Sommaire

Liste des Figures

Liste des Tableaux

Liste des Algorithmes

Liste des (sigles) et acronymes

LR-WPAN : Low-Rate Wireless Personal Area Networks

RCSF : Réseau de Capteurs Sans Fil

WSN : Wireless Sensor Network

NL : Neighbor List, F-ID: Faked node Identifier and AM: Alert Message.

CL-MAC : Cross Layer Medium Access Control protocol

Sink : Station de base

RTT : Round Trip Time

RDP : Réseau de Petri

RDP-T : Réseau de Petri Temporel

TPN : Time Petri Net

PDA : Personal Digital Assistant

RF : Radio Fréquence

DARPA :Defense Advanced Research Projects Agency

RFID : Radio Frequancy Identification

CTS : Clear To Send

RTS : Ready to Send

NAV : Network Allocation Vector

DIFS : Interframe Space duration

SIFS : Short Interframe Space

1. INTRODUCTION GENERALE

1.1. Contexte général

Incontestablement, les avancées technologiques dans les domaines de l'électronique, la micromécanique et la communication sans fil ont placé ce nouveau XXI^{e} siècle sous le signe de *technologie de l'information et de la communication (TIC),* et *la nanotechnologie*. La communication sans fil mariée à la nanotechnologie ont permis le développement d'un nouveau domaine hybride: les réseaux de capteurs sans fil ou RCSF (Wireless Sensor Network ou WSN). Un RCSF est constitué de milliers de composants technologiques granulés où chaque élément du réseau est une station miniaturisée dotée de capteurs, µ-processeur, RAM, ROM, émetteur-récepteur (transceiver), système embarqué, des applications et protocoles de communication, le tout est alimenté par une source d'énergie, dans la plupart des cas, une batterie embarquée. De plus, cette nano-station (noeud de capteur) peut être équipée d'un sous-système de mobilité et d'un autre pour l'approvisionnement en énergie. Le noeud de capteur utilise toutes les puissances de la micro-technologie électronique pour récolter des grandeurs physiques environnementales plus fiables. Le µ-processeur ainsi que la RAM sont très limités, ce qui exige des applications assez spécifiques en matière de : 1) traitement des mesures acquises par les capteurs, 2) compression et agrégation des données, 3) routage et accès au medium. Ces nœuds de capteurs ont une puissance énergétique et des ressources relativement basses. En conséquence, leurs logiciels systèmes doivent être optimisés pour gérer au mieux la puissance et les ressources. Ces considérations influenceront tous les aspects de conception de logiciels systèmes, y compris les protocoles de communication qui y figurent, la gestion du réseau et les dispositions de sécurité.

La réduction des coûts matériels, la diminution de la taille des nœuds, ainsi que l'élargissement de la gamme des capteurs disponibles, ont permis d'étendre le champ d'application des RCSFs. Le domaine militaire a été le premier bénéficiaire de ce fruit technologique et le moteur initial de son développement pour l'analyse de terrains dangereux et la surveillance des mouvements des troupes ennemies. Les applications environnementales se sont, ensuite, succédées et multipliées pour la détection de feux de forêts, la surveillance d'activités volcaniques ou sismiques, ou encore le suivi du déplacement d'animaux. On utilise aussi les réseaux de capteurs pour des applications médicales comme la veille épidémiologique, ou dans un but commercial pour l'optimisation des processus de stockage, ou encore dans l'agriculture de précision, et la construction de maisons intelligentes

Les RCSFs, malgré la diversité de leurs applications telles que dans les situations critiques et militaires, leurs succès dépend de leur propre sécurité. La vulnérabilité des nœuds de capteurs et celle du canal de communication face aux attaques malicieuses constitue un obstacle majeur freinant leur prolifération. En effet, les nœuds de capteurs sont soumis à une forte contrainte de consommation d'énergie en raison de leur miniaturisation ainsi qu'à l'environnement hostile où ils sont déployés. D'un autre côté, assurer des services de sécurité pour ces applications nécessite un surcoût en termes de calcul CPU, consommation énergétique, et la surcharge du trafic réseau causé par les paquets de contrôle supplémentaires. En fait, la consommation d'énergie des nœuds de capteurs joue un rôle moteur dans la

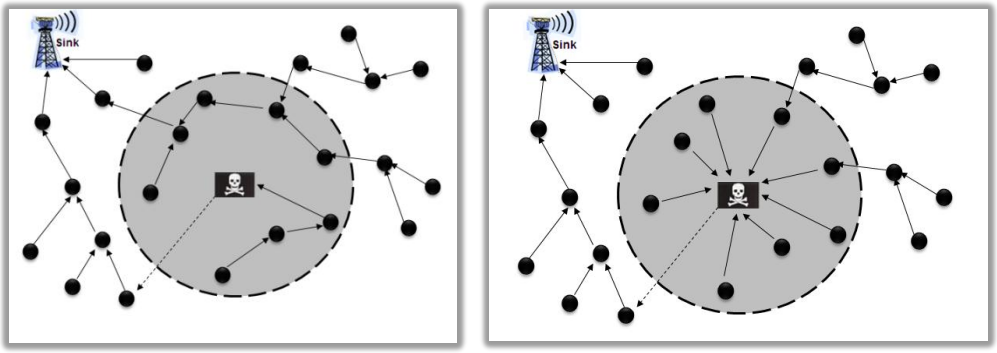
pérennité du réseau qui est devenue le critère de performance déterminant dans ce genre de réseaux. De ce fait, la sécurisation des RCSFs est un défi technique. Un des enjeux principaux est de pouvoir trouver des solutions de sécurité adaptées à leurs caractéristiques spécifiques. Dans cette optique, un protocole de sécurité, dédié à ce type de réseau, doit pouvoir établir des sessions sécurisées sans porter atteinte aux performances globales du réseau tout en assurant les services de sécurité appropriés pour chaque type d'application.

1.2. Contributions

Entre 2008 et 2010, au sein du laboratoire d'informatique industrielle et réseaux (LIIR), nous avons développé le protocole CL-MAC pour les RCSFs qui a fait l'objet d'un ensemble de publications [1-5]. CL-MAC est un protocole à efficacité énergétique et à couches croisées dédié aux applications de détection des feux de forêts. Le côté sécuritaire du protocole n'a pas été pris en compte lors de sa phase de conception, et cela a été mentionné comme perspectives [6]. Dans notre thèse, nous avons tenté de soulever le problème de sécurisation de ce protocole. Nous avons étudié deux types d'attaques trés dangereux du point qu'ils provoquent le disfonctionnement du protocole même dans le cas où la communication est cryptée ; à savoir l'attaque ***Sinkhole*** et l'attaque ***Wormhole***. Chacune de ces deux attaques a fait l'objet de quelques contributions scientifiques.

- **L'attaque Sinkhole**: Le protocole CL-MAC présente des vulnérabilités face à l'attaque Sinkhole. Un nœud malveillant n'a pas besoin de posséder la clé de la cryptographie pour absorber le trafic le traversant. Il suffit, à ce type de nœud malicieux placé sur un chemin de communication multi-sauts, de se présenter comme un nœud routeur avec des critères de performance idèales (plus court chemin, réserves d'énergie assez conséquentes) pour attirer vers lui le trafic réseau.Les paquets de données ainsi que ceux du contrôle, sont tout simplement absorbés, créant ainsi une fracture du chemin passant par ce nœud (Sinkhole).

  Quelle que soit la nature de cette attaque, passive là où le nœud se limite à l'absorbation du trafic, active lorsque le nœud génère de faux paquets, ou selective (*Selective Forward*) quand le nœud laisse passer certains paquets et stoppe les autres dans l'idée de feinter les techniques de détection. Le RCSF opérant avec le protocole CL-MAC se trouve dans l'incapacité à assurer ces fonctions.



(a) Positionnement du nœud malicieux *(b) l'attaque sinkhole*

Figure 1-1 : *Effet de l'attaque Sinkhole*

Il est visible par le biais de la figure 1-1, qu'une partie du réseau est déconnectée du reste du réseau. Notre solution face à ce type d'attaque se basera sur la technique RTT (Round Tripe Time) [7].

- **L'attaque Wormhole**: Le protocole CL-MAC souffre aussi d'une énorme carence en matière de mesures de sécurité contre l'attaque wormhole. Deux nœuds intrus malveillants provoqueront un disfonctionnement du protocole CL-MAC. Ces nœuds intrus construisent un tunnel liant deux parties disjointes du réseau, créant l'illusion que les deux parties font référence au même endroit et que les nœuds des deux parties du réseau sont des voisins, contrairement à la réalité. Ce type d'attaque n'a pas aussi besoin d'avoir possession de la clé de la cryptographie pour nuire au fonctionnement du réseau. Un des deux nœuds malicieux absorbe le trafic de son voisinage et le transmet au point distant pour être générépar le second nœud du wormhole (voir la figure1-2). Ce dernier devient plus dangereux sous sa forme active, lorsque les deux extrémités du wormhole modifient les entêtes des paquets de données et de contrôle communiqués d'une zone où un évènement s'est produit, et la destination.

  Notre proposition pour remédier à cette carence, se base sur la technique du RTT. La version sécurisée de l'algorithme CL-MAC (SCL-MAC) a été testée en utilisant une série de simulations sous NS2 puis modélisée et validée par un modèle formel basé sur les réseaux de Petri temporels. Les propriétés étudiées sont la vivacité, l'inter blocage, le caractére borné et l'accessibilité.

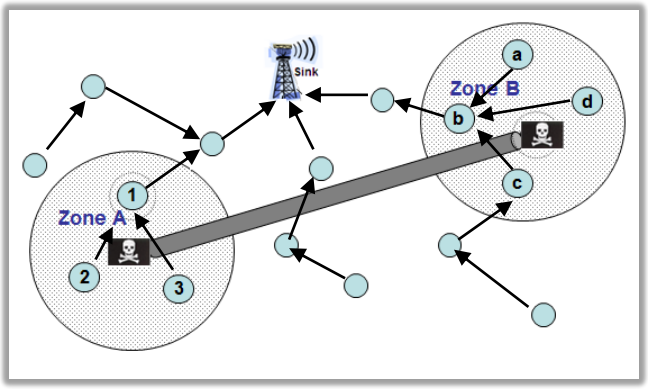


Figure 1-2: *L'attaque wormhole*

1.3. Organisation de la thèse

Notre thèse est composée de cinq chapitres suivis d'une conclusion générale. Le positionnement de nos travaux est présenté dans les trois premiers chapitres et nos contributions sont détaillées dans les deux derniers.

Dans le deuxième chapitre, nous présenterons les concepts de base des RCSFs et ceux de la sécurité. Nous commençerons d'abord par la définition des différents concepts gravitant autour de cette thématique, ensuite nous exposerons les problèmes de sécurité en insistant sur les vulnérabilités et les caractéristiques intrinsèques des RCSFs. Enfin, nous présenterons une taxonomie des attaques et discutons les besoins de sécurité requis par les protocoles.

Dans le troisième chapitre, nous continuerons notre étude bibliographique sur la sécurité. Nous aborderons tous les concepts et les challenges rencontrés lors de la conception

d'un protocole sécurisé et nous survolerons les travaux existants, recensés dans la littérature, en comparant les différentes solutions.

Le quatrième chapitre sera consacré au protocole CL-MAC. Une discussion sur sa vulnérabilité face à tous les types d'attaques y sera présentée. Nous introduirons les réseaux de Petri temporels nécessaires à la modélisation et à la validation des solutions proposées.

Notre première contribution sur la sécurisation du protocole CL-MAC face à l'attaque Sinkhole est présentée dans le cinquième chapitre. Nous exposerons les détails de notre protocole CL-MAC sécurisé, et nous terminerons par une évaluation de performances ainsi qu'une validation formelle da la solution proposée.

Le sixième chapitre est dédié (consacré) à notre deuxième contribution liée à la sécurisation du protocole CL-MAC face aux attaques de type Wormhole. Nous présenterons l'algorithme proposé en tenant compte du problème d'énergie. Plusieurs simulations sont effectuées afin de consolider nos résultats et prouver l'efficacité de la version sécurisée de notre protocole. Nous utiliserons les réseaux de Petri tempoels pour valider nos propositions.

Cette thèse est clôturée par une conclusion générale et des perspectives futures dans lesquelles nous rappelons les différentes contributions réalisées et des perspectives de recherche comme extension à ce travail.

2. LES RESEAUX DE CAPTEURS SANS FIL ET PROBLEME DE SECURITE

2.1. Introduction

Le développement rapide et croissant dans les domaines de la technologie des Systèmes Micro-Electro-Mécaniques (SMEM), la nanotechnologie, la communication sans fil, et l'électronique digitale, ont permis le développement d'un nouveau genre de réseaux sans fil appelé réseau de capteurs sans fil (RCSF). Ce dernier est formé d'un très grand nombre de nœuds, voire des milliers ou des millions, opèrant de façon autonome et communiquant entre eux via des transmissions radio à courtes portées. Ces nœuds sont capables, grâce à des capteurs qu'ils portent, de récolter les grandeurs physiques de leurs environnements sans aucune intervention humaine, telles que la température, la pression, la vitesse, l'humidité, etc. Parmi les verrous majeurs de ces nœuds de capteurs, nous pouvons distinguer la limitation de leurs ressources en termes de capacité de calcul, l'espace de stockage des données et la faible portée radio de leurs transceivers. Les ressources limitées de ces nœuds et les environnements hostiles dans lesquels ils pourraient être déployés, fragilisent ce type de réseaux en les rendant vulnérables à tout type d'attaque. Dans ce contexte, une partie de la communauté des chercheurs a tenté d'apporter des solutions en termes de mécanismes de sécurité pour la prévention et la détection de tout type d'attaque prenant en compte les contraintes techniques spécifiques aux RCSFs.

Dans ce chapitre, nous commençerons par donner un bref aperçu sur les réseaux Ad-hoc qui forment la classe mère des RCSFs. Nous enchaînerons par les RCSF's et leurs domaines d'application. Nous exposerons ensuite les deux types de topologies existants dans les réseaux de capteurs: topologie plate et à base de cluster. La dernière partie est consacrée aux problèmes de sécurité gênant la prolifération de ce type de réseau. Dans cette section, nous décriverons un certain nombre d'attaques pouvant cibler les différentes couches de la pile protocolaire, par la suite nous définissons quelques mécanismes de sécurité proposés dans la littérature afin de protéger le réseau contre les différentes menaces visant à perturber son bon fonctionnement.

2.2. Les réseaux adhocs

Ces dernières années, les réseaux sans fil de tout genre continuent de gagner du terrain et deviennent de plus en plus populaires. Les périphériques sans fil portables d'aujourd'hui, bien qu'ils soient petits et légers, ils possèdent de très grandes capacités de calcul. Ceci est dû essentiellement aux progrès technologiques qui ne cessent de se développer. La prolifération massive de ces périphériques et la demande croissante en matière de communication en continue par les utilisateurs sont deux facteurs qui ont contribué énormément au déploiement des réseaux sans fil. Nous distinguons fondamentalement deux grandes classes de réseaux sans fil.

- La première est connue sous le nom de *réseaux cellulaires ou à infrastructure* dans laquelle les stations (*hosts*) mobiles communiquent directement avec une station de base connectée à une infrastructure réseau fixe et couvrant une région appelée cellule (voir figure 2.1 (a)) [8]. Dans le cas où un nœud mobile se déplace

en dehors de la portée radio d'une station de base, un mécanisme de *handoff* ou *handover* (passage du contrôle d'une station de base au contrôle d'une autre station de base adjacente) se produit. Les paquets de données dans ce cas prennent un chemin filaire et seulement le dernier saut qui est sans fil.

- La seconde classe des réseaux sans fil est la classe des *réseaux adhocs*, nommée aussi *classe des réseaux multi-sauts sans fil* (*wireless multihop networks*) pour décrire réellement cette classe [9]. Ces réseaux sont simplement formés par un ensemble de stations sans fil pouvant se déplacer librement et qui peuvent communiquer sans présence d'aucune infrastructure préétablie ou une administration centralisée. Il est donc naturel que ces réseaux doivent disposer des capacités d'auto-configuration et d'auto-organisation. Puisque les nœuds source et destination peuvent ne pas être dans la même portée radio, des chemins de routage multi-sauts sont donc inévitables (voir Figure 2.1 (b)). Chaque nœud du réseau peut être émetteur, récepteur ou routeur. En tant que routeur, un nœud est capable de relayer le trafic réseau vers d'autres nœuds assurant une communication à travers tout le réseau.

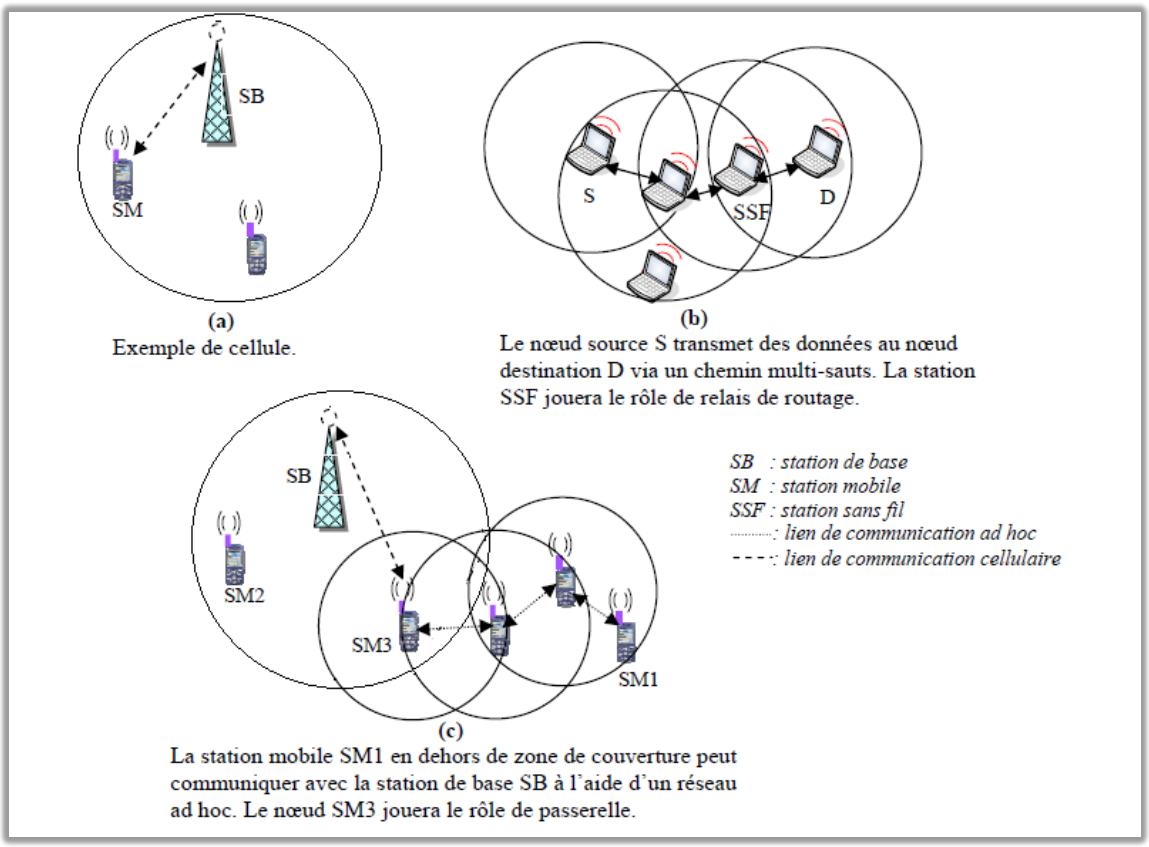


Figure 2-1: *Exemples des réseaux Adhoc*

Les réseaux Adhoc possèdent plusieurs avantages en les comparants aux réseaux traditionnels à infrastructure pour la diversité de leurs applications et scénarios. Les causes de cette supériorité sont induites essentiellement de leur simplicité et rapidité de déploiement, leur robustesse, leur indépendance vis-à-vis de toute infrastructure, et surtout leur coût relativement faible. Les réseaux Ad-hoc peuvent servir également à étendre les réseaux cellulaires en formant ce qu'on appelle réseaux cellulaires hybrides ou réseaux cellulaires multi-sauts (hybrid or multi hop cellular networks) [10]. Un réseau hybride, comme le montre

la figure 2.1 (c), peut aider énormément à étendre la couverture et à améliorer la redondance et la performance des réseaux cellulaires [11].

Le terme latin « *Adhoc* », qui peut être littérairement translaté en « *pour ceci* », veut dire « *pour cette objectif seulement* ». Souvent, on emploie le terme « *mobile* » pour dire « *réseaux adhocs mobiles* » ou MANET (*Mobile Ad hoc Networks*) pour désigner toutes sortes de réseaux multi-sauts sans fil, sans infrastructure et parfaitement auto-organisés [6].

2.2.1. Caractéristiques

Plusieurs caractéristiques distinguent les réseaux Ad-hoc des réseaux cellulaires. Ces caractéristiques doivent être prises en considération dans tout processus de conception protocolaire. Elles sont entièrement spécifiées dans laRFC 2501 [WEB1] et résumées par les paragraphes suivants.

1. *Topologie dynamique* : les raisons principales aux changements topologiques dans ces réseaux sont liées à des facteurs non contrôlables tels que la mobilité des nœuds, les interférences et le bruit, et à des facteurs contrôlables tels que la puissance de transmission et la direction de l'antenne [6] et [12]. S'ajoutant à cela, le mécanisme de mise en veille des nœuds pour la préservation de l'énergie. Ainsi, la topologie du réseau peut changer fréquemment et d'une façon imprévisible.

2. *Contrainte d'énergie* : les nœuds dans un réseau Ad-hoc sont typiquement alimentés par des batteries à capacité en puissance est souvent limitée. Par conséquent, elle ne peut satisfaire les demandes d'énergie d'un nœud pour un fonctionnement normal durant une période de temps raisonnable.

3. *Capacité des liens limitée et variable* : celle-ci est limitée, par rapport à la capacité des réseaux filaires, et peut varier au cours du temps pour au moins deux raisons principales : le changement des conditions de propagation et la variation des distances entre les nœuds.

4. *Sécurité physique limitée* : les réseaux Ad-hoc mobiles sont plus vulnérables par rapport aux autres réseaux filaires et cellulaires. Cette vulnérabilité est due essentiellement à la nature du canal de communication sans fil qui rend possibles certaines attaques malicieuses allant de l'écoute clandestine passive aux interférences actives. D'autres attaques redoutables, dues à la topologie du réseau, peuvent aussi être envisagées comme par exemple l'attaque Sinkhole, Wormhole, etc. [13].

5. *Bande passante limitée :* perte de données, perte de routes, erreur de transmission, interférences, et le problème des nœuds cachés.

2.2.2. Exemples de réseaux Ad-hoc

Les catégories des réseaux présentés dans cette partie, sont fondées sur le paradigme réseaux Ad-hoc. Ces réseaux sont en réalité des réseaux hybrides du moment où certains de leurs nœuds sont connectés à des réseaux à infrastructure, et ne possèdent pas des définitions claires et définitives dans la littérature. Les définitions données par les auteurs, ci-après à ces réseaux peuvent changer à travers le temps et leurs contextes d'utilisation. Nous décrivons chacun de ces réseaux en se basant sur des descriptions les plus communément utilisées [14].

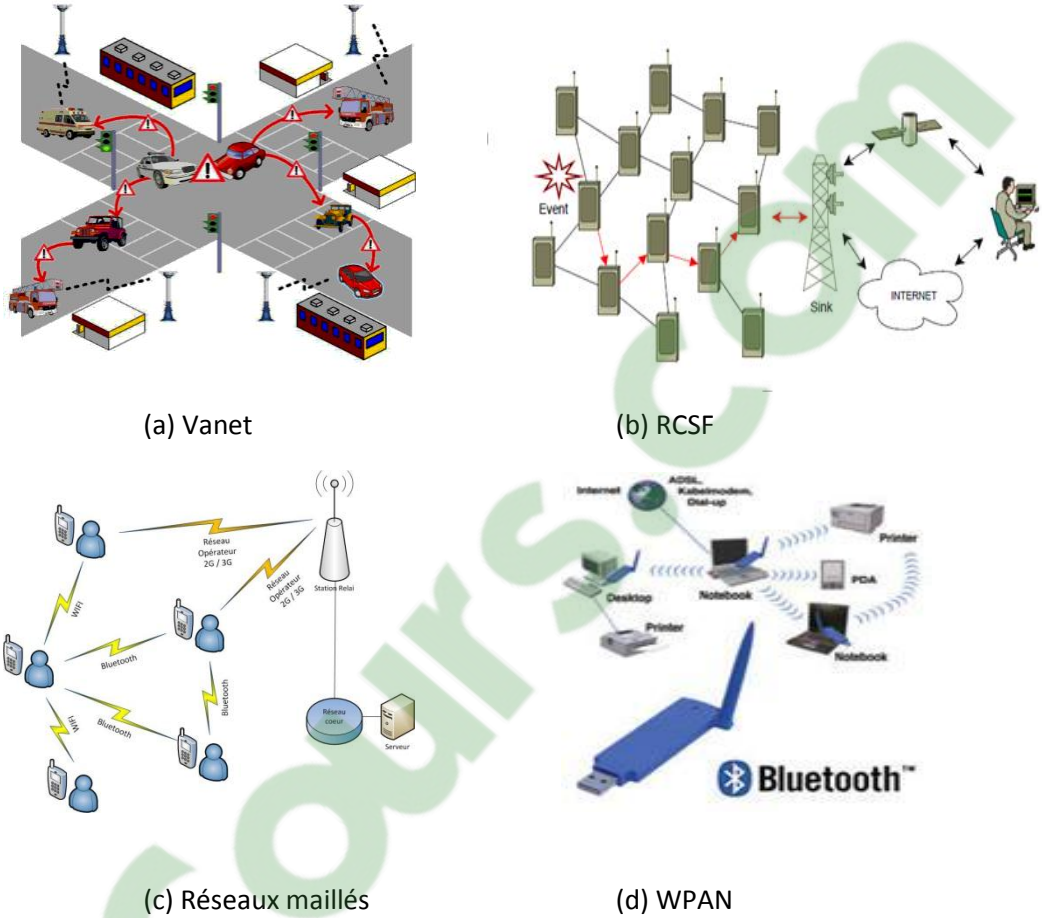
- *Réseaux maillés (Mesh networks)* : ces réseaux peuvent être utilisés dans des cas extrêmes où les solutions basées sur le câblage se révèlent impossibles ou très chères, ou juste en tant que technologie alternative en prévision de pannes. Les noeuds sont déployés d'une manière dense dans une région donnée pour permettre un accès sans fil à bande réduite (*broadband*) aux utilisateurs à partir de chez eux. Ces réseaux sont aussi utilisés dans un contexte métropolitain où des hotspots sans fil sont interconnectés pour offrir des accès sans fil aux utilisateurs. Aujourd'hui, le plus souvent la norme IEEE 802.11b [Web02] est utilisée en tant que technologie sans fil sous jacente. D'autres technologies, à l'image de WiMAX (*Worldwide Interoperability for Microwave Access*) IEEE 802.16 [Web03] et MBWA (*Mobile Broadband Wireless Access*) IEEE 802.20 [Web04], sont proposées pour compléter et/ou remplacer les réseaux IEEE 802.11b en offrant des débits meilleurs et des portées radio plus importantes.

- *Réseaux personnels sans fil (WPAN : Wireless Personal Area Networks)* : un réseau personnel *sans* fil est un réseau utilisé pour permettre une communication entre différents périphériques d'une machine appartenant à une seule personne physique. Ces périphériques regroupent entre autres des téléphones portables et des PDA. La portée radio dans ce cas est de l'ordre de quelques mètres et assure une communication entre les périphériques eux-mêmes et une connexion au réseau internet. Deux technologies commerciales, typiquement pour ce type de réseaux, dominent le marché : Bluetooth [Web05] et le standard Zigbee/IEEE 802.15.4 [15].

- *Réseaux Adhoc de véhicules (VANET: Vehicular Area Networks)* : un réseau adhoc de véhicules est un type de réseau adhoc émergent dans lequel les véhicules constituent les nœuds mobiles (forte mobilité) du réseau [16]. Ces réseaux sont utilisés principalement pour la gestion *virtuelle* du trafic routier, collecte d'information sur le trafic et diminution des congestions de trafic, des accidents et des délais d'attente. Ils sont également employés dans d'autres applications commerciales mettant en œuvre des communications réseau de type véhicule-véhicule et véhicule-infrastructure routière dans le but d'offrir une distribution efficace et rapide des données afin d'améliorer le confort et la sécurité des passagers. Ces réseaux connaissent aujourd'hui une utilisation très intense pour le développement *des systèmes de transport intelligents et intégrés* (*integrated Intelligent Transportation Systems (ITS)*) basés sur les communications sans fil [17]. Ils possèdent certaines caractéristiques qui les distinguent des autres réseaux Ad-hoc : les ressources en énergie sont presque infinies et les capacités de calcul sont très importantes.

- *Réseaux spontanés (Spontaneous networks)* : ces réseaux, considérés également comme des réseaux Ad-hocs émergents, peuvent être décrits comme l'intégration de services et de périphériques, *avec* comme objectif que ces services soient offerts d'une manière instantanée aux utilisateurs sans aucune intervention manuelle. Un exemple typique est celui d'un réseau spontané déployé au sein d'une ville où les touristes découvrent instantanément des services touristiques là où ils se trouvent, par exemple : des informations sur des musées existants dans un boulevard peuvent s'afficher sur le PDA d'un touriste visitant ce boulevard sans que celui-ci ne le demande.

- *Réseaux de capteurs sans fil (WSN : Wireless Sensor Networks)* : ces réseaux, objet de notre *étude*, représentent actuellementla technologie la plus attractive et la plus célèbre des réseaux fondés sur le paradigme réseaux.



(a) Vanet (b) RCSF

(c) Réseaux maillés (d) WPAN

Figure 2-2: *Exemples de réseaux Ad-hoc*

2.2.3. Modélisation d'un réseau Ad-hoc

Un réseau Ad-hoc peut être modélisé par un graphe comme le montre la figure 2.3.

$Gt = (V_t, E_t)$où :

V_t : représente l'ensemble des nœuds (les unités ou les hôtes mobiles) du réseau et
E_t : modélise l'ensemble les connections qui existent entre ces nœuds.

Si **e** = (**u**,**v**) $\in \mathbf{E_t}$, cela veut dire que les nœuds u et v sont en mesure de communiquer directement à l'instant t.

