusieurs champs du paquet avant de le retransmettre ;
- ✓ Fabriquer une réponse à la réception d'une demande de route RREQ ;
- ✓ Fabriquer activement des paquets de routage sans même avoir reçu de messages de routage.

Le tableau 2.1 récapitule les actions malveillantes sur les messages de routage en fonction de l'objectif :

Action élémentaire		Perturbation de route	Invasion de route	Isolation nœud	Consommation des ressources
RREQ	Effacer	Oui *	Non	Non	Non
	Modifier	Oui	Oui	Partiel	Non
	Fabriquer A.	Oui	Oui	Partiel	Non
RREP	Effacer	Oui *	Non	Non	Non
	Modifier	Oui	Oui	Non	Non
	Fabriquer R.	Oui	Oui	Non	Non
	Fabriquer A.	Oui	Oui	Non	Oui
RERR	Effacer	Oui *	Non	Non	Non
	Modifier	Oui	Non	Non	Oui
	Fabriquer A.	Oui	Non	Non	Oui

*dans certains cas

Tableau 2.1 : Classification des actions malveillantes sur les messages de routage en fonction de l'objectif

2.1.5.1.Classifications des attaques spécifiques au protocole AODV

2.1.5.1.1. Attaques élémentaires :

i. Attaques élémentaires portant sur les demandes de route

Nous traitons dans ce qui suit l'effet de chaque action élémentaire sur les paquets de demande de route.

Suppression d'une demande de route :

Un nœud malhonnête pourrait simplement **effacer** la demande de route reçue. En appliquant ce genre de comportement à tout message RREQ reçu, l'attaquant ne participe pas au routage : c'est comme s'il ne faisait pas partie du réseau. Une autre variante serait d'effacer sélectivement des messages RREQ. Ce comportement peut être comparé à celui d'un nœud égoïste.

Modification d'une demande de route :

À la réception d'une demande de route, le nœud malhonnête **modifie** un ou plusieurs champs qu'il n'est pas supposé modifier avant de retransmettre le message. La modification peut aussi porter sur un champ qu'il a le droit de modifier, mais il ne respecte pas la spécification pour le faire.

Plusieurs champs impliquent des traitements différents lorsqu'ils sont modifiés. Par exemple, le champs identifiant de la RREQ associé à l'adresse de la source permet d'identifier de manière unique une demande de route et indique la fraicheur de la demande de route. Puisqu'un nœud n'accepte que le première copie de RREQ, en augmentant cet identifiant, le nœud malhonnête peut garantir l'acceptation et le traitement de la RREQ modifiée par les autres nœuds.

Champs de la RREQ	Est modifiable ?	Modification	Effet
Type	Non	Changer le type	Rejet du message (non conforme au type déclaré)
Identifiant	Non	Augmenter	Rendre la RREQ acceptable (plus fraiche)
		Diminuer	Rendre la RREQ non- acceptable
Nombre de sauts	Oui (autorisé +1)	Diminuer	Mettre à jour le chemin inverse vers la source
		Augmentation (>1)	Ralentir la découverte de route
Adresse destination	Non	Changer	Fausser la découverte de route (trafic inutile)
Adresse source	Non	Changer	
Numéro de séquence source	Non	Augmenter	Mettre à jour le chemin inverse vers la source
		Diminuer	Ne pas mettre à jour le chemin vers la source
Drapeau G	Non	inverser (0|1)	Envoyer ou supprimer la RREP gratuite
Drapeau D	Non	inverser (0|1)	Seulement la destination a le droit de répondre si non tout nœud intermédiaire

Tableau 2.2 : Modifications possibles sur les champs des RREQ

Fabrication d'une demande de route :

Les attaques décrites au-dessus sont déclenchées par la réception d'une demande de route. En revanche, les attaques par *fabrication* peuvent être effectuées sans avoir reçu de messages RREQ. Le nœud malhonnête a besoin de collecter certaines informations, en écoutant le trafic par exemple, avant d'injecter le message fabriqué. Il est à noter qu'une exécution répétitive de ce type d'attaque peut provoquer l'inondation du réseau par des messages de routage inutiles. Le nœud malhonnête peut orienter le trafic vers une seule destination ou faire croire que le trafic part d'une seule source ou les choisir (source/destination) au hasard.

Rushing d'une demande de route :

Dans d'autres cas, le nœud malhonnête peut utiliser la technique du *rushing* qui consiste à diminuer le temps de traitement des messages RREQ et les retransmettre plus rapidement de telle sorte qu'ils atteignent plus rapidement la destination. Ceci garantira pour le nœud malhonnête une place sur le chemin.

ii. Attaques élémentaires portant sur les réponses de route

Suppression d'une réponse de route :

Ce type d'attaque n'a un sens que si le nœud malhonnête a été choisi sur la route reliant la source à la destination. Dans ce cas, la *suppression* de la réponse de route empêche la formation du chemin vers la destination et entraine des messages de contrôle supplémentaires suite à l'initialisation d'un nouveau processus de création de route, ce qui dégrade la qualité de service.

Modification d'une réponse de route :

Comme pour les demandes de route, un nœud malhonnête peut *modifier* un ou plusieurs champs qu'il n'est pas supposé modifier avant de retransmettre le message. La modification peut aussi porter sur un champ qu'il a le droit de modifier, mais il ne respecte pas la spécification pour le faire. Le tableau 2.3 présente les champs où le nœud malhonnête peut intervenir.

Champs de la RREP	Est modifiable ?	Modification	Effet
Type	Non	Changer le type	Rejet du message (non conforme au type déclaré)
Nombre de sauts	Oui (autorisé +1)	Diminuer	Fausser le nombre de sauts vers la destination
		Augmentation (>1)	
Adresse destination	Non	Changer	Rediriger le paquet (trafic inutile)
Adresse source	Non	Changer	
Numéro de séquence destination	Non	Augmenter	Forcer à garder le chemin avec le plus grand numéro de séquence en cas de RREP multiples
		Diminuer	Diminuer les chances en cas de RREP multiples
Drapeau A	Non	inverser (0|1)	Nécessité d'un accusé si égal à 1

Tableau 2.3 : Modifications possibles sur les champs des RREP

Fabrication d'une réponse de route :

– Fausse réponse : à la réception d'une demande de route, le nœud malhonnête fabrique une réponse de route même s'il n'a pas de chemin valide vers la destination. Dans un autre cas de figure, le nœud malhonnête répond avec une réponse de route même s'il n'est pas supposé le faire lorsque le drapeau D est à 1 (indiquant que seule la destination doit répondre).

– Réponse active : des réponses de route sont fabriquées et injectées dans le réseau même sans avoir reçu une demande de route au préalable. Dans ce cas le nœud malveillant peut jouer sur tous les champs précédemment présentés pour produire l'effet désiré. Une variante de cette attaque vise à déborder la table de routage d'une cible en proposant des routes (via RREP) vers des nœuds (nouveaux ou inexistants).

– En écoutant la transmission d'une réponse de route qui ne lui est pas destinée, un nœud malhonnête peut fabriquer et injecter un paquet RREP proposant un chemin plus court et plus frais provoquant la mise à jour du chemin vers la destination qui passe dorénavant par le nœud malveillant.

iii. Attaques élémentaires portant sur les erreurs de route

Suppression d'une erreur de route :

Comme c'est le cas pour les RREQ et RREP, en *effaçant* une RERR, un nœud malhonnête peut retarder la détection des liens défaillants. Cependant, l'impact de cette attaque est restreint du

fait que les nœuds en amont découvrent le problème et demandent l'établissement de nouvelles routes.

Modification d'une erreur de route :

Un nœud malhonnête peut *modifier* des erreurs de route avant de les retransmettre. Le tableau 2.4 indique les champs où le nœud malveillant peut intervenir. Ainsi, il peut supprimer des destinations non-joignables pour faire croire qu'elles le sont encore et ajouter des destinations qui sont joignables et actives pour faire croire qu'elles ne le sont plus et les désactiver.

Champs de la RERR	Est modifiable ?	Modification	Effet
Type	Non	Changer le type	Rejet du message (non conforme au type déclaré)
Nombre de destinations	Non	Incrémenter	Ajouter une destination non joignable
		Diminuer	Supprimer une destination non-joignable
Adresse destination non joignable	Non	Changer	Faire croire qu'une autre destination est non joignable
Numéro de séquence destination non joignable	Non	Augmenter	Mise à jour des entrée des voisins correspondant à cette destination
		Diminuer	Rendre l'entrée correspondante non valide
Ajouter autant de destinations à déclarer comme non-joignables et leur numéro de séquence			

Tableau 2.4 : Modifications possibles sur les champs des RERR

Fabrication d'une erreur de route :

Un nœud malhonnête peut **fabriquer** un message d'erreur de route et déclarer autant de routes non-joignables causant l'invalidation des entrées correspondantes dans la table de routage des nœuds recevant le message de contrôle.

2.1.5.1.2. Attaques composées :

Un attaquant peut combiner des attaques élémentaires pour effectuer des attaques composées potentiellement pour atteindre des objectifs plus évolués.
Nous présentons dans ce qui suit des exemples d'attaques composées exploitant les vulnérabilités du protocole de routage ad hoc AODV.

i. Répétition régulière d'attaques élémentaires

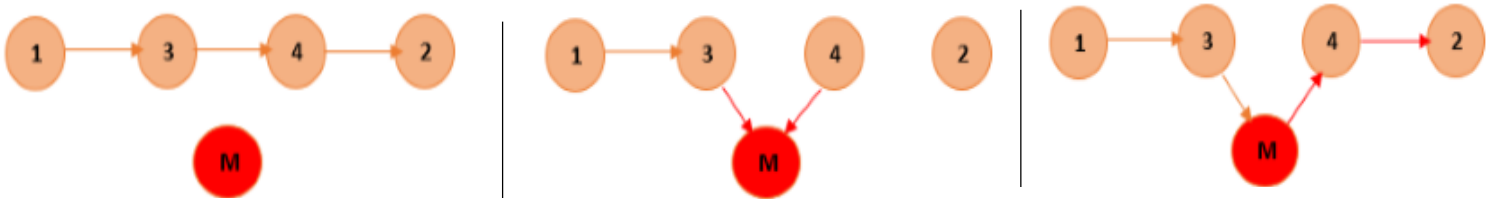
Cette attaque se base sur une répétition régulière d'une attaque élémentaire pour avoir un impact permanent. Par exemple, l'envoi continu de messages de demande de route RREQ fabriqués à destination d'une cible est efficace pour empêcher celle-ci de recevoir les messages

des autres nœuds. De même, l'envoi de réponses de route fabriquées est efficace pour empêcher la cible d'envoyer des messages aux autres nœuds (puisque le nœud malhonnête devient le prochain saut vers les autres nœuds). En combinant ces deux attaques composées, un nœud malhonnête peut isoler sa cible.

ii. Insertion dans une route déjà établie

C'est une attaque composée qui se base sur l'utilisation de deux demandes de route fabriquées pour s'insérer sur une route déjà établie entre une source et une destination.

Dans cette attaque, le nœud malicieux compte sur sa proximité d'au moins deux nœuds faisant partie de la route sur laquelle s'insérer. Nous expliquons le déroulement de cette attaque par l'exemple de la figure au-dessous :



a. Initialement le chemin 1-3-4-2 est établi et M peut entendre les transmissions de 3 et 4

b. Après la transmission de la première RREQ fabriquée

c. Après la transmission de la deuxième RREQ fabriquée

Figure 2.12 : Invasion de route (route déjà existante)

La figure 2.12.a montre l'état initial du réseau où 1 et 2 ont établi une connexion les reliant via les nœuds 3 et 4. Le nœud M, à proximité de 3 et 4, tente de s'introduire sur la route : il fabrique une première demande de route en initialisant les champs comme suit :

- l'adresse source est initialisée à 2,
- l'adresse destination est initialisée à 1,
- le numéro de séquence de la source 2 est fixé à une valeur supérieure à la valeur actuelle.

Il suffit enfin de fixer l'adresse IP de la source dans l'entête IP à l'adresse du nœud malhonnête (M) avant de diffuser le message.

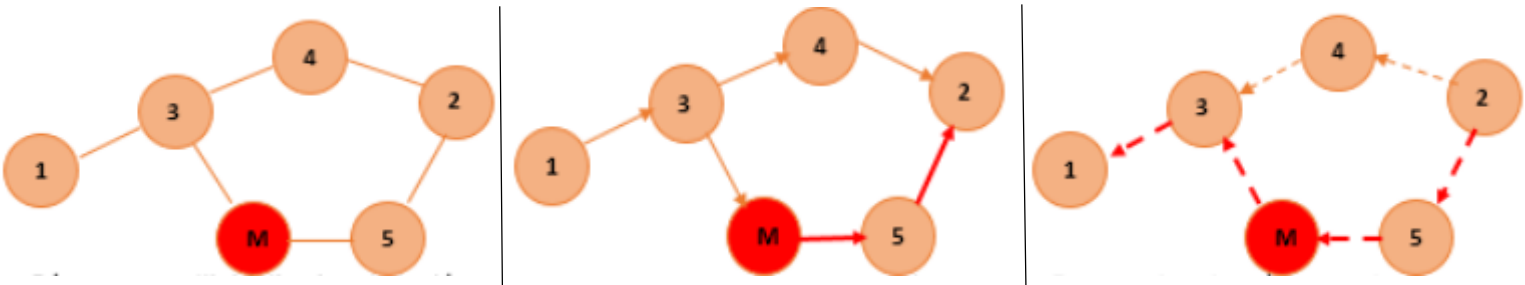
Les nœuds 3 et 4 recevant la demande de route vont choisir M comme prochain saut en direction de la source 2 dans leurs tables de routage (voir figure 2.12.b). Ensuite, le nœud malhonnête fabrique une seconde demande de route pour rétablir le lien vers 2. Cette RREQ fabriquée contient :

- l'adresse source = M,
- l'adresse destination = 2,
- le numéro de séquence de la destination est initialisé à une valeur plus grande que la valeur actuelle. En diffusant cette demande de route, il obtient une route vers la destination (figure 2.12.c).

iii. Insertion dans une route non encore établie

Une autre attaque composée existe pour s'insérer sur un chemin. Celle-ci s'opère lors du processus de découverte de route (route non encore établie).

Elle nécessite l'utilisation d'un paquet de demande de route et un paquet de réponse de route. Nous expliquons cette attaque à travers l'exemple de la figure 2.13. Il est à noter que pour cette attaque, le nœud malhonnête n'est pas obligé d'être à proximité des nœuds qui seront choisis dans le chemin.



a. Réseau avant l'initialisation de la découverte du chemin où les traits entre les nœuds représente la visibilité de chaqu'un d'eux

b. Propagation de la RREQ initialisée par 1 (en orange) et celle initialisée par M (en rouge)

c. Propagation des réponses de route correspondant à $RREQ_1$ en orange et $RREQ_M$ en rouge

Figure 2.13 : Invasion de route (lors de l'établissement du chemin)

À la réception de la demande de route initialisée par le nœud 1, l'attaquant M crée une demande de route normale comme si lui aussi voulait établir un chemin vers la même destination demandée dans la RREQ qu'il vient de recevoir (à savoir 2).
Cette action garantie qu'il obtient un chemin vers la destination quel que soit le chemin qui sera retenu pour la demande de 1.

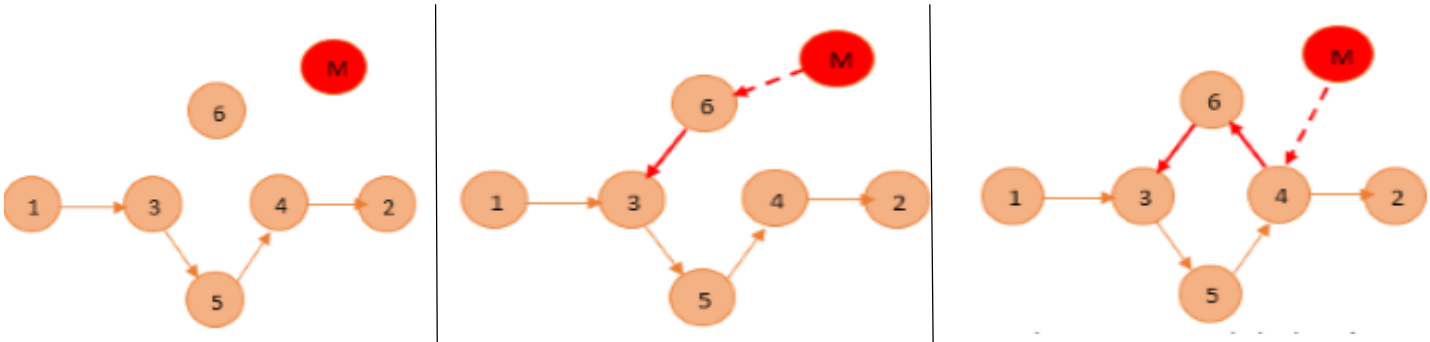
Ensuite, à la réception de la réponse de route, il fabrique une réponse de route à destination de 1 où il augmente le numéro de séquence de la destination qu'il vient de recevoir et transmet le paquet vers le prochain saut en direction de la source 1 (vers le nœud 3).
Ceci assure que le chemin vers la destination proposé par la réponse de route fabriquée est choisi puisqu'il est plus frais.
De cette manière, l'attaquant s'insère entre la source et la destination.

iv. Création d'une boucle de routage

C'est une attaque permettant la formation d'une boucle de routage dans une route déjà établie en fabriquant deux réponses de routes. Nous présentons cette attaque à travers l'exemple de la figure 2.14.a où on suppose l'existence d'un chemin entre le nœud 1 et le nœud 2 passant par 3, 5 et 4.
Le nœud malhonnête M fabrique une première réponse de route où :

- l'adresse source est initialisée à 1,
- l'adresse destination à 2
- un numéro de séquence de la destination 2 supérieur au numéro de séquence actuel.

Il fait croire que ce paquet est envoyé par le nœud 3 à destination du nœud 6 (valeurs à mettre respectivement dans les champs adresse source et destination de l'entête IP). Ceci provoque l'envoi de tout paquet reçu à destination de 2 vers le nœud 3 (voir figure 2.14.b).
Ensuite, M fabrique une deuxième réponse de route équivalente à la première sauf qu'il fait croire qu'elle est à destination du nœud 4, provenant du nœud 6. Cette action garantie que tout paquet transféré vers 2 à travers 4 est envoyé vers 6, ce qui complète la boucle.



a. Initialement le chemin entre 1 et 2 passe par 3-5-4 respectivement

b. Après transmission de la première RREP fabriquée

c. Après transmission de la deuxième RREP fabriquée, la boucle est formée

Figure 2.14 : Création d'une boucle de routage dans une route déjà existante

v. Création d'un tunnel

L'attaque du trou de ver (wormhole) peut aussi être facilement appliquée sur le protocole de routage AODV.
Elle nécessite la coopération de deux ou plusieurs nœuds malhonnêtes qui falsifient la longueur des routes. Ils proposent des routes plus courtes grâce au tunnel qu'ils construisent. Ensuite, le nœud malicieux recevant un message l'encapsule pour le transférer vers l'autre bout du tunnel où il sera décapsulé et rediffusé. De cette manière, les différents sauts formant le tunnel ne sont pas comptabilisés et ainsi le chemin obtenu parait plus court.

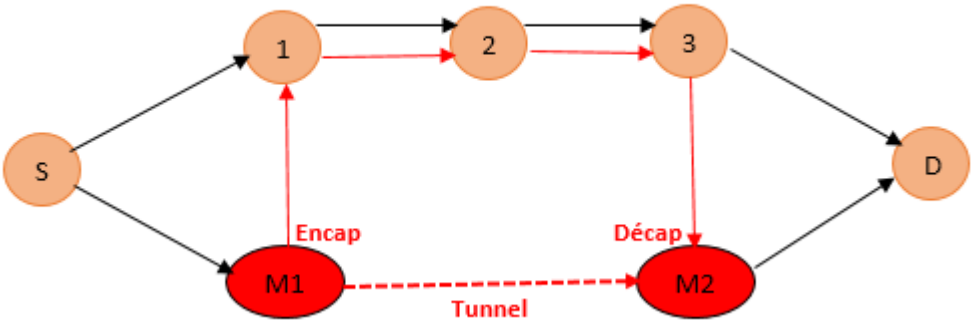


Figure 2.15 : Attaque de tunnel

La figure 2.15 montre un exemple de tunnel créé entre les nœuds malicieux M1 et M2.
D reçoit deux chemins : [S–1–2–3–D] et [S–M1–M2–D]. Le chemin le plus court est choisi (à savoir [S–M1–M2–D]). Or ce chemin est en réalité plus long [S–M1–1–2–3–M2–D].

Chapitre 3 : Mécanismes et techniques de sécurité dans les VANETs

Pour faire face aux attaques décrites dans le chapitre précédent, de nombreux mécanismes de sécurité de routage ont été proposés dans la littérature.

Dans ce chapitre, nous allons présenter un récapitulatif sur les outils et les mécanismes de base de la sécurité dans les VANETs. Ensuite, nous allons étudier quelques extensions sécurisées des protocoles OLSR et AODV.

3.1. Services de sécurité et mécanismes

Pour faire face aux attaques de sécurité, les réseaux ad hoc véhiculaires doivent déployer des services de sécurité tels que la confidentialité, l'authenticité, l'intégrité, la non-répudiation, la disponibilité et le contrôle d'accès[10] [11].
Ces services utilisent un ou plusieurs mécanismes de sécurité conçus pour détecter, prévenir ou contrer une attaque de sécurité.