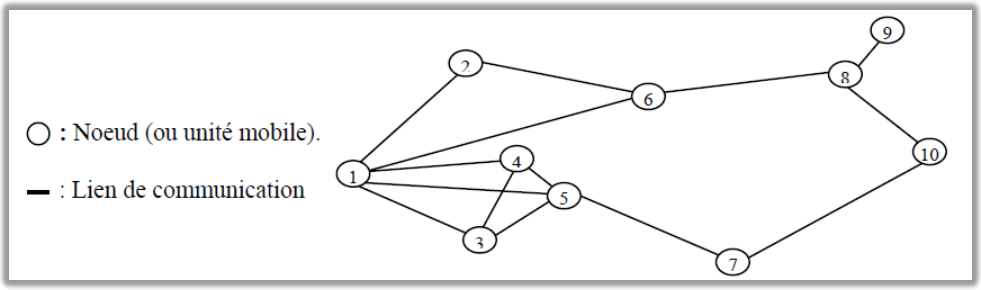


Figure 2-3: *Modélisation d'un réseau Ad-hoc avec 10 stations mobiles*

2.3. Les réseaux de capteur sans fil (RCSFs)

Les RCSFs sont une entrée originale dans le domaine des réseaux. Une pléthore de papiers de recherches a été éditée et de nombreuses recherches espèrent employer la gestion de réseau de capteurs pour un grand nombre d'applications à l'avenir. Le fonctionnement sécurisé de ces réseaux de capteurs a reçu une importance particulière face aux défis uniques que posent de tels réseaux: problème d'énergie, environnement hostile, canal de communication hertzien, etc. Dans cette partie, nous essayerons de décrire les caractéristiques intrinsèques et spécifiques à ces réseaux. La compréhension des caractéristiques de tels réseaux justifie les défis à surmonter afin de sécuriser la communication au sein d'un RCSF.

2.3.1. Introduction aux réseaux de capteur sans fil

La technologie des RCSFs est le résultat de convergence des technologies de fabrications de circuits intégrés (ou nanotechnologie), de communications sans fil et des systèmes micro-électro-mécaniques. Il existe plusieurs définitions pour un RCSF. Celles-ci dépendent souvent de l'application considérée. Selon [32], Akyildz et al l'ont défini comme suit : « *un RCSF est un système distribué de grande échelle mettant en communication un grand nombre d'entités autonomes communément appelées nœuds capteurs sans fil ou simplement capteurs* ». Mais nous avons retenu la définition suivante, inspirée de celle donnée par la DARPA [Web06] et citée dans [6], qui nous semble la plus proche de notre problématique traitée dans cette thèse : « *Un RCSF est un ensemble de petits nœuds capteurs variant de quelque dizaines d'éléments à plusieurs centaines, voir des milliers déployés dans une zone d'intérêt. Les nœuds capteurs sont capables de réaliser les fonctions de capture, de traitement et ne sont pas intégrés à une quelconque architecture pré-existante du réseau, mais communiquent à l'aide d'un réseau adhoc sans fil. L'alimentation électrique de chaque nœud capteur est assurée par une batterie individuelle dont la consommation pour la communication et le calcul lié au traitement de l'information doivent être optimisés*»

Traditionnellement, les capteurs ont été utilisés pour capter séparément des grandeurs physiques de certains phénomènes. Les réseaux de capteurs câblés ont l'utilisation répandue dans les chaînes de fabrications, les canalisations etc. Pour mesurer des phénomènes comme la température et la pression. De tels réseaux utilisent des câbles tordus ou l'Ethernet pour connecter les capteurs à un système d'alarme. Les progrès récents dans les domaines de l'électronique et les communications sans fil ont permis le développement des nœuds de petites tailles, à faible énergie, des noeuds de capteurs accessibles pour être gérés en réseau en

tant qu'un réseau de capteurs sans fil (RCSF), et augmenter, de ce fait, leur efficacité. Ces réseaux sont plus faciles à installer et à maintenir comparés aux réseaux câblés.

Un réseau typique de capteur sans fil a un nombre très grand de noeuds de capteurs (parfois également appelés les motes) déployés pour mesurer certains phénomènes. Ces nœuds emploient des liens sans fil pour transmettre les données collectées. Ils ont également la capacité de traiter les données avant leur cheminement. En intégrant les processus de sensation, traitement et communication, les réseaux de capteur fournissent plusieurs niveaux d'abstraction. L'information collectée peut être interprétée aux niveaux du nœud, local ou global. Chaque capteur est destiné pour récolter seulement des données relatives à une partie du phénomène. C'est le niveau d'abstraction des nœuds. L'agrégation des données (traitées au niveau des noeuds intermédiaires ou au niveau du Sink) collectées par plusieurs nœuds — l'abstraction locale ou globale — peut donner une meilleure compréhension du phénomène.

La gestion du réseau a évolué depuis ses jours de commencement, approvisionnant aux besoins toujours croissants de l'humanité. Elle est devenue une partie intégrante de notre vie, et continue à augmenter son influence au-delà de l'imaginable. Aujourd'hui, la gestion du réseau a apporté le dépôt global de l'information à nos portes sous forme de Web. Les avancées dans la communication ont révolutionné nos styles de vie, amélioré la qualité de vie. Un entrant des plus nouveaux dans ce domaine sont les réseaux de capteurs. Ils ont intéressé le milieu universitaire et industriel au cours de ces dernières années, et par conséquent, ils ont rapidement gagné du terrain.

2.3.2. Caractéristiques des réseaux de capteur sans fil

Certains dispositifs importants des RCSFs sont décrits ci-dessous:

1. ***Consommation énergétique réduite :*** Puisque les nœuds de capteurs sont actionnés par des batteries, et utilisés dans les applications où ils ne peuvent pas être facilement remplacés ou rechargés (ou le fait de les recharger n'est plus rentable), il y a un seuil supérieur critique de la consommation d'énergie à ne pas dépasser.

2. ***Auto configuration et auto-curatif:*** les nœuds sont déployés d'une façon Ad-hoc. Parfois, ils sont mis à l'intérieur et/ou autour du phénomène. Par conséquent, les nœuds doivent avoir la capacité de s'auto-configurer au sein du réseau. En outre, les nœuds doivent collaborer ensemble pour modifier dynamiquement le réseau face à un changement topologique.

3. ***Traitement collaboratif:*** L'utilisation rigoureuse d'énergie exige de la part des nœuds un traitement coopératif de données, avant leurs transmissions. Ceci implique l'agrégation et la compression des données au sien des nœuds du réseau, et parfois s'organiser sous forme de clusters.

4. ***Architecture centrée données:*** Les RCSFs apportent une variation dans le paradigme de la gestion de réseau à partir des architectures centrées nœuds aux architectures centrées données. Puisqu'il est difficile de remplacer les nœuds de capteur, ces derniers sont déployés d'une manière redondante. En tant que tels, l'accent est mis sur l'information rassemblée, plutôt que sur les nœuds eux-mêmes.

5. ***Scalabilité et adaptabilité:***(passage à l'échelle) À la différence des réseaux traditionnels, les RCSFs ont un nombre très grand de nœuds fonctionnant en tandem. Ceci réclame le passage à l'echelle du réseau. En outre, si des nœuds meurent, le réseau devrait continuer à fonctionner. La *densité* des nœuds peut dépasser les 20 nœuds/m3

6. ***Bas débit et les caractéristiques du trafic interrompu:*** Puisque les RCSFs sont habituellement utilisés pour *surveiller* certaines caractéristiques, leur trafic peut être interrompu, fonctionnant à de bas débits.

7. ***Taille et coût réduits:*** Le déploiement à grande échelle des nœuds de capteurs pour les applications à contraintes spatiale, exige la réduction en taille et coût des nœuds. Par conséquent, les capacités de traitement et de mémorisation sont réduites. Ceci influe indirectement sur l'architecture de protocole utilisée pour de tels réseaux.

8. ***Réseaux multi-sauts:*** En raison des besoins stricts en énergie, la topologie multi–sauts est employée

9. ***Supporter l'interrogation:*** Une certaine forme de communication Sink-nœud devrait permettre l'interrogation des nœuds.

2.3.3. Défis faces à la conception d'un RCSF

Certains défis rencontrés lors de la conception d'un RCSFsont énumérés ci-dessous :

1. ***Contraintes matérielles:*** Concevoir les nœuds de capteurs miniaturisés à bas coût et à basse consommation énergétique.

2. ***Gestion de réseau sans fil:*** Fournir un support de gestion réseau sans fil à efficacité énergétique et une communication peer-to-peer robuste dans les réseaux multi-sauts. Les réseaux devraient également être auto-configurables et auto-curatifs.

3. ***Support du développement logiciel:*** Développer des architectures logicielles conçues étroitement avec l'architecture de traitement de l'information.

2.3.4. Comparaison d'un RCSF avec les réseaux semblables

Le tableau 2-1 récapitule la comparaison d'un RCSF avec des réseaux semblables. Il montre clairement que les protocoles utilisés pour les réseaux traditionnels ne peuvent pas être employés dans le contexte des RCSFs.

Tableau 2-1 : *Comparaison d'un RCSF avec des réseaux traditionnels*

Dispositif	**RCSF**	**MANET**	**Réseau cellulaire**	**Bluetooth**
Nombre de nœuds	Très grand	Peu	grand	1 maître et 7 nœuds esclaves
consommation critique d'énergie, ressources de traitements	très haut	bas, puisque les dispositifs peuvent être rechargés	bas, puisque les stations de base ont l'énergie illimitée	Bas
Largeur de la bande passante exigée	Bas	Moyen-haut	Moyen-haut	Moyen-haut
Importance de QOS et d'autres conditions d'exigées	très bas	Haut	haut	Haut
Identification des nœuds	Ne peut pas être présent pour éviter l'overhead	Présent	Présent	Présent
Paradigme de communication utilisé	Émission	Point à point	Mobile à la station de base	esclave à maître

2.3.5. Configuration matérielle d'un nœud de capteur

Le développement et l'implémentation d'un nœud de capteur sans fil doivent prendre enconsidération la conception en concordance avec une bonne étude et analyse de l'application. Ceci exige la spécification et la définition des composants hardware aussi bien que les méthodes (software et modèle de programmation). Ceci offre une meilleure flexibilité et une efficacité durantle fonctionnement. De plus, il faut prendre en considération que le nœud de capteur dans unRCSF doit être de petite taille, moins chère, à efficacité énergétique, équipé de capteur(s),meilleures performances de traitement, taille de rangement (mémoire) adéquate, et des facilités decommunications souhaitables. La figure 2-4 illustre une configuration hardware typique d'un nœud de capteur.

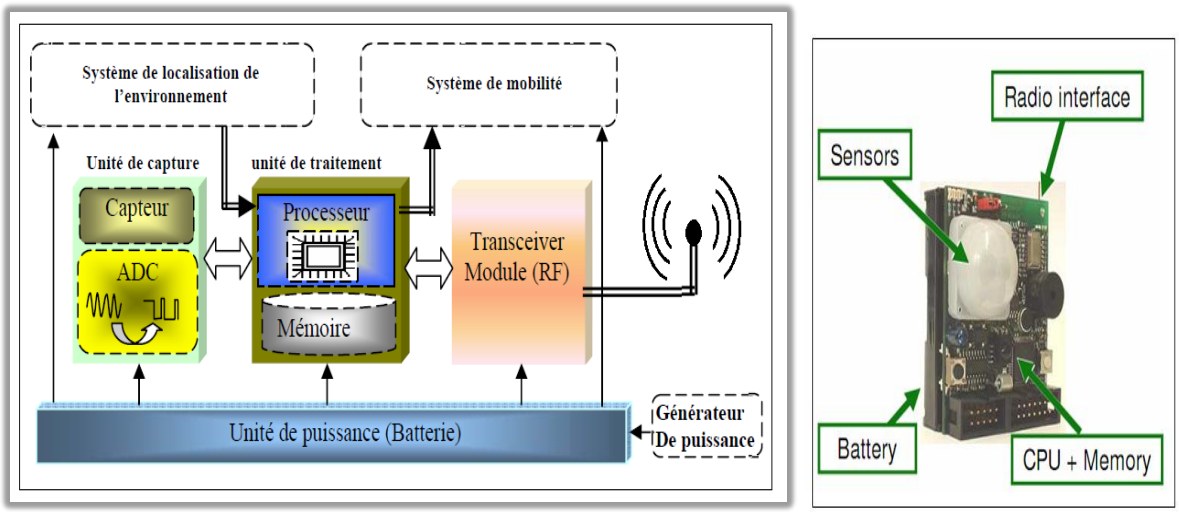


Figure 2-4: *Configuration matérielle d'un nœud de capteur (composants optionnels en pointillés)*

La configuration de base d'un nœud de capteur est comme suit :

- *Unité de traitement* : considérée comme le cœur du nœud capteur et regroupe à la fois le microcontrôleur et la mémoire. Le microcontrôleur exécute les logiciels d'application et la pile protocolaire. Celui-ci est utilisé dans les nœuds capteurs et estconçu en général pour consommer moins d'énergie. Dans plusieurs conceptions de nœuds capteurs, des processeurs 8-bit et 16-bit, tels que le MSP430 de Texas Instruments [Web07] ou ATmega 128 de Atmel [Web08] avec des vitesses d'horloge estimées à quelques dizaines de MHz, sont utilisés. La mémoire représente le plus souvent seulement quelques Ko à 1 Mo [18, 24] de RAM et quelques dizaines de Ko de mémoire flash pour le stockage du code et des données.

- *Capteurs (sensors)*: Comprend un capteur réel (capteur) qui collecte les données en format analogique, et le convertisseur analogique numérique qui convertit les données rassemblées en format numérique. Sur un nœud, on peut placer plusieurs capteurs (thermique, optique, de pression, acoustique…). Il est aussi l'interface avec le monde physique. Il y a une grande variété de capteurs, mais le choix de l'un doit se soucier des métriques de la taille et de la consommation énergétique requises par l'application. Quelques caractéristiques identifient les capteurs autant que :

    - Capteur passif omni directionnel : capte des grandeurs physiques dans un point sans l'actualisation des paramètres environnementaux : capteur optique, microphone, capteur d'humidité ou de pression.

    - Capteur passif redirigé : posséde une bonne définition de l'orientation de la mesure : camera, capteur ultrasonique, etc. Ces capteurs exigent une orientation bien définie pour la capture des grandeurs.

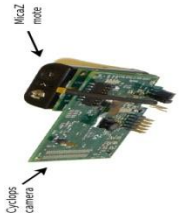
    - Capteur actif : c'est un capteur qui ne cesse de prendre des prélèvements continuellement de son environnement : sonar, radar, sismique, etc.

- *Sous système de communication :* La tâche fondamentale de cet élément est la conversion d'un courant de bits provenant du contrôleur en ondes radio et vice versa. Cela veut bien dire que le transceiver intègre les fonctions de transmission et de réception (Tx/Rx) [19]. Il est équipé d'une antenne omnidirectionnelle et responsable de la modulation en émission et de la démodulation en réception des données digitales sur un canal Hertzien sans fil. La plupart des concepteurs de nœuds capteurs prévoient la bande de fréquence radio sans licence, industrielle, scientifique et médicale ISM à 2.4 GHz, pour laquelle plusieurs circuits transceivers sont disponibles sur le marché. Les concepteurs de ces transceivers dédiés aux nœuds capteurs privilégient l'optimisation de la consommation d'énergie plutôt que le débit ou le taux d'erreurs. Ceci peut être effectué par une conception à faible puissance et un choix relativement petit des puissances de sortie. Les transceivers qu'on trouve aujourd'hui dans les nœuds capteurs (comme par exemple RF Monolithics TR 1001 ou Chipcon CC 1000 et 2420) ont une puissance de sortie de l'ordre de 1mW, alors que celle utilisée dans les réseaux locaux sans fil de la norme 802.11 est plus de 100mW, ou celle des réseaux cellulaires de type GSM qui dépasse largement cette valeur. Ainsi, le débit de transmission des données de ces transceivers varie entre des dizaines et des centaines de Kbps. L'implémentation des techniques de protocoles de routage et/ou MAC est intrinsèquement liée à la conception de cet élément hardware [19].

- *Unité de puissance :* la batterie, en tant que source d'alimentation électrique principale, fournit l'énergie électrique aux différents sous-systèmes d'un nœud capteur. Les batteries utilisées dans ce cas ont une quantité d'énergie très limitée, bien que certains types de batteries arrivent à se recharger eux-mêmes suite à certains processus chimiques sous certaines conditions [20]. Dans la plupart des applications des RCSFs, il est impraticable ou souvent impossible de remplacer ou de recharger les batteries des nœuds capteurs. Ainsi, cette quantité très limitée d'énergie limite par conséquence la durée de vie des nœuds de capteurs et par conséquent une durée de vie très limitée du réseau. Il est vrai que les différentes techniques d'extraction de l'énergie, à partir de l'environnement ambiant (solaire, vibrationnelle, gradients de température, humaine, …) , sont à l'heure actuelle très limitées, mais les combiner aux différentes stratégies effectives de gestion de puissance au sein d'un nœud capteur permet de prolonger d'une manière significative sa durée de vie.

Une comparaison entre quelques plates-formes de nœuds capteurs commerciales est dressée dans le tableau 2-2tandis-que la figure 2-5 présente quelques modèles des nœuds capteurs existant sur le marché. D'autres architectures au stade de prototypage dédiées à la recherche existent, par exemple : BTnode [Web09] (Europe), EYES [Web10] (Europe-Suisse), ZigbeX mote [Web11] (Corée du sud), T-Engine [21] (Japan), LiveNode [22] (France), BEAN (Brazilian Energy-Efficient Architectural Node) [23] (Bresil), Scatterweb [Web12] (Allemagne).

Tableau 2-2: *Comparaison de quelques nœuds capteurs*

Type de Nœud	**Telos** (Intel [Web13]) 	**Mica2**(Berkeley[Web14]) 	**SunSPOT** (Sun [Web15]) 	**Imote2** (Crossbow[Web16]) 
Type de Microcontrôleur	8 Mhz, 8 bits	7.37 Mhz, 8 bits	180 Mhz, 32 bits	13-416 Mhz, 16 bits
RAM	2 Ko	4 Ko	512 Ko	256 Ko
ROM	256 Ko	512 Ko	4 Mo	32 Mo
Bande passante	250 Kbps	38.4 Kbps	250 Kbps	250 Kbps
Capacité de la batterie	Coin cell 1000 mAh	2xAA 5700 mAh	Rechargeable 750 mAh	3xAAA 3750 mAh
Portée radio maximale (m)	100	150-300 m	100	100
Radio	Chipcon CC2420 2.4GHz 250 Kbps IEEE 802.15.4	Chipcon CC1000 315/433/33/868/916 Mhz 38.4 Kbauds	Chipcon CC2420 2.4GHz 250 Kbps IEEE 802.15.4	Chipcon CC2420 2.4GHz 250 Kbps IEEE 802.15.4
Système d'exploitation	TinyOS	TinyOS	Squawk (Machine virtuelle Java)	TinyOS
Connexion au PC	Port série	Carte de programmation	Interface USB	Interface USB

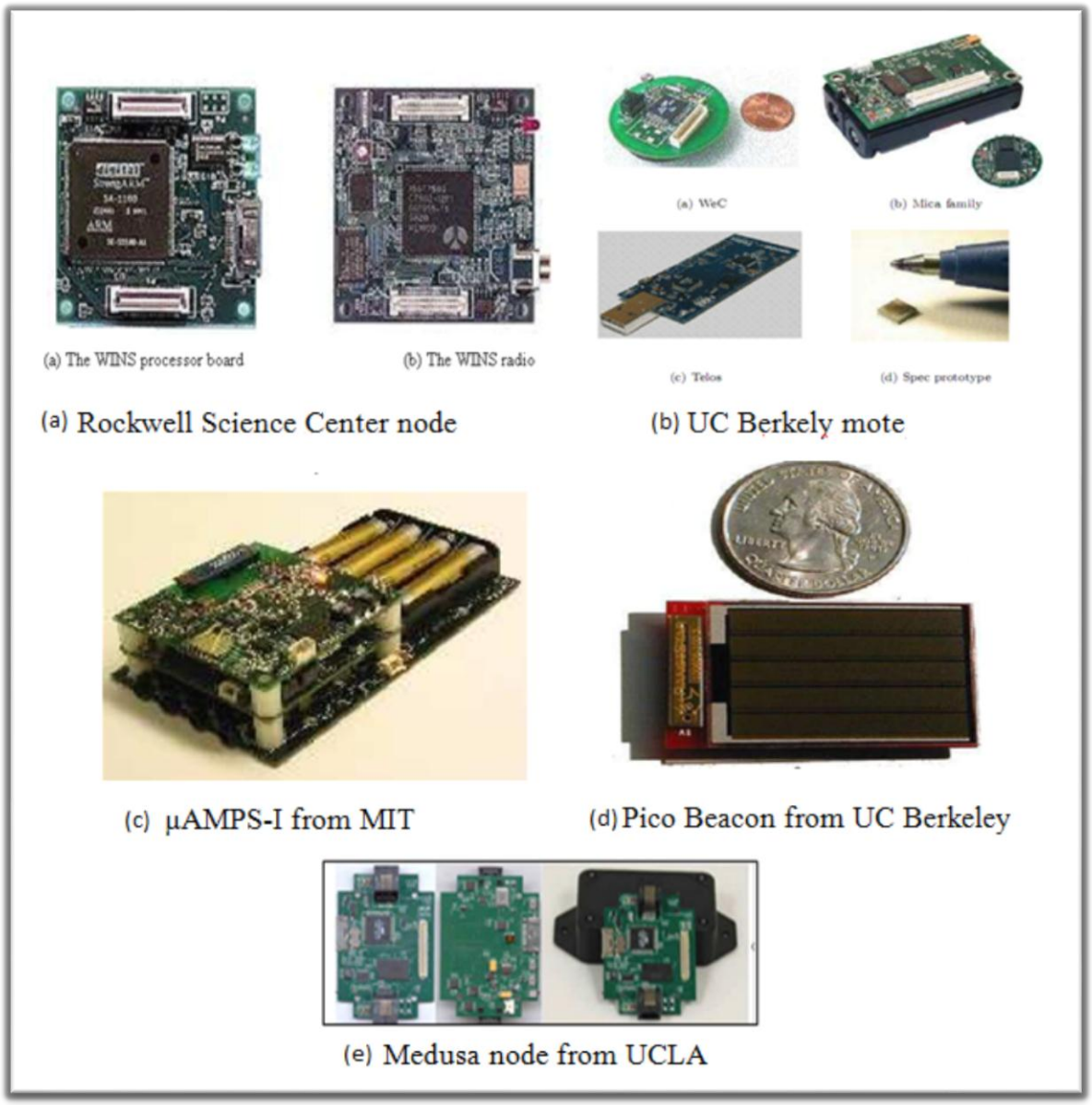


Figure 2-5: *Quelques modèles de nœuds capteurs*

2.4. Pile protocolaire des réseaux de capteurs sans fil

L'architecture protocolaire d'un nœud capteur peut être représentée par deux piles protocolaires : pile capteur et pile réseau (inspirée de [6] et [25]). La Figure 2-6 montre bien que la première pile est liée au canal de capture et la seconde au canal sans fil de communication. A ce niveau, Il est à noter que l'architecture protocolaire du Sink possède au moins une pile réseau mais pas de pile capteur. Comme illustré par la Figure 2-6, un nœud capteur, par le biais de sa couche physique, peut seulement recevoir des signaux des stimuli sur des évènements, produits dans la zone de sa couverture, à travers le canal capteur. Un nœud capteur reçoit et détecte un stimulus seulement si la puissance du signal reçu est dans une fourchette prédéterminée, sinon le signal reçu est considéré comme étant un bruit.

La coordination entre les deux piles protocolaires est réalisée à l'aide des couches application et transport. Par exemple, après avoir reçu le résultat de la capture (i.e. les données brutes), le nœud capteur a la possibilité, soit de relayer directement les paquets de données en direction du Sink, soit de les traiter localement, puis les transmettre ensuite vers le Sink. Tout traitement au sein du réseau s'effectue au niveau de la couche application d'un nœud capteur. Dans ce qui suit, nous explicitons la pile réseau et les cinq plans d'énergie, de mobilité, des tâches, de la qualité de service, et enfin celui de la sécurité [26-30] et [33] :

1. ***La couche physique*** : Cette couche est responsable de la modulation/démodulation, cryptage/décryptage, du choix de fréquence, de génération de la porteuse, et d'autres processus spécifiques au medium tel que les fonctions de détection du signal (CCA : *Clear Channel Assessment*), et la synchronisation des trames de données et le chiffrement. La norme IEEE 802.11 est utilisée.En général, la puissance minimale de sortie exigée pour transmettre un signal au-delà d'une distance d est proportionnelle à d^a, où $2 \leq a \leq 4$.

2. ***LacoucheMAC*** : permet d'interfacer les couches physique et réseau, elle comprend deux fonctions séparées : le contrôle d'erreur de transmission et le contrôle d'accès au médium (MAC). En partant du principe que l'environnement est hostile et non parfait (présence d'interférences) et que les nœuds peuvent être dotés d'un système de mobilité, le protocole MAC de la couche liaison de données doit connaître l'état de consommation de l'énergie résiduelle et il est capable de réduire au minimum toute source de perte d'énergie (par exemple lors des collisions).

3. ***La couche réseau*** : doit être capable de fournir des chemins de communication entre les nœuds sources collecteurs d'informations environnementales de leurs entourages, et la destination qu'est la station de base. Les protocoles de routage dans les RCSFs diffèrent des ceux des réseaux filaires par plusieurs choses : Premièrement, les nœuds capteurs ne possèdent pas d'adresses IP, ce qui fait que les protocoles de routage fondés sur le protocole IP ne fonctionnent plus sur les RCSFs. Deuxièmement, la conception des protocoles réseau pour les RCSFs doit prendre en le facteur du passage à l'échelle (scalability), ces protocoles doivent gérer aisément les communications entre plusieurs nœuds capteurs et acheminer les données captées vers le Sink. En troisième lieu, ces protocoles doivent prendre en considération les contraintes de ressources limitées, telles que l'énergie, la bande passante, la mémoire et les capacités de calcul. Ceci favorise l'assurance de prolongation de la durée de vie du réseau. Et enfin, ces protocoles doivent également évoquer certains problèmes liés par exemple à l'efficacité, tolérance aux fautes, équité et la sécurité.

4. ***La couche transport :*** permet de maintenir le flux de données si le RCSF l'exige. Elle assure la fiabilité et la qualité des données au niveau de la source et au niveau du Sink. Les protocoles de la couche transport pour les RCSFs doivent supporter différentes applications, une fiabilité variable et des mécanismes de récupération des paquets perdus ainsi que le contrôle de congestion. Le développement d'un protocole pour la couche transport doit être générique et indépendant de l'application. Il doit offrir une fiabilité variable des paquets pour différentes applications.

5. ***La couche application*** : emploie différents types de logiciels d'application, selon les tâches de capture. Bien que plusieurs domaines d'application pour les RCSFs soient définis et proposés, les protocoles potentiels de cette couche demeurent un axe de recherche, en grande partie, encore inconnu. Trois protocoles peuvent être considérés

dans cette couche : *Sensor Management Protocol* (SMP), *Task Assignment and Data Advertisement Protocol* (TADAP) et *Sensor Query and Data Dissemination Protocol* (SQDDP) [29].

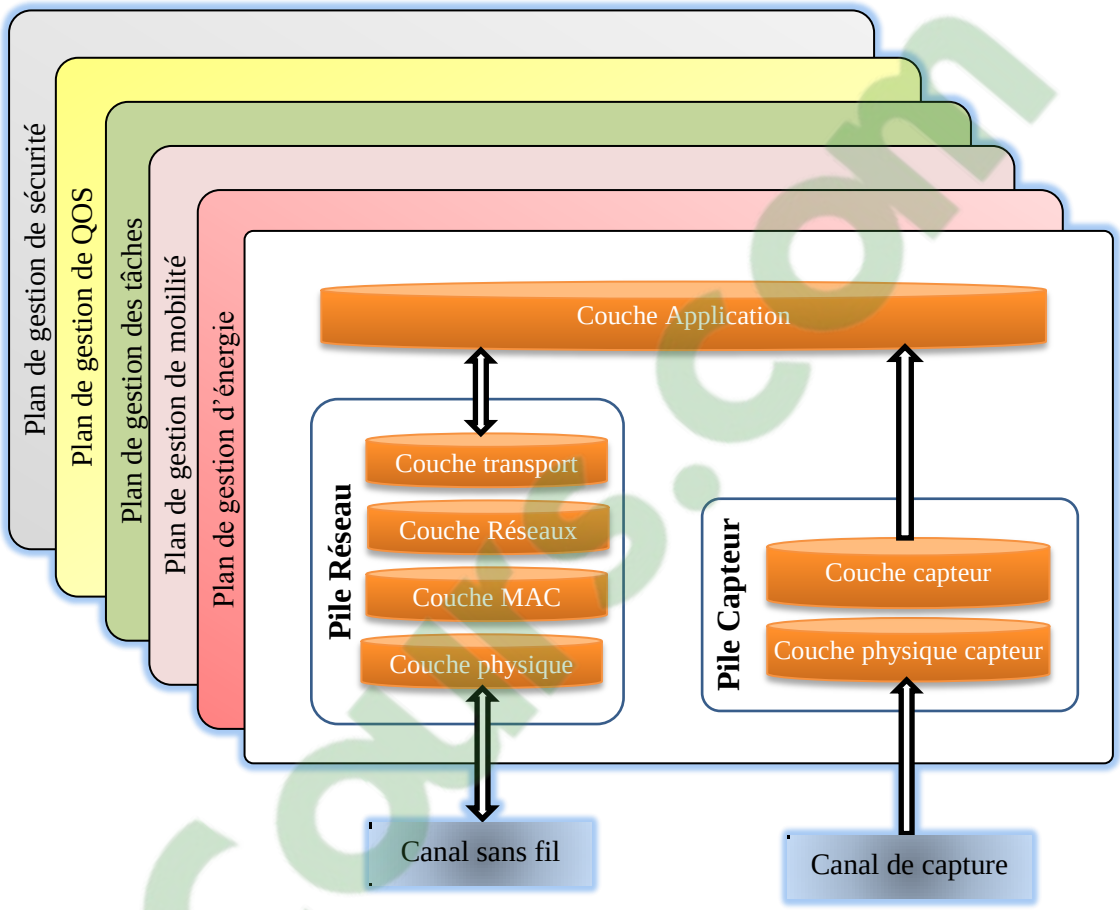


Figure 2-6: *Architecture protocolaire d'un nœud capteur*

1. ***Plan de gestion de l'énergie:*** permet de gérer la distribution et l'utilisation de l'énergie au niveau de chaque nœud capteur afin de favoriser une meilleure gestion d'énergie. Par exemple, lorsqu'un nœud constate que son énergie résiduelle passe au-dessous d'un certain seuil, il sollicite ses nœuds voisins de le priver de dispenser du routage afin de préserver la quantité d'énergie restante pour la capture et la transmission.

2. ***Plan de gestion de mobilité*** : permet de détecter et d'enregistrer les mouvements des nœuds capteurs dans le champ de déploiement, ainsi, un chemin vers l'utilisateur final est toujours maintenu, et les nœuds capteurs peuvent toujours connaître les positions de leurs voisins. Ainsi, les nœuds capteurs peuvent assurer un équilibrage de charge au niveau de la consommation énergétique et la gestion des tâches.