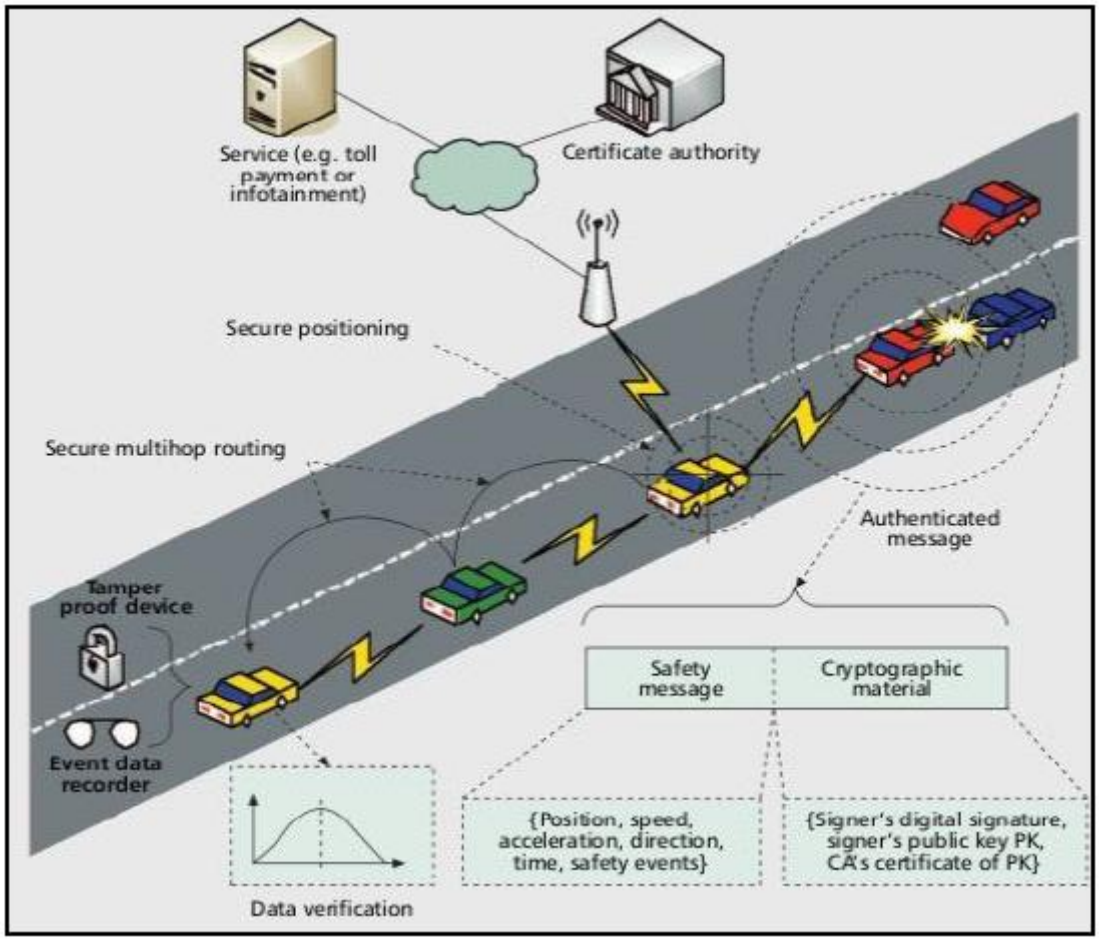


Figure 3.1 : Les mécanismes de sécurité dans les réseaux VANET[11]

3.1.1. Confidentialité

La confidentialité est le fait d'assurer que l'information n'est accessible qu'aux entités qui y sont autorisées. La confidentialité est donc la protection des données transmises contre les attaques passives (écoute clandestine).

Afin d'assurer ce service, le mécanisme proactif utilisé est le chiffrement des données. Le chiffrement peut être réalisé par la cryptographie symétrique ou asymétrique. Dans le cas de la cryptographie symétrique, les parties prenantes doivent partager une clé secrète. Le chiffrement et le déchiffrement sont alors réalisés avec cette clé. Un des problèmes majeurs de la cryptographie symétrique est l'établissement d'un canal sécurisé pour la distribution de la clé privée.

Dans le cas de la cryptographie asymétrique, chaque véhicule a une paire de clés : publique/privée. La clé privée n'est connue que du véhicule émetteur, tandis que la clé publique est partagée avec toutes les entités du réseau.

3.1.2. Authenticité

L'authenticité est le service de sécurité primordial pour un fonctionnement sécurisé des VANETs. En effet, toutes les applications déployées dans les VANETs ont besoin d'avoir confiance en l'information. Cette confiance est assurée par l'authenticité. Par exemple, dans le cas d'un message (tel un signal d'alerte), la fonction du service d'authenticité est d'assurer au destinataire que le message a bien pour origine la source dont il prétend être issu. L'authenticité est assurée par des mécanismes proactifs d'authentification. Il existe deux types d'authentification : l'authentification des messages et l'authentification des entités (identification).
Une première approche de mise en œuvre de l'authentification consiste à utiliser des clés de groupe symétriques. Cette approche ne peut malheureusement concerner qu'un très petit nombre de véhicules placés sous la même autorité. Pour des déploiements à grande échelle, cette approche présente deux inconvénients majeurs :

- Il suffit de compromettre un véhicule pour compromettre la sécurité de tout le réseau,
- Les véhicules ayant la clé peuvent se faire passer les uns pour les autres. Ce qui empêche toute confidentialité et non-répudiation.

Une deuxième approche d'authentification consiste à utiliser des clés symétriques individuelles. Cette approche souffre d'une mise à l'échelle intrinsèquement difficile puisque le nombre de clés à gérer augmente de manière linéaire avec le nombre de véhicules du réseau.

Reste donc la cryptographie à clé publique qui, dans le contexte des réseaux véhiculaires, est la seule à pouvoir permettre la réalisation de l'authentification tout en satisfaisant les exigences de mise à l'échelle, de non-répudiation et de confidentialité. Ainsi, chaque véhicule se voit assigner une paire de clés publique/privée. Chaque véhicule va signer numériquement ses messages et sera ainsi authentifié auprès des récepteurs. Néanmoins, les clés publiques doivent être délivrées et signées par une autorité de confiance. La signature numérique peut être délivrée avec ou sans certificat :

Sans certificat : La signature numérique est un concept basé sur l'application de signature numérique ou de fonction de hachage des messages. Ce concept assure l'authenticité, l'intégrité et la non-répudiation du message. La signature numérique est communément réalisée par le biais de la cryptographie asymétrique. Le message est signé avec la clé privée de l'émetteur, tandis que le récepteur vérifiera l'intégrité et l'authenticité du message en utilisant la clé publique correspondante à l'émetteur. Si l'on suppose que la clé privée n'est connue que de son possesseur, alors un véhicule ne pourra pas usurper l'identité d'un autre véhicule. Mais si l'émetteur utilise la même clé pour signer plusieurs messages, alors le récepteur peut lier ces messages à un seul

émetteur. D'un côté, cela soulève un problème de vie privée, mais de l'autre cela réduit le coût de vérification des signatures. En effet, si le délai entre les deux messages est faible, alors le véhicule récepteur peut éviter de revérifier la signature. L'avantage de la signature sans certificat est qu'elle nécessite peu de prérequis. En effet, les véhicules doivent pouvoir recevoir et stocker les paires de clés, et ils doivent avoir la puissance de calcul nécessaire pour créer et vérifier des signatures. C'est pourquoi ce concept est facilement déployable. L'inconvénient est qu'il ne protège pas des attaques de création de messages ou de déni de service.

Avec certificat : Afin d'améliorer le concept de signature numérique, les signatures peuvent être combinées avec un certificat numérique délivré par un tiers de confiance. Ainsi, le récepteur d'un message pourra s'assurer que l'émetteur a bien utilisé sa clé privée pour signer le message. L'hypothèse de base avec les certificats est que les véhicules doivent être capables de vérifier les certificats, car le certificat atteste l'authenticité de la paire de clés publique/privée. Avec la solution des infrastructures à clé publique auto-organisées (Public Key Infrastructure, PKI), un véhicule V doit signer le message avec sa clé privée et inclure le certificat de l'autorité de certification (Certification Authority, CA). Le récepteur du message doit vérifier la clé publique de l'émetteur V à partir du certificat et vérifie ensuite la signature numérique de V à partir de la clé publique (certifiée). Afin de réaliser ces opérations, le récepteur doit avoir préalablement la clé publique du CA.

Un avantage du certificat est qu'il peut empêcher, ou du moins réduire, l'attaque de réplication de nœud. Bien sûr, dans ce cas, il faut supposer qu'un véhicule ne peut utiliser qu'un seul certificat à la fois. Pour délivrer des certificats, il est nécessaire d'avoir un système de gestion et de distribution des certificats. De plus, les véhicules doivent pouvoir avoir accès à ce système de manière permanente, sporadique ou une seule fois (durant la construction du véhicule par exemple). La fréquence d'accès dépend de la conception des VANETs. Plus la fréquence d'accès est élevée, plus le système sera flexible.

Le concept de signature avec certificat permet donc de se protéger des attaques externes comme l'injection de fausses alertes par un utilisateur non authentifié. De plus, les véhicules qui ne se comportent pas bien peuvent être identifiés puis révoqués. Le certificat joue aussi un rôle de contrôle d'accès. Par exemple, seuls les messages accompagnés du certificat valide seront écoutés. La révocation s'effectue par l'ajout du certificat dans une liste des certificats révoqués (Certificate Revocation List, CRL). Les inconvénients d'un tel système sont la nécessité d'une infrastructure, et que malgré l'ajout de certificats, le réseau n'est toujours pas protégé contre les attaques d'injection par des utilisateurs authentifiés.

3.1.3. Intégrité

L'intégrité se divise en deux concepts : l'intégrité des messages et l'intégrité physique :

- Intégrité des messages : Fonction permettant d'assurer que l'information n'a pas subi d'altération.
- Intégrité physique : Fonction permettant d'assurer que le matériel (destiné aux opérations cryptographiques, à l'envoi de messages, à la collecte d'informations, etc.) n'a pas subi d'altération.

Le service d'intégrité des messages assure que les messages envoyés sont rapidement reçus, sans duplication, insertion, modification, réorganisation ou répétition. À l'instar de la

confidentialité, l'intégrité s'applique à un flux de messages, un seul message, ou à certains champs à l'intérieur d'un message. Là encore, la meilleure solution est la protection totale du flux.

Les mécanismes proactifs utilisés pour assurer l'intégrité des messages sont les chaînes de hachage (SHA1, MD5), et le Message Authentication Code (MAC). Ces mécanismes s'appuient sur des fonctions mathématiques à sens unique. Ces fonctions dépendent d'une clé secrète, et produisent un condensé du message original. Ces fonctions mathématiques sont telles qu'il est difficile de retrouver un message à partir de son condensé ou de produire deux messages ayant le même condensé. De plus, la moindre modification du message original entraine un changement dans le condensé.

Pour assurer l'intégrité physique, les véhicules doivent être équipés d'un équipement robuste dit ***Tamper Proof Device*** **(TPD**). Le TPD est un équipement résistant au sabotage (ou manipulation). Un TPD peut se décliner de plusieurs façons : accès difficile aux composants ou auto-destruction. Son premier objectif est de sécuriser les communications internes aux véhicules en empêchant la récupération de données de capteurs par exemple. Son deuxième objectif est de participer à la sécurisation des communications externes. En effet, cet équipement peut abriter les paires de clés et les certificats, ainsi qu'une boite noire (pour permettre la reconstitution de scénarios d'accidents par exemple). Mais cet équipement n'est pas, le seul, capable de sécuriser les communications externes. On peut donc envisager une combinaison de messages signés avec certificat et un TPD. Néanmoins, la manipulation des capteurs n'est toujours pas protégée, car un attaquant peut mettre une lampe chauffante à proximité d'un capteur de température par exemple.

3.1.4. Non-répudiation

En raison de l'impact que peuvent avoir les applications de sécurité routière sur la sécurité des biens et des personnes, il est indispensable que toute entité générant ou modifiant des messages d'alerte soit toujours identifiable avec certitude. En d'autres termes, toute entité, après avoir émis un message, ne doit pas pouvoir ensuite nier cette action. Assurer la non-répudiation pour les applications de sécurité routière va donc éliminer toute possibilité, pour une entité malveillante d'injecter des informations erronées, et ce sans être confondue.

S'agissant de la mise en œuvre de la non-répudiation, la signature numérique, qui est majoritairement utilisée pour réaliser l'authentification entre des parties étrangères sans qu'il soit besoin de recourir à une entité de confiance en ligne, peut aussi la garantir. Pour ce faire, une signature doit être systématiquement ajoutée aux messages générés ou modifiés. À la différence des applications de sécurité routière qui se distinguent par une exigence forte de non-répudiation, les applications de confort peuvent s'en passer dans la plupart des cas, à l'exception notable de certaines applications sensibles comme celles impliquant des transactions financières.

3.1.5. Disponibilité

L'objectif de la disponibilité est de garantir un accès permanent à un service ou à des ressources. De nombreuses attaques peuvent entraîner une perte ou une réduction de la disponibilité du réseau ou d'un service applicatif. Il n'existe aucun moyen de contrer un déni de service sur le canal radio provoqué par un attaquant ayant les moyens de brouiller efficacement la totalité du spectre radio.

Néanmoins, des techniques proactives comme le saut de fréquence, le changement de canal, permettent de se prémunir contre des attaquants ayant des capacités plus réduites. En effet, pour être efficace, un attaquant devra être capable de brouiller l'étendue des fréquences utilisées.

3.1.6. Contrôle d'accès

Le contrôle d'accès réseau permet de définir les entités autorisées à se connecter à un réseau en bloquant les utilisateurs non autorisés, en contrôlant l'accès des invités et en garantissant que les utilisateurs se conforment aux politiques de sécurité du réseau. Ce service est nécessaire pour les applications qui distinguent différents niveaux d'accès en fonction de l'entité. Par exemple, l'application de contrôle des feux tricolores autorise seulement les véhicules de secours ou de police à échanger des informations avec les feux tricolores pour faciliter leur déplacement.

Ce service est établi grâce à des politiques d'accès qui spécifient ce que chaque entité est autorisée à faire ou à accéder, dans le réseau. Par exemple, un garage agréé peut être autorisé à accéder pleinement à des diagnostics sans fil, tandis que les autres n'auront qu'un accès limité.
Un autre concept réactif pour le contrôle d'accès et l'exclusion de véhicules malveillants est la « vérification de plausibilité » ou « vérification de contexte ». Dans les applications de sécurité du trafic routier, les mécanismes de sécurité doivent détecter les fausses informations et les inconsistances du système. Ainsi, lors de réception d'un message, un véhicule évalue la validité de l'alerte avant de réagir. Le principe est que chaque véhicule collecte des informations de différentes sources pour créer une « vue courante » de son environnement. Les sources d'informations sont les messages d'alerte, les données des capteurs, etc. Ainsi, lorsqu'un véhicule reçoit une alerte de danger local, il peut comparer les informations (localisation, direction, etc.) avec sa « vue courante ».

3.2. La sécurité de routage dans les réseaux VANET

Dans la littérature, il existe plusieurs protocoles de routage sécurisés qui ont été développés pour les réseaux ad hoc de manière générale[16] [17]. Donc, ces protocoles ont été conçus sous des conditions plus ou moins contraignantes que les VANETs, ce qui oblige une reconsidération de leur conception avant leur utilisation dans les VANETs.

Les solutions proposées pour sécuriser les protocoles de routage Ad Hoc peuvent être classifiées en deux catégories[18] [19] :

- Systèmes de sécurité proactifs, dans le sens où des mécanismes sont établis à l'avance pour assurer la sécurité en renforçant la résistance du système aux attaques grâce notamment aux solutions à base de cryptographie,
- Systèmes de sécurité réactifs qui réagissent (adaptation/prise de décision immédiate) selon le comportement du voisinage et qui est lui-même divisé en solutions de gestion de réputation et de confiance.

Dans cette section nous allons décrire quelques protocoles de routage sécurisés connus dans les réseaux ad hoc, et qui ont été proposés comme solutions pour sécuriser les protocoles OLSR et AODV.

3.2.1. Mécanismes de sécurité pour le protocole OLSR

Il existe plusieurs extensions de sécurité pour le protocole OLSR qui ont été proposées dans la littérature[20]. On présentera une revue des principales solutions proposées pour sécuriser le protocole OLSR.

3.2.1.1. Secure OLSR

Ce mécanisme[21] de sécurité se base sur la signature de chaque paquet de contrôle OLSR pour authentifier les messages. La signature numérique est basée sur des clés symétriques.
Tout le trafic de contrôle OLSR est signé pour chaque saut. Ce qui veut dire que les champs variables dans les messages comme le nombre de sauts et le TTL ne sont pas considérés.

Aussi, une seule signature est utilisée puisque plusieurs messages OLSR sont empilés dans un seul paquet. L'utilisation de l'approche saut par saut ne permet pas des signatures de bout en bout, ce qui signifie aussi que le digest ne représente pas une signature de confiance provenant de la source, mais seulement une signature à partir de l'expéditeur en supposant que celui-ci a confiance en la source du message vers le saut précédent.

Un nœud ne disposant pas de la clé secrète partagée ne peut pas produire le bon digest. Tous les récepteurs qui fonctionnent avec « Secure OLSR » ignorent les messages avec des digests incorrects. Les signatures sont transmises dans leurs propres messages. Ceci pour assurer la compatibilité avec les nœuds qui ne fonctionnent pas avec « Secure OLSR », mais aussi parce que le Timestamp est transmis avec la signature.

Signature :

Le message de signature illustré dans la figure 3.2 est attaché à tous les paquets OLSR sortants. Ce message est le dernier à mettre dans le paquet. L'en-tête du paquet OLSR est ajusté pour inclure la taille du message de signature dans le champ « size ».

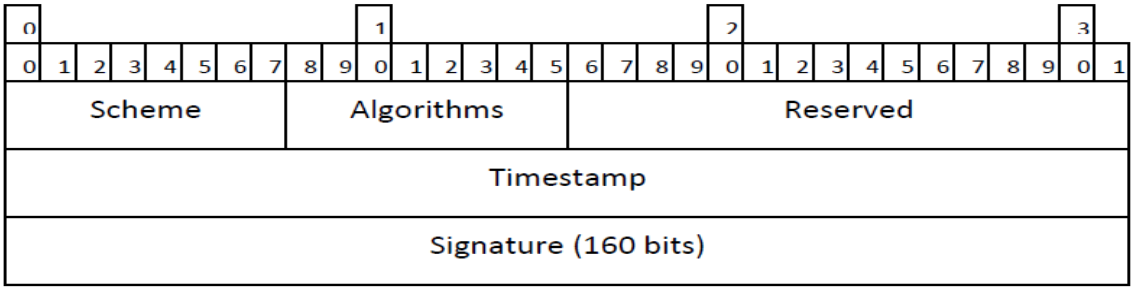


Figure 3.2 : Le message de signature élémentaire (Secure OLSR)[21]

Les champs scheme et algorithms dans l'en-tête du message signature, informent la destination sur la nature du schéma de signature et l'algorithme utilisés. Le champ Timestamp est utilisé pour prévenir les attaques de rejeux. Le digest utilisé comme signature est un haché créé en utilisant l'algorithme de hachage SHA-1, qui produit un digest de 160 bits irréversible. Le haché est basé sur :

- L'en-tête du paquet OLSR (avec sa taille ajustée).
- Tous les messages OLSR dans le paquet sans le message de signature.
- L'en-tête du message OLSR, le sous en-tête et le Timestamp du message de signature.
- La clé secrète partagée.

Aucune considération n'est apportée sur les champs variables des différents en-têtes (comme le TTL ou le hop count) puisque la signature est faite saut par saut.

Timestamp :

Un attaquant peut enregistrer le trafic signé pour le jouer ultérieurement (attaques rejeu). Ceci peut être empêché à un certain degré par les numéros de séquences qui sont déjà utilisés dans OLSR. Pourtant pour le trafic qui est envoyé seulement à un saut, comme les messages HELLO, cela n'aura pas d'effet. Un attaquant peut simplement enregistrer les messages transmis par un nœud, se déplacer dans un autre espace du réseau où les messages HELLO enregistrés ne sont jamais reçus. Ainsi, l'attaquant peut commencer l'attaque de rejeu en transmettant les messages déjà enregistrés. Les numéros de séquences OLSR sont, eux aussi fragiles à cause de leur longueur. Ils ont une longueur de 16 bits seulement et le problème de bouclage peut arriver assez fréquemment.

Pour prévenir les attaques de rejeux, les Timestamps sont utilisés dans cette extension de sécurité pour OLSR. Pour échanger ces Timestamps sur une connexion initiale entre deux nœuds, un mécanisme d'échange de Timestamp à deux directions est utilisé. La solution ne repose pas sur la synchronisation du temps entre les nœuds du réseau.

3.2.1.2. Architecture de sécurité pour OLSR

L'objectif de ce système[22] est d'isoler les nœuds malicieux en utilisant un système de signature et d'authentifier les messages OLSR de bout-en-bout. Ce cryptosystème repose sur l'ajout d'une signature aux messages de contrôle d'OLSR. En effet, chaque nœud génère la signature lors de la création de chaque message de contrôle (HELLO/TC/HNA/MID).
La signature est envoyée par la suite avec le message de contrôle dans le même paquet.

Cette architecture de sécurité n'est pas interopérable avec OLSR standard. En effet, un nœud dans lequel tourne OLSR sécurisé n'accepterait pas des HELLOs non signés de la part des nœuds OLSR standard, en conséquence il ne pourrait pas y avoir de lien symétrique entre les deux, et donc aucune sélection des MPRs qui est le mécanisme principal pour la diffusion des messages dans OLSR.

L'architecture PKI (Public Key Infrastructure) est basée sur une autorité centralisée qui est soit proactive ou réactive. La version proactive envoie périodiquement les certificats à tout le réseau. Par contre, la version réactive répond sur demande aux requêtes d'obtention de certificat par les nœuds.

Une simple PKI pour OLSR : version proactive :

Cette PKI pourvoit trois classes de nœuds :

- ***les autorités de signature*** dont la clé publique est connue par tout autre nœud du réseau, et qui ont la responsabilité d'enregistrer les clés publiques des autres nœuds participants et de distribuer périodiquement des certificats signés contenant la liste des clés publiques des nœuds fiables,
- ***les nœuds fiables,*** qui sont ceux dont la clé publique est connue et certifiée par une autorité de signature,

- ***les nœuds non fiables***, qui sont ceux dont la clé publique n'est pas connue ou n'est pas certifiée par une autorité de signature, il faut remarquer qu'au démarrage du réseau tout nœud (sauf les autorités de signature) est non fiable.

Pour garantir la confiance dans l'information topologique qui est distribuée dans le réseau, tout nœud doit choisir ses MPRs (et accepter d'être choisi comme MPR) parmi les seuls nœuds fiables, accepter les messages TC qui proviennent des seuls nœuds fiables, et faire suivre seulement les messages qui ont été reçus des voisins fiables. Une règle simple pour exclure les nœuds non fiables du réseau serait celle de refuser tout message envoyé par un nœud non fiable. Toutefois, ce comportement porterait à une situation d'interblocage au moment de l'initialisation du réseau, car tout nœud est non fiable à ce moment, et donc la sélection des MPRs (et la distribution des messages dans le réseau entier) serait impossible. Pour éviter cette situation, on établit qu'un nœud accepte les messages HELLO qui proviennent d'un voisin non fiable, et que ce nœud inclut ses voisins non fiables dans ses HELLOs, avec la condition que les liens MPR soient considérés simplement comme liens symétriques. En conséquence, l'autorité de signature transmettra ses certificats à ses voisins, ces voisins, après échange de messages HELLO, accepteront les voisins à deux sauts mais ne sélectionneront pas leurs MPRs parmi eux, ensuite, l'autorité de signature choisira ses MPRs parmi ses voisins pour que sa prochaine émission de certificat rejoigne tous les voisins à deux sauts.

Le message signature :

Le message de signature est encapsulé et transmis comme une portion de données du format standard d'un paquet OLSR. Le champ « Message Type » est mis à la valeur **SIGNATURE-MESSAGE**. Les champs TTL et Vtime sont mis aux mêmes valeurs que ceux des champs TTL et Vtime du message auquel est associée la signature.

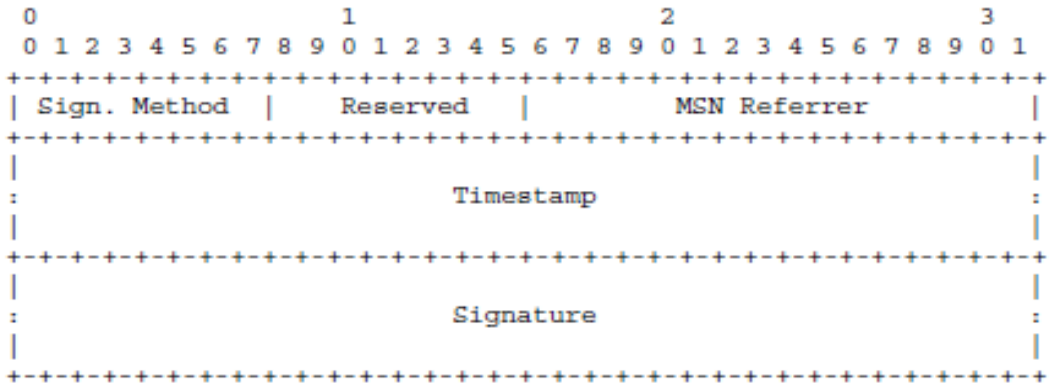


Figure 3.3 : Format de signature associé à chaque message du paquet[22]

Le couplage est identifié grâce au Numéro de Séquence (MSN) du message de contrôle en question et à un champ homologue dans le message de signature, cela permettait d'envoyer les messages dans un ordre quelconque.

Le champ Sign. Method spécifie les fonctions utilisées pour le digest et la signature du message de contrôle, les clés à utiliser, ainsi que des informations sur les méthodes de timestamp.

Le message SIGNATURE contient aussi une estampille temporelle (timestamp) obtenue de l'horloge interne du nœud, pour éviter les attaques de rejeu, la synchronisation des horloges ne nécessite pas d'être très précise, puisque les messages qui seraient des doublons peuvent être reconnus aussi par leur Numéro de Séquence.
Le Timestamp est inséré lors de la création du message en même temps que la signature. Ainsi, lorsqu'un nœud reçoit un message de contrôle, il contrôle le Timestamp et vérifie la signature du message. Si le Timestamp et la signature sont corrects, le nœud traite le message, sinon ce dernier le rejette et alors le nœud malicieux, originaire de ce message, se trouve isolé du reste du réseau.

3.2.1.3. ADVSIG (An Advanced Signature System for OLSR)

Les mécanismes de signature et de *Timestamp* ne sont pas suffisants pour faire face aux différents types d'attaques. En effet, cette solution n'est pas efficace dans le cas où un nœud légitime est compromis car ce nœud malicieux peut alors générer des messages signés correctement avec son identité et ainsi envoyer de faux messages de contrôle à travers le réseau. Dans ce contexte, un mécanisme additionnel ADVSIG (ADVanced SIGnature) a été proposé. Le but du mécanisme ADVSIG[23] est de garantir l'authenticité de bout-en-bout de l'ensemble des informations concernant l'état des liens à travers le réseau, ç-à-d, assurer l'intégrité du réseau et des messages échangés dans ce dernier.

Lors de la création des informations de la topologie et durant l'échange des messages OLSR, chaque nœud attache la signature ADVSIG et doit certifier l'authenticité des informations qu'il fournit à ses voisins. Il doit aussi générer une preuve qui sera utilisée par ses voisins pour prouver l'authenticité de ce lien avec le nœud originaire du message.

Information atomique sur l'état de lien :

La quantité minimale d'information échangée sur l'état de lien, générée par le nœud *A* concernant le nœud *B*, consiste en :

- l'adresse du nœud *B*
- l'état de lien de *B* par rapport à *A*
- une estampille temporelle
- la signature de ces trois champs, calculée par *A*

Les deux premiers champs sont tirés du message HELLO (ils sont tirés respectivement des champs Neighbor Interface Address et Link Code), tandis que les deux derniers sont contenus dans un message ADVSIG couplé à ce HELLO. Cette information atomique est appelée un ***Certificat*** ou une ***Preuve,*** selon respectivement qu'elle est reçue comme information topologique nouvelle ou qu'elle est réutilisée pour prouver un état de lien.

Quand un nœud reçoit un HELLO avec son ADVSIG, il extrait des deux messages les informations qui le concernent (à savoir, celles où l'adresse du nœud annoncé est son adresse), et ces informations constituent donc un *Certificat*. Les Certificats sont mémorisés par chaque nœud du réseau dans une table locale appelée ***Certiproof Table***.

Ensuite, quand le nœud envoie un HELLO ou un TC, il sélectionne dans son ***Certiproof Table*** une Preuve appropriée, qu'il inclura dans son ADVSIG couplé.

Certiproof table

Quand un nœud B reçoit un HELLO avec son ADVSIG d'un nœud A, il extrait des deux messages les informations qui le concernent, et stocke dans sa Certiproof Table le tuple :
< originator, address, linkstate, timestamp, signature>

Avec :

« originator » est l'adresse de A.

Les éléments restants constituent les champs du certificat :

« address *»* est l'adresse du B,

« linkstate *»* est l'état de lien de B par rapport à A,

« timestamp *»* est le temps dans lequel le nœud A a généré les messages HELLO et ADVSIG,

« signature *»* est la signature calculée par A sur les trois champs « *address* », « *linkstate* » et « *timestamp* ».

La clé de tuple est le champ «originator». Pour chaque «originator», un seul tuple est maintenu dans la table. Quand B reçoit un nouveau message HELLO (avec son ADVSIG) de la part de A, il met à jour l'entrée de tuple par les informations les plus récentes (en comparant les champs timestamp). De cette manière le nœud B stocke dans sa Certiproof Table seulement le certificat le plus récent envoyé par un voisin.

Preuves requises

Quand un nœud *A* veut déclarer un lien avec le nœud *B* dans un message HELLO ou TC, la preuve requise est construite en utilisant un HELLO et son ADVSIG couplé qui ont récemment été envoyés par *B* :

- une preuve que le paquet a été entendu, si *A* veut déclarer un lien de type ASYM_LINK avec *B*,
- une déclaration de ASYM_LINK ou SYM_LINK, si *A* veut déclarer un SYM_LINK avec *B*,
- une déclaration de SYM_LINK ou SYM_NEIGH, si *A* veut déclarer un SYM_NEIGH ou un MPR_NEIGH avec *B*,
- une déclaration de SYM_NEIGH ou MPR_NEIGH, si *A* veut déclarer *B* comme voisin.

Le message ADVSIG

Un message ADVSIG doit être généré et envoyé avec chaque message HELLO ou TC. Cependant il y a une différence entre les HELLOs et les TCs: les messages TCs ne peuvent pas contenir des certificats(les informations réutilisable de la topologie), ainsi la signature des champs du certificat existe uniquement dans les messages ADVSIG couplés aux messages HELLOs.

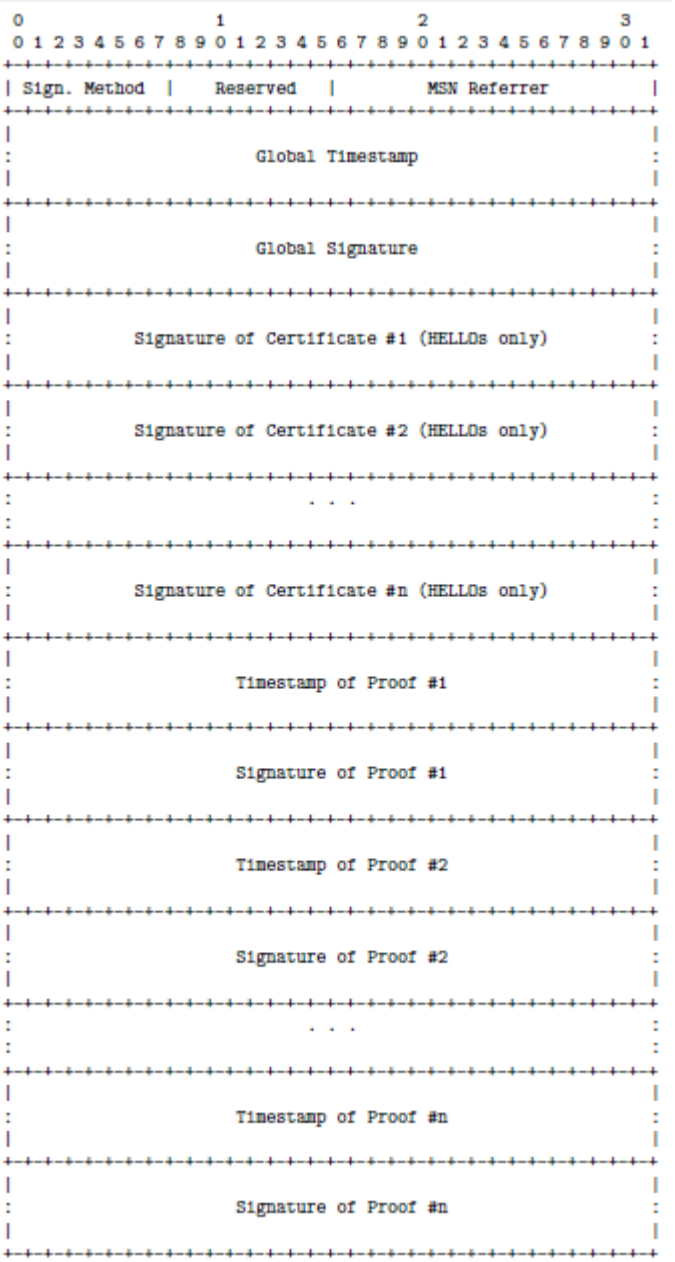


Figure 3.4 : Le message ADVSIG[23]

Sign. Method : spécifie les fonctions utilisées pour le digest et la signature du message de contrôle ainsi que des informations sur les méthodes de timestamp.

MSN Referrer : contient la valeur du champ Message Sequence Number du message Hello/TC auquel le message ADVSIG est associé.

Global Timestamp : est le timestamp de ce message (ADVSIG) et de HELLO/TC avec lequel il est associé.

Global Signature : est la signature du message HELLO/TC couplé, elle est calculée sur une séquence de bits composée du message HELLO/TC et de message ADVSIG associé (sauf le champ Global signature).

Signature of Certificate #i : existe seulement si le ADVSIG est associé à un message de type HELLO. Ce champ contient la signature du Certificate #i lié respectivement à «Neighbor Interface Address » de la position i du message HELLO couplé.

Timestamp of Proof #i et Signature of Proof #i : sont le timestamp et la signature de la preuve liée :

- Au « Neighbor Interface Address » de la position i si le message couplé est de type HELLO.
- « Advertised Neighbor Main Address» de la position i si le message couplé est de type TC.

Notons que dans les messages ADVSIG envoyés par A, chaque signature du certificat est signée par A, alors que chaque signature de la preuve est signée par les autres nœuds (qui sont, ou ont été récemment, les voisins de A).

Le protocole :

Quand un nœud génère un message HELLO ou TC, il doit générer aussi un ADVSIG, en suivant ce protocole :

1. créer le HELLO/TC
2. générer le timestamp global
3. si le message est un HELLO alors, pour chaque lien déclaré, calculer la signature du certificat et joindre la Preuve requise appropriée
4. sinon si le message est un TC alors joindre la Preuve requise appropriée
5. calculer la signature globale
6. envoyer le HELLO/TC et le ADVSIG

Quand un nœud reçoit un message de contrôle, il doit suivre ces étapes :

1. identifier correctement le HELLO/TC avec son ADVSIG couplé
2. contrôler la validité de timestamp global
3. contrôler la validité de la signature globale
4. si le message est un HELLO alors, pour chaque lien déclaré, contrôler la validité de la Preuve, et extraire le Certificat relatif au nœud lui-même le cas échéant
5. sinon si le message est un TC alors, pour chaque voisin déclaré, contrôler la validité de la Preuve.

Une Preuve n'est valable que si elle concerne le bon nœud, si le lien inclus est correct par rapport à la preuve requise, et si l'estampille temporelle (timestamp) n'est pas périmée. Si une erreur survient lors d'une de ces étapes, le HELLO/TC et son ADVSIG doivent être rejetés.

3.2.1.4. GPS-OLSR (OLSR with GPS information)

GPS-OLSR[24] est une extension sécurisée du protocole OLSR, qui inclut dans les messages de contrôle la position géographique du nœud émetteur. Cette information est ensuite retenue par les nœuds destinations pour évaluer la véracité des informations incluses dans le même message de contrôle. Tout nœud mémorise la dernière position connue de chaque autre nœud du réseau dans sa *Position Table*.

Une modification basée sur la géo-localisation a été apportée aux mécanismes précédents. Cette solution présente une approche pour faire face aux attaques de type *wormhole* et aux attaques par mystification des liens. En effet, les trois mécanismes regroupés ensemble (PKI, *Timestamp* et la localisation géographique) permettent de détecter tout relayage incorrect du trafic d'un point du réseau vers un autre éloigné.

Le message SIGLOC (SIGnature and LOCalization)

La position géographique de chaque nœud doit être incluse dans un nouveau type de message signé appelé SIGLOC. Ce message est construit comme un message SIGNATURE avec un champ supplémentaire qui contient la position du nœud, et est envoyé avec tout HELLO ou TC.

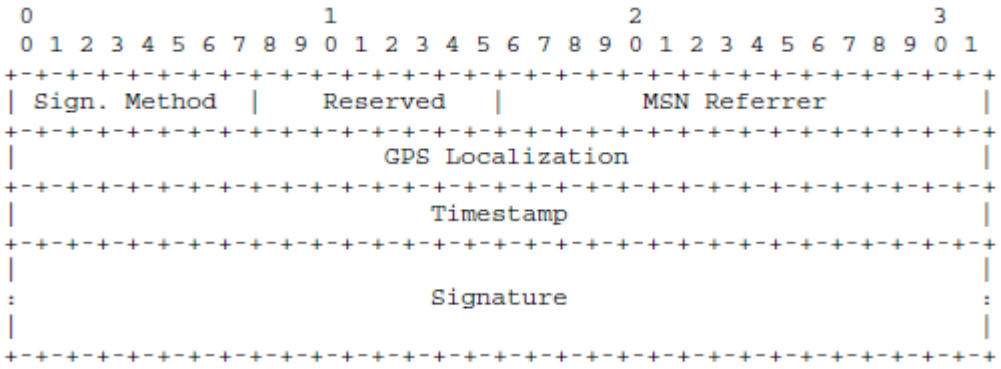


Figure 3.5 : Format de message SIGLOC[24]

Le champ **GPS Localization** contient la position géographique actuelle du nœud émetteur tel qu'elle est obtenue à partir du système GPS (Global Positioning System) intégré dans le matériel. Ce champ est de 32 bits, ce qui est suffisant pour définir une position sur une superficie de plus de 4200 kilomètres carrés avec une granularité de 1 m.

Ainsi, chaque nœud doit informer les autres nœuds de sa position géographique dans un message SIGLOC.
Quand un nœud reçoit un message HELLO/TC avec son SIGLOC couplé, il doit vérifier l'exactitude de l'horodatage et de signature. Ensuite, il extrait les informations sur la position géographique et stocke le tuple <node address, position, timestamp> dans un tableau **Position Table**.
Tout nœud mémorise la dernière position connue de chaque autre nœud du réseau dans sa *Position Table*.

Avantages

La connaissance des positions géographiques de chaque nœud du réseau permet au nœud récepteur de savoir si les communications entrantes sont susceptibles d'être entendues.
En effet, en connaissant les positions géographiques d'un nœud émetteur *S* et d'un nœud récepteur *R* à des moments définis, en calculant la variation de leur position (qui est à son tour limitée par la vitesse maximale d'un nœud), et en prenant en compte les erreurs dans la synchronisation des horloges et dans d'autres variables, on peut calculer leur distance au moment de la transmission. Cette distance ne peut pas être supérieure à la portée maximale de transmission : si c'est le cas, le lien est probablement faux. Cela permet à un nœud d'évaluer non seulement les transmissions qu'il reçoit (et de savoir si elles sont, par exemple, acheminées à travers un wormhole) mais aussi d'évaluer les déclarations de voisinage d'un autre nœud : si un nœud déclare avoir un lien avec un nœud qui est très loin, cette déclaration est fortement suspecte.

En conséquence ce protocole sécurise le réseau contre les attaques de link spoofing et wormhole. Il faut remarquer que ce mécanisme offre aussi des possibilités d'amélioration du protocole OLSR standard, telles qu'une sélection plus efficace des MPRs ou la prévision de rupture des liens.

De surcroît, l'utilisation d'une antenne directionnelle permettrait, avec des simples calculs de géométrie planaire, de savoir avec plus de précision si les informations reçues sont correctes ou fausses : un nœud peut vérifier si le secteur d'antenne dans lequel la transmission est entendue s'accorde avec la direction vers laquelle le nœud émetteur devrait se trouver (direction obtenue en évaluant sa position relative).

Le protocole

Quand un nœud A génère un message HELLO/TC, il doit aussi générer un message SIGLOC en suivant les étapes suivantes :

1. Créer le HELLO/TC et le SIGLOC
2. Insérer la localisation GPS retournée par le système GPS intégré dans le matériel
3. Insérer le Timestamp du temps réel
4. Calculer la signature sur le HELLO/TC + SIGLOC
5. Envoyer le HELLO/TC et le SIGLOC

Quand un nœud reçoit un message de contrôle de la part de A, il doit suivre les étapes suivantes :

1. Coupler le message HELLO/TC avec le SIGLOC associé en faisant correspondre le numéro de séquence du message avec MSN referrer
2. Vérifier la fraicheur du timestamp
3. Vérifier la validité de la signature en utilisant la clé publique du nœud A
4. Dans le cas d'utilisation d'une antenne directionnelle, il faut vérifier si la localisation GPS reçue s'accorde avec la direction de transmission
5. Pour chaque **Neighbor Address I** listé dans le message HELLO/TC : si **I** appartient à position Table, Vérifier si la distance entre les nœuds A et I n'est pas supérieure à la portée maximale de transmission
6. Stocker le tuple <adresse de A, localisation GPS, Timestamp> dans position Table.

Si l'une de ces étapes a échoué, les deux messages HELLO/TC et SIGLOC doivent être supprimés.

3.2.1.5.TOLSR (Trust system for OLSR)

C'est un protocole[25] de détection pour OLSR qui utilise un système d'évaluation du taux de confiance des nœuds. Une ***Trust Table*** globale, dont tout nœud maintient une copie en sa mémoire, associe à chaque nœud une valeur numérique qui représente son niveau de confiance. Quand un nœud détecte un autre nœud qui ne respecte pas le protocole, ce premier diffuse en inondation un message d'accusation signé, s'il y a un nombre suffisant de nœuds qui envoient une accusation pour un même nœud dans le même laps de temps, les nœuds réduisent la valeur du niveau de confiance du nœud accusé. Toutefois, s'il n'y a pas assez d'accusations pendant le temps établi, c'est le nœud accusé qui voit remonter son niveau de confiance tandis que ses dénonciateurs sont pénalisés : cela pour éviter les abus de la part d'un nœud malveillant. Le niveau de confiance de tous les nœuds est périodiquement haussé d'une valeur prédéterminée pour parer les collisions, les erreurs en transmission et les pertes physiologiques de paquets qui s'avèrent même dans un réseau dépourvu de nœuds malveillants. Une fois que le niveau de

confiance d'un nœud accusé tombe à zéro, ce nœud est exclu du réseau, par effacement de son adresse dans les tables de routage.

Spécifications

Le tableau global ***trust table*** associe à chaque nœud du réseau une valeur numérique représentant son niveau de confiance.
Lorsqu'un nœud détecte un mauvais comportement du voisin, il avertit immédiatement le réseau en diffusant un message d'accusation contenant :

- Son adresse
- L'adresse du nœud voisin accusé
- Un code indiquant le type de la mauvaise conduite
- Le timestamp
- La signature

Le format possible d'un message d'accusation :

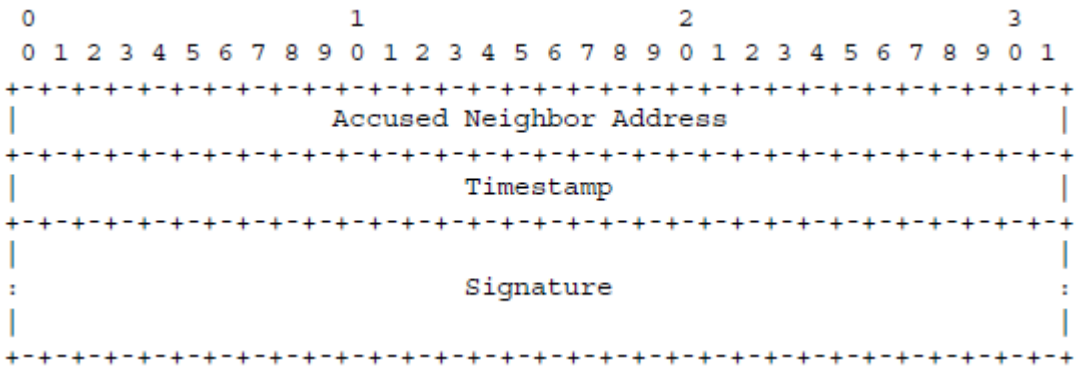


Figure 3.6 : Format du message d'accusation[25]

Les comportements interdits peuvent aller de la négligence dans le relayage, à l'envoi d'un message de contrôle difforme, à une fasse signature dans le message, à une usurpation d'identité jusqu'à l'essai d'un Déni de Service par bombardement de messages, à chaque comportement est associée une réduction différente du niveau de confiance.

Indice	Mauvaise conduite
i=0	L'échec dans la déclaration d'une mauvaise conduite
i=1	L'échec de transfert
i=2	message de contrôle mal formé
i=3	timestamp vicié
i=4	Signature non valide
i=5	usurpation d'identité
i=6	Déni de Service (bombardement de messages)

Tableau 3.1 : Les mauvaises conduites[25]

Chaque nœud maintient une table d'accusation qui stocke toutes les dernières accusations (diffusés dans le réseau) entendues.

A la réception d'une accusation, Le nœud doit l'évaluer d'une façon indépendante pour éviter la nécessité de maintenir une entité centralisée permettant l'évaluation de la confiance.

Si la diffusion est faite correctement, chaque nœud doit avoir la même copie de la table accusation.

3.2.1.6.CSS-OLSR (Cooperative Security Scheme for OLSR)

Cette solution[26] a été proposée afin d'assurer qu'un nœud génère et relaye correctement le trafic de contrôle, pour atteindre cet objectif, le principe est de récompenser les nœuds qui ont un comportement normal et pénaliser les nœuds qui ne se comportent pas bien (nœuds malicieux), ainsi, pour juger le bon comportement d'un nœud quelconque, il faut qu'il génère et relaye correctement le trafic de contrôle.