3. ***Plan de gestion des tâches :*** planifie l'équilibrage et le traitement des tâches de capture relatives à une région spécifique. Il n'est pas nécessaire de laisser simultanément tous les nœuds capteurs dans une telle région pour effectuer la collecte des données, il suffit de désigner un sous ensemble de nœuds pour assurer ces tâches selon les niveaux de capacité énergétique dont ils disposent**.**

4. ***Plan de gestion de la qualité de service (QoS management plane):*** Permet d'assurer une qualité de service exigée par l'application telle que la latence, streaming et la bande passante. Il traite aussi la tolérance aux fautes, contrôle d'erreur et l'optimisation des métriques QOS [31, 33].

5. ***Plan de sécurité (Security Plane) :***la gestion de la sécurité est un processus de gestion, de surveillance, et de contrôle lié à la sécurité du comportementdu réseau.La fonction primaire de la gestion de la sécurité est le contrôle des points d'accès aux données critiques et sensibles.Elle intègre les modules de sécurité tels que le cryptage, décryptage, authentification, et la détection des intrus [31, 33].

2.5. Quelques applications des RCSFs

La combinaison des capacités de capture avec ceux de traitement et de la communication des RCSFs a naturellement attiré plusieurs domaines d'applications quiont mis en œuvre les facilités que les capteurs leur offrent. Une liste non exhaustivede telles applications est donnéeci-dessous et résumée par la figure 2-7 :

- *Le domaine militaire :* comme pour beaucoup d'autres domaines, les applications militaires ont été les locomotives de la recherche pour les réseaux de capteurs. Pour les militaires, un réseau de capteurs offre des avantages très précieux. Il s'agit d'un réseau qui s'installe rapidement, dynamiquement et sans aucune infrastructure. Ainsi, il offre un atout de taille pour surveiller les mouvements de l'ennemi, communiquer à bas coût entre les unités avec une logistique peu compliquée.
- *Surveillance de l'environnement* : la petite taille et les capacités relativement grandes en matière de calcul et de communication des nœuds capteurs, permettent de les placer aux endroits là où les humains ne peuvent ou ne veulent pas y accéder, comme par exemple les forêts denses, les volcans, les profondeurs des océans, les régions polaires, ou encore d'autres planètes que la terre [32]. On peut aussi utiliser les RCSFs pour la surveillance du degré de maturité des récoltes (raisin), la mesure de la qualité de l'eau ou de l'air : surveillance du trafic, surveillance d'habitat, agriculture etc.
- *Sensation et diagnostiques industriels :* les industriels s'intéressent au potentiel des capteurs pour diminuer les coûts du contrôle et de la maintenance des produits, de la gestion de l'inventaire, de la télésurveillance après-vente, etc. [32]. En particulier, l'intégration de la technologie RFID avec les réseaux de capteurs est une des directions prometteuses de recherche dans l'industrie.
- *Les domaines urbains et domotiques :* les capteurs entrent de plus en plus dans nos vies quotidiennes. Dans le milieu urbain, les capteurs sont déjà utilisés pour la localisation des bus, pour des tickets électroniques et pour la sécurité. Une des applications est la surveillance du trafic routier avec les réseaux de capteurs déployés sur les autoroutes. De plus, les maisons, les bâtiments, les bureaux équipés de capteurs intelligents permettent de construire des systèmes pervasifs où l'information est omniprésente [34].
- *Le domaine médical :* la recherche sur l'usage des capteurs intelligents dans le domaine médical inclut les moyens d'hospitalisation à domicile, l'intégration des micro-capteurs "dans" le corps (e.g. construire un BAN-Body Area Network) et la gestion des urgences [35]. Parmi les applications les plus utiles,

on cite la télésurveillance des signes vitaux et des niveaux d'activité à domicile des personnes âgées ou handicapées ainsi que le contrôle à distance des données physiologiques : p*rotection d'infrastructur, emplacements de desastre, sports, etc.*

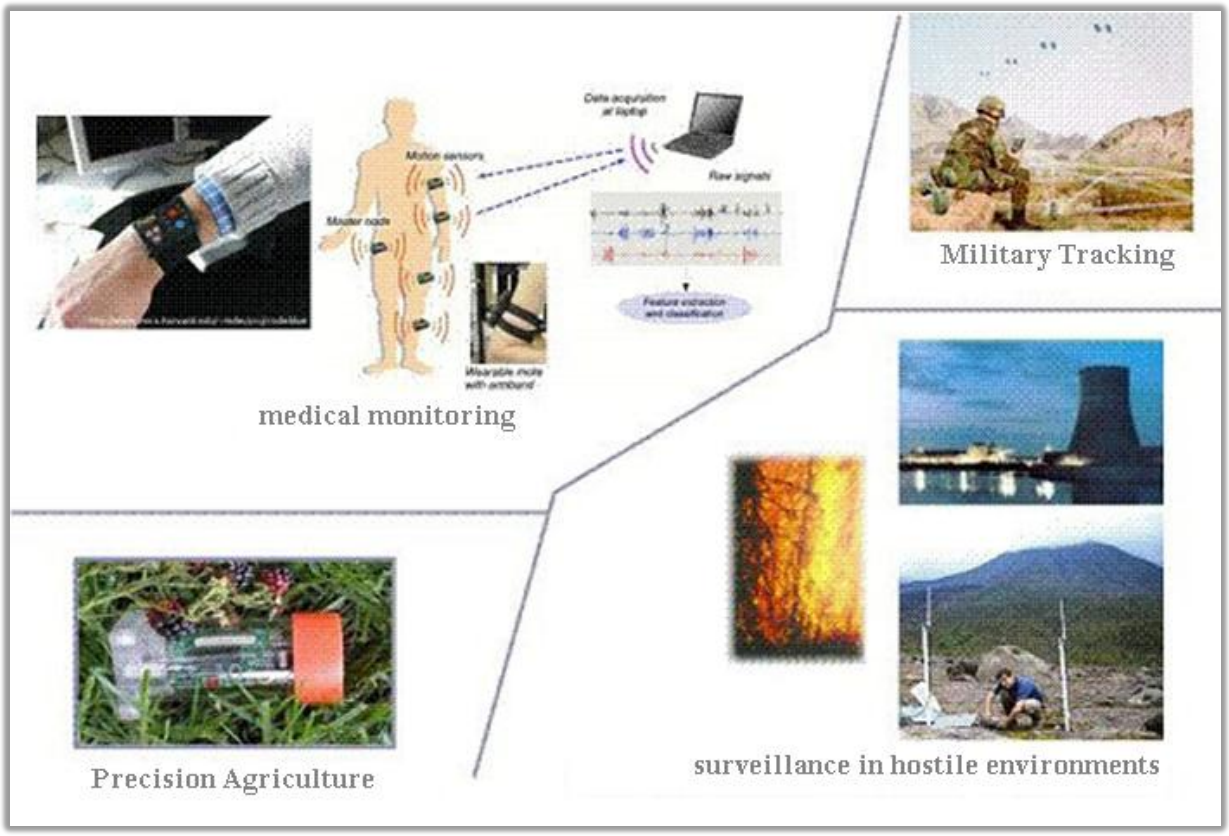


Figure 2-7: *Domaines d'applications des RCSFs*

2.6. Architecture d'un RCSF

L'architecture typique d'un RCSF est illustrée par la figure 2.10. Deux types de réseaux sont représentés : le réseau d'acquisition des données environnementales, et le réseau de transport. Ces deux réseaux permettent une communication dans les deux directions par rapport au RCSF. La première, appelée direction ascendante ou *Uplink direction*, a pour origine les nœuds capteurs et comme destinataire final le centre de traitement exploité par un utilisateur final. La seconde direction, appelée direction descendante ou *Downlink direction*, assure la communication inverse c'est-à-dire du centre de traitement vers les nœuds capteurs. Les communications dans les deux directions utilisent un ou plusieurs nœuds Sink et un réseau de transport comme moyen de communication intermédiaire [6].

Comme le montre la figure 2-10, un RCSF est déployé soit de façon géométrique soit d'une manière arbitraire. Dans les deux cas, les nœuds doivent transmettre les grandeurs physiques, récoltées de leur environnement vers la station de base (Sink) ou répondre à des requêtes du sink. Pour ce faire, les nœuds, selon les requis de l'application, s'organisent dans des groupes appelés clusters (dans ce cas on parle de réseau à architecture hiérarchique), chaque cluster est gouverné par un clusterhead élu parmi l'ensemble des nœuds membres du cluster par un algorithme d'élection selon certains critères, et qui a la responsabilité de collecter les données, gérer le groupe, agréger les données, et les communiquer au sink (voir

figure 2-8). Les communications des nœuds membres (voisins situent dans la portée de sa radio) convergent vers lui. Les données collectées et/ou reçues par le clusterhead seront agrégées et/ou compressées puis transmises vers le Sink avec un signal plus puissant. Les clusters sont de deux natures : les clusters physiques dans le cas des nœuds immobiles et dont les coordonnées sont fixes, et les clusters virtuels (ou logiques). Dans ce dernier cas,les nœuds sont mobiles et le changement topologique est fréquent, car un nœud et a tous moment quitte un cluster pour migrer vers un autre cluster, ce qui exige des synchronisations périodiques en vue d'identifier les nœuds actuellement présent dans le cluster. Ce type d'organisation permet d'augmenter la durée de vie du réseau de capteurs [36]. Il n'y'a qu'un seul nœud dans un groupe qui prend en charge l'acheminement des données agrégées vers le Sink, contrairement à l'architecture plate ou tous les nœuds y participent. Parmi le grand nombre de protocoles de routage basés sur le concept du cluster et proposés dans la littérature, nous citons: LEACH (Low Energy Adaptive Clustering Hierarchy) [37], HEED (Hybrid Energy-Efficient Distributed Clustering) [38], PEGASIS (Power-Efficient Gathering in Sensor Information Systems) [39], TEEN (Threshold-sensitive Energy Efficient sensor Network protocol) [40].

Les RCSFs sorganisent sous deux formes: topologie hiérachique ou clustering et la topologie plate.L'essence de la topologiehiérarchique, repose sur le fait que la remontée des prélèvements des mesures physiques vers le Sink passe soit directement et par amplification du signal comme le montre la figure 2-9 (a), ou via des nœuds intermédiaires, et dans ce cas il s'agit du mode de communication multi-sauts (*multihop communication mode*), voir figure 2-9 (b). Les nœuds intermédiaires figurants dans le chemin liant la source au Sink sont des nœuds routeurs.En voici quelques protocoles de routage de ce type de topologie: Directed Diffusion [41], SAR (Sequential Assignment Routing) [42] et SPIN (Security Protocols for Sensor Networks) [43]. Cependant, une fois la taille du réseau devienne un peu plus grande, sa gestion sera compliquée et l'acheminement des données de la source à la station de base devient difficile. Cefait engendre un gaspillage d'énergie réduisant ainsi la durée de vie du réseau [36].

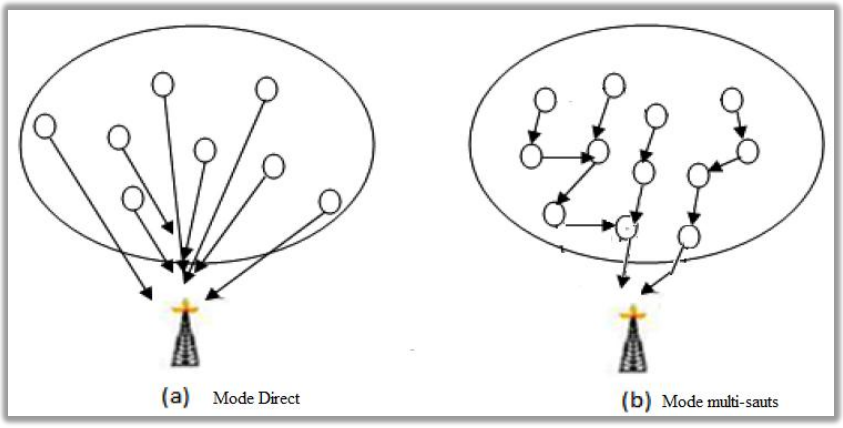


Figure 2-8: *Architecture Plate*

2

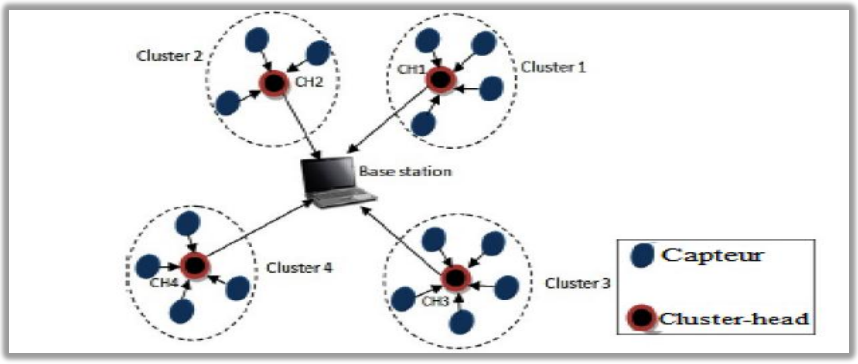


Figure 2-9: *Architecture hiérarchique à base de cluster*

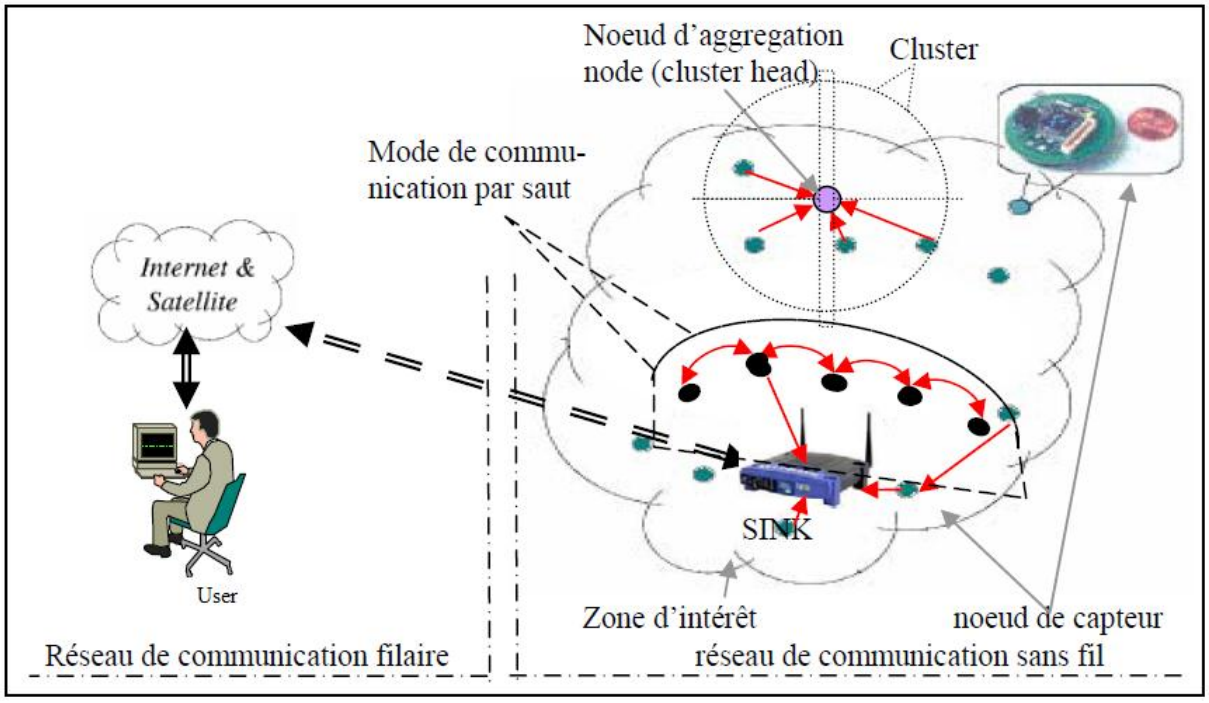


Figure 2-10: *Architecture typique d'un RCSF*

2.7. Vulnérabilité des RCSFs

LesRCSFs sont déployés dans de nombreux domaines d'application à temps réel, tel que les champs de batailles, les catastrophes, surveillance industrielle, et autres applications, ce qui peut être une cible attrayante pour les attaquants. Par conséquent, il est impérativement nécessaire d'intégrer un mécanisme de sécurité dans ces réseaux vulnérables. Pour de tels divers scénarios à temps réel, la sécurité des RCSFs devient une tâche complexe, nécessaire, et une considération majeure à prendre en compte lors du déploiement de ce type de réseaux. Pour chaque application, chaque implémentation, et chaque scénario il y'a différents sortes d'attaques possibles ciblant la nuisance aux données environnementales captées et/ou fonctionnement direct du réseau lui-même. Chaque attaque requière des mécanismes et des niveaux de sécurité différents pour y faire face.

Le principal défi à la réalisation d'un mécanisme de sécurité efficace pour les RCSFs provient des contraintes naturelles des ressources physiques des nœuds capteurs eux-mêmes [44]. Comme il est détaillé dans la section 2-2-8, un nœud capteur est considéré comme un nano-computer doté d'une unité de traitement limitée, d'une mémoire restreinte, d'un tranceiver à une bande de fréquence basse, et d'une batterie à faible énergie [45-46].

La nature du canal de communication sans fil fragilise les RCSFs et les rends plus vulnérables face à de multiples attaques sévères telque : (*DOS, Sybil, Sinkhole, Wormhole, selective forward, Black hole, etc.*) [47-50], Il suffit à un intrus doté d'un laptop équipé d'une antenne radio adaptée pour intercepter les communications.

Les contraintes environnementales et les requis du déploiement d'un RCSF rend la sécurité pour ce type de systèmes une tâche assez compliquée comparée à celle pour les réseaux conventionnels [51]. Lors de la conception d'un protocole de sécurité pour les RCSFs il faut garder en tête une liste de défis, car chaque application possède ses propres challenges, propriétés et buts [52, 65].

- Resource efficaces des services réseau sécurisés telque : découverte de voisinage, initialisation du réseau, routage multi-chemins, etc.
- Utilisation des services de cryptographie telque l'authentification en mode diffusion et la gestion des clés par des algorithmes légers.
- Le mécanisme de sécurité doit être garanti pour tous les services fondamentaux telque : l'agrégation des données, formation des clusters, localisation, découverte, etc.
- Nature d'environnement de déploiement hostile.
- Nature du canal de communication ainsi que le mode de communication multi-sauts.
- La topologie des RCSFs change aléatoirement et rapidement. Les nœuds capteurs quitent et entrent dans le réseau continuellement.
- Les RCSFs sont des réseaux auto-configurables et auto-organisés.

Plusieurs autres problèmes requièrent plus de travaux de recherches. Par exemple: comment sécuriser un lien de communication sans fil face aux attaquesde sécurité telle que l'écoute clandestine, falsification, analyse du trafic et denie de service.

2.8. Conclusion

Dans ce chapitre, nous avons présenté d'une manière générale les RCSFs, ce qui nous a mené à survoler les réseaux Ad-hoc, ensuite les réseaux de capteurs sans fil, dont nous avons abordé l'architecture, le mode de communication et la topologie. Les exigences des applications et la nature hostile des zones de déploiement de ce type de réseaux, fragilisent leurs fonctionnements face à de fortes probables attaques. Les contraintes physiques des nœuds capteurs augmentent d'un cran les challenges à réaliser un système de détection d'intrus (IDS) efficace.

3. SECURITE : NOTIONS DE BASE, TAXONOMIE, PROTOCOLES DE SECURITE ET IDS

3.1. Introduction

Dans le chapitre précédent nous avons mentionné, l'utilité des RCSFs et leurs intérêts particuliers pour les applications militaires, médicales, et bien sûr les applications liées à la surveillance des infrastructures critiques. Par conséquent, la conception des protocoles de communication dédiés à ces applications suppose que tous les nœudsdu réseau engagés sont coopératifs, identiques, et dignes de confiance. Cependant, ceci n'est pas toujours le cas dans les déploiements du monde réel, où les nœuds sont exposés à de multiples attaques causant ainsiun endommagementdu bon fonctionnement du réseau. Ces attaques exploitent essentiellement l'incertitude du canal de communication et le déploiement aléatoire des nœuds capteurs dans des zones difficiles à contrôler et à surveiller.

Garantir la sécurité pour ce type de réseau reste une tâche très difficile, du moment où les paramètres réseau, contraintes physiques des nœuds, nature du canal de communication, et les exigences des applications sont des paramètres orthogonaux. Si on voit, les nœuds sont constitués d'engins électroniques peu onéreux avec des capacités matérielles limitées. Le cas échéant, utiliser des protections physiques est, dans beaucoup de situations, quasiment impraticable. Prendre le contrôle physique et/ou logique des nœuds est alors une possibilité intéressante pour les attaquants.

Néanmoins, les RCSFs ne peuvent plus compter sur l'intervention humaine pour faire face aux tentatives d'attaques pour compromettre le réseau ou gêner son comportement normal. Dans cette section, nous présenterons un aperçu sur les problèmes de sécurité dans les réseaux de capteurs sans fil. Nous commencerons par présenter les défis de sécurité pour les RCSFs qui rendent la sécurité pour ce type de réseaux une tâche presque impossible et assez compliquée. Ensuite, nous ferons une taxonomie des attaques et discuterons les besoins de sécurité requis par les protocoles.

3.2. Definition de la Sécurité

Avant de donner une définition de la notion de *sécurité*, il y'a des termes qui doivent être mis en évidence : ***Le risque*** en terme de sécurité est généralement caractérisé par l'équation suivante :

$$Risque = \frac{menaces \times vulnérabilité}{contre-mesure} \qquad (1)$$

La menace (en anglais « threat ») représente le type d'action susceptible de nuire dans l'absolu, tandis que la ***vulnérabilité*** (en anglais « vulnerability », appelée parfois faille ou brêche) représente le niveau d'exposition face à la menace dans un contexte particulier. Enfin la ***contre-mesure*** est l'ensemble des actions mises en œuvre pour prévenir la menace. Pour débuter, nous avons besoin de donner une définition appropriée qui convienne à notre problématique. Dans ce cas nous définins la sécurité informatique comme celle donnée dans [53]:

"Computer security is the protection of personal or confidential information and/or computer resources from individuals or organizations that would willfully destroy or use said information for malicious purposes."

La sécurité est definit comme : "freedom from risk or danger."[Web 18]. Et dans le context de l'informtique, la sécurité est la prévention de ou la protection contre :

- L'accès à l'information par une personne non autorisée.
- l'altération ou la destruction intentionnelle non autorisée de l'information.

Ceci peut être redefinis par:"Security is the ability of a system to protect information and system resources with respect to confidentiality and integrity." cette seconde définition inclue additionnellement l'information, les resources systèmetelles la CPUs, disks, and programs [Web 18, Web 19].

La sécurité informatique est trop souvent vue uniquement comme un ensemble de mesures opérationnelles de sécurité appliqué aux systèmes informatiques. En fait, il faut la voir comme un ensemble de mesures organisationnelles prises au niveau d'un organisme (entreprise, école, etc.) pour gérer le risque informatique d'une façon formelle. Cela est très proche de la gestion de la qualité et la gestion des risques.

Le but ultime est d'atteindre le niveau de confidentialité, de disponibilité, et d'intégrité des données, requis par le centre de décision, qui est l'ultime responsable. Ce but doit être ré-évalué constamment par des analyses de risques et la proposition de contre-mesures.

3.3. Contrôle d'accès et origine de la sécurité informatique

En 1976, Saltzer et Schroeder [54] enregistrent le début du concept du contrôle d'accès. En utilisant la théorie disant, qu'il est mieux de réduire l'accès aux ressources par défaut en arrière-plan et explicitement autoriser les usagers à y accéder à ces ressources que de réduire les droits et privilèges d'accès des utilisateurs.

Au cours des années, la communauté scientifique a formulé l'idée sur le contrôle d'accès, en construisant de modèles mathématiques prouvant les différentes politiques et règles de sécurité. Le modèle le plus complet et largement utilisé est appelé Matrice de contrôle d'accès (*access control matrix*). Visualisé par le tableau 3-1, la matrice de contrôle d'accès est sous forme d'une grille avec les ressources sur un axe et les entités pouvant accéder à ces ressources sur l'autre axe. Les éléments de la grille représentent les droits d'accès des entités aux ressources correspondantes. En utilisant ce modèle, nous pouvons représenter toutes les situations de sécurité pour n'importe quel système. Le nombre complet des possibilités rend difficile sa mise en pratique dans sa frome entière (*correspondre toutes les ressources et les utilisateurs possibles*) [53].

Tableau 3-1: *Matrice du contrôle d'accès*

	Alice (manager)	Bob (IT admin)	Carl (normal user)	Donna (temporary user)
C:\Users\Alice	RWG	RWG		
C:\Users\Bob	R	RWG		
C:\Users\Carl	R	RWG	RWG	
C:\Users\Donna	R	RWG		RW

R : Can read from directory
W:Can write to directory
G : Can grant other users read, write, or grant permission

3.4. Politiques de sécurité

La matrice du contrôle d'accès fournis des fondements théoriques définissant la notion de sécurité, et non pas des méthodes pratiques pour la mise en œuvre des systèmes de sécurité. Pour cela, nous avons besoin d'une politique de sécurité. L'idée derrière est simple : un ensemble de règles qui doit être appliqué et imposé par le système afin de garantir certains niveaux de sécurité prédéfinis. Par analogie au système du code pénal, une politique de sécurité définit quelle entité (personne ou autre système) doit et ne doit pas faire telle ou telle chose. Lors de la conception d'une application sécurisée, une partie des conditions est une liste des objets du système qui doivent être protégés. Cette liste doit former la politique de sécurité qui devrait être une partie intégrale dans la conception.

Lorsque vous déployez votre politique de sécurité, pensez que votre application est un château. Vous aurez besoin de protéger les occupants (*les habitants*) des attaques de l'extérieur et de s'assurer qu'ils sont bien (la figure 3- 2). Votre politique est donc une liste de contrôle (*checklist*) formée à partir des choses qui peuvent nuire ou endommager les habitants. Certaines choses ou pratiques sont claires et triviales tel que la fermeture des portes du château et la vérification des identités des entrant avant de leurs permettre d'entrer, ou s'assurer que les habitants fassent leurs devoirs. Malgré cela et afin de développer une vraie politique de sécurité, vous devez penser comme un ennemi. Quelles sont les autres possibilités? Pour un ennemi, prendre le contrôle total du château peut ne pas l'intéresser, par contre empêcher son fonctionnement correctest une bonne attaque en empêchant l'approvisionnement du château (denial of service attack). Certaines attaques subtiles non notées au départ, deviennent un problème sérieux, tel que : engager un habitant de l'intérieur pour saboter les lignes internes de la défense, ou empoisonner les ressources en eaux (virus) [31].

L'ennemi compte aussi sur l'intelligence. Le cheval de Troie est un exemple vivant. Les virus Trojans offrent aux hackers une porte noire ou porte dérobée (accès invisible) vers le système.L'ennemi peut ne pas être reconnu autant qu'ennemi, dans le cas des refugiés de guerre, soudainement ils commencent par consommer l'approvisionnement et l'épuisement des ressources (*Worms*). Les possibilités se multiplient et on se sent découragé de penser à toutes les choses pouvant être un problème ou générer des problèmes.

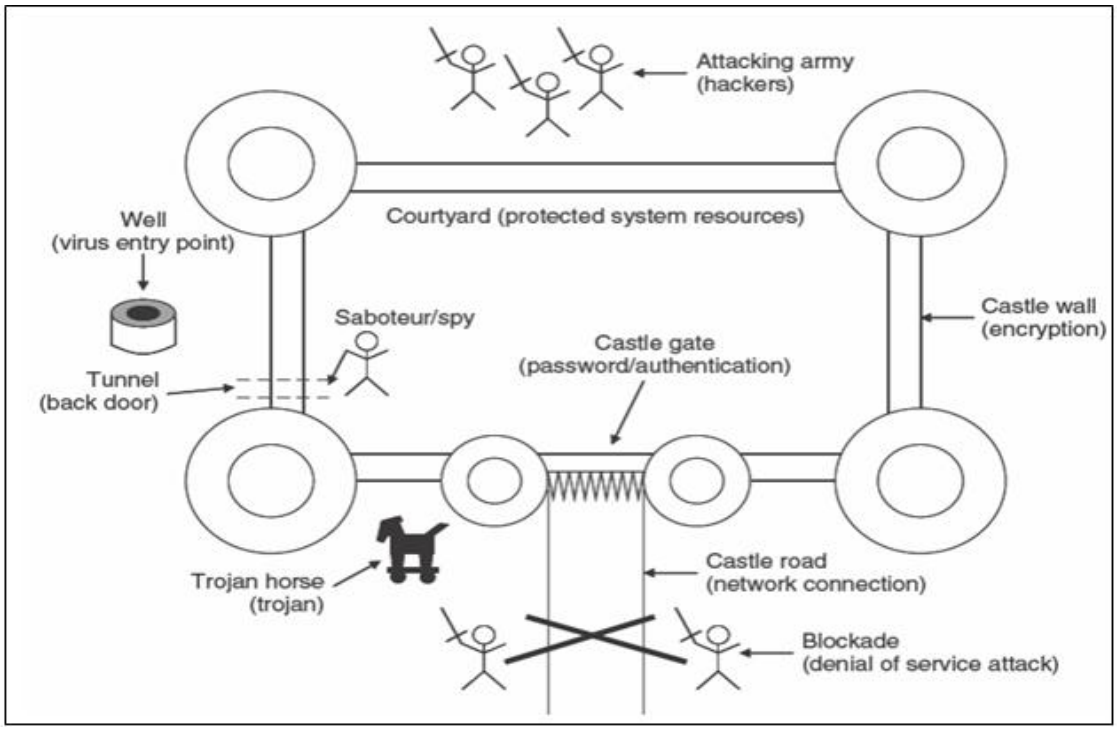


Figure 3-1: *Application château*

3.5. Les besoins de sécurité typiques aux RCSFs

Afin de pouvoir déterminer des objectifs de sécurité, il faudra connaitre ce qu'on doit protéger. Les RCSFs partagent certaines caractéristiques et mode opérationnel avec les réseaux ordinaires filaires et les réseaux mobiles Ad hoc mais aussi possèdent des propriétés intrinsèques aux RCSF, abordées dans les sections précédentes (2-2-8 et 2-7).