Cette solution ajoute au protocole OLSR standard les trois éléments suivants :

- ***Message CPM*** (Complete Path Message) : utilisé pour transmettre le chemin parcouru par un message à travers le réseau. Dès la réception de message TC, chaque MPR envoie un message de retour CPM à l'expéditeur du TC contenant le chemin parcouru par le message TC.

- ***Table de classification*** (Rating table) : chaque nœud du réseau a sa propre table de classification qui contient des informations sur le comportement de ses voisins d'un saut et 2 sauts. Chaque entrée dans la table contient l'ID de nœud, classification primaire (primary rating) et classification secondaire (secondary rating). L'ID identifie le nœud d'une manière unique, secondary rating est une classification à base de l'observation directe de comportement de nœuds et primary rating qui est une classification qui dépend des informations fournies par le message CPM (analyse de CPM) et de secondary rating.
  La table rating permet aux nœuds de décider comment gérer les nœuds qui se comportent mal.

- ***Message d'alerte*** : Ce type de message est utilisé pour avertir les nœuds voisins de la mauvaise conduite des autres nœuds du réseau.

Spécification du protocole

CSS-OLSR est une extension de sécurité du protocole OLSR qui utilise la procédure suivante :

1. Lors de la formation du réseau, une autorité de certification distribuée[27] est utilisée pour garantir l'authentification de chaque nœud.
2. Chaque fois qu'un nouveau nœud entre dans le réseau, l'autorité de certification distribuée est utilisée pour assurer l'authenticité du nœud
3. Lors de la diffusion du message Hello pour assurer la connaissance des voisins de premier et deuxième niveau, seuls les nœuds correctement authentifiés sont considérés.
4. Pour chaque nœud authentifié trouvé, une nouvelle entrée dans la table de classification est ajoutée avec la valeur 100 pour la classification secondaire et la valeur 50 pour la classification primaire.
5. En utilisant les informations de message Hello chaque nœud effectue la sélection de leurs ensembles MPR.
6. Les MPR sélectionnés sont déclarés dans les prochains messages Hello.

7. En utilisant cette information, chaque nœud peut construire la table des MPR selector (les nœuds qui ont sélectionnés un nœud comme MPR).
8. Un message TC est envoyé périodiquement par chaque nœud et relayer dans le réseau pour déclarer les MPR selector.
9. Dès la réception d'un message TC, un message CPM contenant le chemin parcouru par le message TC peut être renvoyé à l'origine.
10. En utilisant les informations des différents messages TC reçus, chaque nœud maintient une table de topologie qui se compose d'une entrée d'une adresse de destination, une adresse de dernier saut vers cette destination et un numéro de séquence.
11. La table de topologie est utilisée pour calculer la table de routage.

Détection de nœud malicieux

Deux méthodes possibles :

- ***Détection par observation directe***

Cette détection se fait par l'écoute de chaque nœud aux transmissions de ses MPRs. Si le nœud source S d'une communication détecte qu'un MPR n'a pas relayé son message, il diminue la classification secondaire de ce MPR par 2 et notifie les voisins d'un saut par un message décrivant la mauvaise conduite, A la réception de ce dernier, chaque voisin de S décrémente la classification secondaire par 1 , si non si, le MPR a bien relayé le message, sa classification secondaire est incrémenté par « 1 » mais seulement par le nœud S.
Pour encourager la coopération, la punition doit être supérieure à la récompense, ceci peut être assuré par le fait que seul le nœud source S augmente la classification secondaire par l'observation directe, et l'ensemble de ses voisins d'un saut, le diminue si le nœud ne se comporte pas bien.

- ***Détection par analyse de message CPM***

Il y'a des cas où les nœuds ne peuvent pas détecter les nœuds malicieux par l'observation directe de ses voisins (collision de paquets, puissance de transmission limitée…). La classification secondaire est utilisée uniquement comme état de nœud non fiable tandis que la classification primaire est utilisée pour classer les nœuds qui se comportent mal. La classification primaire est obtenue grâce à la corrélation de la classification secondaire et les informations tirées des messages CPM.

Algorithme de traitement de message CPM

Fondamentalement, l'algorithme stipule que si le nœud A est le destinataire prévu de message CPM et il a envoyé un message TC dans un laps de temps δ , A trouve le MPR auquel il a transmis le paquet, disant M1 et vérifie si :
Le premier saut après M1 dans le path du message CPM appartient à la liste des relais multipoints de M1. Et Si ce nœud (le premier saut après M1 dans le path du message CPM) est bien celui attendu par la table de routage de A :

- Si c'est le cas, et si la classification secondaire est plus grande que la classification primaire de nœud M1(on peut dire qu'il s'agit d'un nœud légitime), on incrémente la classification primaire de M1 par ***round(2/3 × (secondary rating - primary rating))*** , et par la suite on affecte la valeur de la classification primaire à la classification secondaire du même nœud.

Si c'est le contraire, c'est-à-dire que la classification secondaire est plus petite que la classification primaire du nœud M1 (le nœud M1 a été signaler comme malicieux), on la considère comme une erreur de la classification secondaire (parce que l'observation directe des nœuds peut être une source d'erreur), donc la classification secondaire est augmenté par «1».

- Si non, si l'information dans CPM n'est pas compatible avec ce que M1 annonce et la classification secondaire est inférieure à la classification primaire de M1 (nœud malicieux), on décrémente la valeur de la classification primaire en lui affectant la valeur de la classification secondaire, et si la classification secondaire est supérieure à la classification primaire (c'est-à-dire que M1 semble être un nœud légitime), puisque l'information de CPM montre le contraire, la classification secondaire de M1 est décrémentée par 2, par la suite A transmet le paquet à tous les voisins d'un saut pour le même traitement.

Algorithm 1 CPM processing

$SR_{MPR} \leftarrow$ secondary rating of the MPR in A's rating table
$PR_{MPR} \leftarrow$ primary rating of the MPR in A's rating table
if A is the intended receiver of the CPM and A has sent a TC message to the network within a short period of time δ **then**
    if the information in the CPM is consistent with the information obtained from the MPR by A **then**
        if $SR_{MPR} > PR_{MPR}$ **then**
            $PR_{MPR} \leftarrow PR_{MPR} + \text{round}(\frac{2}{3} \times (SR_{MPR} - PR_{MPR}))$
            $SR_{MPR} \leftarrow PR_{MPR}$
        else
            $SR_{MPR} \leftarrow SR_{MPR} + 1$
        end if
    else
        if $SR_{MPR} < PR_{MPR}$ **then**
            $PR_{MPR} \leftarrow SR_{MPR}$
        else if $SR_{MPR} > PR_{MPR}$ **then**
            $SR_{MPR} \leftarrow SR_{MPR} - 2$
        end if
    end if
    A forwards the CPM to all one-hop neighbors.
else
    Forward the CPM as usual.
end if

Figure 3.7 : Algorithme du traitement de message CPM[26]

Avantages de la solution CSS-OLSR.

- CSS-OLSR hérite les avantages d'autorité de certification distribuée permettant d'identifier chaque nœud et l'origine exacte de chaque paquet sans l'utilisation d'une approche centralisée. De cette façon des attaques d'usurpation d'identité sont contrées.

- Insérer de faux messages (par l'envoi d'informations d'état de lien erronées soit à partir de Hello ou TC) de routage va pénaliser le nœud malicieux: les chemins reçus dans les CPM seront incompatibles avec les informations fournies par le nœud malicieux ce qui diminue sa classification principale et par la suite réduire sa capacité de communiquer.

- Refuser de relayer le trafic peut être détecté par une corrélation d'un ensemble de CPMs reçus, la probabilité du nœud envoyant le message CPM et la densité du réseau.

- Les attaques wormhole peuvent être partiellement détectées par la même technique utilisée pour détecter l'attaque au-dessus si le nœud malveillant décide de supprimer les paquets.

- Pour se défendre contre les attaques de rejeu, l'usage traditionnel du timestamp peut être invoqué.

Inconvénients de la solution CSS-OLSR

- L'ajout de message CPM va entrainer une surcharge dans le réseau.

- Cette solution (CSS-OLSR) ne permet pas de détecter les nœuds malicieux qui ne sont pas sélectionnés comme MPRs.

- Si le nœud malicieux est sélectionné comme MPR et ses MPR selector ne sont pas sélectionnés comme MPR, la solution CSS-OLSR ne sera pas capable de détecter le nœud malicieux. Les MPR selector de nœud malicieux doivent être aussi sélectionnés comme MPR pour qu'ils puissent lancer un TC.

- La modification des messages de contrôle est très difficile à résoudre par cette solution.

3.2.2. Comparaison entre les extensions sécurisées du protocole OLSR

Dans cette section nous allons comparer entre les variantes sécurisées du protocole OLSR déjà présentées dans la section précédente au niveau des mesures de sécurité standards implémentées et de protections offertes par chacune d'entre eux contre les attaques spécifiques au protocole OLSR.

3.2.2.1.Les mesures de sécurité standards implémentées pour chaque variante

	Approche d'authentification des nœuds du réseau		Entité signée			Schéma d'authentification		Autorité de certification		Interopérabilité avec OLSR standard	
	Saut en saut	Bout en bout	Paquet	Message	Informations topologiques	Clé symétri-que	Clé asymétri-que	centralisée	distribuée	Oui	Non
SOLSR	✓		✓			✓				✓	
Messages de signatures		✓		✓			✓	✓			✓
ADVSIG		✓		✓	✓		✓	✓			✓
GPS-OLSR		✓		✓			✓	✓			✓
TOLSR							✓	✓		✓	
CSS-OLSR									✓		✓

Tableau 3.2 : Les mesures de sécurité standards implémentées par les variantes sécurisées du protocole OLSR

3.2.1.1. Les protections offertes par chaque variante

	Génération incorrecte du trafic					Relayage incorrect du trafic			
	Génération incorrect des HELLOs		Génération incorrect des TCs		Attaques ANSN	Relayage incorrect du trafic de contrôle	Attaque par retransmission des messages de contrôle	Attaque wormhole	Attaque du Blackhole
	Identity spoofing	Link spoofing	Identity spoofing	Link spoofing					
SOLSR	✓		✓		✓		✓		
Messages de signatures	✓		✓		✓		✓		
ADVSIG	✓	✓	✓	✓	✓		✓		
GPS-OLSR	✓	✓	✓	✓	✓		✓	✓	
TOLSR	✓		✓		✓	✓	✓		✓
CSS-OLSR	✓	✓	✓	✓		✓	✓	○	

○ : Partiellement

Tableau 3.3 : Les protections offertes par les variantes sécurisées du protocole OLSR

3.2.3. Mécanismes de sécurité pour le protocole AODV

3.2.3.1.SAODV (Secure Ad hoc On-Demand Distance Vector)

SAODV[28]est une extension du protocole AODV pour assurer l'authenticité et l'intégrité des messages de routage.

Dans AODV, les messages de routage (Route_Reply et Route_Request) ont une partie non modifiable et une autre modifiable. La partie non modifiable est protégée par une signature numérique et elle inclut les champs suivants : le numéro de séquence, les adresses des nœuds source et destinataire et l'identifiant de requête. Tandis que la partie modifiable qui inclut le compteur de sauts est protégée par une technique basée sur les chaînes de hachage, qui permet aux nœuds intermédiaires (selon les concepteurs de ce protocole) de vérifier que sa valeur n'a pas été décrémentée abusivement.

Cette extension permet de signer le paquet AODV par la clé privée de l'expéditeur originaire du message de routage.
Quand un RREQ est envoyé, l'expéditeur signe le message. Les nœuds intermédiaires vérifient la signature (en utilisant la clé publique de l'expéditeur), s'elle est valide, ils créent ou met à jour un chemin inverse à cet émetteur. Le nœud de destination finale signe le RREP avec sa clé privée. Les nœuds intermédiaires et finaux, encore une fois vérifient la signature avant de créer ou mettre à jour une route à cet hôte, stockent également la signature avec l'entrée de l'itinéraire.

3.2.3.2.ARAN (Authenticated Routing for Ad hoc Networks)

Sanzgiri et al. ont proposé le protocole sécurisé ARAN[29] qui prévoit l'utilisation de la cryptographie à clé publique pour sécuriser la construction des chemins des protocoles réactifs tels que AODV. Il suppose l'existence d'un serveur d'authentification, dont le rôle est de gérer la distribution des certificats pour les nœuds autorisés dans le réseau. Ce certificat, signé par le serveur d'authentification, contient l'identité du nœud (i.e. l'adresse IP), sa clé publique, une date de création et une date d'expiration. Ainsi, avant de rejoindre le réseau, chaque nœud doit, au préalable, récupérer un certificat auprès du serveur qui lui servira à signer les messages de contrôle.

ARAN s'appuie sur deux mécanismes d'authentification. Le premier consiste en une authentification de bout en bout afin qu'un nœud destinataire puisse d'une part authentifier l'origine d'un message de contrôle, et d'autre part vérifier la non-modification des données statiques (i.e. l'adresse du nœud source et destinataire) pendant le transit. Le second est une authentification de saut en saut dans lequel chaque nœud sollicité dans un processus de recherche ou de maintenance de chemin utilise un certificat pour s'authentifier auprès d'autres nœuds voisins. En particulier, un nœud doit, pour chaque message de contrôle qu'il reçoit, vérifier le certificat fourni par le nœud précédent, puis s'il est valide, l'utiliser pour vérifier la signature. Ensuite, s'il n'est pas le destinataire du message, il supprime le certificat et la signature du nœud précédent, signe le message avec sa propre clé privée, et appose son certificat avant de le rediffuser.

ARAN offre des services d'authentification, d'intégrité et de non-répudiation. Une particularité d'ARAN concerne les informations incluses dans le message de demande de chemin.
Contrairement aux spécifications d'AODV dont il s'inspire, ce message ne contient ni compteur de sauts devant être incrémenté au fur et à mesure des retransmissions, ni chemin spécifique associé à la source. Ainsi, ARAN ne garantit plus le chemin le plus court (exprimé en nombre de sauts), mais le plus rapide, c'est-à-dire celui qui donne lieu en premier à une réponse de chemin. En raison de cette approche, les messages de contrôle ne sont finalement formés d'aucun champ variant au fil de leur retransmission. L'authentification des messages effectuée de saut en saut assure alors qu'un attaquant indépendant, qu'il soit interne ou externe au réseau, ne peut ni créer des boucles de routage, ni rediriger le trafic en insérant par exemple des identités de nœuds illégitimes dans les messages de découverte de chemins.

Néanmoins, en termes de sécurité, une limitation vient du fait que dans la phase de maintenance, la véracité d'une information de rupture de lien n'est pas vérifiée. Cette absence de vérification offre l'opportunité à un attaquant interne d'inoculer de fausses annonces de ruptures, ceci dans le but d'invalider des liens (et des chemins) pourtant opérationnels. Un tel comportement est un déni de service qui peut entrainer une surconsommation des ressources, car des procédures de recherche de chemins devront être reconduites.

Une autre limitation vient du fait que les mécanismes proposés ne permettent pas de déterminer si les nœuds intermédiaires relaient les messages pour lesquels ils ont été sollicités. En d'autres termes, il ne permet pas de contrer les attaques par non-participation.

Puisqu'aucun compteur de sauts n'est utilisé dans les messages de contrôle, cette caractéristique offre l'opportunité à un attaquant d'augmenter les délais lors de la formation d'un chemin, soit en retardant la propagation de la demande, soit dans le pire des cas, en supprimant la demande. Dans la phase de maintenance, la suppression d'une annonce de rupture de lien peut également s'avérer très néfaste pour le réseau, car des délais supplémentaires seront occasionnés pour d'une part détecter, et d'autre part corriger la rupture.

3.2.3.3.SEAR (Secure Efficient Ad hoc on demand Routing)

Le protocole SEAR[30] est une extension sécurisée du protocole AODV, il permet d'assurer l'authenticité des messages de routage via l'utilisation des fonctions de hachage unidirectionnelles permettant de construire et associer à chaque nœud du réseau une liste de hachés appelés authentificateurs, et de sécuriser simultanément le numéro de séquence et le nombre de sauts.

Dans SAODV, un nœud malveillant peut passer un RREQ aux autres nœuds sans incrémenter le nombre de sauts, ce problème est évité par SEAR en codant l'identité du nœud dans la liste des hachés. Par conséquent, chaque nœud ne peut pas transmettre les messages de routage avec des authentificateurs encodés avec l'identité d'un autre nœud, et ils doivent augmenter le nombre de sauts de RREQ et RREP.
Au fur et à mesure de la propagation d'un message de découverte, un MAC est apposé par chaque nœud intermédiaire, il s'ensuit que la taille de ce message augmente linéairement avec la longueur du chemin.

Les messages RERR sont protégés par une variante du protocole TESLA[31] (un protocole d'authentification « broadcast »).

3.2.4. Comparaison entre les extensions sécurisées du protocole AODV

Le tableau 3.4 présente une comparaison entre les solutions de sécurité du protocole AODV déjà présentées dans la section précédente au niveau des mesures de sécurité standards implémentées par chacune d'entre eux.

	Approche d'authentification des nœuds du réseau		Authentification de la source du message par :		Protection du champ modifiable		protection TESLA pour les messages RERR		Les nœuds intermédiaires répondent par RREP	
	Saut en saut	Bout en bout	Authentifi-cateurs de hachage	Signature numérique	Authentifi-cateurs de hachage	Chaine de hachage	Oui	Non	Oui	Non
SAODV		✓		✓		✓		✓	✓	
SEAR	✓		✓		✓		✓			✓
ARAN	✓			✓				✓	✓	

Tableau 3.4 : Les mesures de sécurité standards implémentées par les variantes sécurisées du protocole AODV

N.B : Les trois protocoles sécurisés SAODV, SEAR et ARAN utilisent tous des mécanismes cryptographiques qui assurent l'authenticité et l'intégrité des messages. Toutefois, ces mécanismes sont d'une part assez lourds à mettre en place notamment lors de la distribution des clés, et d'autre part, ils n'empêchent pas des nœuds ayant le matériel cryptographique nécessaire de se comporter malhonnêtement. Ces solutions sont donc plutôt destinées à fournir une protection contre les attaques externes.

Chapitre 4 : Simulations et résultats

L'émergence de réseaux de véhicules a encouragé la conception d'un ensemble de nouvelles applications et de protocoles spécifiquement pour ces types de réseaux. L'évaluation de ces protocoles en plein air, en utilisant des réseaux à grande échelle pour obtenir des résultats significatifs, est extrêmement difficile en raison des coûts de construction, c'est pourquoi la simulation est devenue un outil indispensable car il permet de modéliser un VANET et ensuite de récupérer des données statistiques sur l'utilisation du réseau au cours de la simulation afin de mesurer les performances des protocoles.

Ce chapitre s'articule autour de deux parties, en premier lieu, nous allons présenter les différentes étapes qui doivent être suivies pour simuler un réseau véhiculaire. En dernier lieu, nous allons présenter notre objectif et les outils utilisés pour réaliser nos simulations, expliquer les différentes simulations effectuées et analyser les résultats obtenus.

4.1. Contexte et objectif

Après avoir comparé entre les protections offertes par les extensions sécurisées du protocole OLSR, on a décidé de travailler sur la solution CSS-OLSR et on a réalisé des simulations pour étudier l'impact du taux des messages CPM (Voir chapitre 3) sur les performances des paramètres de qualité de service (QoS) dans un réseau véhiculaire.

4.2. Processus de simulation dans les VANETs

Le processus de simulation dans les VANETs exige quatre étapes qui doivent être exécutées (voir la figure 4.1). Dans ce qui suit, nous essayons d'expliquer les différentes étapes d'exécutions.

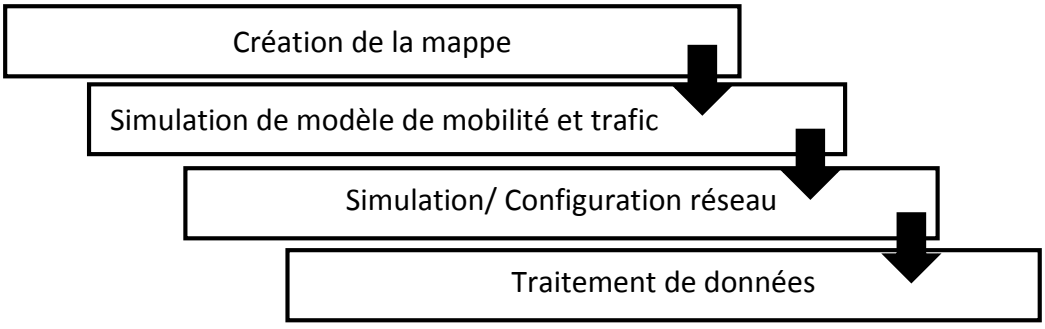


Figure 4.1 : Le processus de simulation dans les réseaux VANET

4.2.1. Génération d'une Mappe de simulation

Dans la première étape, nous avons besoin d'une zone de simulation appropriée au scénario désiré pour visualiser le mouvement des nœuds. Dans le cas des VANETs, la zone de simulation se présente comme étant une carte routière, cette carte peut être créée de deux manières. Manuellement en donnant la possibilité à son utilisateur de la générer ou bien de la dessiner d'une façon manuelle, dans ce cas-là, la topologie obtenue n'est pas réelle. Ou bien automatiquement en se basant sur l'importation automatique d'une carte réelle à partir des bases de données en ligne, elle est prise soit sous forme des images satellites par Google Earth, soit tirée de la base de données TIGER (Topologically Integrated Geographic Encoding and Referencing system) ou

l'extraire de site OpenStreetMap sous forme d'un fichier .osm, en tous les cas, l'utilisateur personnalise la carte selon ses besoins de son scénario.

4.2.2. Simulation de modèle de mobilité et la génération du trafic

Un modèle de mobilité reflète le comportement et les déplacements des nœuds dans un réseau, où le but est de représenter au mieux les conditions de cette mobilité dans un contexte particulier du monde réel. Les déplacements des véhicules ne peuvent pas être représentés par les modèles de mobilité MANET, cela est dû d'une part à la liberté restreinte des véhicules dans leurs mouvements car ils doivent obéir aux règles de la circulation, et d'une autre part aux interactions entre véhicules.

Dans les VANETs, les déplacements et les vitesses des véhicules sont délimités et prédéfinis par les routes et le comportement des conducteurs qui doivent obéir aux règles de la circulation. Le modèle de mobilité dans un VANET se compose de méthodes qui gèrent les déplacements au sein de la mappe qu'on a défini précédemment, et de méthodes de communication qui permettent la connexion aux stations de base, la communication des véhicules entre eux et la circulation de l'information entre les nœuds du réseau. La représentation de la mobilité est très importante pour les simulations de réseaux de véhicules. Le développement ou la configuration de la simulation de modèles de mobilité et de trafic dans les VANETs suit deux approches différentes. La première approche consiste à mettre en application un nouveau modèle de mobilité i.e. la carte routière (les nœuds, les routes, les voies, les jonctions et les véhicules) et les règles de la circulation (la distance de sécurité et les priorités entre les véhicules) sont programmés à nouveau par l'utilisateur, la deuxième approche se base sur un modèle de mobilité déjà conçu ou un modèle de trafic déjà existant. L'exécution et la simulation de modèle de mobilité génère des fichiers TRACE qui décrivent les déplacements des nœuds mobiles du réseau, et les requêtes échangées par ces derniers. Ces fichiers traces sont ensuite utilisés par un simulateur réseau dans l'étape de la simulation réseau.