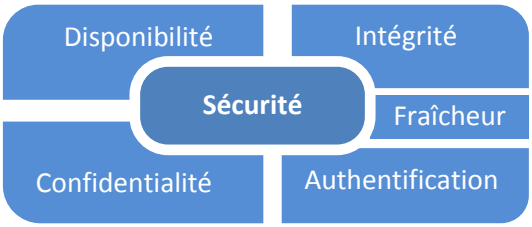


Figure 3-2: *Enjeux de la sécurité*

La figure 3-2schématise les objectifs de sécurité, tandis que la figure 3-3 est sous forme d'un organigramme listant quelques types d'attaques ciblant ces objectifs. Pour ce type de RCSF, ils englobent ceux des réseaux traditionnels et les objectifs issusdes contraintes intrinsèques aux RCSFs. Parmi les principaux objectifs de sécurité, nouscitons:

3.5.1. L'authentification

Chaque nœud du réseau aura besoin de connaitre si les paquets reçus proviennent des émetteurs dignes de confiance. L'authenticité est une assurance de l'identité du nœud communiquant. Elle apparaît comme la pierre angulaire d'un réseau de capteur sans filsécurisé. En effet, on ne peut assurer une confidentialité et une intégrité des

messageséchangés si, dès le départ, on n'est pas sûr de communiquer avec le bon nœud. Si l'authentification est absente ou mal gérée, un attaquant peut se joindre au réseau et s'autoproclamer membre du réseau, il lui suffit d'injecter des messageserronés pour tromper le comportement correct du réseau.

L'utilisation de Code d'Authentification de Message (CAM), ou MAC en anglais (Message Authentication Code), permet d'assurer à la fois l'authentification de l'origine et l'intégrité du message. Un exemple de MAC est HMAC [32].

3.5.2. La confidentialité

La confidentialité est une assurance d'accès autorisée à l'information. Dans le contexte du réseau, la confidentialité est synonyme de garder en secret les informations échangées sans une permission d'accès autorisée [30]. La confidentialité reste un point important, étant donné la communication sans fil dans les RCSFs. Elle consiste à préserver le secret des messages échangés et ne pas les révéler aux nœuds non authentifiés (*adversaires*). La confidentialité peut être assurée par l'usagede la cryptographie à clé symétrique ou asymétrique.

3.5.3. L'intégrité

Elle assure que les données communiquées n'ont pas été altérées durant leur transmission dans leréseau de manière volontaire ou accidentelle. Ceci est un requis élémentaire pour une communication, le nœud récepteur doit savoir exactement ce qui lui a été demandé par le nœud émetteur. Dans un canal sans fil, cette tâche n'est pas facile à assurer. Elle peut être assurée par l'utilisation desfonctions de hachage cryptographiques qui permettent d'obtenir pour chaque message uneempreinte numérique. Les fonctions MD2 (Message Digest 2), MD5, SHA-1(Secure Hash Algorithm 1) sont des exemples de quelques fonctionsde hashage les plus utilisées [110].

3.5.4. La disponibilité

C'est une métrique qui mesure le taux de garanti et d'assurance des services réseaux requis et conçus d'avance. Elle est un concept trop lié aux aspects du réseau. N'importe quel problème qui peut surgir au sien du réseau peut conduire à une dégradation d'un ou plusieurs services réseau et delà compromettre la disponibilité du réseau en provoquant un déni de service (DOS). Cette propriété reste difficile à assurerdans les RCSFs étant donné les contraintes qui pèsent sur ces réseaux, à savoir: topologie dynamique, ressources limitées des nœuds de transit, communications sans fil pouvant être facilement brouillées ou perturbées.

3.5.5. La fraîcheur

Les données décrivant un état temporaire d'un objet sont seulement valides dans un intervalle de temps limité. Par conséquent, lorsqu'un nœud reçoit un paquet de données, il doit être sûr que ces données sont fraiches. Autrement, le contenu du paquet est inutile. Ce service permet de garantir que les données échangées sur le réseau sont actuelles et ne sont pas une réinjection de précédents échanges interceptés par un attaquant.

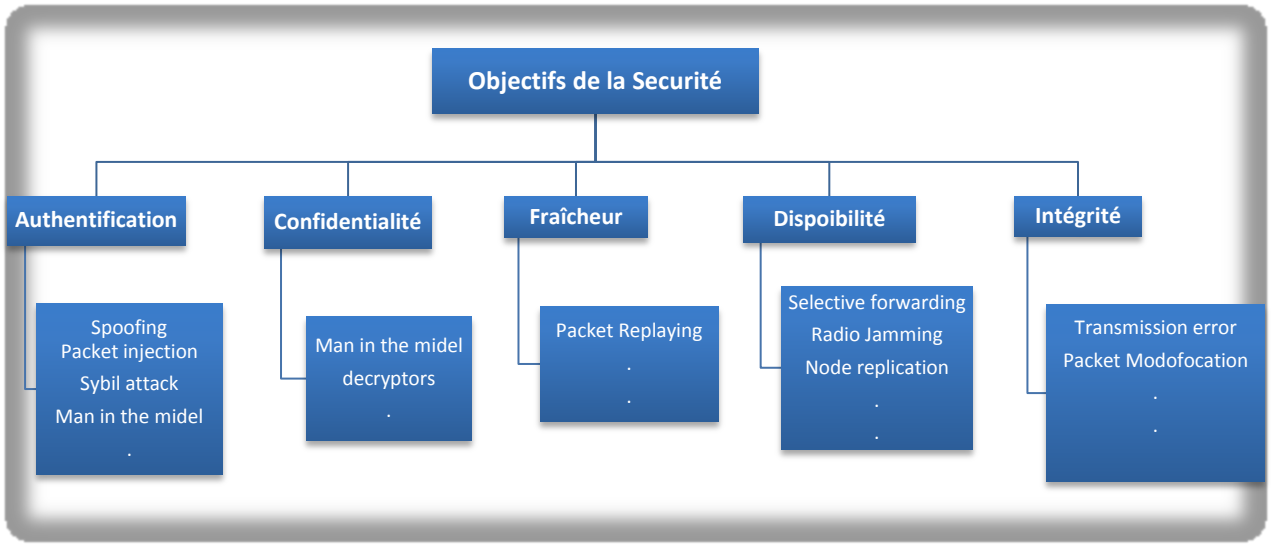


Figure 3-3: *Objectifs de la sécurité et quelques types d'attaques*

3.6. Classification des attaques

Dans les réseaux de capteurs, un attaquant peut effectuer une variété d'attaquesn'ayant pas forcément le même objectif et/ou motivations. Ainsi le choix d'une stratégie de sécurité pour faire face à un type d'attaque doit se baser sur une modélisation de cette dernière; ceci afin d'éviter un déploiement massif de moyens de protection conduisant à des solutions irréalistes. Les attaques sur les RCSFs sont classées selon plusieurs critères, tel que le domaine d'attaque et la technique suivie par l'attaquant. Généralement, les attaques de sécurité sont réparties comme le montre la figure 3-4 ci-dessous, en attaques passives ou actives, externes ou internes, cryptographique ou non-cryptographique, stealthy ou non-stealthy[56-57].

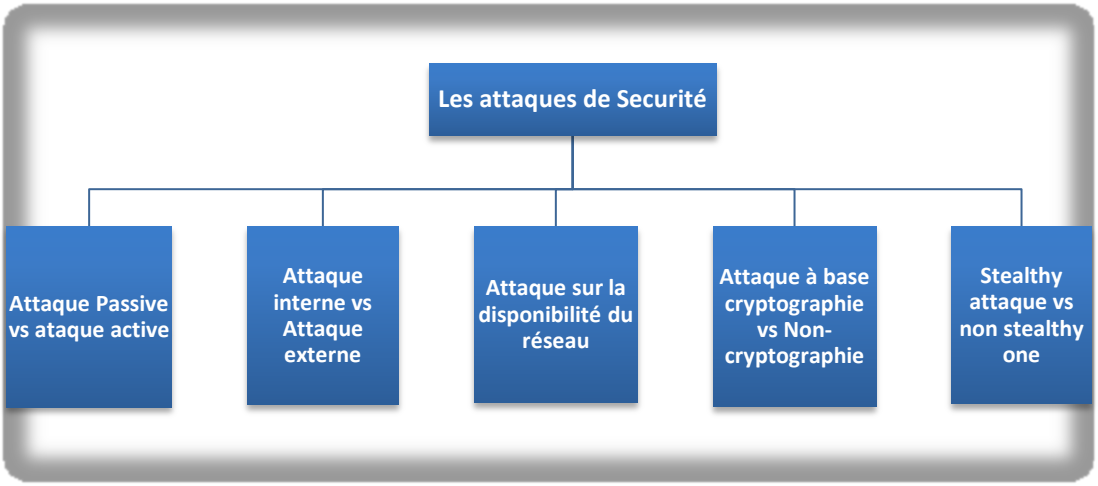


Figure 3-4: *Les attaques de sécurité dans les RCSFs*

3.6.1. Attaques passives contre attaques actives

Les attaques passives "eavesdropping" se limitent à l'écoute de la porteuse et l'analyse du traficréseau échangé. Ce type d'attaque est plus facile à réaliser (il suffit d'être équipé d'un récepteur adéquat), alors que sa détection est difficile du moment que l'attaquant n'apporte aucune modification sur les informations échangées et ne perturbe pas le trafic. L'intention de l'attaquant peut être la connaissance des informations confidentielles ou bien la connaissance des nœuds importants dans le réseau (clusterhead). En analysant les informations de routage, l'attaquant va sepréparer à mener ultérieurement une action précise.

Dans une attaque active, l'attaquant tente de supprimer ou modifier les messages transmis sur le réseau. Comme il peut aussi injecter des paquets qu'il a conçus ou injecter d'anciens messages afin de perturber le bon comportement du réseau ou provoquer un déni de service.

3.6.2. Attaques externescontre attaques internes

Dans le cas d'une attaque externe, le nœud attaquant est un élément externe par rapport au réseau et n'est pas autorisé à participer dansles services (*capture des grandeurs physiques, routage, et communication*) du RCSF. Des techniques de cryptographie et d'authentification protègent l'accèsau réseau vis-à-vis à ce type d'attaque. Cependant ce dernier peut uniquement déclencher des attaques passives telles que l'écoute clandestine, le brouillage radio, ou l'attaque par rejou (Replay attack). Par contre une attaque interne est considérée comme étant la plus *dangereuse* du point de vue sécurité. Puisque l'attaquant qui capture un nœud, peut lire le contenu de sa mémoire et s'emparer des données clés de contrôles et par là avoir accès à son matériel cryptographique ce qui conduit à ce que l'intrus peut s'authentifier comme un nœud légitime (*identity userpage*) et émettre des messages aléatoires erronés sans être identifié comme étant un intrus, puisqu'il utilise des clés valides. Les méthodes cryptographiques s'avèrent donc inefficaces pour ce genre d'attaque. Ilest donc nécessaire d'utiliser d'autres méthodes complémentaires telles que les systèmes de monitoring et les systèmes de réputations.

3.6.3. Attaques à base cryptographique

Une mauvaise implémentation des protocoles de sécurité est la cause de plusieurs failles sécuritaires. Par exemple, les protocoles d'authentification et d'échange de clés sont souvent la cible d'attaques malicieuses. Les primitives de cryptographie sont considérées être incassables; mais récemment la fonction (SHA-1) a fait l'objet d'attaque. Pseudo-random number attacks, digital signature attacks, et hash collision attacks sont des attaques cryptographiques. Dans l'attaque du nombre pseudo-aléatoire les paquets sont générés on utilisant le nombre aléatoire (*nonce*). La clé de session est souvent générée à partir d'un nombre aléatoire. Dans la cryptographie publique infrastructure, la clé privée peut être aussi générée à partir d'un nombre aléatoire. Dans la plus part des langages de programmation, le nombre générateur conventionnel est désigné comme un nombre aléatoire-statique, non résistant aux prédictions des crypto-analystes.

3.6.4. Attaques de l'intégrité des services

Dans une attaque de type furtive, l'objectif de l'attaquant est de pousser le réseau à accepter les fausses données. Par exemple, un attaquant injecte des données érronnées via les capteurs d'autres noeuds. Dans cette attaque, garder le réseau de capteurs disponible pour ses

objectifs est essentiel. L'attaque de déni de service face au RCSF peut causer de réels dommages.

3.7. Modèle de l'attaquant

En général, plus l'attaquant dispose de ressources, plus la défense est coûteuse. La connaissance de la capacité de l'attaquant permet de définir au mieux la défense adéquate. La conception de réseaux de capteurs doit prendre en considération les menaces les plus fréquentes en énumérant leurs capacités d'attaquants (*leur nombre, leur coordination, leurcapacité technique et leur intérêt d'influence*).

Plusieurs attaquants peuvent menacer un réseau au même moment d'une manière autonome ou en coordination, afin de réaliser une attaque commune rendant la défense difficile voire impossible (*état d'échec système*). La définition des capacités techniques des attaquants est importante pour connaitre la nature de leurs menaces, par exemple un attaquant peut seulement intercepter la transmission de données, mais il peut aussi se présenter comme un nœud capteur digne de confiance légale du réseau, et avoir accès à latotalité des services du réseau. Selon [58], un réseau de capteurs peut être attaqué par deux types d'attaquants : un attaquant puissant et un attaquant réaliste.

3.7.1. Attaquant puissant

L'adversaire est considéré comme présent avant et après la phase de déploiement du réseau. Ila la possibilité de surveiller toutes les communications, n'importe où, et à tout instant. Il participe à la phase initiale de découverte de voisinage, et ainsi il sera identifié entant que voisin légitime. Il est aussi courantde le modéliser avec le potentiel de subvertir un sous-ensemble restreint de nœuds.

3.7.2. Attaquant réaliste

Dans [58] il est noté que la première catégorie est trop exigeante en termes de protocoles de sécurité et conduit à des solutions inutiles dans la pratique. Les auteurs préfèrent modéliser l'attaquant comme étant capable de surveiller un pourcentage *fixe* des canaux de communication lors du déploiement du réseau. Pour justifier la vision « réaliste » de l'attaquant, et non pas le modèle de l'adversaire capable d'intercepter toute communication et d'injecter de nouveaux messages comme il le souhaite [32].

3.8. Types de vulnérabilités des RCSFs

Les vulnérabilités sont les faiblesses d'un réseau que l'attaquant exploite afin de prendre possession des privilèges. Il y a deux types de vulnérabilités dans un RCSF :

La vulnérabilité physique est un moyen d'attaque, qui permet à l'attaquant de changer en partie un capteur, en modifiant par exemple son code de programmation, ou en copiant les clés de protection afin de les réutiliser dans une nouvelle attaque. Un réseau de capteurs est vulnérable aussi aux modifications de son environnement, où un attaquant peut modifier les valeurs d'un capteur l ocal, lui permettant ainsi d'avoir un accès aux commandes de contrôledu réseau.

La vulnérabilité logique réside dans les programmes et les protocoles de communication. Elle se présentesous quatre formes : défauts de conception, défauts d'implémentation, erreurs de configuration, et 'épuisement des ressources.

- Les défauts de conception permettent l'utilisation d'un protocole qui viole le mode d'utilisation, tout en se conformant à la spécification du protocole. Par exemple, un manque d'authentification dans un protocole de gestion de puissance peut permettre de mettre n'importe quel capteur en sommeil à plusieurs reprises.
- Les défauts d'implémentation sont des erreurs dans la construction du matériel ou dansle codage du logiciel. Par exemple, une erreur de dépassement de mémoire, peut entraîner une violation d'accès et une mise en panne.
- Les défauts de configuration sont le résultat de défauts de paramétrages pour unattaquant.
- L'épuisement des ressources est possible même si la conception, l'implémentation, et la configuration sont correctes. Un attaquant générant de grandes quantités de trafic peut inonder un des liens réseau de la victime. Une mauvaise authentification de l'allocation de mémoire ou de l'exécution de code peut également permettre à un attaquant de consommer les ressources du capteur subissant l'attaque, et de causer un déni de service.

3.9. Attaques contre les RCSFs

Les RCSFs peuvent faire l'objet d'un grand nombre d'attaques, chacune avec ses objectifs propres. Par exemple, certaines attaques visent à affecter l'intégrité des messages qui transitent dans le réseau, tandis que d'autres visent à réduire la disponibilité du réseau ou de ses composants. Les attaques se produisent souvent par l'insertion d'éléments intrus dans leréseau. Il existe aussi des attaques contre l'environnement extérieur au réseau, les quelles provoquent des altérations ou des interférences sur les signaux transmis. Une bonne classification des attaques est présentée dans [32, 59].

1. **Ecoute du *réseau* (eavesdropping)** : Du fait que les transmissions se font en diffusion par les ondes radio, aucun contrôle d'accès au réseau n'est possible, ce qui est d'autant plus vrai que le réseau peut être déployé dans un environnement ouvert accessible à tout le monde. Il est donc très facile d'intercepter des données échangées sur un réseau de capteurs et d'accéder à leur contenu si aucun service de confidentialité n'est prévu.

2. **Attaque *physique* (tampering)** *:* Comme les RCSFs sont très souvent déployés dans des zones sans aucune protection, ils sont très exposés aux attaques physiques qui peuvent être considérées sous différents points de vue. L'un est lié au matériel qui n'est pas qualifiéd'inviolable. Dans ces conditions, une attaque aura pour but de récupérer du matériel cryptographique comme les clés utilisées pour le chiffrement. Un autre objectif serait dereprogrammer le capteur pour perturber le réseau et l'application en provoquant volontairement un comportement anormal du noeud. La seconde attaque physique consisterait simplement à supprimer le capteur du réseau en le détruisant (on retombe sur la question del'inviolabilité) ou en le subtilisant.

3. **Attaque de *l'identité* multiples** (**sybil attack**) : Dans cette attaque, un noeud malveillant peut revendiquer différentes identités afin de participer à des algorithmes distribués tels que l'élection et de prendre de l'avantage sur les noeuds légitimes. Un noeud malveillant peut être capable de déterminer le résultat de n'importe quel vote en faisant voter toutes ses identités multiples pour une même entité. Les techniques

d'authentification et de chiffrement peuvent empêcher un étranger de lancer une attaque Sybil sur le réseau de capteurs.

4. **Attaque du trou noir (blackhole ou sinkhole)** : Un noeud falsifie les informations de routage pour forcer le passage des données par lui-même. Sa seule mission est ensuite de nerien transférer, créant ainsi une sorte de puits ou trou noir dans le réseau. L'intrus (nœud malveillant, qui s'introduit illégitimement) se place sur un endroit stratégique de routage dansle réseau et supprime tous les messages qu'il devrait retransmettre, causant la suspension du service de routage du réseau dans les routes qui passent par le noeud intrus. La première contribution traite ce type d'attaque avec beaucoup plus de détail.

5. **Attaque du trou gris (grey hole) :** Une variante de l'attaque précédente est appelée trougris, dans laquelle seuls certains types de paquets sont ignorés par le noeud malicieux. Parexemple, les paquets de données ne sont pas retransmis alors que les paquets de routage lesont.

6. **Brouillage radio (jamming)** : L'intrus inonde avec du bruit les fréquences radio utilisées par le réseau de manière à empêcher les transmissions et/ou les réceptions de messages. Ce type d'attaque peut affecter tout ou une partie du réseau selon la portée radio de l'intrus. Dansce cas-là, l'intention est de provoquer un déni de service.

7. **Relais sélective (selective forwarding)** : L'intrus néglige son rôle de routeur et ne transmet pas certains messages qui sont choisis selon certains critères ou même aléatoirement.

8. **Attaque du trou de ver (wormhole)** : L'attaque du trou de ver nécessite l'insertion d'au moins deux noeuds malicieux [82, 83, 87]. Ces deux noeuds sont reliés entre eux par différents moyens, tel qu'uncannal à une band puissante ou comme par exemple une liaison filaire cachée, encapsulation des paquets, ou transmission à haut débit. Le but de cette attaque est de tromper les noeuds voisins sur les distances. Généralement le protocole de routage cherche le chemin le plus court en nombre de sauts (hop). Le trafic arrivant par un noeud passe à travers le tunnel *wormhole* et est réémis par l'autre noeud. Physiquement, le trafic passe par un ensemble de noeuds. Tandis que, logiquement, du point de vue des autres stations, le trafic ne sera passé uniquement à travers 2 noeuds sur cette portion de réseau (l'attaquant crée l'illusion que les deux extrimétés sont proches). Cette attaque permet de contrôler les flux de trafic, en faisant passer du trafic sur une route plus couteûse qu'en réalité, et ainsi favoriser un chemin que delestera ce détournement.

   L'attaque Wormhole est classée comme assez dangereuse et très défficile à détecter surtoutdans les RCSFs utilisant les protocoles de routage où les routes sont identifiées à la base des informations nuancées telles que le niveau énergétique résiduel ou délai estimé de bout-en-bout ou le plus court chemin en matière de nombre de sauts [84-85].

   L'attaque wormhole est aussi independente des protocoles MAC et est immunisée contre les techniques de cryptographie. Autrement, un attaquant n'a pas besoin de connaitre ni le fonctionnement du protocole MAC, ni comment chiffrer et déchiffrer les paquets circulant dans le réseau pour être en mesure d'exercer une attaque. Dans sa forme la plus sofistiquée, l'attaque wormhole peu être lancée au niveau bit à partir de la couche physique [82]. Le processus de réplication est réalisé bit par bit par une

extrémité avant même que le paquet ne soit complètement reçu par l'autre éxtémité. Tout simplement, le signal produit par la couche physique est reproduit dans une autre zone en temps réel. Cette forme d'attaque est difficile à détecter du moment que la reproduction du signal se fait assez rapidement et de là il est impossibe à la détecter par une analyse temporelle. L'attaque s'éffectue même si le réseau assure et offre la confédentialité et l'authenticité des données communiquées et que l'attaquant ne dispose pas des clés de chiffrement [86]. Pour distinguer ces formes d'attaques de la forme la plus simple où les noeuds malicieux instaurent le tunnel, wormhole copie le paquet entier avant de l'injecter dans le tunnel, nous nous référons à cette dernière forme simple d'attaque par l'attaque store-and-forward selon la terminologie mentionnée dans [71].L'attaque *wormhole* est classifiée dans trois types ou classes: l'attque sélective [82], l'attaque active [67], et l'attaque passive.

Dans le cas d'une attaque du trou de ver, les deux noeuds malicieux permettent d'atteindre un lieu éloigné avec un saut unique. Cette possibilité trompera les autres noeuds sur les distances réelles qui séparent les deux noeuds, mais va surtout obliger les noeuds voisins à passer par les noeuds malicieux pour faire circuler les informations. Ainsi les noeuds malicieux qui forment le trou de ver vont se trouver dans une position privilégiée qui va leur permettre d'avoir une priorité sur l'information circulant à travers leurs noeuds proches. Cette attaque est représentée par la figure 3-5 où deux noeuds malicieux X1 et X2, reliés par une connexion puissante, forment un trou de ver. Les noeuds A et B vont alors privilégier la route la plus rapide formée par le trou de ver, et donc l'information pourra être récupérée par l'attaquant.

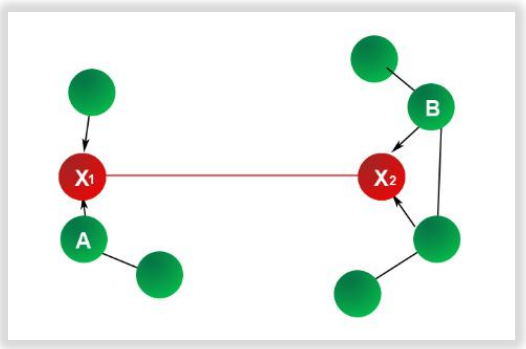


Figure 3-5: *Exemple d'une attaque wormhole*

9. **Rejeu, Délai et Altération de données** : L'intrus répète, retarde ou altère le contenu des messages en transit. Les messages peuvent contenir des données de perception prélevées etdes données de configuration ou de routage. Ce type d'attaque vise entre autres à créerdes boucles, attirer vers lui ou éloigner du trafic, augmenter ou diminuer le nombre de routes, générer de fausses erreurs, partitionner le réseau, et augmenter la latence de distribution des données.

10. **Attaque par *chantage***: Elle est connue sous le nom Anglais de "Blackmail attack". Un nœud malicieux fait annoncer qu'un autre nœud légitime est malicieux pour éliminer cedernier du réseau. Si le nœud malicieux arrive à attaquer un nombre important de nœuds, il pourra perturber le fonctionnement du réseau.

11. **Attaque de *l'inondation* de "HELLO"** : De nombreux protocoles de routage utilisentdes paquets "HELLO" pour découvrir les nœuds voisins et ainsi établir une topologie duréseau. La plus simple attaque pour un attaquant consiste à envoyer un flot de tels messages pour inonder le réseau et empêcher d'autres messages d'être échangés.

12. **Epuisement de la batterie (exhaustion)** : Cette attaque de déni de service est redoutablecar elle vise à épuiser les batteries des nœuds composant le réseau de manière à réduire ladurée de vie du réseau. Elle peut consister à injecter de nombreux messages dans le réseau quiconduisent les nœuds à gaspiller leur énergie en retransmissions inutiles

Le tableau 3-2 1 qui suit classe les attaques et les mesures défensives par couche protocolaire [60].

Tableau 3-2: *Différentes couches protocolaires: Attaques et contre-mesures*

Protocol Layer	Attacks	Defences
Physical	Jamming	Detect and sleep Route around jammed regions
	Node destruction	Hide or camouflage nodes. Tamper-proof packaging
MAC Layer	Denial of sleep	Authentication and anti-replay protection. Detect and sleep method Broadcast attack protection
	Interrogation	Authentication and anti-replay protection
Network	Spoofing, replaying or altering clustering messages	Authentication and replay protection. Secure cluster formation
	Hello Floods	Pairwise authentication Geographical routing
	Homing	Header encryption Dummy packets
	Sybil	Radio resource testing, key validation, position authentication
	Wormhole	Location based routing protocols
Transport	Synchronise flood	Synchronised cookies
	Desynchronised attack	Packet authentication
Application	Overwhelming Sensors	Sensor tuning Data aggregation
	Path-based DoS	Authentication and anti-replay protection
	Re-programming attack	Authentication and anti-replay protection. Authentication streams

3.10. Protocoles de sécurité pour les RCSFs

La conception de protocoles de sécurité pour les réseaux de capteurs n'est pas anodineet peut même avoir des conséquences a impact majeur et dévastateur sur les capteurs eux même, en particulier sur les services applicatifs assurés et sur leur durée de viedu moment que la mise en œuvre de mesure de sécurité engendre une consommation supplémentaire en matière d'énergie. L'échange de paquets de contrôle, le chiffrement et le déchiffrement des messages est un traitement CPU de plus pour un nœud de capteur.

Il est à retenir que le problème du traitement CPU à réaliser au niveau de chaque nœud de capteurs afin d'assurer les fonctions de sécurité (algorithmes de cryptographie) est un défit à soulever. Le choix de ces fonctions estun facteur important déterminé par la taille réduite de la mémoire des nœuds, ce qui rend le code de ces fonctions en fonction de la capacité mémoire réduite. Ces fonctions ou algorithmes de sécurité ne doivent pas aussi être gourmand en terme de traitement CPU et ce, de manière à s'intégrer aisément dans les nœuds de capteurs (sans perturber leur fonctionnement de base [32]). En particulier, les algorithmes cryptographiques à clés publiques qui sont très consommateurs en CPU et en mémoire sont écartés de la liste des choix. On conseillera plutôt l'usage d'algorithmes symétriques de type RivestCipher 5 (*RC5*) ou Skipjack du fait de la taille réduite du code source, de sa rapidité d'exécution etde la mémoire réduite utilisée pendant son exécution (RAM).

La réduction de la taille du code d'une solution de sécurité est une préoccupation et souci majeur des concepteurs. Pour limiter la taille du code inséré au sein des capteurs, l'idée de base en générale sera d'utiliser les mêmes outils cryptographiques pour chiffrer les données émises (RC5 par exemple) et pour générer un MAC (Message Authentification Code) et protéger en intégrité les données. Ce MAC s'appelle classiquement CBC-MAC car il consiste à fragmenter les données à protéger en plusieurs blocs de données (voir figure 3-6), et à faire intervenir l'opération XOR dans le chiffrement d'un bloc X_i, le bloc chiffré précédent H_{i-1} ; de la sorte, le MAC final obtenu correspond au dernier bloc chiffré ; il dépend de tous les blocs constituant les données à protéger et constitue une empreinte des données [32].

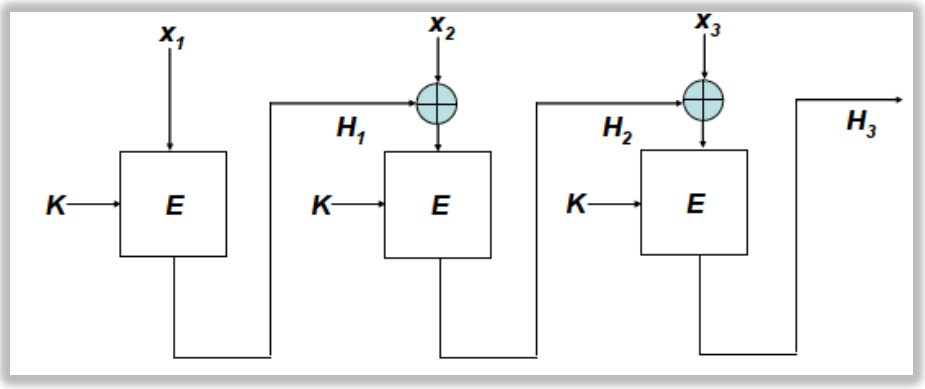


Figure 3-6: *Authentification par CBC-MAC avec l'opérateur XOR*

D'autre part, dans la figure 3-7, la communication est la fonction (*l'activité*) qui consomme le plus d'énergie dans un nœud de capteur. Le traitement CPU ne représente que 3% à 4% de la consommation totale des ressources énergétiques. Ainsi, il apparaît d'autant plus important que les solutions de sécurité envisagées n'insèrent pas un nombre important de paquets de contrôle dans le trafic réseau et n'ajoutent pas de champs aux paquets de données échangés.

Selon [43], le simple fait d'assurer la protection d'un paquet avec entre autre l'ajout d'un MAC, allonge la taille du paquet de 6 octets et aura pour incidence que 20% de la batterie du capteur sera consommée par la simple transmission du MAC. Dès lors, la durée de vie du capteur sera réduite de plus de 27% par la simple introduction des mécanismes de sécurité : MAC et fraîcheur.