4.2.3. Simulation de modèle réseau

La simulation est l'implantation d'un modèle simplifié du système à l'aide d'un programme de simulation adéquat. Cette méthode traduit le comportement du système à évaluer d'une manière réelle. La simulation permet de tester et visualiser à moindre coût les résultats sous forme de graphes.

4.2.4. Traitement de données

Après avoir fait la simulation du modèle réseau, il ne reste plus qu'à interpréter les résultats de simulation en analysant les graphes obtenus.

4.3. Environnement de simulation

Pour effectuer nos simulations, nous avons utilisé le simulateur réseau **Network Simulator *NS-2.29*** et le simulateur de mobilité ***Vanetmobisim*** sous le système d'exploitation Ubuntu 10.04.

Dans la suite, nous présentons successivement ces outils.

4.3.1. NS-2

Le plus célèbre et le plus répandu des simulateurs de réseaux est sans conteste Network Simulator 2 (NS-2)[32]. Il s'agit d'un simulateur à événements discrets disponible gratuitement et open source. L'ouverture du code source à la communauté a contribué à l'enrichir de nouveaux protocoles et de nouvelles fonctions au fil du temps. Il permet à l'utilisateur de définir un réseau

et de simuler des communications entre les nœuds de ce réseau. La simulation doit d'abord être saisie sous forme de fichier texte que NS-2 utilise pour produire un fichier trace contenant les résultats.

NS-2 nécessite deux langages : OTcl (Object Tools Command Language) et C++. À travers le langage Otcl, l'utilisateur décrit les conditions de la simulation : topologie du réseau, caractéristiques des liens physiques, protocoles utilisés, etc. Bien que les scripts de simulation soient écrits en Otcl, la base du simulateur est en C++ ce qui permet à chacun de modifier à sa guise les différents protocoles. Ces éléments sont résumés sur la figure au-dessous :

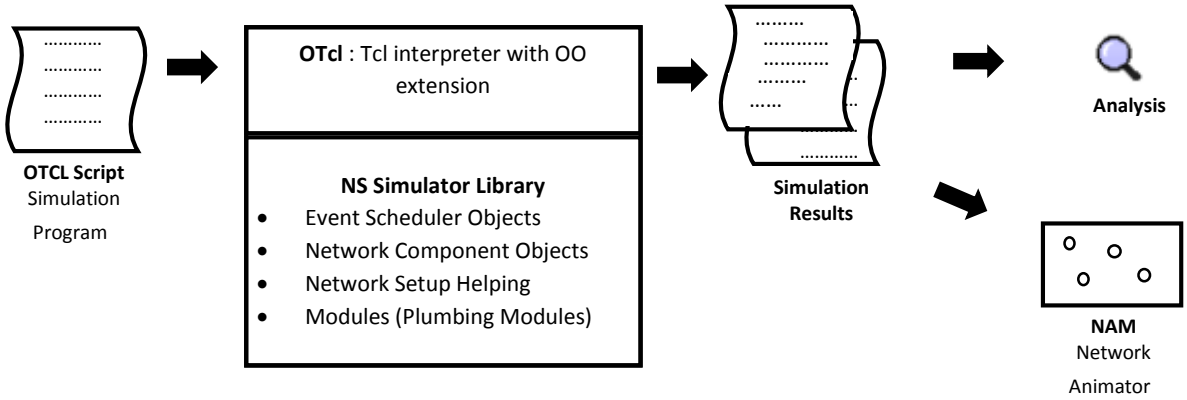


Figure 4.2 : Synoptique de l'interpréteur de script Otcl avec les librairies C++ de simulation de réseau

Comme le montre la figure, le script Otcl qui définit la simulation est interprété et associé aux bibliothèques C++ du simulateur NS-2. L'ensemble permet d'obtenir des fichiers de résultats autorisant l'analyse ou la visualisation de la simulation.

L'avantage de l'utilisation de ces deux langages est de faciliter au maximum l'écriture des scripts de simulation tout en gardant de bonnes performances du code grâce au C++. De plus, comme Tcl est un langage interprété, les changements dans les scripts de simulation ne nécessitent aucune compilation. Le désavantage majeur vient de la vitesse d'exécution des scripts. L'utilisation de deux langages entraîne aussi la nécessité de définir les éléments utiles à la simulation dans les deux langages pour pouvoir fonctionner.

Le simulateur NS-2 fonctionne sur le principe des événements discrets. Ainsi, le temps simulé est discrétisé, c'est-à-dire que la simulation change d'état à chaque « événement ». Un événement est un point du temps simulé référencé dans la simulation. Les changements d'états interviennent au temps simulé spécifié par l'événement. C'est le gestionnaire d'événements qui effectue cette tâche. Il est donc possible d'ordonnancer plusieurs événements dans le temps simulé. La simulation s'arrête lorsqu'il n'y a plus d'événements à interpréter. NS-2 modélise alors tous les éléments du réseau avec des classes hiérarchisées.

Le simulateur NS-2 imite le flot des paquets dans le réseau. L'environnement simulé contient un certain nombre de nœuds et chaque nœud contient des objets réseaux qui représentent des applications, des couches du modèle OSI ou tout autre élément du réseau nécessaire à la simulation. Ces objets réseaux interagissent les uns avec les autres en se passant des « paquets ». Comme dans un réseau réel, les paquets sont tout d'abord enrichis avec des en-têtes lorsqu'ils passent d'une couche du modèle OSI à l'autre dans un nœud (sens descendant), puis, après avoir

été transmis via un médium simulé, chacun de ces en-têtes est décodé et désassemblé dans le nœud récepteur lors de la phase ascendante du modèle OSI.

4.3.2. VanetMobiSim

VanetMobiSim[33] est une extension de CanuMobiSim. Ce dernier est un simulateur de mobilité en Java qui peut générer des simulations de mobilité dans différents formats (NS-2, GloMoSim, QualNet, etc.). Il inclut également différents modèles de mobilité aléatoire ainsi que des modèles plus réalistes. Le problème de CanuMobiSim est qu'il ne peut être utilisé que dans certains environnements, à cause de son manque de modèles adaptés. Il n'est pas à même de fournir un grand niveau de détails dans certains scénarios spécifiques. VanetMobiSim vient compléter les lacunes de CanuMobiSim. La modélisation des VANETs inclut des relations entre véhicules mais aussi entre véhicules et infrastructures. Au niveau mobilité, elle doit également inclure les panneaux stop, les feux de croisements et une mobilité basée sur l'activité humaine. VanetMobiSim regroupe tous ces éléments. Il permet aussi d'extraire des cartes des bases de données TIGER et GDF, ainsi que de créer des cartes manuellement ou aléatoirement. VanetMobiSim permet de générer les parcours en fonction de points (ou de zone) d'attractions ou de répulsions. Le parcours entre le point de départ et d'arrivée peut être configuré sur la base de l'algorithme de « Dijsktra » (le plus court chemin) ou en fonction des routes les plus rapides ou encore en fonction de la densité du trafic. Tout comme son prédécesseur, VanetMobiSim permet de générer des résultats dans différents formats. Ce simulateur offre donc de bonnes performances, une flexibilité vis-à-vis du format des résultats pour le simulateur de réseau Ad Hoc et une prise en main rapide et intuitive.

4.4. Paramètres d'évaluation

Pour étudier l'impact du taux des messages CPM sur les performances des paramètres de qualité de service (QoS), je me suis basé sur 5 métriques :

4.4.1. Taux de paquets délivrés : PDR (Packet Delivery Ratio)

C'est le rapport, multiplié par 100, entre le nombre de paquets de données reçus par les destinations et le nombre de données émis par les sources.

Si ce taux est élevé, alors le taux de perte des paquets est faible et par suite, le protocole de routage utilisé offre des routes valides entre la source et la destination. Les causes de pertes de paquets sont nombreuses, nous citons par exemple la collision, la surcharge des files d'attente, les interférences, absence de routes vers la destination, les routes inactives, etc.

- **Le délai**

Le délai de bout en bout est le temps écoulé entre l'envoi d'un paquet par un émetteur et sa réception par le destinataire.

- **La gigue**

C'est la différence du délai de deux paquets successivement reçus appartenant au même flux de données.

- **Cout de routage**

Le cout du routage évalue le surcout induit par l'échange des messages de routage dans le réseau en présence d'un trafic applicatif.

Il est défini par le rapport entre le nombre total de paquets de contrôle émis ou transférés et le nombre total de paquets de données reçus avec succès par la destination.

- **Efficacité**

Cette métrique mesure l'efficacité du protocole de routage. Elle est définie par le rapport entre le nombre de paquets livrés qui sont transmis (les paquets de données) et (les paquets de données + les paquets de routage).

4.5. Scénario de simulation

4.5.1. Le modèle d'attaque :

Les simulations ont été effectuées afin d'étudier l'impact du paramètre cpmrate (le taux CPM) sur les performances des paramètres à QoS (PDR, délai, gigue, cout de routage, efficacité).

Nous avons considéré un attaquant actif qui injecte dans le réseau des paquets contenant des fausses informations de routage (link spoofing) pour perturber le fonctionnement du protocole de routage. Nous avons supposé que l'attaquant ne peut ni usurper l'identité des autres nœuds (l'utilisation d'une autorité de certification distribuée) ni retransmettre les anciens messages dans le réseau (l'utilisation de timestamp).

Cet attaquant effectue 2 types d'attaques :

Faux messages Hello : Le nœud malicieux oblige ses voisins à le choisir comme relai multipoint en déclarant dans ses messages HELLO une fausse relation de voisinage avec ses voisins à 2 sauts, ce qui va lui permettre d'empêcher les messages envoyés par les nœuds attaqués d'arriver à leurs destinations.

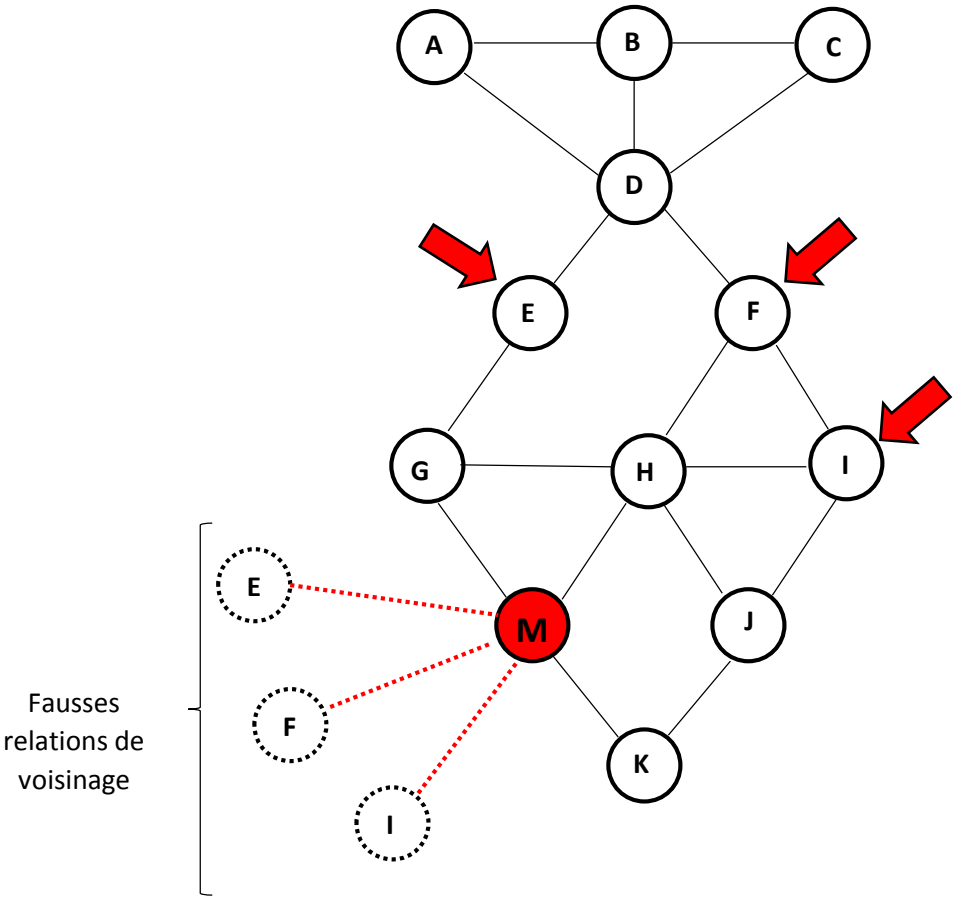


Figure 4.3 : Insertion de faux message HELLO

Faux messages TC : Le nœud malicieux effectue une attaque de type link spoofing en choisissant au hasard un ou plusieurs nœuds éloignés du réseau (voisins à 3 sauts et plus) et annoncer une connectivité directe avec eux, ce qui va favoriser une perte de connectivité.

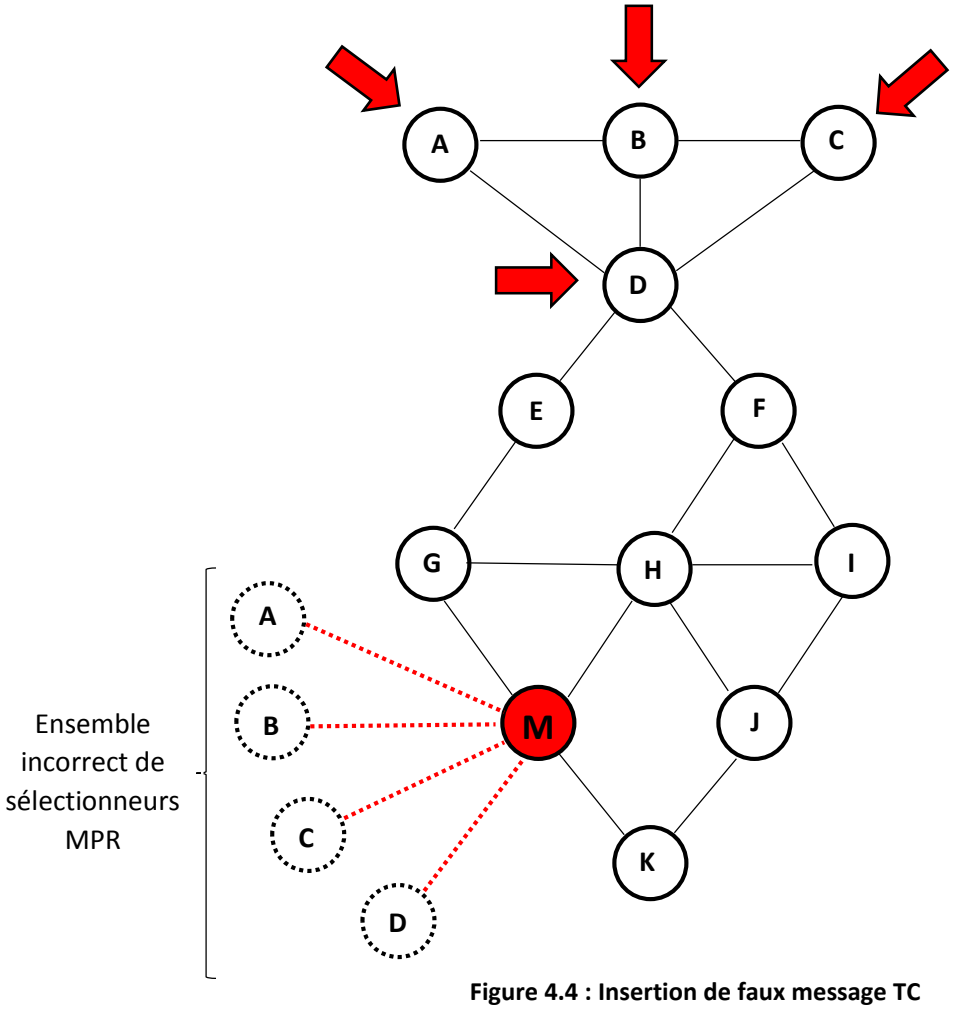


Figure 4.4 : Insertion de faux message TC

Nous avons supposé que l'attaquant est actif à partir de 50 à 100 secondes du temps de simulation, et on a testé les deux types d'attaques séparément. Le deuxième type a été testé en générant des messages avec un seul lien incorrect, et en générant des messages avec quatre liens incorrects.

4.5.2. La sécurité avec CSS-OLSR :

Pour détecter un nœud du réseau comme malicieux et définir le type d'attaque (Faux message HELLO/TC), La solution CSS-OLSR ajoute les éléments suivants :

- ✓ Dans le message TC :
  - Le champ id : pour identifier le message, il combine les deux champs de numéro de séquence et adresse origine.
  - Le champ path : pour sauvegarder le chemin parcouru par le message TC.
- ✓ La structure Nids : pour sauvegarder l'état du nœud lors de l'envoi de message TC.
- ✓ Le message CPM : qui contient le champ id et le champ path comme le message TC.

4.5.2.1. Procédure de CSS-OLSR :

Chaque nœud du réseau envoie régulièrement en diffusion un message TC à destination de l'ensemble de ses voisins à un saut, et seuls ses relais multi points qui vont le relayer. Lors de l'envoie, la source du message TC doit enregistrer dans sa structure Nids : ses nœuds MPR, l'identité du message TC ainsi que ses voisins à 2 sauts. Le message TC va être relayé par les nœuds MPR et à chaque passage le champ path enregistre le chemin parcouru.

Pour qu'il y ait un retour de message CPM, deux principales conditions doivent être satisfaites :

- ✓ Le nombre de saut dans le message TC doit être supérieur ou égale à 4.
- ✓ Le dernier nœud de chemin de message TC ne doit pas être voisin ni de premier ni de deuxième nœud dans le même chemin.

Finalement, c'est à partir du message CPM retourné et la structure Nids que nous pouvons détecter le nœud malicieux.

4.5.2.2. Détection du nœud malicieux :

Faux message HELLO :

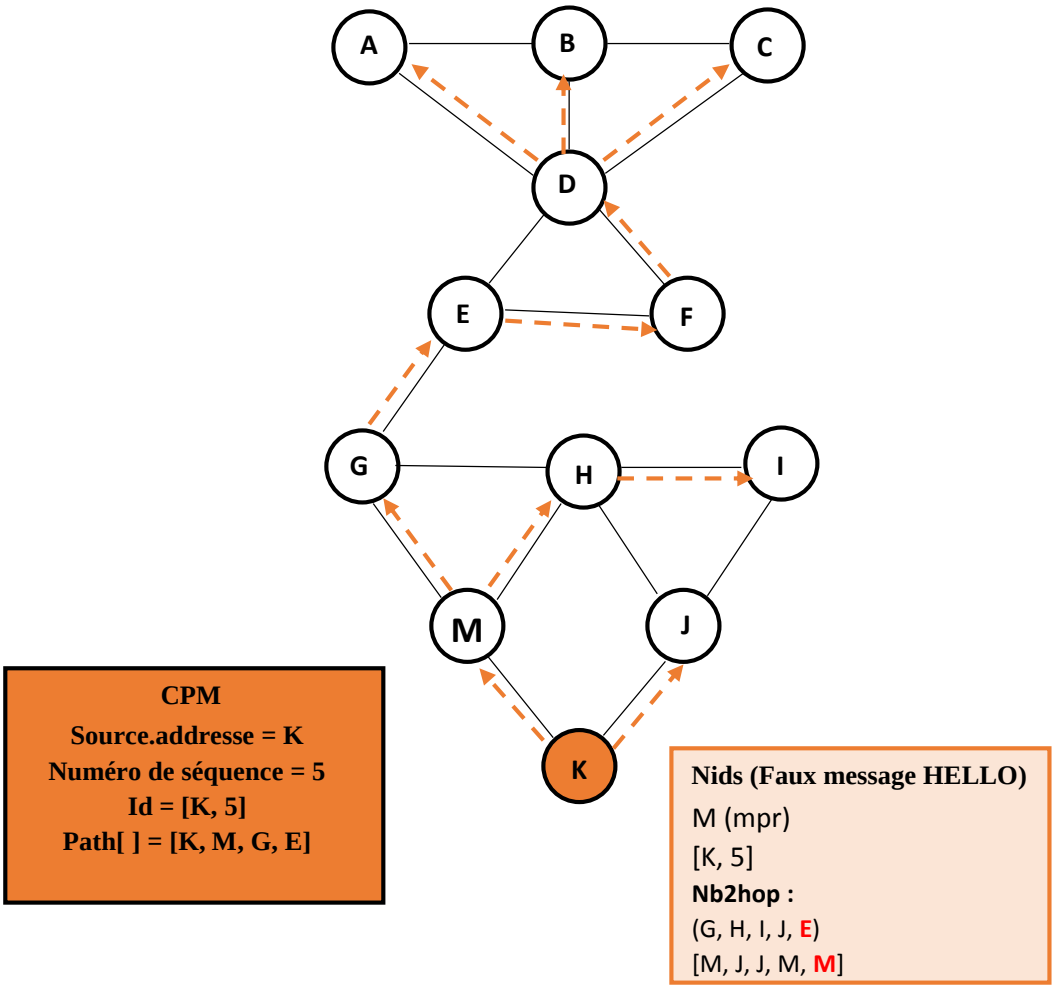


Figure 4.5 : Détection de faux message HELLO

Le message CPM a retourné comme chemin : « KMGE » c'est-à-dire que « E » n'est pas voisin ni de « K » ni de « M » (deuxième condition de retour de message CPM).
Dans la figure au-dessus, on a supposé que le nœud « M » est le nœud malicieux, donc il va envoyer un faux message Hello dans lequel il déclare le nœud « E » comme voisin, et « K » va enregistrer le nœud **E** avec l'ensemble de ses voisins à deux sauts.
A la réception du message CPM, le nœud « K » générant le message TC compare les informations contenues dans le message CPM avec celles stockés dans sa structure Nids :
D'Après le message CPM, le nœud « E » n'est pas voisin ni de « k » ni de « M », Or d'après la structure Nids « E » est un voisin de « M ». Cette contradiction va permettre à la solution CSS-OLSR de détecter le nœud malicieux « M » et définir le type d'attaque.