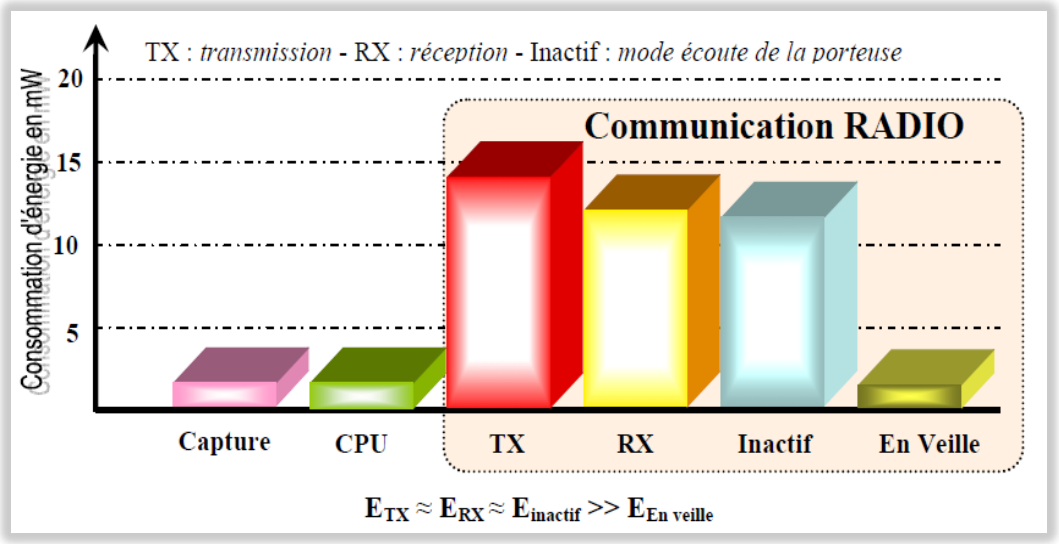


Figure 3-7: *Niveaux de consommation d'énergie au sein d'un nœud capteur*

Delà, dans la conception des solutions de sécurité, quatre critères sont à considérer: en premier lieu le Coût en stockage. Ici il faut distinguer les deux types de mémoires ROM et RAM de faibles tailles et indispensables à la réalisation des solutions de sécurité. La ROM contient le système d'exploitation TinyOS du capteur et le code du programme associé à la sécurité et à la gestion des communications. La RAM est un conteneurdes données en cours. Le second critère est celui de l'énergie. Selon la figure 3-7 et le paragraphe précédent, la consommation énergétique se voit étroitement liée aux fonctions du transceiver (émission + réception). Chaque bit de donnée communiqué entre deux nœuds épuise les ressources en énergie des deux nœuds. Par conséquent, une bonne mesure de sécurité est celle qui injecte moins de paquets de contrôle ou ajoute moins de champs aux paquets de données. En troisième position se classe les failles résiduelles de sécurité : Les protocoles de sécurité ne permettant pas de résoudre tous les problèmes de sécurité, en particulier les attaques par épuisement de la batterie, il est intéressant de déterminer les failles les plus importantes qui persisteront même avec l'introduction de services de sécurité. Le dernier point est lié aux applications du RCSF lui-même : Certaines fonctions habituellement utilisées sur les réseaux de capteurs ne sont pas compatibles avec certaines solutions de sécurité. Par exemple, la fonction d'agrégation a pour but de réduire le volume de données transmises par un capteur, mais elle suppose que le capteur soit en mesure d'accéder au contenu des paquets de données

et de modifier ces paquets, ce qui n'est pas toujours envisageable dans le cas d'une protection de paquets en confidentialité ou en intégrité.

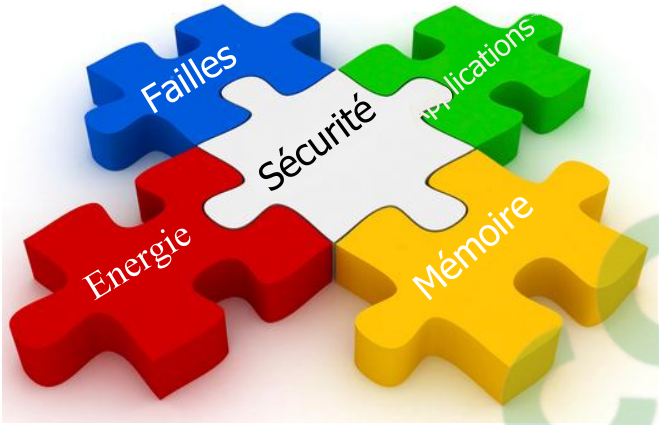


Figure 3-8: *Critères de conception d'une solution de sécurité*

3.11. Travaux existants sur la sécurité des RCSFs

Quelque examples d'algorithmes de détection des attaques seront présentés dans cette sous section. La plupart des travaux traitent les attaques wormhole et/ou autres attaques similaires liées à notre recherche. Nous nous somme limités juste aux méthodes de sécurité à partir des quelles nous nous sommes inspirés, à savoir la technique du RTT, DAWWSEN et UDG. Un tableau récapitulatif des contres mesures face aux attaques sur les RCSF est laissé en fin de cette section.

3.11.1. Mecanisme de Défense contre l'attaque Wormhole

DAWWSEN ou bien (Defense mechanism Against Wormhole attacks in Wireless SEnsor Networks) est un protocole de routage proactif pour les RCSFsdédiés à la construction d'un réseau arborescent hierarchique où le Sink est le nœud racine et les autres nœuds sont soit des liens internes ou des feuilles. Il est doté d'un mécanisme de défense contre l'attaque *wormhole*. La construction de l'arbre est initiée par le *Sink* qui diffuse une requête de découverte des nœuds fils [63, 65].

La requête porte l'ID du nœud originaire du paquet et le nombre de sauts qui est égale à un lorsque le Sink est l'initiateur de la requête. Les nœuds qui reçoivent le 1er paquet ne peuvent pas décider de leurs parents, ils temporisent pendant une période afin de collecter un nombre de requêtes car il est impossible de décider s'il s'agit d'une requête émise par un attaquant (*wormhole*) ou non. Donc chaque nœud, reçevant la requête, insert une entrée dans sa liste (*request-list*) triée par ordre croissant selon le nombre de sauts. Cette dernière contient les IDs des nœuds éméteurs des requêstes reçues ainsique le nombre de sauts correspondant, (voir la figure3-9).

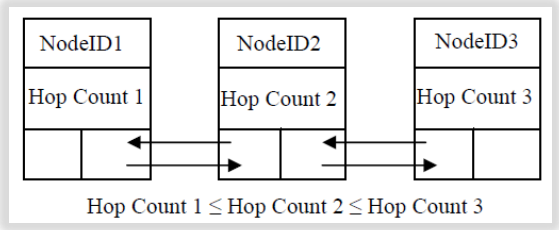


Figure 3-9: *Request-list d'un nœud recevant trois requêtes*

Une variable *Replay-timer* est expirée après une période de *Replay-delay* secondes à partir du moment de la réception de la 1ère requête. Le nœud envoie un paquet Replay contenant son ID une fois *Replay-timer*a expiré en destination du nœud dont ID est la 1ère entrée de la Request-liste, et mis à jour sa table *Replay-Table* contenant les champs suivants : ID de la destination, nombre de saut correspindant, nombre des paquets Replay (Mun_Rep), et le champ Recv_Accept. Les deux premiers champs prennent les valeurs ID destination et nombre de sauts respectivement et les deux autres sont mis à zéro (voir la figure 3-10).

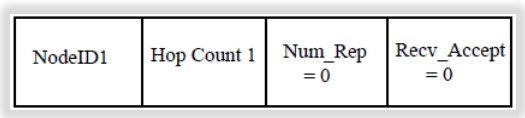


Figure 3-10: *Structure de la table Replay-Table*

Une autre variable temporelle *Chek-Timer*, qui expire après une période *Check-Delay* secondes àpartir de l'instant d'envoie du paquet *Replay*. Durant cette période, le nœud reste en écoute aux paquets *Replay* transmis et incrémente le champ *Num_Rep* pour chaque paquet reçu avec un ID = ID1 (Id de la première entrée de la liste *Request-List*). D'autre part, le nœud recevant un paquet *Replay* insert dans sa liste (*Replay-List*, voir la figure 3-11) un nouvele entrée avec l'ID du nœud émétteur du paquet *Replay*, son nombre de saut = *Num_Replay*.

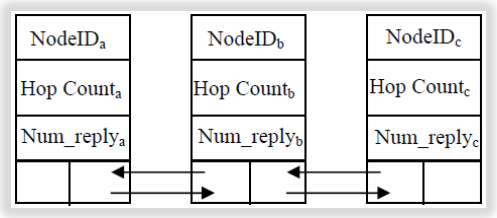


Figure 3-11: *Structure de Replay-List*

Dés la réception du 1er paquet *Replay*, le nœud initialise une troisième variable temporelle *Accept-Timer* autodécrémentable par la valeur *Accept-Delay* secondes. Pour chaque paquet *Replay* reçu durant cette période, le nœud cherche dans sa liste *Replay-List*, le NodeID. S'il existe, son *Num-Replay* sera incrémenté, sinon un nouvel enregistrement sera ajouté à la liste avec *Num-Replay*= 1. Une fois *Accept-Timer* est éxpiré, le nœud envoie, en destination de chaque entrée de sa *Replay-Table*, un paquet *Accept* contenant son ID comme source, le NodeID comme destination, et *Num-Replay* designant le nombre de paquets *Replay* reçus par ce nœud.

Le nœud reçevant un paquet *Accept* doit vérifier le champ source qui doit être égal à NodeID dans sa liste *Replay-Table*. Si ce n'est pas le cas, cela signifie que le paquet a été enregistré par un attaquant durant le processus de construction de l'arbre de routage, et donc il doit être omis. Sinon, le nœud recevant ce paquet met à jour sa liste *Replay-Table* (*Resc-Accept* = 1) et vérifie si *Num-Reply* = *Num-Rep* + 1.

Si les conditions ci-dessus ne sont pas valides, une attaque wormhole est détectée par ce nœud qui doit:

- Omettre le paquet *Accept* reçu
- Inserer le contenu du champ source (*ID* originaire du paquet *Accept*) dans la liste noire (*NAP* = not accepted paquets table)
- Mettre à zéro toutes les entrées de son *Replay-Table*
- Envoyer un paquet *Replay* à destination de sa seconde entrée dans la liste Request-List ou temporiser pour une autre requête.

Le protocole DAWWSEN a été évalué sous l'environnement NS-2. Les simulations ont montré sa capacité à détecter les attaques de type wormhole, et qui'il a besoin seulement de 0,14 secondes pour finaliser la construction des routes arborescentes. Ce protocole n'a besion d'aucune information géographique pour faire face aux attaques. L'insufisance de cette solution réside dans l'hypothèse topologique du réseau qui suit une hérarachie arborescente, ce qui n'est pas le cas dans la plupart des solutions.

3.11.2. Round Trip Time (RTT)

Dans [66] un mécanisme de détection des attaques wormhole pour la sécurisation du protocole de routage AODV dans les RCSFs a été présenté. La technique repose sur le temps éstimé pour un aller et retour d'une requête (rout request) (the Round Trip Time ou RTT). La détection de l'attaque est faite après une série de calculs des intervalles de temps successives des RTT associées aux messages circulant entre des nœuds successifs et leurs voisins. La considération ici et que l'attaquant (l'adversaire) augmente le nombre de voisins d'un noeud dans un rayon de portée radio du transceiver, raccourcit le chemin et augmente la valeur du RTT entre les nœuds successifs. Cette méthode proposée suit trois phases: i) La construction de la liste des voisins pour chaque noeud du réseau, ii) la recherche du chemin de routage liant la source à la destination (Sink), iii) et la localisation des extrémités du tunnel *wormhole* afin de permettre le déclenchement des mesures nécéssaires.

L'attaque typique du wormhole est que l'attaquant reçoit des paquets à un endroit du réseau, via un lien sans fil ou filaire, les expédient avec moins de latence comparé à un lien ordinaire, et ensuite les transmettent vers un autre point du réseau. Ici nous supposons que l'attaque wormhole est:

- Bi-directionelle avec deux noeuds extrémités, quoique l'attaque wormhole à multi-extrémités soit possible en théorie.
- L'attaque est selective.
- L'attaque est active de telle sorte que le champ destination des paquets est modifié par l'attaquant.

La première étape du protocole AODV consiste en la construction des listes des voisins pour chaque noeud. La seconde étape, le protocole entame la tâche de la construction des tables de routage. Ici nous utilisons le mécanisme du *RTT* pour la detection de la présence

des noeuds intrus dans le voisinage (échangent des paquets route request et route replay). Après que le réseau est établi, l'algorithme se déclenchera. La valeur du RTT et celle du RTT entre deux noeuds successifs (noeuds "*A*"et "*B*") est calculée comme suit:

$$RTT = Time_{replay} - Time_{request} \quad (2)$$

$$RTT_{A,B} = RTT_A - RTT_B \quad (3)$$

RTT_A est le temps du RTT entre le noeud "*A*" et la déstination (voir figure 3-12), RTT_B est le temps du RTT entre le noeud "*B*" et la déstination. Par exemple, supposons que le chemin entre la source "*S*" et la déstination "*D*" passe par les nœuds "*A*" et "*B*": S →A → B →D.

Avec $T(S)_{REQ}$, $T(A)_{REQ}$, $T(B)_{REQ}$, $T(D)_{REQ}$ respectivement les instants où les noeuds S, A, B, D envoyent le paquet RREQ et $T(S)_{REP}$, $T(A)_{REP}$, $T(B)_{REP}$, $T(D)_{REP}$ sont respectivement les instants où les nœuds S, A, B, D envoyent le paquet RREP. Alors, les valeurs du RTT entre les noeuds S, A, B et D seront calculées à la base de l'équation (2) précédente:

$$\begin{cases} RTT_S = T(s)_{REP} - T(s)_{REQ} \\ RTT_A = T(A)_{REP} - T(A)_{REQ} \\ RTT_B = T(B)_{REP} - T(B)_{REQ} \\ RTT_D = T(D)_{REP} - T(D)_{REQ} \end{cases} \quad (4)$$

Les valeurs du RTT entre chaque deux noeuds successifs lelong de ce chemin sont calculées pa le système d'équation (5):

$$\begin{cases} RTT_{S,A} = RTT_S - RTT_A \\ RTT_{A,B} = RTT_A - RTT_B \\ RTT_{B,D} = RTT_B - RTT_D \end{cases} \quad (5)$$

Pour un RCSF opérant dans un environnement saint, les valeurs des $RTT_{S,A}$, $RTT_{A,B}$, et $RTT_{B,D}$ se convergent vers la même valeur ($RTT_{S,A} \approx RTT_{A,B}$, $\approx RTT_{B,D}$). Dans le cas où il y'aurai une attaque (de type wormhole ou sinkhole) dans un voisinage d'un des noeuds membres du chemin de routage, la valeur du RTT sera lue un peut plus grande par rapport aux autres valeurs RTTs successives. Le nœud dont la valeurdu RTT semble éloignée des autres valeurs, se trouve dans la portée radion d'un attaquant. Alors l'attaque est ainsi détectée du moment que la valeur du RTT dépasse un certin seuil accepté ($> X \pm \Delta$, ou X est la valeur moyenne des RTT, et Δ est liée à la nature de l'environnement).

La figure 3-13 suivante illustre les étapes de la procédure de la recherche d'un chemin de routage. Les instatnts d'envoie et de réception des paquets PREQ et RREP pour chaque nœud sont marqués sur l'axe du temps.

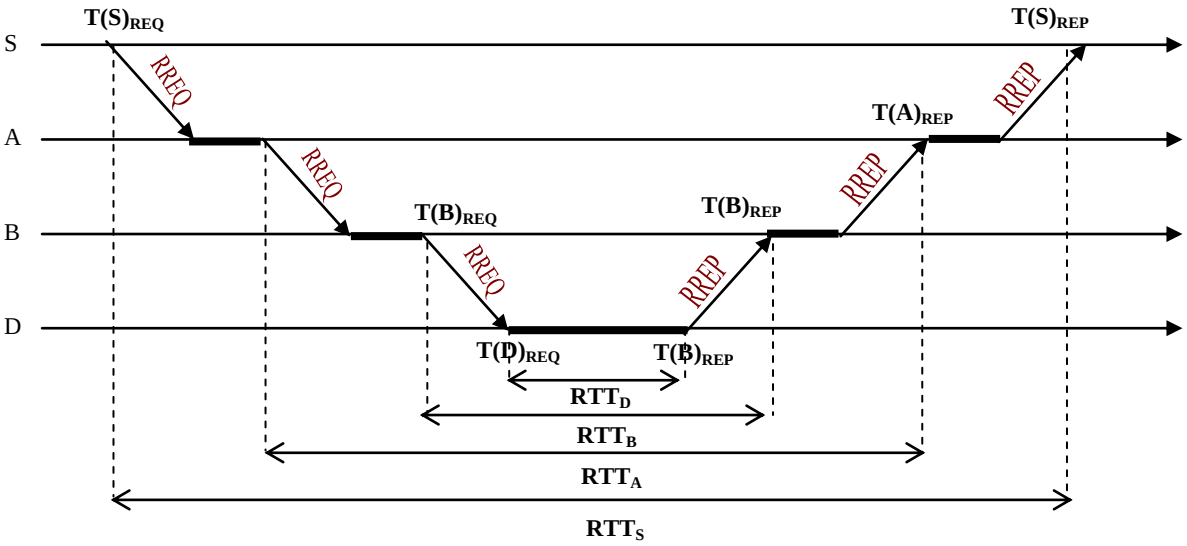


Figure 3-12: *Round Trip Time for finding route*

Discussion : En termes de mémoire, chaque noeud aura besoin de stocker la liste des voisins. Il a été supposé que l'ID du nœud est codifié sur 2 bytes et que la taille de la liste est égale à $\|Liste\| = (N-1)\,\pi\,R^2/A$ éléments (avec R est la portée radio, A la surface de la région et N le nombre de nœuds dans cette région). La liste des voisins a besoin de 20 bytes suplémentaires, et pour le calcul du RTT, chaque noeud doit disposer de n (4+4) byte (n est le nombre maximum de RREQ arrivant en mêm temps à un noeud, généralement n = 4) ce qui fait que chaque nœud aura besoin d'un suplément de mémoire de 32 bytes pour le calcul des valeurs des RTT. En termes de consommation d'énergie, il est clair que cette méthode ne consomme pas plus d'énergie, alors la durée de vie du réseau est presque la même avec ou sans son utilisation.

3.11.3. Cell-based Open Tunnel Avoidance (COTA)

Dans [90], un protocole MAC pour se défendre contre les attaques wormhole est présenté. L'attaque wormhole a été classée en trois classes selon l'emplacement des extérmités du tunel wormhole (voir la figure 3-13) le long du chemin de routage. Ce protocole nommé COTA (*Cell-based open tunnel Avoidance*). De même dans [91], les auteurs traitent le même problème avec le même principe. Le point de départ est que le wormhole possède son propre canal et le processus de tunneling peut être conduit en temps réel. Le chiffrement ou l'authentification seules ne peuvent pas prévenir l'attaque. Les autres nœuds ne peuvent pas confirmer l'origine des paquets injectés dans le réseau.Les auteurs du protocole COTA ont emis les hypothèses suivantes :

1. L'attaque wormhole est bi-directionnelle,
2. Les mecanismes de sécurité tels que le chiffrement sont implantés dans toutes les couches de la pile protocolaire,
3. Les nœuds sont équipés d'un système de repérage géographique (GPS) qui renvoie une position précise, et que l'erreur de calcul de la distance entre deux neouds est de l'ordre de δ ($d_{A,B} \leq \|Position_A - Position_B\| + \delta$).

4. Les nœuds sont synchronisés (différence entre les horloges de deux nœuds est inférieure à Δ),
5. La mobilité des nœuds est limitée (la vélocité maximun d'un nœud = υ).

Dans le protocole COTA, si le nœud mobile déclare avoir reçu un paquet à l'endroit $M1$ et qu'il le fait suivre à l'endroit $M2$. Sachant le temps du buffering "t ", la distance entre les deux points $M1$ et $M2$ doit satisfaire l'inégalité suivante : $d_{M1,M2} \leq \upsilon \times t$. Lors de l'éxamen des paquets de contrôle, si le nœud déclare sa position $P1$ au moment $t1$ et sa position $P2$ au moment $t2$, la déstination peut éstimer sa vitèsse moyenne. Si l'inégalité $\frac{\|P_1-P_2\|-\delta}{\|t_1-t_2\|+\Delta} > \nu$ est vérifiée, le nœud ment àpropos de sa position et de là une attaque au long du chemin est détectée.

Discussion : le mécanisme proposé impose surcertaines hypothèses qui ne vont pas avec notre étude de cas. La dotation des nœuds par des systèmes de positionnement GPS engendre un supper usage d'énergie pour les RCSFs. Du moment que l'attaque wormhole surmonte les mesures de cryptographie et perturbe le comportement ordinaire du réseau, l'hypothèse 2 n'a pa de ségnification.

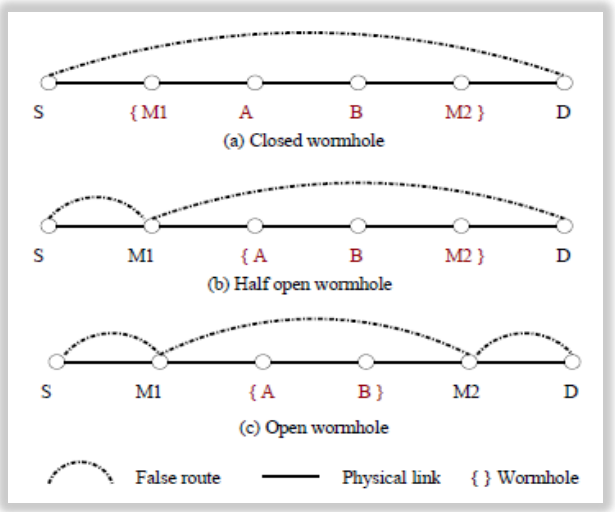


Figure 3-13: *Classification des attaques wormhole selon [90]*

3.11.4. Unit Disk Graph Model (UDG)

Rites et al. [71] ont présenté un protocole pour la détection des attaques wormhole dans les RCSFs basé sur l'usage des informations de connectvité. L'algorithme est indépendant du modèle de communication sans fil et utilise l'information de connectivité pour chercher des sous-structures interdites dans le graphe de connectivité. Le protocole détecte la présence d'attraque même si les nœuds malicieux se sont introduits avant l'initialisation du réseau avec un taux de 100%. L'algorithme repose sur un modèle mathématique UDG (*Unit Disk Graph Model*) : dans UDG, un nœud est modèlisé par un disque planaire d'un rayon unitaire. Chaque nœud est voisin de tous les nœuds localisés dans son disque. Si le graphe de conectivité observé n'a pas un UDG plat fixé, ceci permet de déduire la présence d'un wormhole dans le réseau. Ceci se produit lorsque le tunnel wormhole est assez long (plus long qu'une unité) qui

ne deverait pas exister dans un UDG. Contrairement, lorsque le lien wormhole est très court ou infériеur à une unité, aucun algorithme ne poura détecter l'attaque, et cette dernière n'aurra aucun effet.

L'idée de base exploitée par les auteurs, est le nombre de nœuds qu'on peut mettre dans une région. On denote par

1. $p(S,r)$ le *packing number* = le monbre maximun de points dans une région S tel que la distance séparant chaque paire de points est strictement égale à r.

2. $D_R(u)$ le disque de rayon R centré au point u. (dans un disque unitaire « R=1» il y'a au maximum 5 points avec une distance 1 entre chaque deux points [92]),P(D_1, 1) = 5.

3. Deux disques de rayon R centrés en u et v distancés de r, l'intersection de ces deux disques est la lune $\mathcal{L}(r,R) = D_r(u) \cap D_R(v)$. quand $R = r = 1$ on simplifé par$\mathcal{L}$.

4. Il a été prouvé que : $p(\mathcal{L},1) = 2$ [71]. Ainsi, il y'a seulement deux nœuds dans $\mathcal{L}$ avec la distance 1 (voir la figure 3-14).

Le résultat de répartition des points dans une surface donnée est utilisé pour definir les structures non autorisées pour un UDG. Le wormhole connecte tous les nœuds dans une région A avec tous les nœuds de la région B (voir la figure 3-15). Il y'a deux nœuds indépendants a et b dans la région A qui sont voisins de trois nœuds c, d, e de la région B. Ceci constitue une structure interdite. Les trois voisins (c, d, e) doivent être dans l'intersection des deux disques centrés dans a et b, et puisqu'ils sont indépendants, la distance qui les sépare est plus grande qu'une unité (selon la formule précédente$(\mathcal{L},1) = 2$) cela ne se produira pas, ainsi la découverte de cette structure interdite révèle l'éxistence du wormhole.

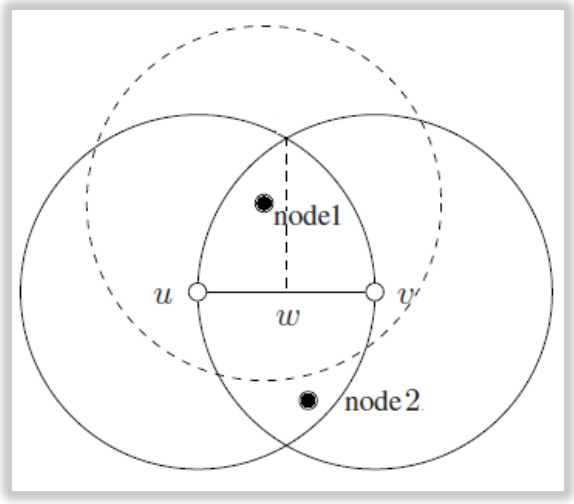


Figure 3-14: *Deux au maximun distancés par plus d'une unité dans l'intersection de deux UDG*

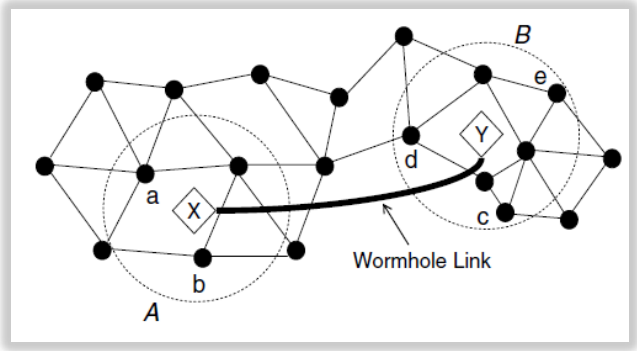


Figure 3-15:*L'attaque wormhole*

Discussion : les auteurs du protocole ont présenté un mécanisme de détection des attaques wormhole en se basant uniquement sur les données de connectivité. Les tests de simulation sous TOSSIM ont montré l'efficacité de la solution proposée à un taux très élevé. L'inconvénient est quil faut plus de calcul mathématique afin de pouvoir déduire les structures interdites et delà la détection de l'attaque. La deuxième insufissance de cet algorithme, est qu'il foctionne pour une topologie bien déterminée où la distribution des nœuds est homogène, les nœuds sont équi-distants, le réseau moins dence. Ce qui n'est pas toujours le cas dans la réalité.

3.11.5. Autre solutions de sécurité proposées pour RCSFs

Le tableau 3-3 résume les différentes techniques de sécurité appliquées aux RCSFs [93].

Tableau 3-3: *Résumé des déffirents mécanismes de sécurité dans les RCSF*

Security Schemes	Attacks Deterred	Network Architecture	Major Features
JAM	DoS Attack (Jamming)	Traditional wireless sensor network	Avoidance of jammed region by using coalesced neighbor nodes
Wormhole based	DoS Attack (Jamming)	Hybrid (mainly wireless partly wired) sensor network	Uses wormholes to avoid jamming
Statistical En-Route Filtering	Information Spoofing	Large number of sensors,highly dense wireless sensor network	Detects and drops false reports during forwarding process
Radio Resource Testing,Random Key Pre-distribution etc.	Sybil Attack	raditional wireless sensor network	Uses radio resource, Random key pre-distribution, Registration procedure, Position verification and Code attestation for detecting sybil entity
Bidirectional Verification, Multi-path multi-base station routing	Hello Flood Attack	Traditional wireless sensor network	Adopts probabilistic secret sharing, Uses bidirectional verification and multi-path multi-base station routing
On communication Security	Information or Data Spoofing	Traditional wireless sensor network	Efficient resource management, Protects the network even if part of the network is compromised

TIK	Wormhole Attack, Information or Data Spoofing	Traditional wireless sensor network	Based on symmetric cryptography, Requires accurate time synchronization between all communicating parties, implements temporal leashes
Random Key Predistribution [96]	Data and information spoofing, Attacks in information in Transit	Traditional wireless sensor network	Provide resilience of the network, Protect the network even if part of the network is compromised, Provide authentication measures for sensor nodes
Key management scheme for large-scale DSN [98]	Data and Information Spoofing	Distributed Sensor Network, Large-scale wireless sensor network with dynamic nature	Suitable for large wireless sensor networks which allows addition and deletion of sensors, Resilient to sensor node capture
REWARD [97]	Blackhole attacks	Traditional wireless sensor network	Uses geographic routing, Takes advantage of the broadcast inter-radio behavior to watch neighbor transmissions and detect blackhole attacks
TinySec [94]	Data and Information spoofing, Message Replay Attack	Traditional wireless sensor network	Focuses on providing message authenticity, integrity and confidentiality, Works in the link layer
SNEP &μTESLA [95]	Data and Information Spoofing, Message Replay Attacks	Traditional wireless sensor network	Semantic security, Data authentication, Replay protection, Weak freshness, Low communication overhead

3.12. Sécurité Holistique dans les RCSFs

Une approche holistique de sécurité pour les RCSFs vise à améliorer les performances réseaux d'un point de vue sécurité, pérennité et connectivité des nœuds capteurs sous des conditions environnementales variantes. L'approche holistique de la sécurité concerne l'implication de toutes les couches de la pile protocolaire afin d'assurer et d'une manière globale la sécurité du réseau. Pour un réseau, une simple solution sécuritaire au niveau d'une seule couche peut ne pas être efficace contrairement à l'emploi d'une solution globale qui se voit comme une bonne option [60].

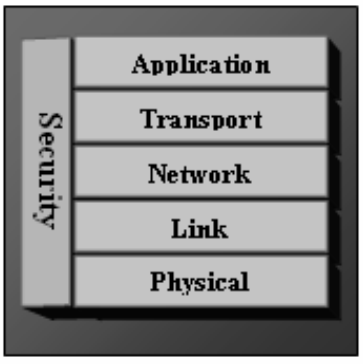


Figure 3-16: *Vue Holistique de la sécurité dans les RCSFs*

Une approche holistique a des principes de base telsque dans un réseau donné, la sécurité doit être prise en charge au niveau de chaque couche protocolaire et le coût engendré ne doit pas dépasser le risque évalué à un instant.

En l'absence d'une sécurité physique des nœuds, les meures de sécurité doivent être capable d'exhiber de légères dégradations si des nœuds se sont compromis ou capturés par l'ennemi. Les mesures de sécurité doivent être déployées pour fonctionner d'une manière décentralisée et autonomes. Si la sécurité n'est pas conçue pour toutes les couches, par exemple : si un nœud est capturé quelque part, la sécurité du réseau complet tombe en faille malgré la présence d'un mécanisme de sécurité implémenté sur d'autres couches. Par la construction d'une sécurité selon une approche holistique, la protectionsera établie pour le réseau entier.

3.13. Conclusion

Ce chapitre a été consacré aux notions de bases liées à la sécurité de l'informtion en générale et à la sécurité des réseaux de capteurs sans fil en particulier. Nous avons présenté quelques définitions, les attaques sur les RCFs, classification des attaques, les points de vulnéabilité des RCSFs vis-à-vis des attaques, et quelques protocoles de contres-mesures face aux différentes attaques. Ici, nous nous sommes aiguillés essentiellement vers deux types d'attaques à savoir l'attaque sinkhole qui fera l'objet de notre première contribution et celle du wormhole à laquelle nous consacrerons notre seconde contribution dans nos travaux de recherches.

Une liste de protocoles et mécanismes de défense pour sécuriser les RCSFs a été donnée dans la section 3.11. Le contexte de la problématique de notre travail a été en quelque sorte tracé par ce chapitre. Le dernier point abordé dans ce chaptre est une forme d'interogation sur l'éffiacité d'une bonne méthode de lutte contre les attaques dans les RCSFs. Notre vision est de prévoir un méchanisme inter-couches là où tous les niveaux de la pile protocolaire peuvent bénificier et en même temps enrichir la défense.

4. Cross-Layer Medium Access Control (CL-MAC) et Modélisation Formelle

4.1. Introduction

Dans ce chapitre nous allons presenté en détail le protocole CL-MAC, sa modèlisation sera largement déscutée. Ses modules fonctionnels prenderont aussi leurs part dans cet espace. Le modèle UML de ce protocole sera aussi présenté [6, 77]. Nous aborderons aussi dans ce chapitre les vulnérabilités du protocole CL-MAC. Il est à signaler que ce protocole a été conçu pour résoudre le problème de gaspillage d'énergie dans les réseaux de capteurs et delà accroître la durée de vie de ces réseaux. Lors de sa conception, nous n'avons pas pris en consédiration tous les paramètres environementaux ni envisagé tous les sénarios possibles. Le propblème de la sécurité de CL-MAC face à l'attaque sinkhole et l'attaque wormhole est l'objet de nos travaux.

4.2. Présentation du protocole CL-MAC

Le protocole CL-MAC proposé fait partie de la classe des protocoles MAC fondés sur la notion d'inter-couches, faisant interagir par collaboration (non pas par fusion) les deux couche adjacentes : la couche réseau et la couche MAC. Son objectif primordial est l'efficacité énergétique. Il permet d'éliminer les principales causes de gaspillage d'énergie et de diminuer sensiblement la latence en présence d'applications particulières sensibles au délai de délivrance de données de bout en bout et pour les quelles le protocole proposé peut être un candidat potentiel. Les applications sensibles au délai, lorsqu'un évènement anormal aura lieu, existent en pratique : monitoring environnemental (par exemple la détection de feu de forêt [99] choisi ici comme un exemple typique, détection d'intrus), assistance à des personnes âgées ou handicapées et surveillance à distance de la santé d'édifices publics, et surveillance à l'aide de capteurs vidéo d'une infrastructure importante. Dans ces applications, l'évènement détecté est considéré comme un message urgent qui devait être transmit rapidement au Sink pour une intervention urgente. Afin de répondre aux exigences imposées sur le comportement du CL-MAC, il est nécessaire de réduire considérablement la latence au niveau MAC lorsqu'il s'agit de transmettre ce type de données urgentes depuis une source vers le Sink.

La couche MAC assure l'accès au medium selon un calendrier à deux états par alternance Veille/Reveil (sleep/weakup) adaptatif identiquement au protocole T-MAC décrit dans [101] et [102], et les nœuds sont localement et périodiquement synchronisés comme c'est le cas du protocole Z-MAC [100]. Les nœuds capteurs du réseau sont déployés aléatoirement dans une surface géométrique de forme linéaire ou sous forme de grille. La phase de synchronisation des nœuds suit celle du déploiement.

Avant de donner plus de détails illustrant le comportement de notre protocole CL-MAC, nous avons jugé utile d'énumérer les hypothèses suivantes que nous avons supposées valides pour faire fonctionner CL-MAC:

i) Scénario typique d'utilisation : dans un souci de clarté, la figure 4-1 présente un exemple typique d'application où le protocole peut être utilisé. Lorsqu'un évènement sensible au délai est détecté (ici il s'agit de la détection des feux de forêt dans une

région couverte par le nœud capteur S1). Les données relatives à cet évènement sont considérées comme étant un trafic urgent qui devait être transmises rapidement au Sink. CL-MAC, en agissant au niveau de la couche MAC, permet de réserver un chemin de routage à économie d'énergie et de faible latence entre la source et le Sink pour assurer la délivrance de ce trafic urgent. Deux types de nœuds peuvent être distingués sur ce chemin, en plus du Sink : (i) un nœud capteur source ayant capté des donnés urgentes, donc sensibles au délai de bout en bout (il s'agit du nœud S1) ; (ii) Les nœuds relais (A ou B dans la figure 4-1) dont la tâche principale consiste à relayer le trafic urgent dans un chemin multi-sauts. Dans le scénario de la figure 4-1, le chemin S1-A-B-C-D-E-F-Sink est un exemple d'un chemin urgent réservé à l'aide de CL-MAC. Chaque nœud appartenant à ce chemin comme par exemple B et ses voisins comme B' et S4 utilisent CL-MAC au niveau de la couche MAC.

ii) Le déploiement du réseau dans l'espace de couverture est aléatoire et est suivi par une phase de synchronisation.

iii) Les nœuds capteurs sont synchronisés localement et d'une façon périodique comme c'est le cas du protocole Z-MAC.

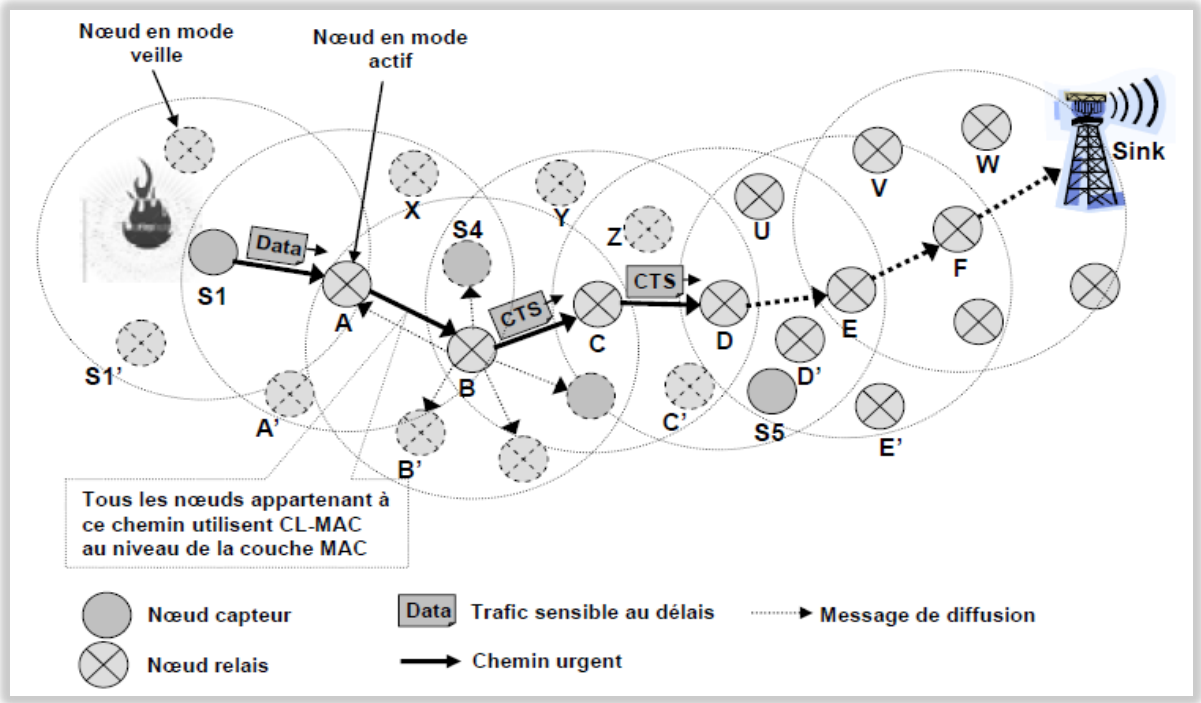


Figure 4-1: *Fonctionnement modèle du protocole CL-MAC*

Dans ce qui suit, nous présenterons par ordre: (i) la couche abritant le protocole CL-MAC dans son contexte intercouches, (ii) les nouvelles structures de données des paquets de contrôle RTS et CTS et leur interprétation, (iii) L'algorithme détaillé de CL-MAC et ses principaux avantages, (iv) et enfin d'autres détails relatifs au protocole CL-MAC.

4.2.1. La couche MAC opérant avec le protocole CL-MAC

Chaque nœud construit une liste de voisinages contenant des informations relatives à ses voisins (identificateur du voisin, sa position, son calendrier sleep-weakup) et la table de routage maintenue par un agent de la couche réseau adjacente. Le protocole de routage utilisé

est basé sur l'approche de Greedy, connue sous le nom de routage à base de position ou de localisation. Dans cette approche, la décision de relayage des paquets est prise en tenant compte des nœuds voisins candidats et la position de la destination finale. Le schéma de routage de Greedy basé sur la distance, proposé par Finn [103] et par K. Johnson [104], a été adopté pour l'étude du protocole CL-MAC afin de calculer le plus court chemin selon les métriques d'énergie et de latence. Dans cette approche, le nœud choisi pour le prochain saut, dans le mécanisme de routage, est le nœud voisin le plus proche de la destination finale (le Sink dans notre cas).

La figure 4-2 illustre le fonctionnement du protocole CL-MAC au niveau de la couche MAC. Un nœud qui l'implémente peut transmettre deux types de trafic unicast (mono destinataire) : un trafic sensible au délai généré localement (le nœud qui implémente CL-MAC agit en tant que nœud source) ou un trafic sensible au délai reçu de l'un de ses voisins (le nœud qui implémente CL-MAC agit dans ce cas en tant que nœud routeur). Lorsqu'un nœud source agit en tant que relais et en même temps il désire transmettre un trafic urgent, les deux trafics générés au sein du nœud seront donc transmis vers le Sink.

CL-MAC exploite les informations de routage à travers l'approche inter-couches appliquée selon la stratégie par interaction. Dans cette stratégie, CL-MAC interagit avec la couche réseau en utilisant un mécanisme simple de type « Get/Store » employant une mémoire de stockage commune dédiée à stocker les informations de routage à un instant donné (voir figure 4-2). Ce mécanisme permet à l'agent de routage d'écrire « Store » dans la mémoire commune, partagée par les deux couches MAC et Réseau, l'information sur le prochain saut auquel le nœud actuel appartenant au chemin urgent doit lui relayer le paquet. Cette information de routage peut être lue « Get » et exploitée par CL-MAC pour gérer l'accès au médium (plus exactement, réserver un chemin urgent d'une durée déterminée pour favoriser l'accélération des transmissions multi-sauts successives qui suivent). Il est à noter que dans la figure 4-2, nous avons proposé au niveau de la couche physique le standard IEEE 802.15.4 [105] dédié à l'origine pour les réseaux LR-WPAN (Low Rate Wireless Personnel Area Networks), mais il est aussi très adapté pour les RCSFs car il est conçu pour les applications à faible débit, à faible consommation énergétique et à faible coût.

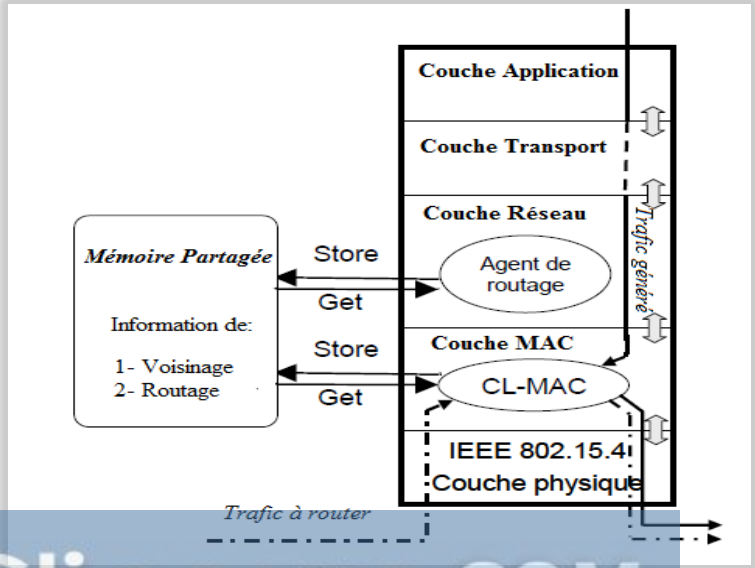


Figure 4-2: *Schémas inter-couches de CL-MAC*

4.2.2. Nouvelles structures des paquets

Les structures des paquets échangés entre les nœuds de capteurs ont subi des modifications pour des raisons d'implémentation et de simplification des traitements tout en restant proche de la représentation normalisée IEEE 802.11 [Web 2] (voir figure 4-3). Pour le paquet de données nous avons préservé la même structure avec seulement un changement d'interprétation des champs d'entête. Les champs des adresses de la trame MAC sont au nombre de quatre ToDs, FromDs, Adresse3, Adresse4. Nous les avions utilisés pour mettre les adresses des nœuds appartenant au chemin vers le Sink. Les champs Adresse3 et Adresse4 pour les adresses des nœuds précédants et suivants respectivement. Le schéma qui suit montre la structure de notre trame MAC modifiée.

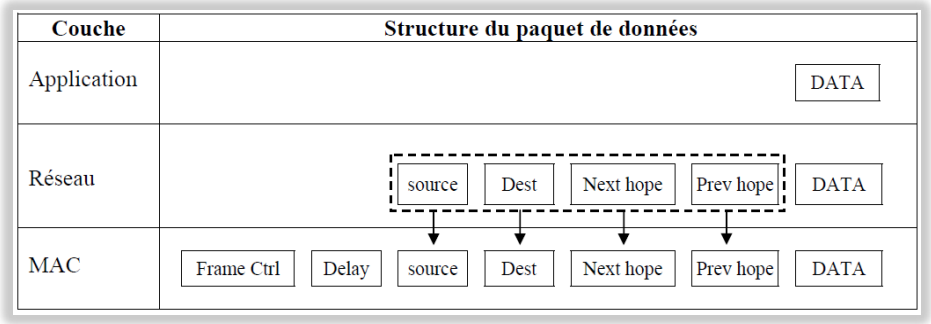


Figure 4-3: *Structure d'un paquet DATA dans CL-MAC*

Delay : est un champ de type entier dont la valeur représente le temps, estimé, nécessaire pour la transmission complète d'un message (tous les fragments du message). Il est calculé à partir du nombre des paquets formant le message. (Delay = nombre de paquets * temps nécessaire pour l'envoie d'un seul paquet). Sachant qu'un paquet nécessite une unité temporelle de transfèrt.

Ceci aide la couche MAC, du nœud en possession du paquet, pour endormir ses nœuds voisins ne faisant pas partie du chemin réseau vers le Sink, le basculement en mode mise en veile (Sleep) permet d'économiser l'énergie de l'écoute à la porteuse à vide « idle listening power » et libère le medium pour surmonter le problème de la collision. Les nœuds voisins non impliqués dans la présente communication doivent hiberner pour une durée mentionnée dans leurs variables NAVs (Network Allocation Vector).

Path info : est formé de quatre champs adresse préservant, à un niveau de profondeur, la trace du chemin réseau. Ce quadruplet d'adresses est constitué de:

- 1er : identificateur du nœud source.
- 2ième : identificateur du nœud destination finale (dans notre cas c'est le Sink).
- 3ième : identificateur du nœud suivant (next hop node).
- 4ième : identificateur du nœud précédant (previous hop node).

Les quatre champs du pathinfo sert à tracer la route vers le sink (champ 3), et si le chemin se brise à un moment donné, il est possible de faire un feedback vers le nœud précédent pour emprunter un autre chemin alternatif différent de celui passant par le nœud mentionné dans champs 3 (le nouveau chemin passera par champ4 et non pas par champ3).

Frame control : est un champ information MAC ajouté au paquet (figure 4.4). Il comporte le numéro du paquet à envoyer (il est nécessaire pour le contrôle des nouveaux noeuds voisins, dû au changement topologique, en les obligeant à basculer vers le mode mise en veille pour le temps restant de la communication) et le type de message, qui est l'un des ces quatre possibilités {ACK, Wake-up, Resend, Data}.

Nb digit	2	2	4	1	1	1	1	1	1	1	1
Filed	Protocol version	Type	Sub Type	To DS	From DS	More Frag	Retry	Power Management	More Data	Wep	order

Figure 4-4: *Structure du contrôle frame*

Afin de maintenir une certaine compatibilité avec le standard IEEE 802.11 [Web 2], une légère modification est apportée à la structure des trames RTS et CTS de la couche MAC. La figure 4-3 illustre les nouvelles structures proposées pour les messages RTC et CTS. Le nouveau champ ajouté au message RTS est le champ 'Next_Node_Adress', obtenu à l'aide de l'agent de routage de l'émetteur et désigne l'adresse du prochain nœud dans le chemin de routage auquel les paquets doivent être transmis. Les nouveaux champs du message CTS sont 'Next_Node_Adress' et 'Sender_Adress'. Le champ 'Next_Node_Adress' a la même signification que celui du message RTS, mais dans ce cas il est déterminé par l'agent de routage du récepteur du message CTS. Le champ 'Sender_Adress' permet à un nœud capteur de prendre la décision d'ignorer ou non un message CTS, selon que son adresse est spécifiée ou non en tant qu'adresse d'un nœud prédécesseur. Si le nœud ignore le message CTS cela voudrait dire qu'il a déjà reçu un message CTS et qui l'a relayé à un autre nœud du voisinage, qui lui va le retransmettre à nouveau. Le seul cas où il l'accepte est le cas où il le reçoit pour la première fois [6, 77].

Le changement de 'Sender Adress ' à 'Previous Adress' est effectué par le récepteur du message CTS. Afin de permettre à CL-MAC de fonctionner correctement, l'adresse du Sink est supposée être connue au niveau de chaque nœud capteur du réseau dans le cas d'un RCSF mono-Sink.

RTS	Contrôle de trame	délai	Adresse Précèdent	Adresse suivant	Adresse émetteur	CRC
CTS	Contrôle de trame	délai	Adresse précèdent	Adresse suivant	Adresse émetteur	CRC

Figure 4-5: *Nouvelle structure des paquets RTS et CTS*

Toutefois, il est important de noter que les deux messages ont une structure identique, la différence réside au niveau de leur interprétation par le nœud récepteur. Ainsi, si le récepteur est:

i) le nœud suivant : le message agit comme un RTS (virtuel) provenant de l'émetteur.

ii) le nœud précèdent, le message joue le rôle d'un CTS provenant de l'émetteur comme une réponse à un RTS (effectif).

iii) un autre nœud (i.e. du voisinage), le message contrôle le comportement du nœud et le force à basculer en mode veille.

Durant la période de réveil, un nœud continue à écouter la porteuse pour une période relativement courte. Deux cas peuvent être considérés si le medium n'est pas alloué :

Cas 1. Le nœud est en possession de données urgentes à transmettre au Sink. Il prend le contrôle du médium et informe ses voisins de sa décision.

Cas 2. Le nœud n'a pas de données à transmettre. Dans cette situation, il éteint son transceiver afin d'épargner de l'énergie due à l'écoute de la porteuse à vide ou à l'interception de paquets destinés à d'autres nœuds. Dans ce cas, le nœud passe plus de temps en mode veille que dans le cas de S-MAC : ½ frame + le temps de la communication et reste dans ce mode jusqu'au prochain cycle correspondant à la planification actuelle. Si dans cette période le médium est occupé, cela signifie que soit une communication est en cours, soit qu'un autre nœud tente d'obtenir le contrôle du médium. Dans ce cas, deux situations peuvent se présenter :

Situation 1. Un autre nœud désire accéder au médium. Dans cette situation l'algorithme de Backoff [90] est activé pour résoudre ce contention afin d'élire le nœud à qui il faut donner les droits d'accès au médium. Tous les autres nœuds non élus basculeront en mode veille à l'exception du récepteur du paquet. Celui-ci restera en mode réveil (actif) afin d'établir la communication avec le nœud élu.

Situation 2. Le nœud n'a pas de données à communiquer. Dans ce cas, il reste en mode réveil (actif) s'il est concerné par la communication en cours (il est le récepteur), sinon il passe en mode veille jusqu'au nouveau cycle.

Un nœud récepteur d'un paquet est identifié par l'émetteur en se référant à sa table de routage. Cette dernière contient les informations relatives au chemin vers le Sink.

4.2.3. Principe de fonctionnement de CL-MAC

Un noeud dans le réseau de capteurs sans fil se trouve à chaque instant dans un état parmi quatre états possibles : (i) Mise en veille, (ii) Transmission, (iii) Réception, et (iv) Ecoute de la porteuse à vide.

L'algorithme se constitue de trois modules opérant en deux phases:

Phase1. Module : Découverte des voisins.
Module : Synchronisation des calendriers entre les noeuds voisins.
Phase2. Module : Contrôle d'accès au medium.

Les deux modules de la 1ère phase d'initialisation sont intégralement identiques à ceux du Z-MAC (voir référence [100]).Le module Contrôle d'accès medium est conçu comme suit:

Côté émetteur : Cette partie décrit le comportement du nœud émetteur en possession d'un paquet à transmettre vers le Sink.

1. **Début**
2. Émetteur est le nœud en possession de données générées par ses capteurs ou reçu pour être routées.
3. Émetteur génère deux paquets RTS et DATA.
4. Émetteur envoie le paquet RTS pendant l'intervalle de temps [0, SIFS].
5. Émetteur reçoit le paquet CTS du noeud récepteur après un temps égale à
   T_{cts} = 2*SIFS + 2*Delay(RTS) …… (3-I)
   (delay(RTS) est le temps de propagation radio du paquet RTS, et puisque les tailles des paquets de contrôle RTS et CTS sont égaux alors le temps de leurs propagations est le même)
6. Émetteur envoie le paquet DATA, via le réseau, juste après la réception du CTS, dans la période de T_{cts} à T_{cts}+ SIFS; ……..…… (3-II)
7. Émetteur reçoit l'ACK du noeud récepteur en repense au paquet DATA. Après un temps T_{ack} = 4*SIFS + 2*Delay(RTS)+Delay(Data)+Delay(ACK). ……(3-III)
8. Émetteur bascule vers le mode mise en vielle (sleep mode) en fin de la communication.
9. Émetteur reste en hibernation tant qu'il n'est pas concerné par une communication. Là,
10. l'émetteur reprend le calendrier weakup/sleep défini lors de la phase synchronisation ;
11. **Fin**

Algorithme 4-1: *Emission des données*

Remarque:*les instances de temps sont calculées par rapport à l'instant de la prise du contrôle du médium par le nœud de capteur voulant communiquer.*

Côté recepteur : l'algorithme suivant décrit d'une manière chronologique les étapes (événements) d'un nœud destinataire provisoire (routeur) d'un paquet.

1. **Début**
2. Récepteur est le nœud destinateur des paquets de contrôle RTS et celui des données.il est le nœud finale (Sink) ou (ou exclusif) routeur (next hop) ;
3. Récepteur reçoit le paquet RTS de l'Émetteur ;
4. Récepteur génère le paquet CTS en repense au paquet RTS;
5. Récepteur envoie le paquet CTS vers le nœud Émetteur via le réseau;
6. Récepteur reçoit le paquet DATA du nœud Émetteur, et il le sauvegarde dans sa pile;
7. Récepteur génère le paquet ACK suite à la bonne réception du paquet DATA ;
8. Récepteur envoie le paquet ACK vers le nœud Émetteur via le réseau;
9. **Si** est-routeur(Récepteur) **Alors**
10. Récepteur ← Émetteur ;
    (*Récepteur devient nouveau expéditeur des données pour pouvoir les routées*)
11. **Sinon** fin de la communication (le message a atteint sa destination finale) ;
12. **Fin si**
13. **Fin**

Algorithme 4-2: *Réception des données*

Remarque:*La fonction booléenne est-routeur, prend comme argument l'identificateur du nœud est décide si ce dernier figure dans le champ next-hop de l'entête des paquets de contrôle reçus par ce nœud.*

Côté réseau : les nœuds voisins des nœuds impliqués dans la présente communication n'appartenant pas au chemin réseau entre la source et la destination finale : description chronologique du comportement du voisinage des noeuds membre du chemin adopté par le nœud initiateur de la communication pour transmettre les grandeurs récoltées par ses capteurs.

1. **Début**
2. Autre-nœud est le nœud voisin non concerné par la présente communication ;
3. Autre-nœud reçoit un paquet quelconque (RTS, CTS, DATA, ou ACK);
4. Autre-nœud pivote vers le mode mise en vielle (Sleep) pour une durée estimée égale au temps nécessaire pour achever la présente communication;
5. Autre-nœud se réveil après l'achèvement de la communication;
6. **Si** une nouvelle communication dont autre-nœud est un membre communicant **Alors**
7. **Si** is-source (autre-nœud) **Alors**
8. Autre-nœud ← Émetteur ;
9. **SinonSi** is-destination (autre-nœud) **Alors**
10. Autre-nœud ← Récepteur ;
11. **Sinon**
12. Autre-nœud ← autre-nœud ;
13. **Fin si**
14. **Fin si**
15. **Sinon** (le canal est vide= absence de communication)
16. Autre-nœud reprend le calendrier wakeup/sleep défini lors de la phase synchronisation
17. **Fin si**
18. **Fin**

Algorithme 4-3: *Comportement d'un voisin non concerné par la communication*

Le comportement global d'un noeud de capteur, quelque soit sa nature (initiateur d'une communication, routeur, ou voisin) est détaillé dans ce qui suit par un algorithme et un schéma descriptif qui ont servi de base pour l'implémentation du protocole CL-MAC. La figure 4-15 suivante schématise graphiquement le comportement des noeuds voisins et ceux membre du chemin de routage lors de la transmission d'un paquet.

```
1.  Entrée : Table; // table de routage
2.  Début
3.  Construire la liste des voisins (Liste);
4.  Synchroniser les horloges;
5.  Construire la table de routage(Table); // En récupérant des infos générées par l'agent de routage
6.  Etique1: Nav := 0;
7.  Etat := Réveil; //WakeUp
8.  Répéter
9.  ┌ Si existe des données à transmettre ET (état = Réveil) Alors
10. │   ┌ Si canal est libre Alors
11. │   │   Si autres noeuds désirent accéder au médilum Alors
12. │   │       Procédure-Backoff-pour-résolution-contention ;
13. │   │   FinSi
14. │   │   Destination := Lire-destination-de-Table(Table);
15. │   │   Envoyer RTS à la Destination;
16. │   │   Etat := Attente-pour-CTS;
17. │   Sinon
18. │   │   Se mettre en mode veille (Sleep) et se réveiller au prochain cycle (frame) ;
19. │   │   Aller à Etique1;
20. │   FinSi
21. │ Sinon
22. │   ┌ Si (canal est libre) Alors
23. │   │   Recevoir (massage);
24. │   │ ┌ Si (message→destination = ID) Alors // ID du nœud récepteur du paquet
25. │   │ │ ┌ Case Type (message) Of
26. │   │ │ │ 'RTS' : Construire-et-transmettre(CTS);
27. │   │ │ │         Etat :=Attente-pour-DATA;
28. │   │ │ │ 'DATA' : Construire-et-transmettre-à-la-source(ACK);
29. │   │ │ │         Reconstruire-et-transmettre-DATA-auprochain- saut(DATA);
30. │   │ │ │         Etat := Attente-pour-ACK;
31. │   │ │ │ 'ACK' : Se mettre en mode veille ;
32. │   │ │ │         Aller à Etiqu1;
33. │   │ │ │ 'CTS' : Si (Etat = Attente-pour-CTS) Alors
34. │   │ │ │ │         Transmettre(DATA); Etat := Attente-pour-ACK;
35. │   │ │ │ │       Sinon ┌ Si Etat ≠Attente-pour-DATA Alors
36. │   │ │ │ │             │     Reconstruire-et-transmettre (CTS);
37. │   │ │ │ │             │     Etat := Attente-pour-DATA;
38. │   │ │ │ │             └ FinSi
39. │   │ │ │ │       FinSi
40. │   │ │ │ └ FinCaseOf
41. │   │ │ │   Sion
42. │   │ │ │       Nav := message→durée;
43. │   │ │ │       Se mettre en mode veille; Aller à Etique1;
44. │   │ │ └ FinSi
45. │   │ Sinon // cas où le canal est libre
46. │   │       Se mettre en mode veille et se réveiller au prochain cycle ;
47. │   │       Aller à Etique1;
48. │   └ FinSi
49. └ FinSi
50. Jusqu'à ce que (Niveau-batterie < seuil) ;
51. Fin CL-MAC
```

Algorithme 4-4: *Algorithme CL-MAC implémenté au sien d'un nœud*

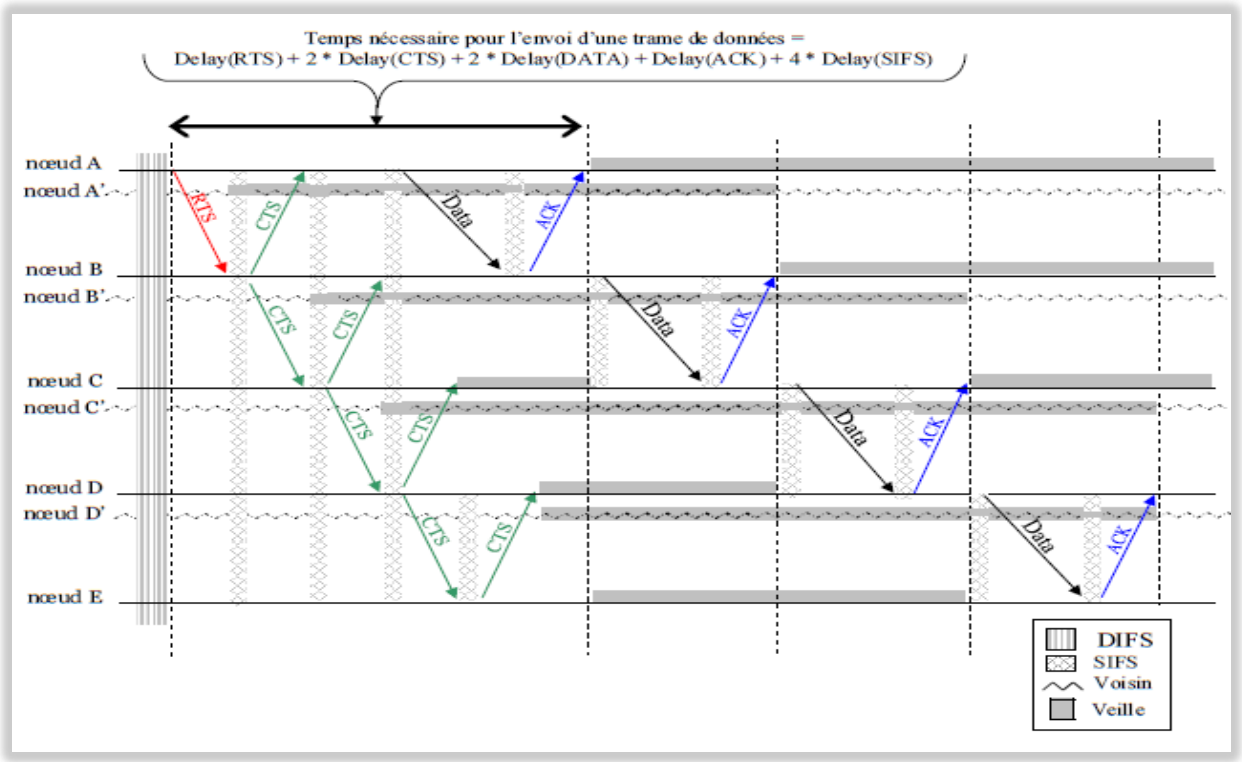


Figure 4-6:*Schéma illustratif du comportement du CL-MAC [77]*

4.3. Vunérabilité de CL-MAC face aux attaques sinkhole et wormhole

Le protocole CL-MAC est un protocole à couches croisées et à efficacité énergétique. La pérénité du réseau était le premier parametre de la qualité de service. Le chemin de routage vers le sink ainsin trouvé, force les noeuds membres d'assurer un lien depuis l'initiateur de la communication j'ausqu'au sink à des moments bien précis.Tout autre nœud voisin et hors ce chemin doit étteindre son transceiver durant la communication. Sauf que notre protocole CL-MAC soufre d'une grande fragilité. Il a été conçu pour opérer dans des conditions assez idéales. Le menaces de sécurité non pas été prises en consédiration tout au long du processus de sa conception.