Faux message TC :

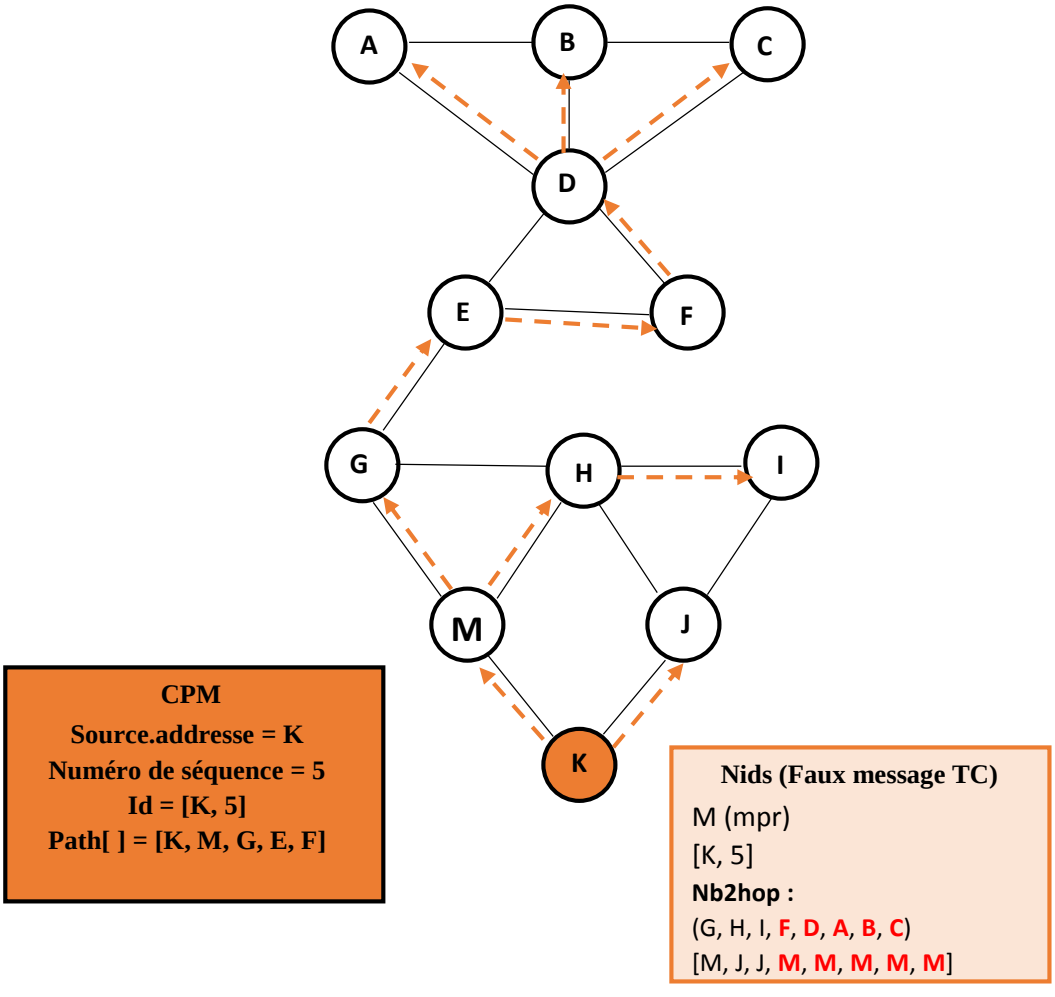


Figure 4.6 : Détection de faux message TC

Pour la détection de faux messages TC, c'est presque le même principe que la détection de faux message Hello : le nœud malicieux « M » déclare ses voisins à 3 sauts et plus comme MPR selectors, et le nœud « K » générant le message TC enregistre ces informations dans sa structure Nids afin de les comparer avec les informations qui seront retournés dans le message CPM, ce qui va lui permettre par la suite de détecter le nœud malicieux « M » et définir le type d'attaque.

4.6. Etapes de simulation

4.6.1. Modèle de mobilité

Pour créer une topologie appropriée au scénario désiré, on a saisi au format XML à l'aide des balises l'ensemble des paramètres de mobilité nécessaires. Ensuite, on a lancé l'exécution de simulateur VanetMobisim[34] en générant un fichier de mobilité au format NS2 qui sera intégré par la suite dans le script TCL (voir Annexe 3). Ce fichier de mobilité a été généré à chaque variation du nombre des nœuds dans le fichier xml (voir Annexe 2).

Paramètre	Valeur
Aire du réseau Ad hoc	1000 m x 1000 m
Temps de simulation	300s
Panneaux de signalisation	6
Nombre de voies	2
Temps de pause	[5,30] seconds
Nombre de nœuds	25, 50, 75
Vitesse des nœuds	20 km/h __ 100km/h
Le modèle utilisé	IDM_LC

Tableau 4.1 : Paramètres utilisés pour le modèle de mobilité

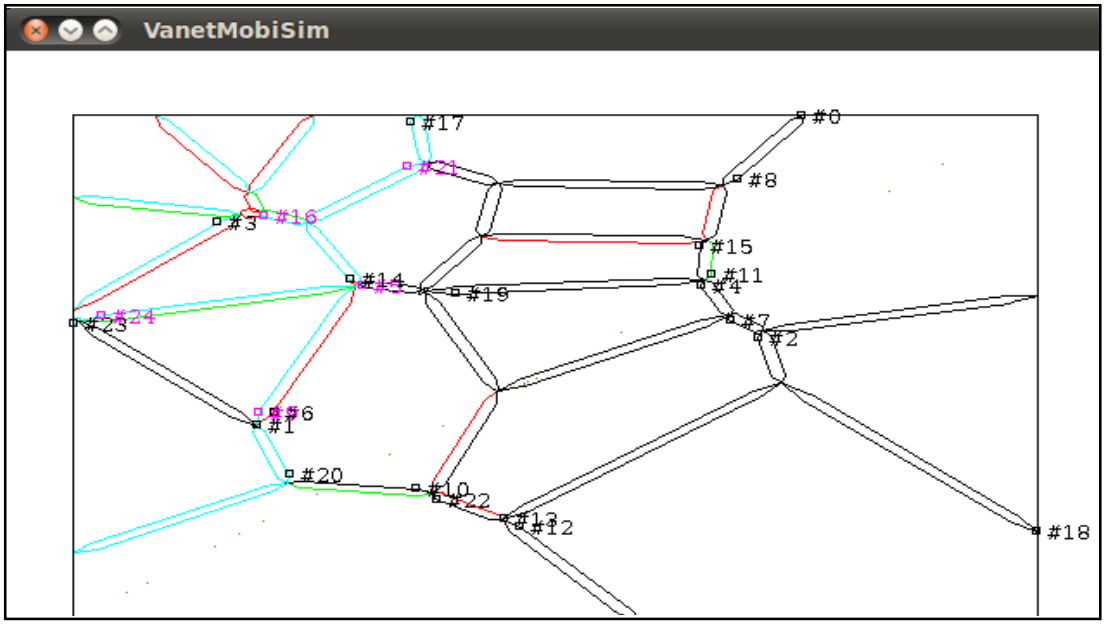


Figure 4.7 : Visualisation de la mobilité des nœuds par VanetMobisim

Les paramètres de la couche physique et liaison de donnée utilisés dans les simulations sont résumés dans le Tableau au-dessous :

Paramètre	Valeur
Protocole MAC	IEEE 802.11p
Modèle de réflexion	Two-ray ground[35]
Portée de communication	250 m
Taille max des files d'attente	50 paquets

Tableau 4.2 : Paramètres utilisés pour la couche physique

4.6.2. Modèle de trafic

Avant de lancer l'exécution du script TCL, un fichier de trafic doit être généré pour chaque nombre de nœuds choisi et injecté dans le script.

Le trafic entre les nœuds est produit en utilisant un générateur de trafic (cbrgen) qui crée aléatoirement des connexions de type CBR qui commencent à des instants distribués uniformément entre 10 et 300 secondes. La taille des paquets de données est 512 octets. Le taux d'émission des paquets est fixé à 8 paquets par seconde et le nombre de connexions à 15.

Paramètre	Valeur
Type du trafic	CBR
Nombre de connexions	15 connexions
Taux de transmission	8 paquets/second
Taille des paquets	512 octets
Nombre de nœuds	25, 50, 75 nœuds

Tableau 4.3 : Paramètres utilisés pour le modèle de trafic

4.6.3. Simulation réseau

Une fois les fichiers de mobilité et les fichiers de trafic générés. On a lancé la simulation de notre scénario (voir Annexe 3) trente fois pour chaque type d'attaque, en variant le taux CPM (CPMrate) dix fois pour chaque nombre de nœuds choisi (25, 50, 75). Le tableau au-dessous résume les paramètres du protocole CSS-OLSR utilisés pour effectuer les simulations.

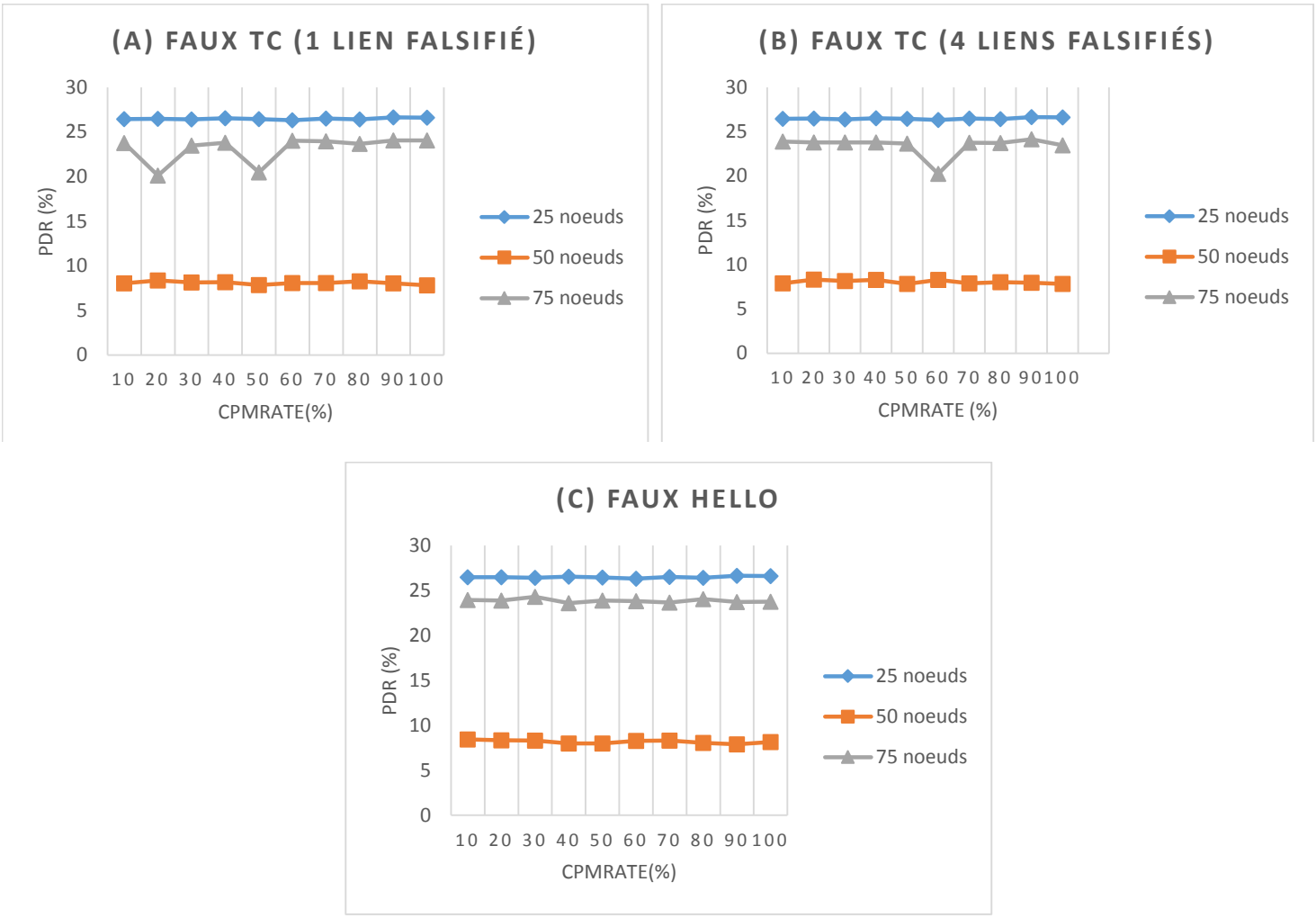
Paramètre	Valeur
CPMrate	10, 20, 30, 40, 50, 60, 70, 80, 90, 100 %
Primary rating	100
Secondary rating	100
Secondary rating increase	1
Secondary rating decrease	2

Tableau 4.4 : Paramètres de CSS-OLSR

Pendant la simulation, les informations des flux de paquets ont été collectées à travers deux types de fichiers traces d'extensions .tr et .nam. Après la simulation, une phase nommée Post-processing est utilisée. Elle consiste, d'une part, à analyser les fichiers traces d'extension .tr pour extraire les paramètres de performance avec un programme écrit en AWK (voir Annexe 4). Et d'autre part, à lancer l'animation du réseau à partir des fichiers d'extension .nam en utilisant le programme NAM (Network AniMator) intégré dans NS-2. Ce dernier permet de visualiser le déplacement des nœuds, le parcours des paquets dans le réseau et l'état des files d'attente des nœuds.

4.7. Résultats et Analyse

4.7.1. Courbes de visualisation du PDR



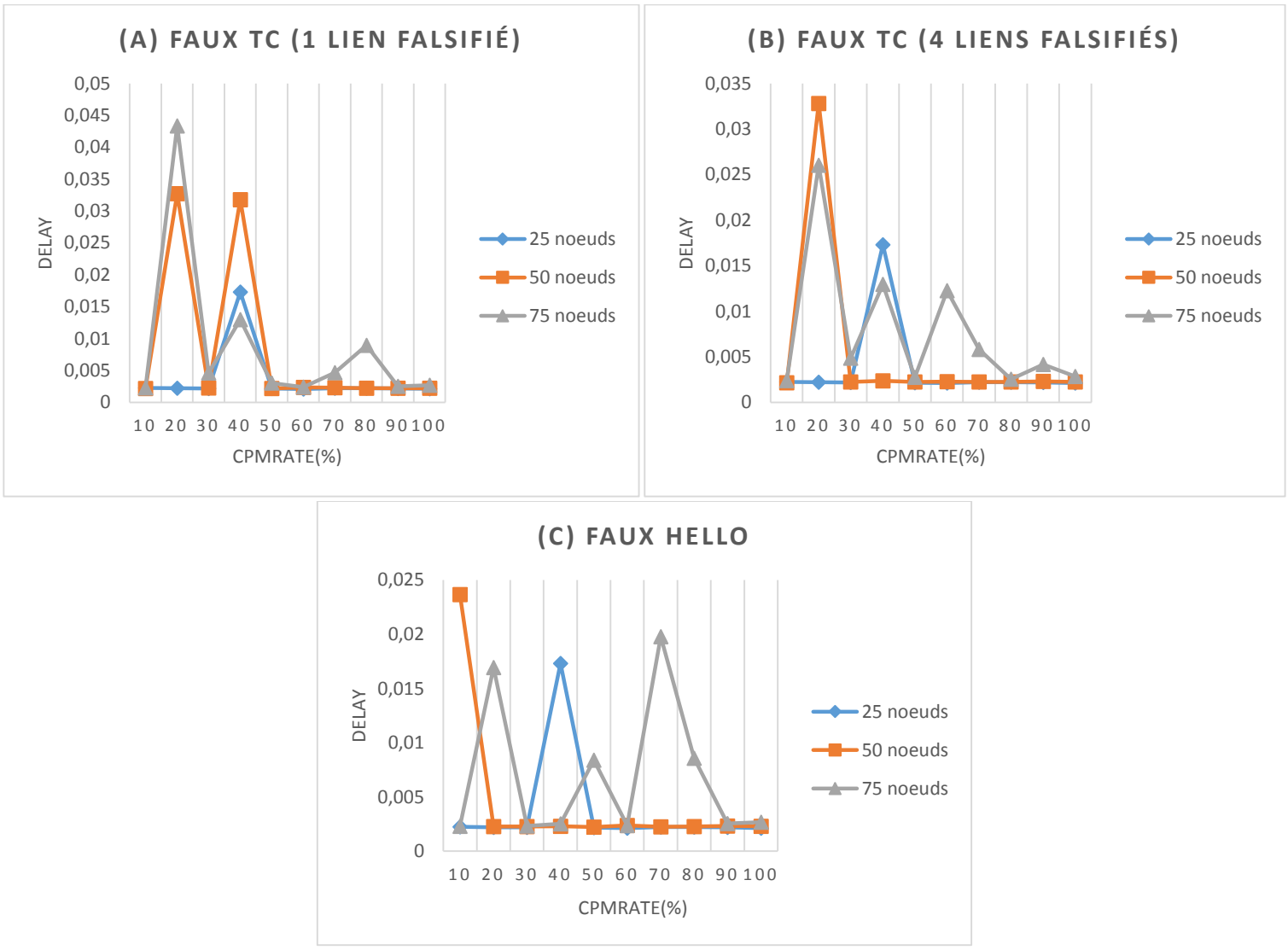
Figures 4.8 : Taux de paquets délivrés

Les figures 4.8 montrent le taux de paquets délivrés pour trois modèles de mobilité qui diffèrent en termes de nombre de nœuds constituant le réseau pour deux types d'attaques (Faux TC et Faux HELLO).

D'après les figures, on remarque que la variation du taux CPMrate n'a aucune influence sur le taux de paquets délivrés pour les deux types d'attaques quel que soit la densité des nœuds.

Donc pour avoir un taux important de paquets délivrés, on peut choisir n'importe quelle valeur du taux CPM dans le cas des attaques de type « Faux HELLO », et éviter de choisir les valeurs 20%, 50% ou 60% dans le cas des attaques de type « Faux TC » puisqu'elles donnent des taux un peu plus faibles dans un réseau constitué de 75 nœuds. La diminution brusque du PDR pour ces valeurs de CPMrate dans les scénarios d'attaques de type « Faux TC » peuvent être expliquées par la suppression de certaines messages CPM par l'attaquant, Ou bien les nœuds sélectionnant l'attaquant comme MPR dans ces cas ne sont pas des relais multipoint, donc ils n'ont pas généré des messages TC sur lesquels se base la solution CSS-OLSR pour détecter les nœuds malicieux.

4.7.2. Courbes de visualisation de Délai



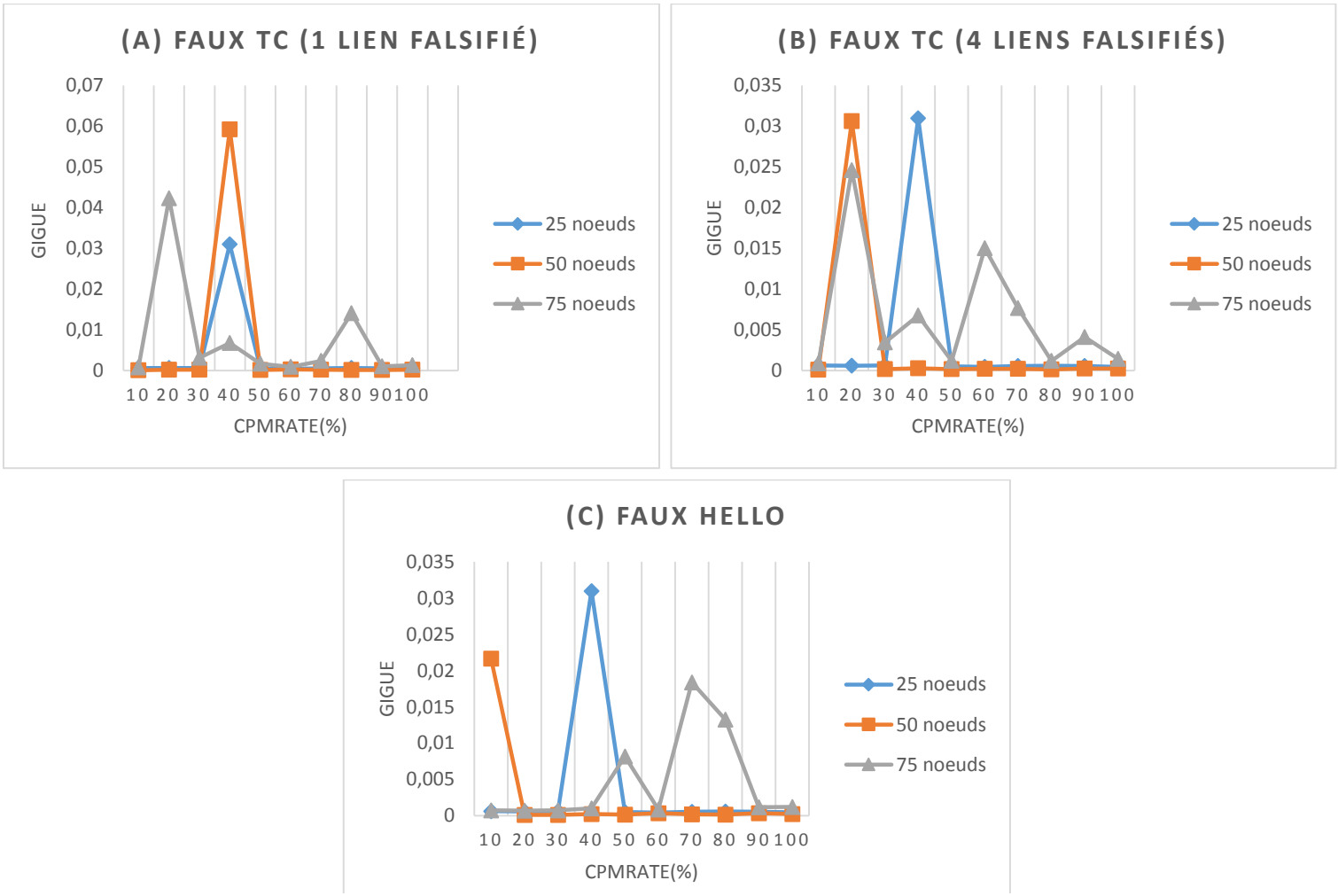
Figures 4.9 : Le délai de bout en bout

On commence notre analyse par les attaques de type « Faux TC » représentées dans les figures 4.9 (a) et (b) en comparant les graphes obtenus respectivement avec ceux du taux de paquets délivrés représentés dans les figures 4.8 (a) et (b) et chercher les valeurs de CPMrate qui donnent simultanément les meilleurs Taux de paquets délivrés et les plus petits délais pour les trois modèles de mobilités choisis.
On constate que les taux CPMrate permettant d'avoir simultanément des valeurs acceptables de PDR et du délai sont 10%, 90%, 100%.
Si on fait la même analyse pour les attaques de type « Faux HELLO », on constate que les taux CPMrate permettant d'avoir simultanément des valeurs acceptables de PDR et du délai pour les trois modèles de mobilité sont 30%, 60%, 90%, 100%.

On peut expliquer la grande variabilité des valeurs du délai d'un taux CPMrate à l'autre pour la même densité des nœuds (nombre de nœuds) par le faite que la solution CSS-OLSR engendre

parfois des délais plus long pour détecter les comportements malhonnêtes et bloquer ses initiateurs.