Selon les références liées au CL-MAC, le protocole fonctionne convenablement selon les descriptions imposées. Les hypothèses faites lors de son développelment etaient en nombre de trois. (i) nous avons supposé un réseau saint sans aucun noeud intrus ou malviellant. (ii) le réseau avait une topologé plate, (iii) et en troisième point, la combinaison des informations de contrôle des deux couches réseau et MAC.

Ces hypothèses faites sur le protocole CL-MAC nous mènent à mettre en exergues ses failles de sécurité :

- La nature hostile de l'environnement du déployement du réseau operant avec le protocole CL-MAC, n'a pas été prise en charge comme paramètre de conception.

- Les problèmes de securité rencontrés dans tous les protocoles de communication. Les attaques, la prise du contrôle d'une partie du réseau ou de sa totalité a été écartée dans les phases du développement du CL-MAC.
- Le canal de communication hertzième (ondes radio) pose de sérieuses difficultés pour le sécuriser. Il sufit de se mettre dans la portée radion et d'être équiper d'une antenne adéquate pour intercepter les communications sans aucune peine.
- Dans CL-MAC, les noeuds apartenant au même chemin doivent participer à faire acheminer et sans le moindre arrêt des paquets de la communcation en cours. Donc aucun noeud dans le voisinage ne perturbera la communication. Les voisins des members du chemin de routage se mettent en mode veille pour une période estimée sufisante pour achever la commicantion courante. Supposons qu'une communication a reservé le chemin "path1" et une autre a été initialisé sur le chemin disjoint "path2" (voir la figure 4-1). Alors un bon emplacement d'un attaquant créant un lien entre ces deux chemins ou absorbant le traffic dans un coin, ou simplement injectant des paquets dans le voisinage d'un noeud membre du chemin interrompera une ou les deux communications selon le type de l'attaque exercée.
- Une étude statistique simple a montré que 50% des communications sur deux chemins liés par une attaque de type wormhole seront perdues [66, 81].

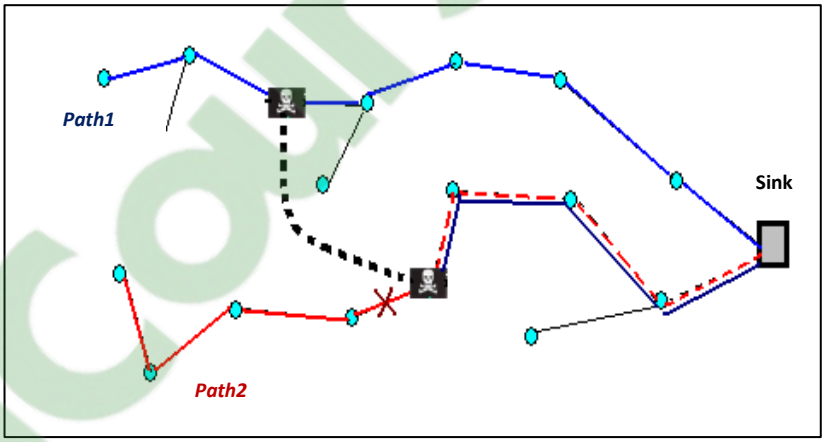


Figure 4-7: *Tunnel wormhole liant deux chemins disjointsdans CL-MAC*

4.3.1. Vulnérabilité deCL-MAC face à l'attaque wormhole

La figure 4-8 montre un comportement anormal du protocole CL-MAC en présence d'un lien de trou de ver. Selon la confguration et l'emplacement du wormhole présenté dans la figure 4-8, le noeud malicieux "*H*1" se comporte comme étant un voisin légitime du noeud "*A*"et par conséquent, il reçoit tous les paquets transmis par "*A*". Parmi ces paquets, se trouve le paquet *CTS* qui est envoyé par le noeud "*A*" à son successeur (le saut suivant) sur le chemin de routage (nœud "*B*" dan la figure 4-8) selon la table de routage. Le noeud malicieux "*H*1" utilise le lien wormhole pour transmettre le paquet *CTS* intercepté au niveau du second nœud malicieux "*H*2". Quand ce dernier aura reçu le paquet *CTS* téléporté, il le diffusera dans son voisinage (nœuds *D, U, E, V, F, E, E', D', S*5). L'essence du CL-MAC lorsqu'il est dans un mode saint, il permet à un seul nœud "S1", dans une portée radio et inclus dans le chemin de routage depuis la source jusq'au sink, de rester éveillé, tandis que tous les autres nœuds du voisinage restent en mode endormi le temps necéssaire pour acheminer la donnée au sink. Comme conséquence de ce comportement, le nœud "*D*" qui est supposé appartenir au chemin

de routage et en même temps un voisin du nœud "*A*" et grâce au wormhole installé, rentre en mode endormi. Quand le nœud "*C*", à son tour tente de transmettre le paquet *CTS* au noeud "*D*", il trouvera ce dernier en mode endormi. De cette manière, la reservation du chemin de routage sera bloquée au niveau du noeud "*C*"et le sink (destination) ne sera pas atteint. Même le paquet de données *DATA* qui est derrière, sur le chemin, sera lui aussi bloqué.

Il faut aussi noter que d'autres phénomènes de blockage peuvent avoir lieu dans ce scénario dans le cas où le paquet *RTS* est transmis via le tunnel wormhole à la place du *CTS*. Dans ce scénario, le tunnel wormhole se comporte comme une attaque selective car l'attaquant choisit la reproduction, uniquement des paquets *RTS* et *CTS*, d'un endroit à un autre et rejette tous les autres paquets circulant dans son entourage (i.e. *DATA*, *hello* messages).

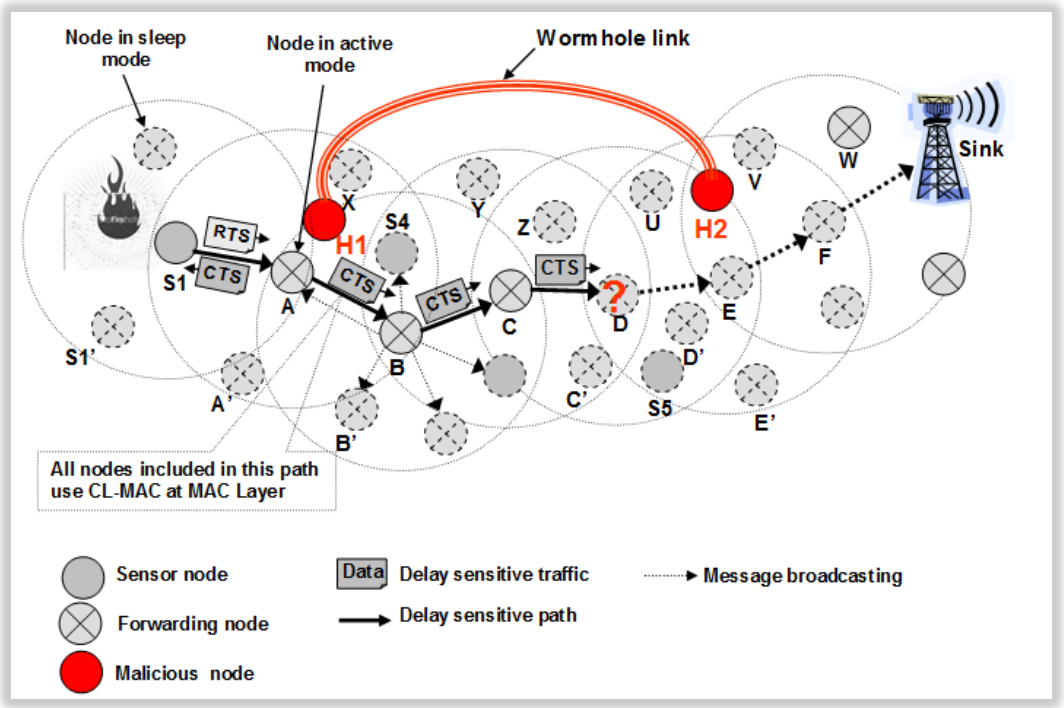


Figure 4-8: *L'attaque Wormhole et son effet sur le fonctionnement de CL-MAC*

En dépit de la présence de multiples types d'attaques dans un réseau, n'importe lequel peut perturber le fonctionnement de base du protocole. Alors pour le CL-MAC, un simple attaquant passif engendrera des dommages irréversibles. Lorsque les nœuds voisins s'échangent les paquets RTS et CTS afin d'établir une communication dans la zone1 (voir figure 4-9), un nœud compromis ou intrus (formant un lien wormhole avec un autre nœud distant) passivement injecte une copie du paquet RTS/CTS assez loin dans la zone2.

Indépendemment de l'emplacement des deux nœuds extrémité du wormhole, que se soient connectant deux chemins disjoints ou sur le même chemin; le second nœud attaquant dans la zone2 injecte une copie du paquet RTS qui n'est pas adressé à aucun nœud se trouvant dans cette zone (zone2). Cette situation, conduit tous les nœuds dans la zone2 (voisins de la seconde extrémité du wormhole) à entrer en mode sommeil (sleep mode). Par conséquent, les communications venant vers le sink seront bloquées suite à la rupture du chemin dans la zone2 comme le schématise la figure 4-9.

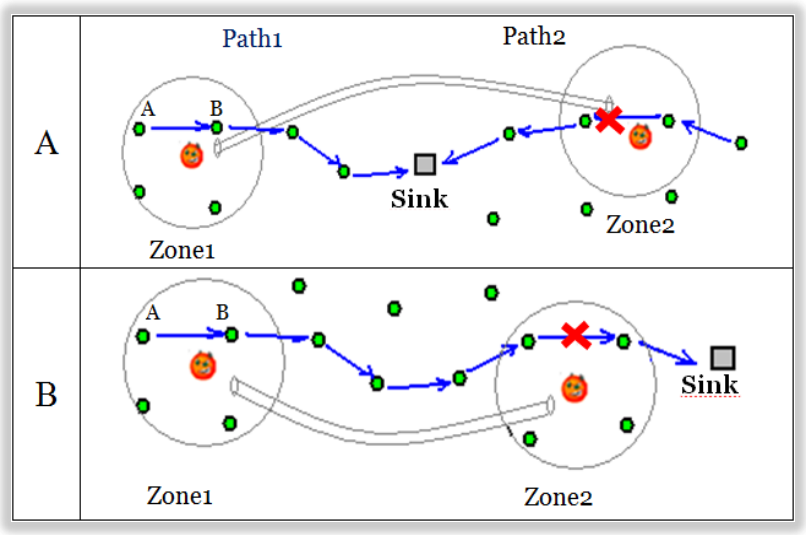


Figure 4-9: *Effet de l'attaque wormhole sur CL-MAC:*
(A) *: le Wormhole joint deux chemins disjoins,*
(B) *: le Wormhole sur le même chemin*

1. Effet de l'attaque wormhole sur CL-MAC au niveau de la couche réseau :

L'attaque wormhole est classée très dangereuse et est difficile à détecter quand l'attaque survient au niveau de la couche réseau utilisant un protocole de routage où la découverte des chemins de routage est basée sur les informations annoncées tels que le niveau d'énergie résiduel, la fiabilité de bout-en-bout, nombre minimal de sauts vers le sink. La figure 4-9 illustre dans sa partie gauche un protocole de routage opérant sur un RCSF et les chemins formés par ce protocole. Dans la partie droite, le même réseau avec la même topologie, protocole de routage, et répartition des nœuds dans la zone d'intérêt. Deux nœuds malicieux se sont introduits restaurant un tunnel wormhole. Les paquets communiqués par n'importe quel nœud dans la zone colorée seront redirigés vers la seconde extrémité du tunnel wormhole. Le plus simple dommage que peut causer cette attaque est de casser le tunnel pour que tous les nœuds se trouvant dans la zone colorée soient déconnectés du réseau.

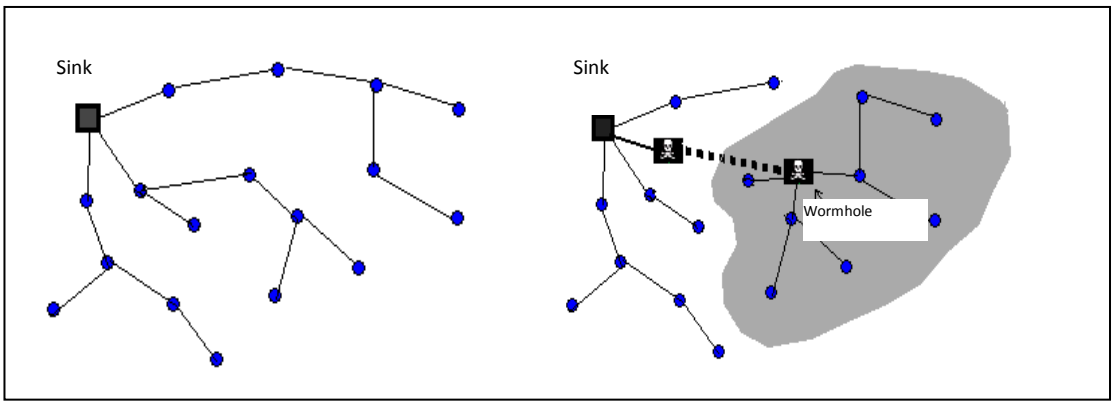


Figure 4-10: *Effet de l'attaque wormhole sur la couche réseau.*

2. Effet de l'attaque wormhole sur CL-MAC au niveau de la couche MAC :

Une fois les deux nœuds malicieux forment le tunnel wormhole et prennent contrôle de la couche MAC, ils peuvent causer de sérieux problèmes pour affecter le fonctionnement du réseau. Sur la figure 4-11, X et Y sont deux nœuds liés par un lien wormhole. Comme conséquence de cette attaque, les nœuds de la zone A considèrent ceux de la zone B come voisins légitimes et vice versa. Cela doublera la taille des listes de voisins pour chaque nœud et engendrera un trafic supplémentaire pour la maintenance des listes des voisins et les table de routage.

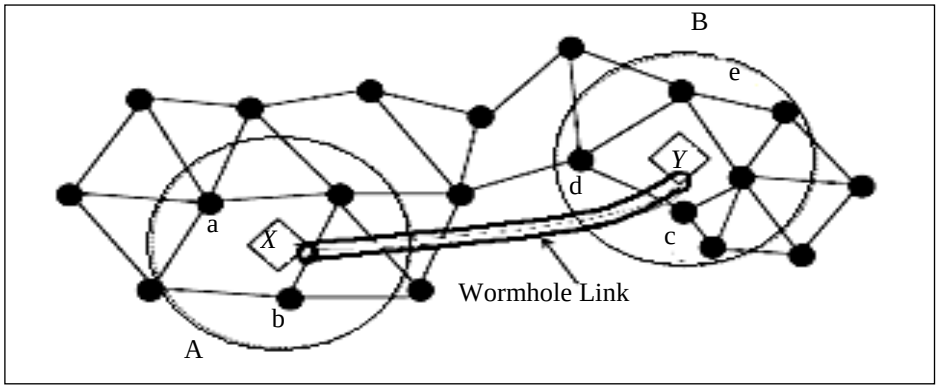


Figure 4-11: *Effet de l'attaque wormhole sur la couche MAC.*

4.3.2. Vulnérabilité de CL-MAC face à l'attaque sinkhole

Dans la figure 4-12, un nœud "*H*2" maliciex exercant l'attaque sinkhole est placé, à mi-chemin, entre la source et la distination (*S1-A-B-C-**H2**-E-F-Sink)*. Selon cette confguration, l'attaquant "*H*2" est un voisin légitime du noeud "*C*". Il se presente comme un nœud favori pour le prochain saut dans le chemin de routage. Lors de l'échange des paquets de contôle pour la découverte et/ou la mise-à-jour du voisinage, et le calcul du chemin de routage, il informe ses voisins qui'il est le plus proche du sink et ayant une réserve sufissante d'énergie. Cette fausse information permet son insertion dans le chemin de routage et la liste des voisins. Par conséquent, tout le trafic transitant par "*C*" en direction du sink passera par l'attaquant "*H*2". Le nœud malicieux "*H*2" peut exercer trois formes d'attaque : (i) Sinkhole passive où

le nœud malicieux se contentera d'absorber les paquets en provenance du nœud "*C*". Dans cette situation, le nœud "*C*" retransmettra trois fois le paquet pour lequel il n'a pas reçu de réponse avant de déclarer que le nœud placé au prochain saut (ici il s'agit du nœud "*H*2") dans sa table de routage est défaillant. (ii) Sinkhole active : ce mode d'attaque est très dangereux et est indétectable. L'attaquant répond aux paquets empruntant le chemin liant la source à la destnation (Sink) sans les faire suivre. Cette situation bloque tout le trafic à destination du Sink et une partie du réseau sera isolée (détachée). L'attaquant est vu comme un barrage séparant deux parties du réseau (voir la figure 4-12). (iii) Selective forwarding sinkhole : l'attaquant laisse passer certains paquets et bloque d'autres, il transmet les paquets de contôle afin de masquer sa présence et échappe aux mécanismes de sécurité employés.

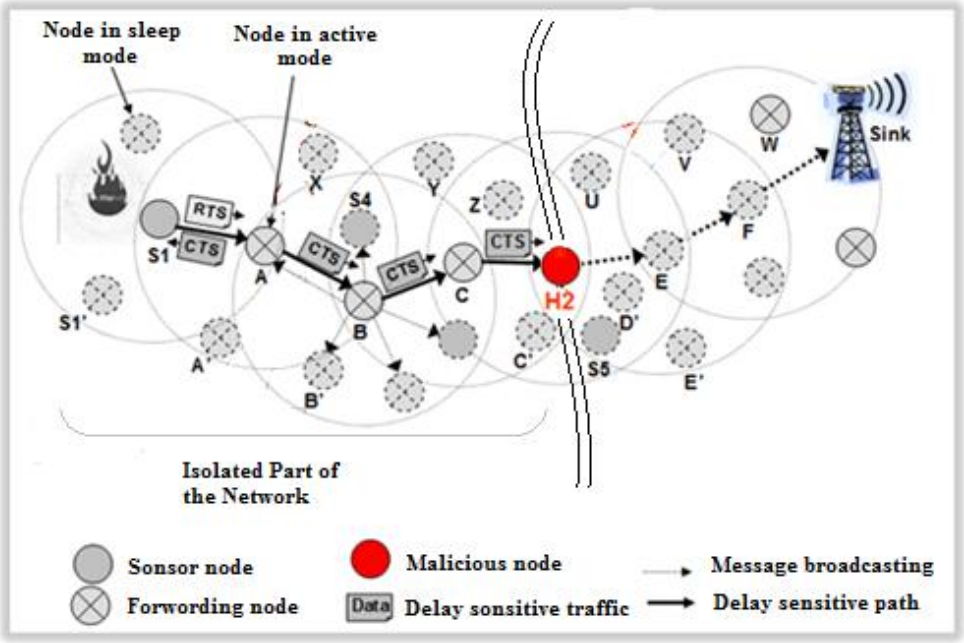


Figure 4-12: *l'attaque Sinkhole et son effet sur le fonctionnement de CL-MAC*

1. Effet de l'attaque sinkhole sur CL-MAC au niveau de la couche réseau :

Quand l'attaque sinkhole survient au niveau de la couche réseau, dont les nœuds capteurs utilisent un protocole de routage où la découverte des chemins de routage est basée sur les informations annoncées comme le niveau d'énergie résiduel, la fiabilité de bout-en-bout, le nombre minimal de sauts vers le sink, etc. Dans sa partie gauche, la figure 4-13 illustre un protocole de routage opérant sur un RCSF ainsi que les chemins construits par ce protocole. Dans la partie droite, la même topologie du même réseau est reproduite avec l'insertion d'un nœud malicieux. Le réseau semble avoir deux sinks (deux station de base). Les chemins de routage qui passaient par l'attaquant, se sont redirigés vers lui et le trafic qui deverait être acheminé vers le sink sera absorbé par l'attaquant. Sur la figure 4-13, les chemins de confiance sont colorés en bleu par contre ceux conduisant vers le sinkhole portent la couleur rouge. Nous remarquons deux choses : (i) une partie du réseau est hors contrôle, formée par les nœuds membres des chemins représentés en rouge. (ii) une partie de la zone d'interêt (la zone grise sur la figure 4-13) n'est plus couverte et toutes les données récoltées seront redirigées vers le sinkhole.

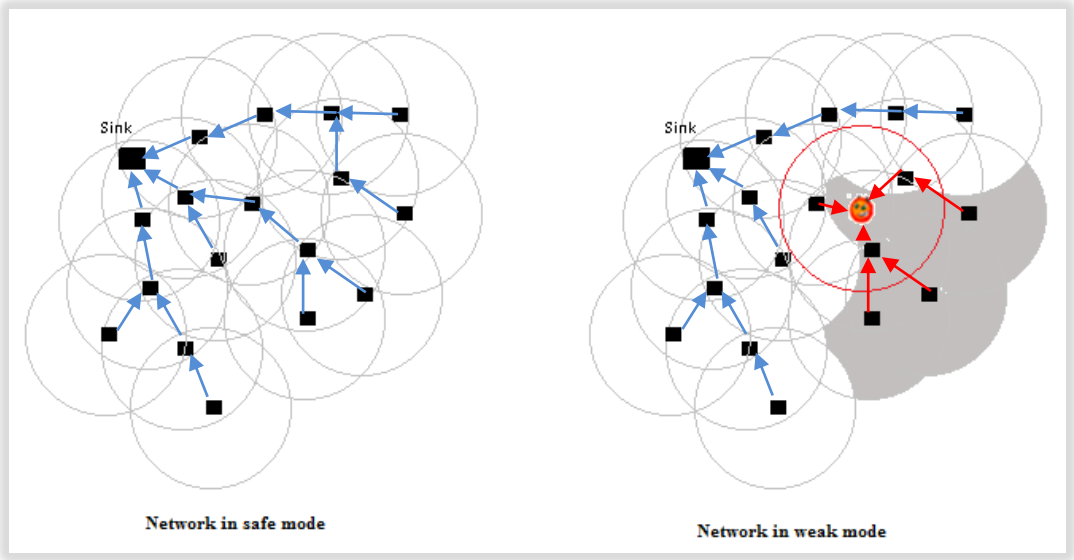


Figure4-13: *Effet de l'attaque sinkhole sur la couche réseau.*

2. Effet de l'attaque sinkhole sur CL-MAC au niveau de la couche MAC :

Un nœud malicieux, intrus ou compromis, exerçant l'attaque sinkhole dans l'une de ses formes citées précédemment dans la section4-3-2, fractionnera le réseau en deux parties. Le fait que les voisins de l'attaquant (le nœud "*H*2" dans la figure 4-12) le choisissent comme voisin potentiel (plus court chemin + reserve d'énergie sufisante), causera un gaspillage énergétique du coté des voisins du nœud "*H*2" suite à la retransmission continue des paquets (du moment que ces derniers non pas été accusés).

4.4. Modélisation UML

Cette partie est destinée à compléter l'étude du protocole CL-MAC ainsi que sa sécurité. Nous présenterons le modèle UML d'un nœud de capteur et quelques diagrammes liés à notre méthode de sécurisation du CL-MAC.

4.4.1. Méthodologie de conception UML.

UML (Unified Modeling Language, que l'on peut traduire par « langage de modélisation unifié ») est une notation permettant de modéliser un problème de façon standard. Ce langage est né de la fusion des trois méthodes qui s'imposaient dans le domaine de la modélisation objet au milieu des années 1990 [106-109].

UML 2.0 définit treize types de diagrammes répartis en trois catégories [108] et [Web20]. Dans le cadre de notre travail, nous avons utilisé certains de ces diagrammes répondant à nos objectifs de modélisation. Dans les paragraphes qui suivent, nous donnerons quelques exemples pratiques de ces diagrammes utilisés dans cette thèse.

4.4.2. Modèle UML d'un noeud capteur

Conformément à la définition de la structure interne d'un noeud capteur évoquée dans le chapitre 1, nous proposons dans la figure 4-14, le diagramme de classes pour un noeud capteur en tenant compte même d'un nœud wormhole et/ou sinkhole.

4.4.3. Diagramme états-transitions d'un neoud capteur

Le mécanisme de détection d'attaque de type wormhole ou sinkhole introduit au sein d'un noeud capteur est modélisé par le diagramme d'états-transitions d'un noeud capteur selon la figure 4-15. Un noeud capteur opérationnel se trouve dans un des trois états suivants après une phase de déployement: actif, inactif, et cible d'attaque. Le noeud capteur dans un état actif est capable de communiquer (transmettre ou recevoir des paquets de données), dans l'état inactif le nœud passe en mode sommeil ordinaire (fonctionnement de base du CL-MAC). Dans l'état cible d'attaque, le nœud lance une alerte et pivote vers le mode hibernation (pour une période assez longue jugée suffisante pour que l'attaquant juge inutile de se planquer dans ce voisinage)

Avec son voisinage, dans un état inactif, le noeud capteur bascule entre une phase de mise en veille d'une durée Tsleep et une phase d'écoute de la porteuse d'une durée de TIdle. Lorsqu'un noeud capteur en état actif termine l'opération de communication avec son voisinage, il passe en mode inactif pour économiser son énergie électrique.

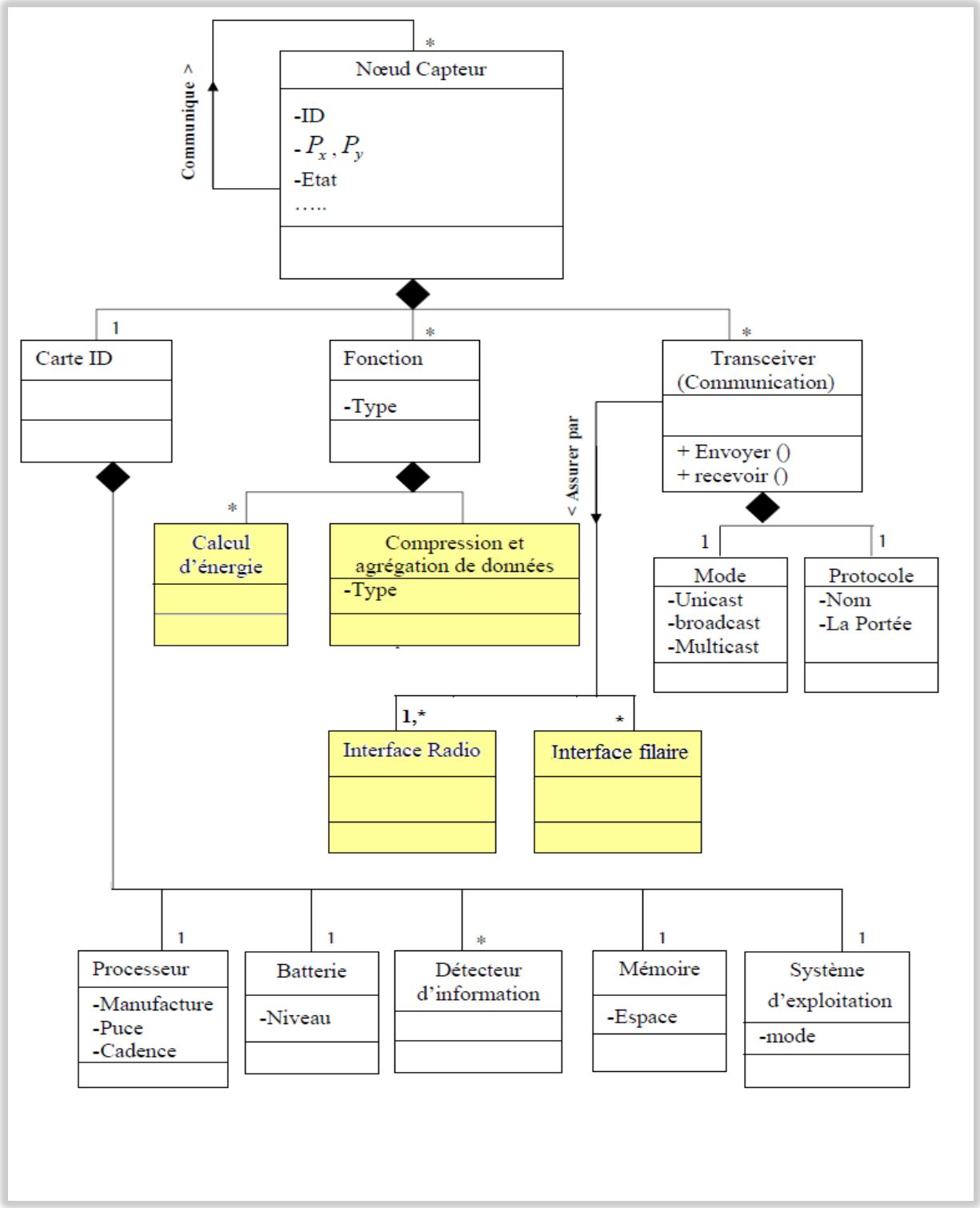


Figure 4-14: *Diagramme de classe d'un noeud de capteur avecun nœud malcieux (wormhole) peut avoir deux interfaces radio et/ou interface filaire*

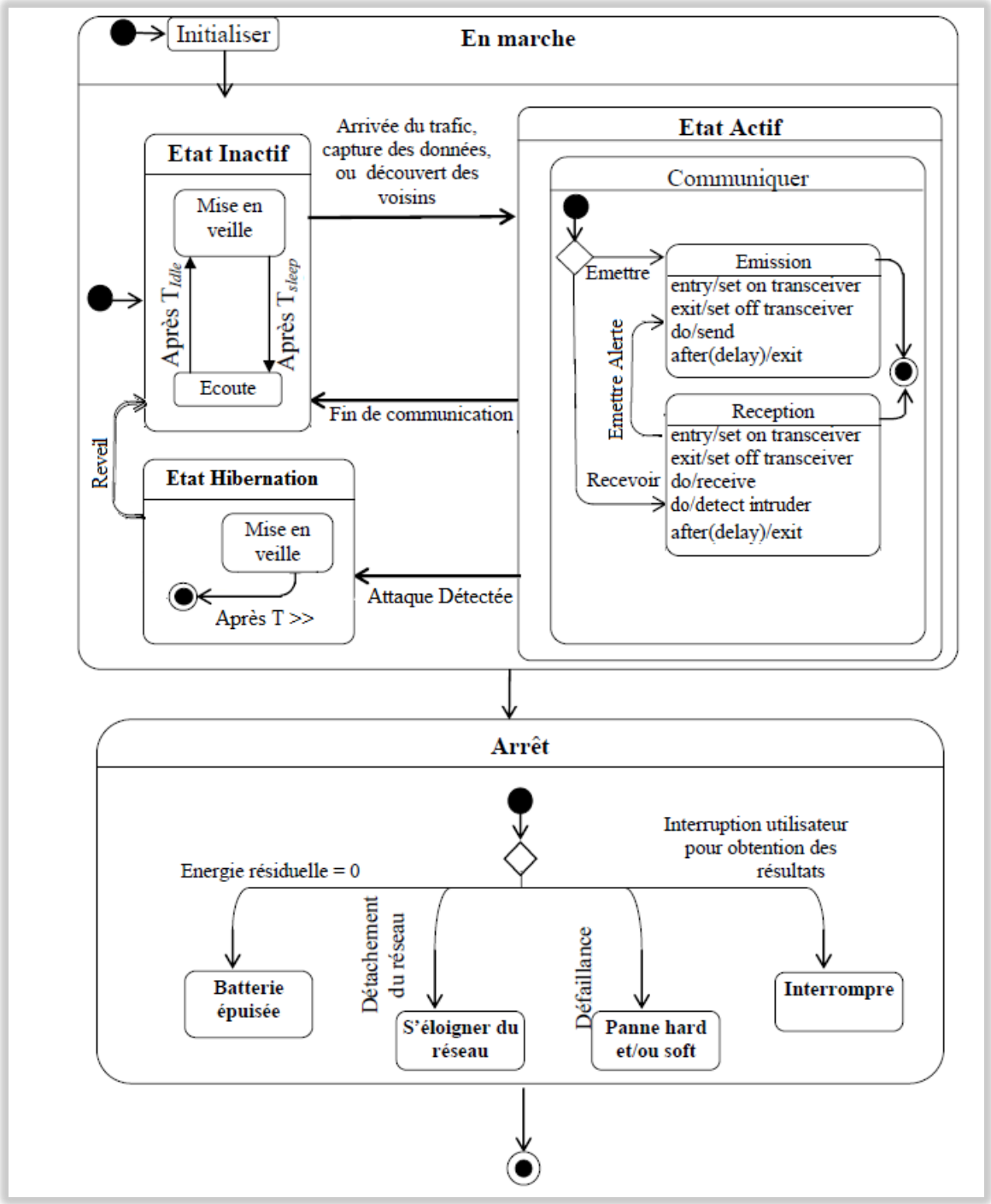


Figure 4-15: *Diagramme états-transitions d'un noeud capteur*

4.4.4. Diagramme de séquences

Le diagramme de sequences décrit le comportement temporel des acteurs (procédures). Une interaction est un acte qui consiste en un ensemble de messages échangés entre les objets du système. Les interactions peuvent être synchroniséesou asynchrones. La figure 4-16 montre le diagramme de sequences résumant le fonctionnement de base d'un noeud opérant avec le protocole CL-MAC sécurisé contre l'attaque wormhole.