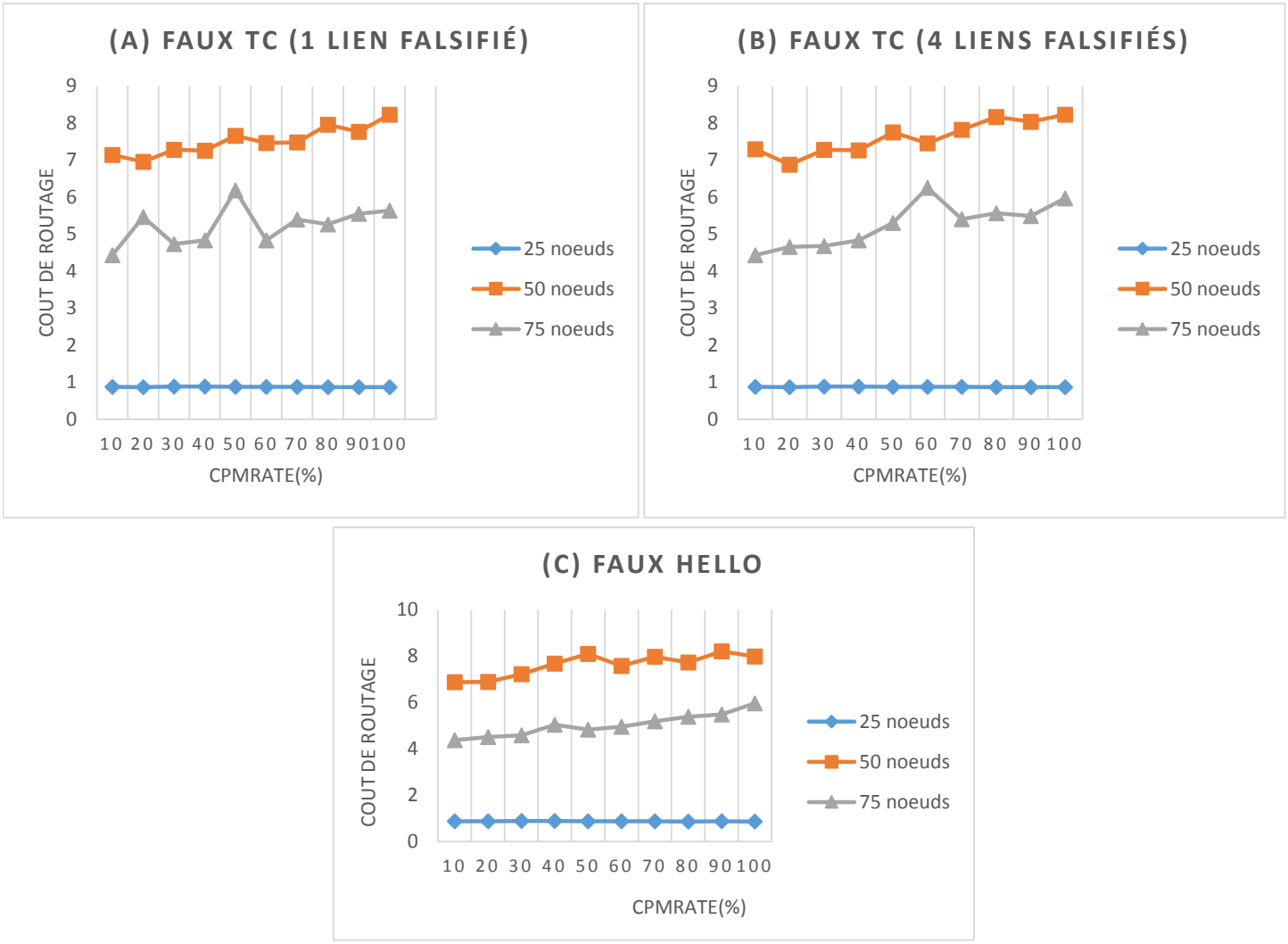
4.7.3. Courbes de visualisation de la gigue



Figures 4.10 : la gigue

La même chose que pour le délai, et en prenant en compte l'analyse des graphes de figures 4.9, on remarque que les taux CPMrate qui donnent les meilleures valeurs de gigue pour les trois densités sans causer une dégradation des performances du réseau sont 10%, 90%, 100% pour le cas des attaques de type Faux TC et 30%, 60%, 90%, 100% pour le cas des attaques de type Faux HELLO.

4.7.4. Courbes de visualisation du cout de routage



Figures 4.11 : Le cout de routage

Les trois figures au-dessus représentent la variation du cout de routage en fonction de nombre des messages CPM pour les deux types d'attaques (Faux TC et Faux HELLO) en utilisant trois modèles de mobilité.

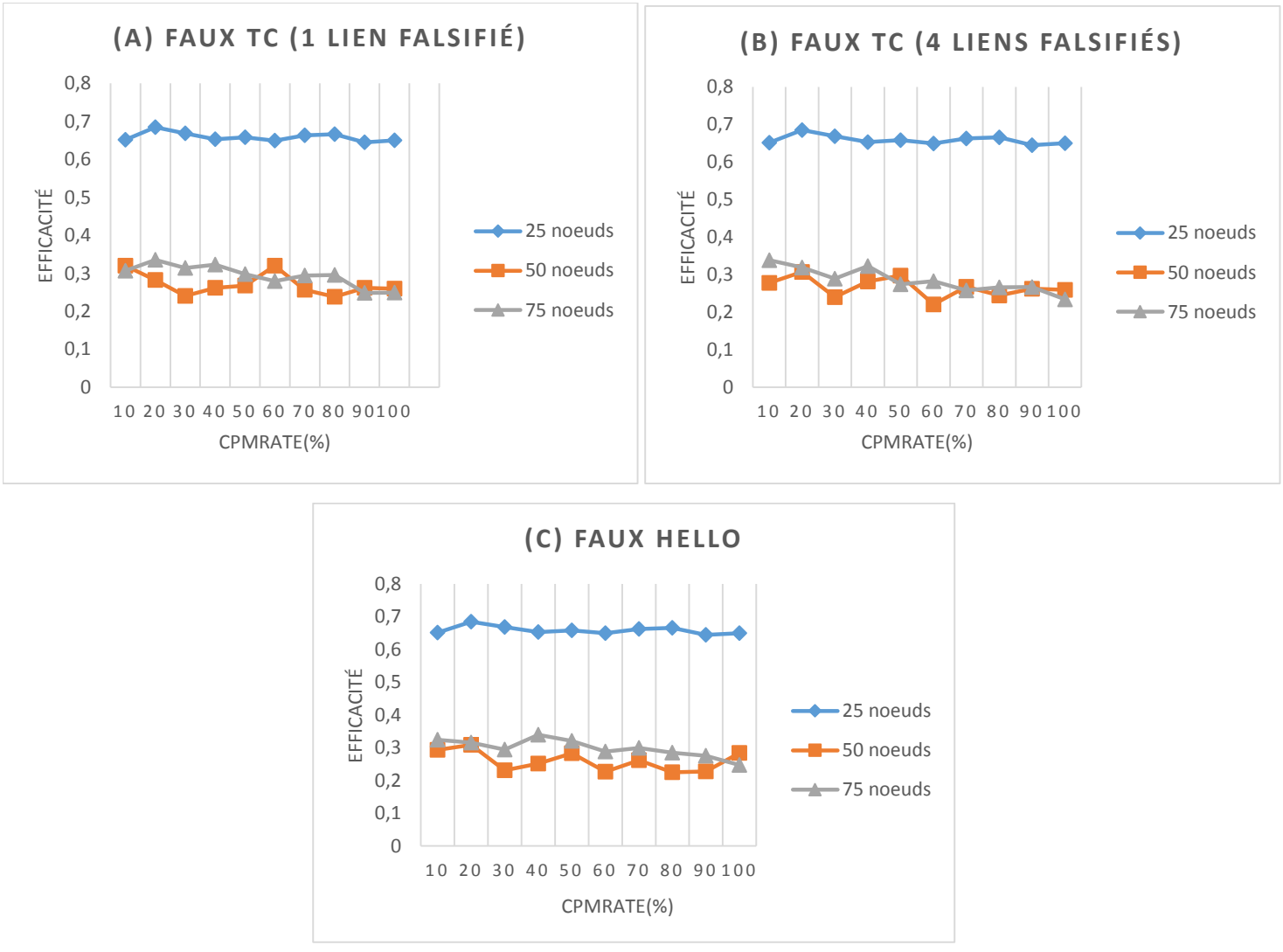
Pour les attaques de type « Faux TC », des dégradations du cout de routage sont remarquées pour un taux CPMrate égale à 20% dans un réseau constitué de 75 nœuds dans le cas où un seul lien incorrect est déclaré dans le message TC, et quand le taux des messages CPM envoyés dépasse 40% pour les réseaux constitués de 50 et 75 nœuds.
Dans le cas des attaques de type « Faux HELLO », une dégradation du cout de routage est remarquée quand le taux des messages CPM envoyés dépasse 30% dans les réseaux constitués de 50 et 75 nœuds.

Donc, et d'après les analyses précédentes, on peut conclure que les valeurs de CPMrate qui donnent un cout de routage acceptable sans dégrader les performances des autres paramètres de qualité de service sont 10% pour les attaques de type « Faux TC » et 30% pour les attaques de type « Faux HELLO ».

4.7.5. Courbes de visualisation de l'Efficacité

D'après les études qu'on a fait précédemment pour étudier l'impact du paramètre cpmrate sur les performances du PDR, le délai, la gigue et le cout de routage, on a trouvé que les taux CPMrate qui donnent des meilleurs résultats par rapport aux autres sont 10% pour les attaques de type Faux TC et 30% pour les attaques de type Faux HELLO.

Les figures 4.12 représentent la variation de l'efficacité en fonction de nombre des messages CPM envoyés pour les deux types d'attaques (Faux TC et Faux HELLO) en utilisant trois modèles de mobilité, Et montrent que les taux CPMrate 10% pour les attaques de type « Faux TC » et 30% pour l'attaque de type « Faux HELLO » donnent des valeurs acceptables d'efficacité par rapport aux autres taux.



Figures 4.12: L'Efficacité

Pour conclure, afin d'étudier l'impact du taux CPMrate sur les performances des paramètres de qualité de service, on a cherché au début les taux qui donnent des valeurs du PDRs plus élevés. Ensuite, on a essayé étape par étape, d'éliminer ceux qui dégradent les performances des autres paramètres jusqu'à trouver le taux qui serait le meilleur choix pour chaque type d'attaque (10% pour Faux TC et 30% pour Faux HELLO).

Conclusion

Les réseaux ad-hoc véhiculaires constituent un nouveau type de réseaux issus des réseaux ad-hoc mobiles (MANETs). Leur particularité vient de ce qu'ils permettent la communication entre les véhicules sur les routes. Cette communication peut être réalisée autant dans une topologie à infrastructure que dans une topologie ad-hoc dans laquelle les nœuds communiquent les uns avec les autres sans besoin de tiers d'aide à la communication.

Grace aux informations échangées, les véhicules peuvent s'informer les uns des autres sur l'état de la route et ainsi éviter les situations malencontreuses aux automobilistes. L'implémentation des VANETs donne lieu à d'autres applications telles que la maintenance à distance, le payement et l'accès à des services à distance. Il est également possible de penser à des applications de confort telles que les jeux en réseaux, le téléchargement d'applications multimédia comme la musique ou les séquences vidéo. En d'autres termes, les VANETs permettent de rendre le système de transport plus fiable et sécuritaire tout en permettant aux usagers de la route de voyager dans des conditions agréables.

Dans ce travail de recherche, nous nous sommes intéressés aux menaces et aux problèmes de sécurité dans les réseaux VANET, et plus particulièrement à la sécurité des communications véhiculaires au niveau de la couche réseau du modèle OSI. En effet, les informations échangées entre les véhicules sont des données informatiques susceptibles de subir à tout type d'attaque. Que ce soit une attaque sur les données partagées en modifiant les informations par exemple, soit en écoutant et en collectant les informations à propos des usagers de façon à les utiliser par la suite à mauvais escient, soit en injectant des informations erronées dans le réseau de façon à modifier le comportement des usagers de la route, soit encore en empêchant la communication entre les véhicules par une attaque de déni de service par exemple.

Dans ce mémoire, la plupart des solutions de sécurisations des protocoles de routage étudiées sont basées sur des primitives cryptographiques (chiffrement, signature, etc.) pour assurer les propriétés de base de la sécurité (confidentialité, intégrité, non répudiation, etc.). Cependant, ces primitives sont souvent utilisées comme un moyen de prévention et constituent souvent le premier rempart de sécurité du réseau. Toutefois, quand l'attaquant est à l'intérieur du réseau et corrompt des nœuds légitimes afin qu'ils se comportent de manière malfaisante ou quand la cryptographie est considérée trop coûteuse, il faudra alors penser à d'autres solutions réactive permettant de réagir selon le comportement des nœuds voisins.

Le protocole CSS-OLSR sur lequel nous avons travaillé, est une extension sécurisée du protocole OLSR basée sur un mécanisme de réputation qui évalue l'intégrité du trafic de contrôle en corrélant les informations locales de routage avec des messages retournés par les destinations du trafic de contrôle. Cette évaluation permet de détecter les nœuds malicieux d'une manière fiable et les pénaliser en réduisant leurs capacités à communiquer à travers le réseau.

Pour étudier l'impact du taux des messages retournés sur les performances des paramètres de la qualité de service (QoS) dans un environnement VANET, nous avons fait une implémentation plus proche des conditions réelles sous ns2 dans laquelle nous avons considéré un attaquant qui peut injecter des faux messages HELLO/TC dans le réseau. La conclusion et le résultat auquel on a abouti montrent que les taux de messages de retour qui ont un niveau d'impact acceptable par rapport aux autres sur les performances des paramètres de la qualité de service sont 30% pour le mécanisme de détection de faux messages HELLO et 10% pour le mécanisme de détection de faux messages TC.

Comme perspective à ce travail, nous envisageons de comparer le protocole CSS-OLSR avec le protocole sécurisé réactif TOLSR au niveau de leur impact sur les performances des paramètres de la qualité de service et de leur rapidité de détection des nœuds malveillants dans un environnement VANET.

Références

[1] N. Bouchemal, R. Naja, et S. Tohme, « Traffic Modeling and Performance Evaluation in Vehicle to Infrastructure 802.11p Network », in *Ad Hoc Networks*, M. H. Sherif, A. Mellouk, J. Li, et P. Bellavista, Éd. Springer International Publishing, 2013, p. 82‑99.

[2] B. Paul, M. Ibrahim, M. Bikas, et A. Naser, « VANET Routing Protocols: Pros and Cons », *ArXiv Prepr. ArXiv12041201*, 2012.

[3] S. Giordano et I. Stojmenovic, « Position Based Routing Algorithms for Ad Hoc Networks: A Taxonomy », in *Ad Hoc Wireless Networking*, X. Cheng, X. Huang, et D.-Z. Du, Éd. Springer US, 2004, p. 103‑136.

[4] L. K. Qabajeh, M. L. M. Kiah, et M. M. Qabajeh, « A scalable and secure position-based routing protocol for ad-hoc networks », *Malays. J. Comput. Sci.*, vol. 22, n° 2, p. 100, 2009.

[5] S. Carter et A. Yasinsac, « Secure position aided ad hoc routing », 2003.

[6] E. Amar et S. Boumerdassi, « A Location Service for Position-based Routing in Mobile Ad Hoc Networks », in *Proceedings of the 8th International Conference on New Technologies in Distributed Systems*, New York, NY, USA, 2008, p. 48:1–48:4.

[7] T. Clausen, P. Jacquet, C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, et L. Viennot, « Optimized Link State Routing Protocol (OLSR) », 2003.

[8] E. W. Dijkstra, « A note on two problems in connexion with graphs », *Numer. Math.*, vol. 1, n° 1, p. 269‑271, déc. 1959.

[9] C. Perkins, E. Belding-Royer, et S. Das, « Ad hoc On-Demand Distance Vector (AODV) Routing », RFC Editor, RFC3561, juill. 2003.

[10] M. Razzaque, A. Salehi, et S. M. Cheraghi, « Security and privacy in vehicular ad-hoc networks: survey and the road ahead », in *Wireless Networks and Security*, Springer, 2013, p. 107‑132.

[11] M. K. Nasir, A. D. Hossain, M. S. Hossain, M. M. Hasan, et M. B. Ali, « Security challenges and implementation mechanism for vehicular ad hoc network », *Int. J. Sci. Technol. Res.*, vol. 2, n° 4, p. 156‑161, 2013.

[12] T. Clausen, « Comparative Study of Routing Protocols for Mobile Ad-hoc networks », INRIA, report, 2004.

[13] T. Clausen, A. Laouiti, P. Muhlethaler, D. Raffo, et C. Adjih, « Securing the OLSR routing protocol with or without compromised nodes in the network », INRIA, Research Report RR-5494, 2005.

[14] B. Kannhavong, H. Nakayama, N. Kato, Y. Nemoto, et A. Jamalipour, « Analysis of the node isolation attack against OLSR-based mobile ad hoc networks », in *2006 International Symposium on Computer Networks*, 2006, p. 30‑35.

[15] P. Ning et K. Sun, « How to misuse AODV: a case study of insider attacks against mobile ad-hoc routing protocols », *Ad Hoc Netw.*, vol. 3, n° 6, p. 795‑819, 2005.

[16] (Prénom) C.BURGOD, « Contribution la sécurisation du routage dans les réseaux ad hoc », 2009.

[17] P. Michiardi, « Mécanismes de sécurité et de coopération entre nœuds d'un réseaux mobile ad hoc », 2004.

[18] L. Zhou et Z. J. Haas, « Securing ad hoc networks », *Netw. IEEE*, vol. 13, n° 6, p. 24‑30, 1999.

[19] M. A. Ayachi, « Contributions à la détection des comportements malhonnêtes dans les réseaux ad hoc AODV par analyse de la confiance implicite », 2011.

[20] C. Adjih, T. Clausen, A. Laouiti, P. Muhlethaler, et D. Raffo, « Securing the OLSR routing protocol with or without compromised nodes in the network », *HIPERCoM Proj. INRIA Rocquencourt Tech Rep INRIA RR-5494*, 2005.

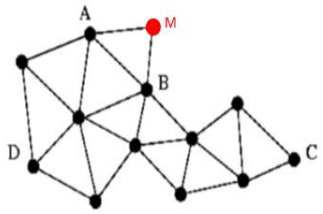
[21] A. Hafslund, A. Tønnesen, R. B. Rotvik, J. Andersson, et Ø. Kure, « Secure Extension to the OLSR protocol », présenté à Proceedings of the OLSR Interop and Workshop, San Diego, 2004.

[22] C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, et D. Raffo, « Securing the OLSR protocol », présenté à Proceedings of Med-Hoc-Net, 2003, p. 25‑27.

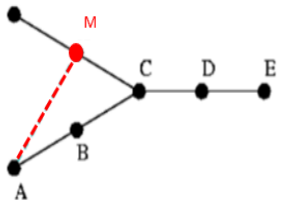
[23] D. Raffo, C. Adjih, T. Clausen, et P. Mühlethaler, « An Advanced Signature System for OLSR », in *Proceedings of the 2Nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, New York, NY, USA, 2004, p. 10–16.
[24] D. R. C. A. T. Clausen et P. Mühlethaler, « OLSR with GPS information ».
[25] D. Raffo, « Security Schemes for the OLSR Protocol for Ad Hoc Networks », phdthesis, Université Pierre et Marie Curie - Paris VI, 2005.
[26] J. P. Vilela et J. Barros, « A cooperative security scheme for optimized link state routing in mobile ad-hoc networks », *Proc. 15th IST Mob. Wirel. Commun. Summit Mykonos Greece*, 2006.
[27] D. Dhillon, T. S. Randhawa, M. Wang, et L. Lamont, « Implementing a fully distributed certificate authority in an OLSR MANET », présenté à Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE, 2004, vol. 2, p. 682‑688.
[28] M. G. Zapata, « Secure Ad Hoc On-demand Distance Vector Routing », *SIGMOBILE Mob Comput Commun Rev*, vol. 6, n° 3, p. 106–107, juin 2002.
[29] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, et E. M. Belding-Royer, « A secure routing protocol for ad hoc networks », in *10th IEEE International Conference on Network Protocols, 2002. Proceedings*, 2002, p. 78‑87.
[30] Q. Li, M. Zhao, J. Walker, Y.-C. Hu, A. Perrig, et W. Trappe, « SEAR: a secure efficient ad hoc on demand routing protocol for wireless networks », *Secur. Commun. Netw.*, vol. 2, n° 4, p. 325‑340, juill. 2009.
[31] A. Perrig, R. Canetti, J. D. Tygar, et D. Song, « The TESLA broadcast authentication protocol », *RSA CryptoBytes*, vol. 5, 2005.
[32] T. Issariyakul et E. Hossain, *Introduction to network simulator NS2*. Springer Science & Business Media, 2011.
[33] J. Härri, M. Fiore, F. Filali, et C. Bonnet, « Vehicular mobility simulation with VanetMobiSim », *Simulation*, 2009.
[34] J. Harri et M. Fiore, « VanetMobiSim–Vehicular Ad hoc Network mobility extension to the CanuMobiSim framework », *Inst. Eurécom Dep. Mob. Commu*, vol. 6904, 2006.
[35] C. Sommer, S. Joerer, et F. Dressler, « On the applicability of two-ray path loss models for vehicular network simulation », présenté à Vehicular Networking Conference (VNC), 2012 IEEE, 2012, p. 64‑69.

Annexe 1

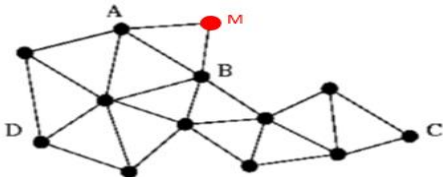
Attaques spécifiques au protocole OLSR

<table>
<tr><th colspan="4">1. Génération incorrecte du trafic</th></tr>
<tr><th colspan="2">Génération incorrect des HELLOs</th><th colspan="2">Génération incorrect des TCs</th></tr>
<tr><th>Identity spoofing</th><th>link spoofing</th><th>Identity spoofing</th><th>link spoofing</th><th rowspan="2"></th></tr>
<tr>
<td> 

▪ M envoie des messages avec C comme origine
➡ A et B vont annoncer leur voisinage avec C

▪ M choisit A et/ou B comme ses MPRs avec l'identité de C
➡ Ces MPRs vont déclarer qu'ils peuvent fournir connectivité vers C
➡ Conflits des routes vers C, perte de connectivité.</td>
<td> 

▪ M déclare un lien symétrique avec A
➡ C choisi comme son MPR set probablement {M,D} au lieu de {M, B,D}
➡ Les messages de E ne vont pas joindre A

▪ M ne déclare pas tous ses voisins
➡ Perte de connectivité avec les voisins ignorés</td>
<td> 

▪ M envoie un message avec l'identité de C et déclarant A comme voisin

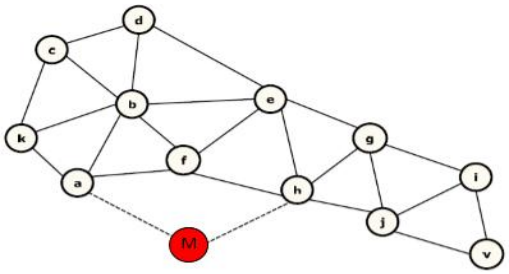
➡ Topologie erronée</td>
<td>▪ M envoie un message déclarant D comme voisin

➡ Topologie erronée</td>
<td>▪ Un nœud malveillant choisi comme MPR peut refuser de générer les messages TC ou déclarer dans ses TC un ensemble incomplet de voisins qui l'ont choisi comme MPR

➡ Topologie non diffusée</td>
</tr>
<tr><th colspan="4">2. Relayage incorrect du trafic</th></tr>
<tr><th>Relayage incorrect du trafic de contrôle</th><th>Attaque par retransmission des messages de contrôle</th><th>Attaque du blackhole</th><th colspan="2">Attaque Wormhole</th></tr>
<tr>
<td>▪ Un nœud malveillant peut être choisi comme MPR légitime par d'autres nœuds mais il refuse de relayer les messages TC des autres MPR.

➡ Perte de connectivité de certains nœuds (Dans le cas où il n'existe pas de routes qui ne passe pas par le nœud malicieux)</td>
<td>▪ Un nœud malicieux peut renvoyer à d'autres nœuds des messages de contrôle (TC ou HELLO) déjà envoyés dans le passé par d'autres nœuds qu'il a pu écouter à travers le réseau.
▪ Nécessite de changer MSN(HELLO ou TC) et/ou ANSN(TC)

➡ Perte des messages selon leurs MSN/ANSN</td>
<td>▪ Un nœud malicieux qui a été choisi par ses voisins comme MPR peut rejeter tous les paquets de données reçus de ses voisins.

➡ Perte de connectivité et la dégradation de la communication.</td>
<td colspan="2"> 

▪ M relaye les messages entre a et h
▪ Création d'un faux lien sous le contrôle de M

➡ Topologie fausse, perte de messages</td>
</tr>
</table>

Annexe 2

Fichier de topologie au format XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Cars in a City Center using the SpaceGraph Traffic Light. -->
<universe>
 <dimx>1000.0</dimx>
 <dimy>1000.0</dimy>
 <seed>1</seed>
 <extension class="de.uni_stuttgart.informatik.canu.mobisim.extensions.NSOutput"/>
 <extension class="de.uni_stuttgart.informatik.canu.mobisim.simulations.TimeSimulation"
param="300.0"/>
   <extension name="NewSpatialModel"
class="de.uni_stuttgart.informatik.canu.spatialmodel.core.SpatialModel"
traffic_light="NewTrafficLight" min_x="0" max_x="1000" min_y="0" max_y="1000">
     <max_traffic_lights>6</max_traffic_lights>
     <reflect_directions>true</reflect_directions>
      <number_lane full="false" max="4" dir="true">2</number_lane>
   </extension>
   <extension name="NewTrafficLight"
class="eurecom.spatialmodel.extensions.TrafficLight" spatial_model="NewSpatialModel"
step="10000"/>
 <extension class="eurecom.spacegraph.SpaceGraph" spatial_model="NewSpatialModel"
traffic_light="NewTrafficLight" cluster="true">
   <clusters density="0.000004">
       <cluster id="downtown">
         <density>0.0002</density>
           <ratio>0.1</ratio>
           <speed>16.66</speed>
         </cluster>
         <cluster id="residential">
         <density>0.00005</density>
           <ratio>0.4</ratio>
           <speed>11.11</speed>
         </cluster>
         <cluster id="suburban">
          <density>0.00001</density>
           <ratio>0.5</ratio>
           <speed>27.77</speed>
         </cluster>
      </clusters>
 </extension>
  <extension name="PosGen"
class="de.uni_stuttgart.informatik.canu.tripmodel.generators.RandomInitialPositionGenerator
" spatial_model="NewSpatialModel"/>
  <extension name="TripGen"
class="de.uni_stuttgart.informatik.canu.tripmodel.generators.RandomTripGenerator"
```

```
spatial_model="NewSpatialModel">
    <reflect_directions>true</reflect_directions>
     <minstay>5.0</minstay> <maxstay>30.0</maxstay>
  </extension>
  <nodegroup n="25">
    <extension class="polito.uomm.IDM_LC" spatial_model="NewSpatialModel"
initposgenerator="PosGen" tripgenerator="TripGen">
      <minspeed>5.55</minspeed>
      <maxspeed>27.77</maxspeed>
      <step>0.1</step>
    <b>0.5</b>
    </extension>
  </nodegroup>
  <extension class="de.uni_stuttgart.informatik.canu.mobisimadd.extensions.GUI"
spatial_model="NewSpatialModel">
    <width>640</width>
    <height>480</height>
    <step>1</step>
  </extension>
</universe>
```

50

75

Spécifications des champs du fichier

La zone de simulation est spécifiée en utilisant la balise **<unierse>**

<Dimx> et **<dimy>** : la dimension x et la dimension y de la zone de simulation, ils peuvent être spécifiés en mètres (notez qu'ils ne peuvent être utilisés que dans des scénarios avec des zones de simulation rectangulaire délimitée).

<step> : précise la durée de chaque étape de simulation (en second). En cas d'omission, la valeur 1 ms est utilisée.

<seed> : Précise la semence de la génération du nombre aléatoire utilisé par VanetMobiSim.

<nodegroup> : ajoute un groupe de nœuds à la simulation.

<extension> : ajoute une instance d'une extension globale de la simulation.

- **class** : spécifie le nom de la classe instanciée. La classe doit être une dérivée de *uni_stuttgart.informatik.canu.mobisim.core.ExtensionModule* et accessible par la JVM.
- **name** : spécifie le nom d'instance de classe. Utilisé pour identifier de manière unique et référencer une instance dans la simulation. La plupart des extensions ont des noms prédéfinis.

Avec une instance de la classe ***de.uni_stuttgart.informatik.canu.mobisim.extensions.NSOutput*** un fichier de trace au format NS2 peut être défini.

Le temps de simulation peut également être défini avec une instance de classe ***de.uni_ stuttgart.informatik.canu.mobisim.simulations.TimeSimulation.***

Environnement spatial :

Un environnement spatial est ajouté avec une instance de ***de.uni_stuttgart.informatik.canu.spatialmodel.core.SpatialModel***. L'extension de l'environnement spatial ajoute le support pour les routes multivoies ou Multiflow et feux de circulation aux intersections.

- **name** : spécifie le nom d'instance de classe. Utilisé pour identifier de manière unique et référencer l'instance dans la simulation. Le nom par défaut est "SpatialModel". Changer le nom par défaut n'est pas recommandé.
- **traffic_light** : spécifie le nom du ***eurecom.spatialmodel.extensions.TrafficLight*** si différente de la valeur par défaut.
- **min_x** : spécifie la plus à gauche coordonnée x de "fenêtre écrêtage" (en m). La coordonnée est relative à la coordonnée x minimale de la source. Utilisé pour traiter une partie de la zone géographique.
- **min_y** : spécifie la plus basse coordonnée y de "fenêtre écrêtage" (en m). La coordonnée est relative à la coordonnée y minimale de la source. Utilisé pour traiter une partie de la zone géographique.
- **max_x** : spécifie la plus à droite coordonnée x de "fenêtre écrêtage" (en m). La coordonnée est relative à la coordonnée x maximale de la source. Utilisé pour traiter une partie de la zone géographique.
- **max_y** : spécifie la plus haute coordonnée y de "fenêtre écrêtage" (en m).

<max_traffic_lights> : spécifie le nombre d'intersections gérées par des feux de circulation. Cette balise n'a aucun effet si le ***eurecom.spatialmodel.extensions.TrafficLight*** n'est pas déclaré après cette extension. La valeur par défaut est 5.
<number_lane> : spécifie le nombre et les caractéristiques des routes à plusieurs voies. La valeur par défaut est 1. La valeur maximale est 4.

- **full** : indique si toutes les routes ont de voies multiples ou non.
- **max** : si la valeur de <full> est false, il indique le nombre maximum de routes avec une capacité multi-voies dans le graphique. Si omis, la valeur par défaut est 4.
- **dir** : spécifie si le modèle spatial différencie physiquement les deux flux de trafic. Si la valeur de l'attribut <full> est true, <dir> et <reflect_directions> sont équivalentes. Si non, <dir> permet à l'utilisateur de différencier les flux directionnels de routes à plusieurs voies seulement. Si omis, la valeur par défaut est false.

<Reflect_directions> : spécifie si le modèle spatial différencie physiquement les deux flux de trafic. La valeur par défaut est false. Cette valeur doit correspondre à la valeur du Trip Generator.

Feux de circulation :

Un support pour les feux de circulation aux intersections peut être ajouté à l'aide d'une instance de l'extension ***eurecom.spatialmodel.extensions.TrafficLight***, permettant de définir l'intervalle de temps entre les changements successifs de la lumière des feux de circulation en ms.

Espace graphique :

Un espace graphique est ajouté avec une instance de l'extension ***eurecom.spacegraph.SpaceGraph.*** Cela crée un graphe aléatoire construit en appliquant une tessellation de Voronoï à un ensemble de points répartis au hasard (obstacles) .Il est possible de définir des zones ou des clusters avec différentes densités d'obstacles, en créant un graphe non homogène. L'utilisateur doit être conscient du fait que les feux de signalisation spécifiés par cette extension sont représentés comme des meubles de route, tandis que le ***eurecom.spatialmodel.extensions.TrafficLight*** est en charge de la gestion de leurs tâches.

- **cluster** : spécifie si le regroupement est utilisé pour créer l'espace graphique. La valeur par défaut est false.

<clusters> : spécifie les caractéristiques des clusters si la valeur de cluster a été défini par true.

- **density** : indique la densité de grappes (en grappes/m2). à titre d'exemple : une valeur de densité de grappes de 0.000004 dans une topologie de 1000 m2 signifie que la zone de simulation est divisée en 4 grappes. Ensuite, l'espace graphique enverra les différentes grappes dans la zone de simulation selon l'ordre de la description et les ratios correspondants, et remplira la fente de la grappe restante par la grappe qui la plus faible densité.

<cluster> : précise les paramètres de chaque grappe :

- **id** : spécifie l'identificateur de la grappe.

<density> : spécifie la densité des obstacles dans la grappe (en obstacles / m2), à titre d'exemple, une valeur de densité de 0,0002 signifie qu'il y a 2 obstacles tous les 100 m2.

<ratio> : indique le pourcentage de ce type de grappe dans la zone de simulation : cette valeur peut varier dans l'intervalle [0,1], et doit être compatible avec les valeurs similaires d'autres grappes.

<speed> : spécifie la vitesse maximale en m/s autorisée sur les segments de route créés pour chaque grappe. La valeur par défaut est de 50 km/h.

Position initiale et générateur de route :

Une génération aléatoire de la position initiale de nœuds et de leur modèle de déplacement au cours de la simulation peut être définie respectivement à l'aide d'une instance de l'extension ***de.uni.stuttgart_informatik.canu.tripmodel.generators.RandomInitialPositionGenerator*** et de l'extension ***de.uni_stuttgart.informatik.canu.tripmodel.generators.RandomTripGenerator***.

Définition du nœud :

Plusieurs nœuds sont ajoutés à la simulation en utilisant la balise <nodegroup>. Pour simuler le mouvement de nœuds à l'aide de IDM_LC une instance de l'extension ***polito.uomm.IDM_LC*** est utilisée. Les véhicules se déplaçant selon le modèle intelligent de gestion d'intersection IDM_LC : ralentir et arrêter aux intersections, ou agir selon les feux de circulation, s'ils sont présents. En outre, les véhicules sont capables de changer de voie et effectuer des dépassements en présence de routes à plusieurs voies.

Définition de GUI :

A l'aide d'une instance de l'extension ***de.uni_stuttgart.informatik.canu.mobisimadd.extensions.GUI,*** Une interface graphique apparaît lors de la simulation, avec le modèle spatial et ses éléments tels que les routes et les nœuds en mouvement.

Annexe 3

Script de simulation

```
#check the parameters
if {$argc != 7} {
        puts "Invalid options"
        puts "Usage: "
        puts "ns rp.tcl routing_protocols node_num max_connection sending_rate max_speed
sim_time cpmrate"
        exit
}
# ======================================================================
# Initialization
# ======================================================================
# Default node configuration
set nodeConfig "no-log 0; log-none ; log-route 1"
# Attacker types
set faux_link_hellos 1
set faux_link_tcs    2
set refuser        3
set faux_links 4
#get
set rp             [lindex $argv 0]
set nn             [lindex $argv 1]
set co             [lindex $argv 2]
set ra             [lindex $argv 3]
set v              [lindex $argv 4]
set sim            [lindex $argv 5]
set cpmrate        [lindex $argv 6]
#options for the mobile-nodes
#=================================================================
set val(chan)              Channel/WirelessChannel
set val(prop)              Propagation/TwoRayGround ;#Propagation/Shadowing
set val(netif)             Phy/WirelessPhyExt
set val(mac)               Mac/802_11Ext
#--------------------for IEEE 802.11p-------------------------------------
Phy/WirelessPhyExt set CSThresh_            3.162e-12   ;#-85 dBm Wireless interface sensitivity
(sensitivity defined in the standard)
Phy/WirelessPhyExt set Pt_                  0.001
Phy/WirelessPhyExt set freq_                5.9e+9
Phy/WirelessPhyExt set noise_floor_         1.26e-13    ;#-99 dBm for 10MHz bandwidth
Phy/WirelessPhyExt set L_                   1.0         ;#default radio circuit gain/loss
Phy/WirelessPhyExt set PowerMonitorThresh_     6.310e-14   ;#-102dBm power monitor  sensitivity
Phy/WirelessPhyExt set HeaderDuration_         0.000040    ;#40 us
Phy/WirelessPhyExt set BasicModulationScheme_   0
```

```
Phy/WirelessPhyExt set PreambleCaptureSwitch_   1
Phy/WirelessPhyExt set DataCaptureSwitch_       0
Phy/WirelessPhyExt set SINR_PreambleCapture_    2.5118;     ;# 4 dB
Phy/WirelessPhyExt set SINR_DataCapture_        100.0;      ;# 10 dB
Phy/WirelessPhyExt set trace_dist_              1e6         ;# PHY trace until distance of 1 Mio. km
("infinty")
Phy/WirelessPhyExt set PHY_DBG_                 0
Mac/802_11Ext set CWMin_                        15
Mac/802_11Ext set CWMax_                        1023
Mac/802_11Ext set SlotTime_                     0.000013
Mac/802_11Ext set SIFS_                         0.000032
Mac/802_11Ext set ShortRetryLimit_              7
Mac/802_11Ext set LongRetryLimit_               4
Mac/802_11Ext set HeaderDuration_               0.000040
Mac/802_11Ext set SymbolDuration_               0.000008
Mac/802_11Ext set BasicModulationScheme_        0
Mac/802_11Ext set use_802_11a_flag_             true
Mac/802_11Ext set RTSThreshold_                 2346
Mac/802_11Ext set MAC_DBG                       0
#-------------------------------------------
if {$rp=="DSR"} {
        set val(ifq)        CMUPriQueue ;#for DSR routing protocol
} else {
        set val(ifq)        Queue/DropTail/PriQueue ;#for DSDV and AODV
}
set val(ll)                 LL
set val(ant)                Antenna/OmniAntenna
set val(x)                  1001 ;#the simulation scenario is 1000x1000
set val(y)                  1001
set val(cp)                 "trafic/cbr-$nn-$co-$ra"
set val(sc)                 "scen/scen-$nn-$v"
set val(ifqlen)             50
set val(nn)                 $nn     ;#num of mobile nodes
set val(stop)               $sim    ;#the end time
set val(tr)                 "$rp/$rp-$nn-$co-$ra-$v-$sim-$val(x)x$val(y)-$cpmrate.tr"       ;#trace file
set val(nam)                "nam/$rp-$nn-$co-$ra-$v-$sim-$val(x)x$val(y)-$cpmrate.nam" ;#nam trace file
set val(rp)         $rp     ;#routing protocol
# JP_NEW
# A cpm rate of 50 means that, in average, CPMs are sent in response to TC in 50% of the cases
Agent/OLSR set cpm_rate_    $cpmrate
Agent/OLSR set def_prating_ 100
Agent/OLSR set def_srating_ 100
Agent/OLSR set srating_dec_  -2
Agent/OLSR set srating_inc_  1
Agent/OLSR set detect_faux_hello_ 0
Agent/OLSR set detect_faux_tc_ 1
# Communication range = 250 meters
Phy/WirelessPhy set RXThresh_ 3.65262e-10
```

```
#Main Program
set ns_ [new Simulator]
$ns_ use-newtrace
set tracefd [open $val(tr) w]
$ns_ trace-all $tracefd
set namtracefd [open $val(nam) w]
$ns_ namtrace-all-wireless $namtracefd $val(x) $val(y)
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
set god_ [create-god $val(nn)]
set chan [new $val(chan)]
#configure the mobile node
$ns_ node-config -adhocRouting $val(rp) \
                 -llType $val(ll) \
                 -macType $val(mac) \
                 -ifqType $val(ifq) \
                 -ifqLen  $val(ifqlen) \
                 -antType $val(ant) \
                 -propType $val(prop) \
                 -phyType $val(netif) \
                 -channel $chan \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace ON \
                 -macTrace OFF \
                 -movementTrace OFF
for {set i 0} {$i < $val(nn) } {incr i} {
        set node_($i) [$ns_ node]
        $node_($i) random-motion 0
}
puts "loading traffic file: $val(cp)"
source $val(cp)
puts "loading scene file: $val(sc)"
source $val(sc)
for {set i 0} {$i < $val(nn)} {incr i} {
        $ns_ initial_node_pos $node_($i) 10
}
# -------- MALICIOUS NODES ------------
# Attack type 1 (=) Faux HELLO
# Attack type 2 (=) Faux TC
# $ns_ at 50 "[$node_(5) agent 255] attack-type 1"
# $ns_ at 300 "[$node_(5) agent 255] attack-type 0"
# $node_(5) set willingness 7; # will always (so that he is always selected as mpr)
# node starts misbehaving at the 50 seconds
$ns_ at 50 "[$node_(5) agent 255] attack-type 1"
# number of links fauxd by the malicious node (only for attack-type 1)
$ns_ at 50 "[$node_(5) agent 255] max-faux-links $faux_links"
# node stops misbehaving at the 100 seconds
```

```
$ns_ at 100 "[$node_(5) agent 255] attack-type 0"
#---------------------------------------------
# Print (in the trace file) routing table and other
# internal data structures on a per-node basis
#
for {set i 0} {$i <= $val(stop)} {incr i} {
   $ns_ at $i "[$node_(5) agent 255] print_mprselset"
   for {set j 0} {$j < $val(nn)} {incr j} {
          $ns_ at $i "[$node_($j) agent 255] print_rating_table"
   }
#    $ns_ at $i "[$node_(0) agent 255] print_rtable"
#    # $ns_ at $i "[$node_(0) agent 255] print_linkset"
#    $ns_ at $i "[$node_(0) agent 255] print_nbset"
#    $ns_ at $i "[$node_(0) agent 255] print_nb2hopset"
#    $ns_ at $i "[$node_(1) agent 255] print_mprset"
#    # $ns_ at $i "[$node_(0) agent 255] print_topologyset"
}
#-----------------------------------------------------
for {set i 0} {$i < $val(nn)} {incr i} {
          $ns_ at $val(stop).000000001 "$node_($i) reset"
}
$ns_ at $val(stop).000000001 "finish"
proc finish {} {
          global ns_ tracefd namtracefd
          $ns_ flush-trace
          close $tracefd
          close $namtracefd
          exit 0
}
 proc timeReport {interval} {
 global ns_ tracefd namtracefd
 set now [$ns_ now]
 puts "Time=[clock format [clock second] -format "%H:%M"](min), Sim=[format %.1f $now](sec)"
 flush $tracefd
 flush $namtracefd
 $ns_ at [expr $now+$interval] "timeReport $interval"
}
$ns_ at 0 "timeReport 1"
$ns_ run
```

Annexe 4

Fichier AWK

```
BEGIN {
 totalreceved = 0;
 PDR = 0.0;
 totalsend = 0;
 total_time = 0.0;
 nb_Pacquet_routage=0.0;
 cout_routage=0.0;
 nb_Pacquet_donnee=0.0;
 totalsend_data=0.0;
 gigue=0.0;
 highest_packet_id = 0;
 jitter_sum=0.0;
 last_delay=0.0;
 n=0.0;
 MAX_NODES = 1000;
 flow = 0;
 delay = 0.0;
}
{
 action = $1;
 time = $3;
 layer = $19;
 pkttype = $35;
 packet_id = $41;
 msg=$51;
 if (packet_id > highest_packet_id )
      highest_packet_id = packet_id ;
 if ( action == "s" && layer == "AGT")
      if ( start_time[packet_id] == 0 ) start_time[packet_id] = time;
 if (action != "d" ){
      if (action == "r" && layer == "AGT" && pkttype=="cbr") {
                  if( end_time[packet_id] == 0 ) {
                        split ($31,a,".");
                        split ($33,b,".");
                        src_id[packet_id] = a[1];
                        dst_id[packet_id] = b[1];
                  }
            end_time[packet_id] = time;
      }
 }
 else
      end_time[packet_id] = -1;
```

```
 if ((layer == "AGT") &&(action == "r")) totalreceved ++;
 if ((layer == "AGT") &&(action == "s")) totalsend ++;
 #if ((layer == "RTR") && (action == "s")) nb_Pacquet_routage++;
 if (layer == "AGT") nb_Pacquet_donnee++;
 if (layer=="RTR") {
        if ( (action=="s" || action=="f") ) {
                if (pkttype == "AODV" || pkttype == "DSR" || pkttype == "message" ||
pkttype=="OLSR") {              #message pour DSDV
                        nb_Pacquet_routage++;
                }
                else (pkttype=="cbr") totalsend_data ++;
        }
  }
}
END  {
    for ( packet_id = 0; packet_id <= highest_packet_id; packet_id++ ){
      start = start_time[packet_id];
      end = end_time[packet_id];
      end_to_end = end - start;
      if ( start < end )
           total_time = total_time + end_to_end;
    }
   for ( i = 0; i <= highest_packet_id; i++ ) {

                if( end_time[i] <= 0.0 )
                        continue;
                delay = (end_time[i] - start_time[i]);
                flow = src_id[i] * MAX_NODES + dst_id[i];
                last_delay = last_delay_by_flow[flow];
                if( last_delay > 0.0){
                        if ( delay > last_delay) jitter_sum += delay - last_delay;
                        else jitter_sum +=  last_delay - delay;
                }
                        n++;
                        last_delay_by_flow[flow] = delay;
        }
   delay = total_time/totalreceved;
   PDR  = (totalreceved/totalsend)*100;
   gigue=jitter_sum/n;
   cout_routage=nb_Pacquet_routage/totalreceved;
   efficacite=totalsend_data/(nb_Pacquet_routage+totalsend_data);
   printf("%f %f %f %f %f\n",PDR, delay, gigue, cout_routage, efficacite);
  }
```