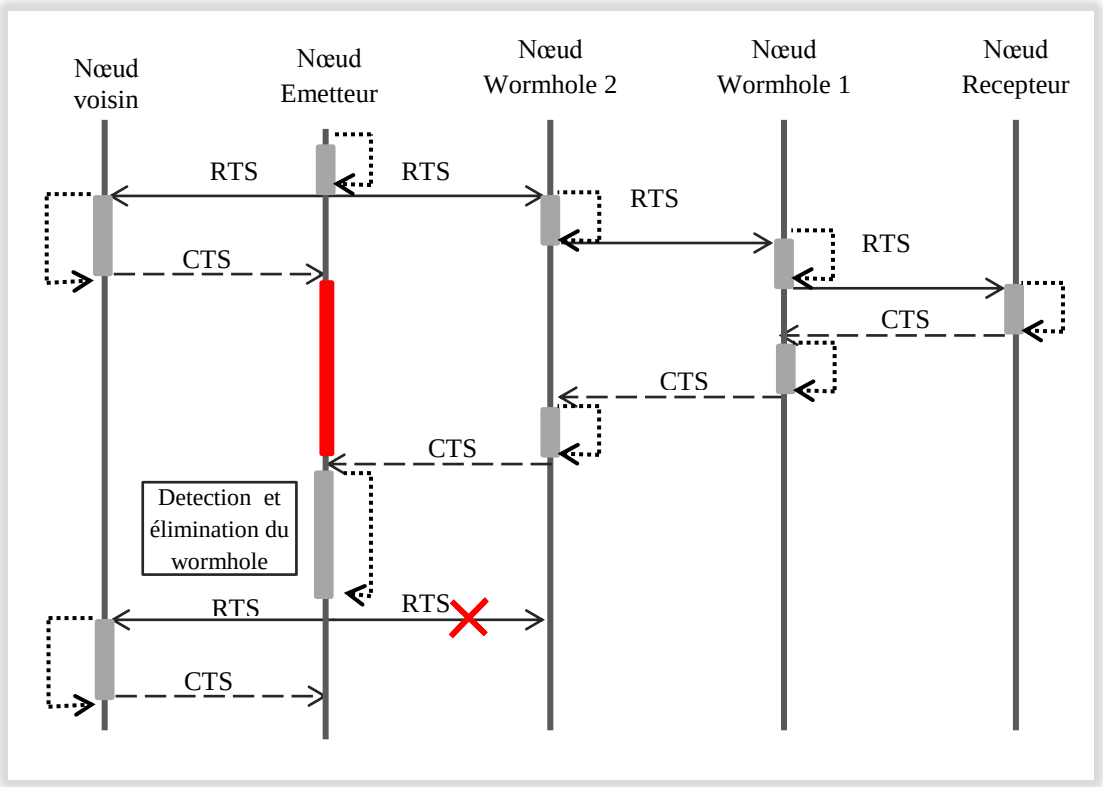


Figure 4-16: *Diagramme de séquence pour CL-MAC sécurisé contre l'attaque wormhole*

4.5. Conclusion

Le protocole CL-MAC a fait l'objet de plusieurs travaux de recherches [2-6] et [77] a été présenté en détails dans ce chapitre. Nous avons expliqué son fonctionnement de base et défini ses composantes. Lors de sa conception, le problème de sécurité n'a pas été pris en charge par les comcepteurs, ce qui nous a permis d'éditer une liste des points de vulnérabilité de ce protocole. Nous avons aussi présenté quelques diagrammes UML pour modéliser le nœud capteur et le fonctionnement du RCSF face aux attaques Sinkholeet Wormhole. Ces deux dernières attaques ont été choisies comme type d'attaques pour sécuriser le protocole CL-MAC. Le choix est justifié par le fait que l'attaque soit : (i) remèdiable à la cryptographie, (ii) simple à restaurer dans un RCSF, (iii) dévastatrice, (iv) difficile à détecter.

5. SECURISATION DU PROTOCOLE CL-MAC CONTRE L'ATTAQUE SINKHOLE

5.1. Introduction

Dans ce chapitre, nous présenterons une nouvelle version du protocole CL-MAC, une version sécurisée SCL-MAC (**S**ecured **CL-MAC** protocol) face aux attaques de type sinkhole. Nous avons adopté le mécanisme du round tripe time (*RTT*) [28] pour notre solution. L'extenssion du protocole CL-MAC est realisée au niveau de la couche MAC, exactement lors de la construction des listes de voisinage (neighbors discovery). L'algorithme de la nouvelle version sécurisée est modèlisée en UML et formellement validée par les réseaux de Petri Temporels (RDP-T) ou (Time Petri Net TPN). Nous avons aussi utilisé l'outil TiNA afin de valider de manière formelle certaines propriétés de la solution proposée. Pour la simulation expérimentale, nous avons utilisé le simulateur CASTALIA (Omnet++ 4.2/Castalia 3.0) sous l'environnement Ubuntu 11.2.

5.2. L'approche proposée.

Les contraintes physiques des RCSFs nous obligent à explorer d'autres solutions de sécurité que celles des réseaux traditionels. La crypthographie à clef asymétrique (clé publique + clé privée), est considérée comme coûteuse en matière de traitement CPU et consommation d'énergie [43] et [73-75]. Par coonséquent, elle parait inconvenable pour les réseaux de capteurs. Alors, nous avons proposé notre propore algorithme de sécurité face à l'attaque Sinkhole basé sur la techniqe du RTT. La solution est une amélioration des performances du protocole CL-MAC [1] et [4]. Le mécanisme de notre nouvelle version du CL-MAC s'articule sur trois étapes. Le principe de la solution est donné par l'algorithme 5-1. La partie déclaration del'algorithme CL-MAC a été enrichie par de nouvelles variables résumées dans le tableau 5-1.

Tableau 5-1: *Description des variables de l'algorithme*

Variables	Déscription
Neighbour_list	Liste de voisinage contient les identificateurs des noeuds se trouvant dans la portée radio.
Black_list	Liste noire: liste des identifiants des noeuds suspects localisées proches du noeud.
Timer	Variable de type temps initialisée au moment d'envoi du paquet *hello* par la valeur d'horloge du CPU du nœud.
Cure	Variable booléenne initianlisée à *true* pour indiquer que le noeud et ses voisins operant dans le mode saint (aucun nœud suspect intrus n'a été détecté jusqu'à présent).

Étape1: Initialisation

Dans cette première étape, certaines variables doivent être initialisées: les deux listes, celle des voisins et la liste noire, sont initialisées par la valeur *nil*, la variable *Timer* reçoit la valeur d'horloge du noeud au moment de la diffusion du message *Hello* à l'aide de la primitive (*gettime()*) et la variable booléenne *cure* est initialisée par la valeur *true* (aucun noeud n'est présent dans la zone de couverture radio du noeud).

Étape2: découverte du voisinage (Neighbour discovery)

Chaque noeud diffuse aléatoirement un message *Hello* à ses voisins dans l'objectif de construire sa liste de voisins. Supposons que le noeud "*A*" envoie un message *hello* pour découvrir ses voisins, immédiatement, il initialise sa variable *timer* par la valeur de son horloge. Les noeuds entourant "*A*" se trouvant dans sa portée radio répondent par un message *Hello-replay* une fois avoir reçu le paquet du message *hello* du noeud "*A*". Le noeud "*A*" construit ainsi sa liste des voisins en se basant sur les paquets reçus des messages *hello-replay* de ses voisins.

La différence temporelle entre l'instant de la réception du message *hello-replay* et celle de l'émission du message *hello* par le nœud initiateur du processus de recherche de voisins (Temps de réception du *hello-replay* – temps de déffusion du *hello*), dans l'équation (5) nous avons (receiving time – timer variable), nous permettra de lancer un jugement de detection d'anomalie ou de dysfonctionnement. Si cette différence est inferieure ou égale un seuil prédefinis (la valeur du seuil = temps de propagation des paquets * 2 + temps CPU nécessaire du traitement de ces paquets) alors le noeud "*A*" emplile l'identificateur du noeud "*B*" dans sa liste de voisins (noeud qui a envoyé le paquet *hello-replay*) et incrémente le nombre de ses voisins (l'identifiant du noeud "*B*": ID-B = contenu du champ source du paquet reçu). Autrement dit, le noeud "*A*" admettra qu'un noeud malicieux ou corrompu soit présent dans le voisinage. En ce moment, le noeud "*A*" alors affectera la valeur *false* à la variable *cure* comme indication d'une présence d'attaque, ajoutera l'identificateur du noeud "*B*" à sa liste noire, diffusera sa liste noire à l'ensemble de ses voisins pour les informer, et enfin, il se mettra en mode hibernation pour une période estimée assez suffisante pour dépasser cette situation (situation d'attaque).

$$RTT = gettime() - Timer \qquad (2)$$

Etape3: Recherche de chemins de routage

Dans cette étape, si le noeud fonctionne dans un environnement saint, le processus de recherche de chemins de routage est initié.

```
1.   Algorithm Secured_CL-MAC ;
2.    Input   : R-Table;    // routing table
3.    Output  : Black_list; // list of faked nodes
4.   Var
5.     Timer : time_type ;
6.     Cure  : Boolean;
7.     Neighbour_list, Black_list : list_type
8.   Begin
9.     Cure←true;
10.    Neighbor_list = Black_list ← Nil
11.  While(Cure)Do
12.  Begin
13.       Construct_Hello_Message;
14.       Broadcast_Hello_Message;
15.       Timer←getTime;
16.       Receive_Hello_Repply;
17.       If ((getTime-Timer) <  Throwshould  ) then
18.         Add node_ID to Nieghbours_list
19.       Else
20.         Cure ← false;
21.       Endif
22.       If ( Not(cure)) then
23.         Black_list ← Black_list+node_ID(Hello_Reply)
24.         Inform_neighbours_Black_list;
25.         Exit
26.       Else
27.         Wait_next_period;
28.       Endif
29.  Endwhile
30.  EndAlgorithm
```

Algorithme 5-1 : *Algorithme de la solution proposée*

5.3. Modélisation formelle du protocole CL-MAC sécurisé

Dans l'ensemble, un RCSF opérant avec le protocole CL-MAC ou sa version sécurisée (SCL-MAC) ou tout autre protocole de communication, se compose de plusieurs nœuds de capteurs tout autour et partageant le médium de communication sans fil. Dans la figure 5-1, les noeuds initiateurs d'une communication et ceux jouant le rôle de routeurs sont présentés par le nœud *sender* alors que le sink et les autres nœuds récepteurs d'une communication, selon la définition traditionnelle (pour un saut), sont représentés par le nœud *receiver*. Le reste des nœuds forment le réseau. Dans la figure 5-2, un modèle plus simplifié du protocole CL-MAC sans retransmission ni détection d'attaque. Le réseau complet est représenté par trois variantes: émetteur sur la partie gauche, recepteur sur la partie droite, et réseau au milieu (voisinage des noeuds communicants).

5.3.1. Hypothèses du modèle RDPT

Les paquets échangés entre deux noeuds successifs (communication à un saut) sont : RTS, CTS, DATA, et ACK. Dans notre modèle, que nous avons proposé pour modéliser le fonctionnement du protocole CL-MAC, nous avons défini les intervalles de temps nécessaires pour la propagation de chacun de ces paquets comme suit :

- Les paquets RTS, CTS,et ACK consomment, chacun, 03 unités de temps.
- Le paquet DATA, consomme 10 unités de temps afin d'être complètement transmis.
- Les intervalles de contrôles SIFS et DIFS consomment une unité de temps.

Tableau 5-2: *Déscription des transitions du modèle RDPT de la figure 5-1*

Transition	Description
*t*1	L'envoi du paquet RTS au nœud du prochain saut
*t*2	L'envoi du paquet DATA par l'émetteur
*t*3	Réception de l'ACK par l'émetteur
*t*4	Débuter une nouvelle communication
*t*5	L'envoi du paquet CTS par le noeud récepteur
*t*6	L'envoi du paquet ACK par le noeud récepteur
*t*7	Le noeud récepteur se prepare pour une nouvelle communication
*t*8,*t*9	*t*8 représente le pivotement en mode veille d'un noeud voisin non concerné par la présente communication, et la transition *t*9 schématise le mode réveil du voisin après que la communication dans son voisinage est achevée.

La figure 5-2 trace la version sécurisée du protocole CL-MAC représentée par un modèle de réseau de Petri Temporel (RDP-T).Le RDP-T modèlise aussi le processus de détection de l'attaque Sinkhole. Avec le marquage initial $M_{0,}$ nous avons ajouté une partie matérialisant la variable *timer*. Cette dernière, formée de : transition *t*12, *t*13, et *t*14et la place *p*0, nous permettra de conclure qu'il y a des intrus dans le voisingae. La valeur du RTT = temps de propagation des paquets RTS et CTS + deux temp SIFS + un delta temps lié à l'environnement.

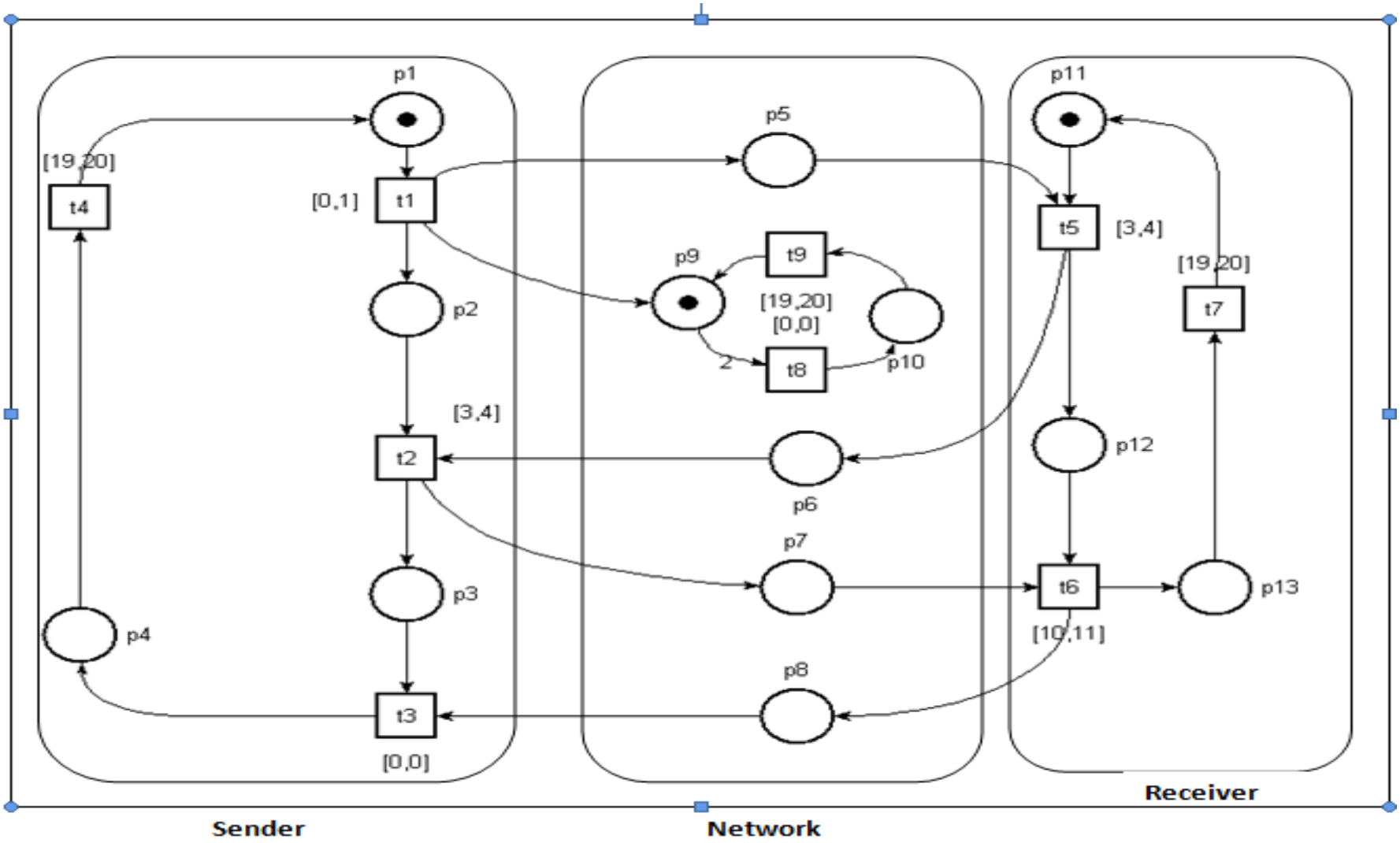


Figure 5-1: *RDP-T du modèle réduit du protocole CL-MAC.*

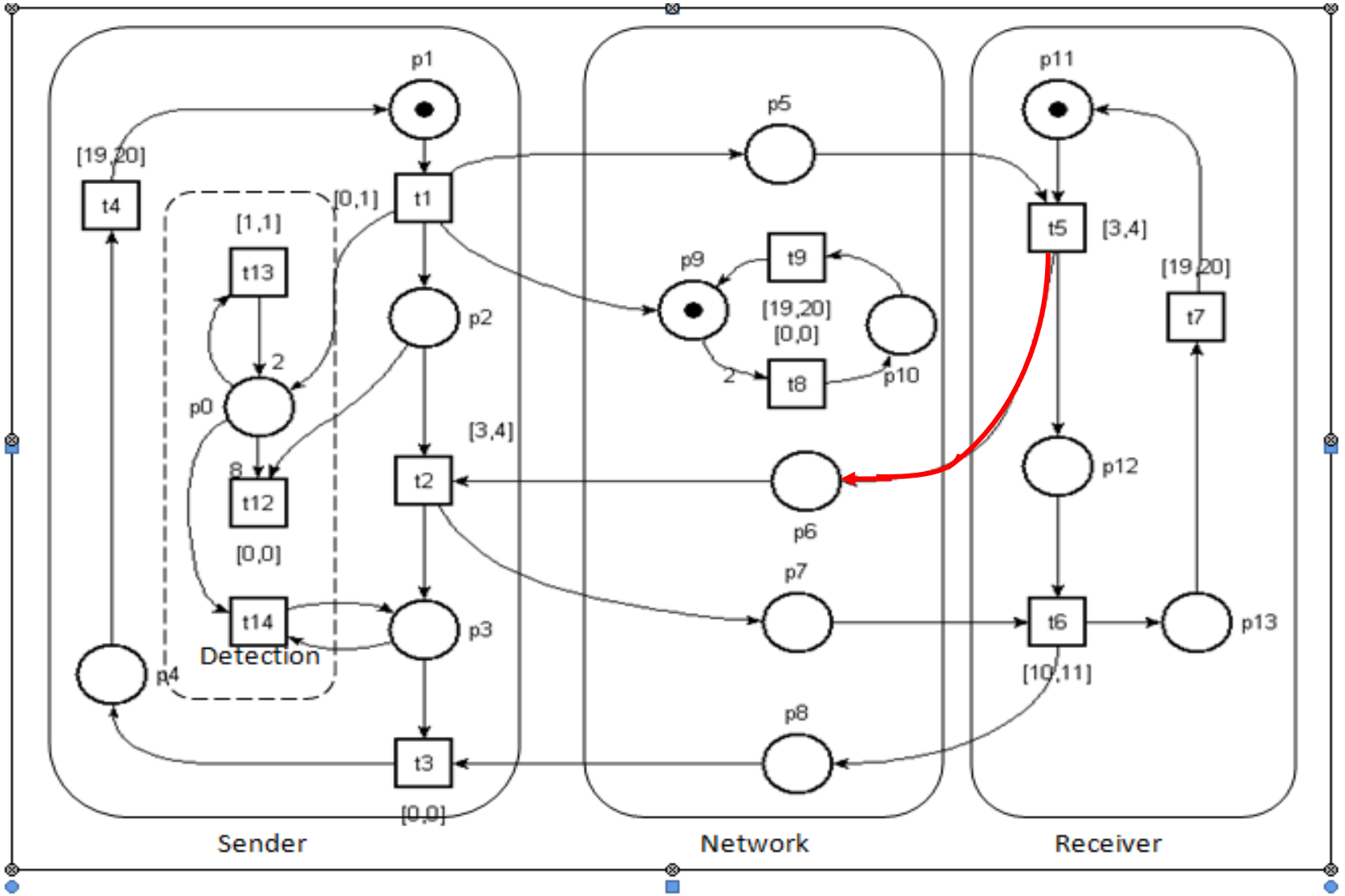


Figure 5-2: *RDP-T du protocole SCL-MAC face à l'attaque sinkhole.*

$$M_0 = \begin{bmatrix} p1 \\ p9 \\ p11 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad (3)$$

Tableau 5-3: *Description des transitions duRDP-T de la figure 5-2*

Transition	Description
*t*1	Envoi du paquet RTS au noeud du prochain saut
*t*2	Envoi du paquet DATA par l'émetteur
*t*3	Réception du paquet ACK par l'émetteur
*t*4	Initiation d'une nouvelle communication
*t*5	Envoi du paquet CTS par le noeud récepteur
*t*6	Envoi du paquet ACK par le noeud récepteur
*t*7	Pivotement vers le mode réveil du nœud récepteur
*t*13	Incrémentation de la valeur de la variable *timer*après chaque unité de temps (time slot)
*t*12	Détection de l'attaque Sinkhole
*t*14	Initialisation de la variable *timer* quand l'attaque n'a pas eu lieu ou elle n'a pas été détectée.
*t*12,*t*13,*t*14	Mécanisme de détection de l'attaque sinkhole
*t*8,*t*9	*t*8 représente le pivotement en mode veille d'un noeud voisin non concerné par la présente communication, et la transition *t*9 schématise le mode réveil du voisin après que la communication dans son voisinage est achevée

Nous supposons, dans cette partie, qu'il y a une attaque de type sinkhole organisée par un noeud malicieux bien implanté dans le réseau. Ce noeud malicieux se présente comme un noeud favori sur le chemin de routage (dans le modèle RDP-T présenté c'est le nœud récepteur qui jouera le rôle du sinkhole). Alors, l'arc liant la transition "*t*5" à la place "*p*6", en rouge sur la figure 5-2, lorsqu'il est ôté, il simulera la fonction d'une attaque sinkhole passive (obsorption des paquets). Dans ce cas, nous avons modélisé ce comportement par la variable *timer*. Cette variable fonctionne comme un compteur. Elle est incrémentée après chaque unité de temps jusqu'à atteindre la valeur de 8 unités (c'est le temps de propagation d'un RTS + CTS + 2 * DIFS). Les huit jetons qui s'accumuleront dans la place "*p*0" représentent la valeur

du seuil (threshold value) à la quelle est comparée la variable timer dans notre algorithme (ligne d'instruction n° 17), cette variable *timer* est imagée par la place "*p*0". La transition "*t*13"incrémente la variable *timer*. Après chaque unite de temps, "*p*0" sera incrémentée par un jeton, la transition "*t*12"modélise la fonction teste (**if** timer > seuil**then**), elle compare le contenu de la variable *timer* (nombre de jetons dans la place "*p*0") avec la valeur pré-calculée seuil (8 unités de temps). Sur le modèle présenté par la figure 5-3, nous voyons que la transition "*t*12" est sensibilisée avec 8 jetons dans la place "*p*0". Si la transition "*t*12" est tirée, celà expliquera que le paquet RTS n'a pas reçu d'échos (aucune réponse n'a été donnée par les voisins), et comme dans notre cas, l'attaque sinkhole est détectée après avoir expiré les 8 unités de temps. La transition "*t*14", quand le réseau opère dans l'absence d'une attaque, met à zero le compteur *timer* pour permettre au nœud de continuer de fonctionner (zéro jeton dans la place "*p*0").

La figure 5-3 visualise le comportement operationel d'une manière globale d'un système de noeuds de capteurs. Supposons que le nœud "*A*" possède des données à communiquer au nœud "*B*". Noeud "*A*" initialise le processus par l'envoi d'une requète (*RTS*) au nœud "*B*". La destination "*B*"répond par un paquet (*CTS*). Dès la réception du *CTS*, le nœud "*A*"envoie son paquet de données (*DATA*). Si le paquet (*DATA*) est bien reçu, le noeud "*B*" informe son interlocuteur, le noeud "*A*", par un message de confirmation sous forme d'un acquitement (*ACK*). Si le nœud "*A*"a un message de données fragmenté en plusieurs fragments, il attend durant des intervalles aléatoires après chaque fragment transféré avec succès.

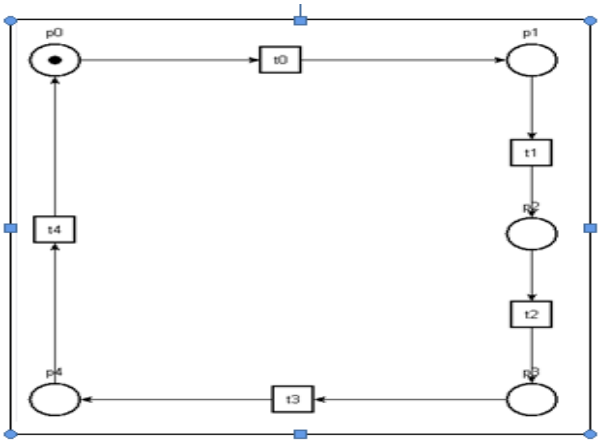


Figure 5-3: *RDP modélisant une connexion entre deux noeuds*

Tableau 5-4: *Description des transitions du graphe de la figure 5-3*

Transition	Description
*t*0	Envoi du paquet RTS au noeud du prochain saut
*t*1	Envoi du paquet CTS par le noeud du prochain saut vers l'initialteur
*t*2	Envoi du paquet DATA au noeud du prochain saut
*t*3	Réception du paquet ACK en provenance du noeud du prochain saut
*t*4	Noeud initiateur attend pour un temps aléatoire avant d'envoyer le fragment suivant, ou pour initier une nouvelle communication

L'analyse d'accessibilité par TiNA avec l'option "Essential States" a donné le résultat illustré par la figure 5-4. Nous voyons que le réseau RDP-T est à la fois : (i) borné (k-Borné avec k=8, le nombre de jetons dans la place P0 avec la présence du nœud malicieux exerçant l'attaque Sinkhole), (ii) non réversible : le nœud capteur détecteur de l'attaque se met en état d'hibernation, cela est modélisé par la non injection d'un jeton dans la place P1 pour pemettre au réseau RDP-T d'initialiser une nouveau cycle (la propriété d'irréversibilité correspond bien au fonctionnement de la solution proposée, (iii) non-vivant : Six transitions sont marquées mortes, il s'agit de t7, t6, t4, t3, t2, et t14. Les transitions t4 et t7 permettent la ré-initialisation du réseau et ,dans cette ce cas le réseau RDP-T ,ne sera plus ré-intialisable. Cette propriété reflète le fait de la diffusion du message d'alerte « MA » et le passge des nœuds du voisinage en mode hibernation.

Le fichier complet de cette analyse est présenté dans l'annexe C. Le fichier texte généré par TiNA lors du lancement de l'analyse du modèle RDP-Tde la solution proposée est illustré par la figure 5-2 et est présenté dans l'annexe D.

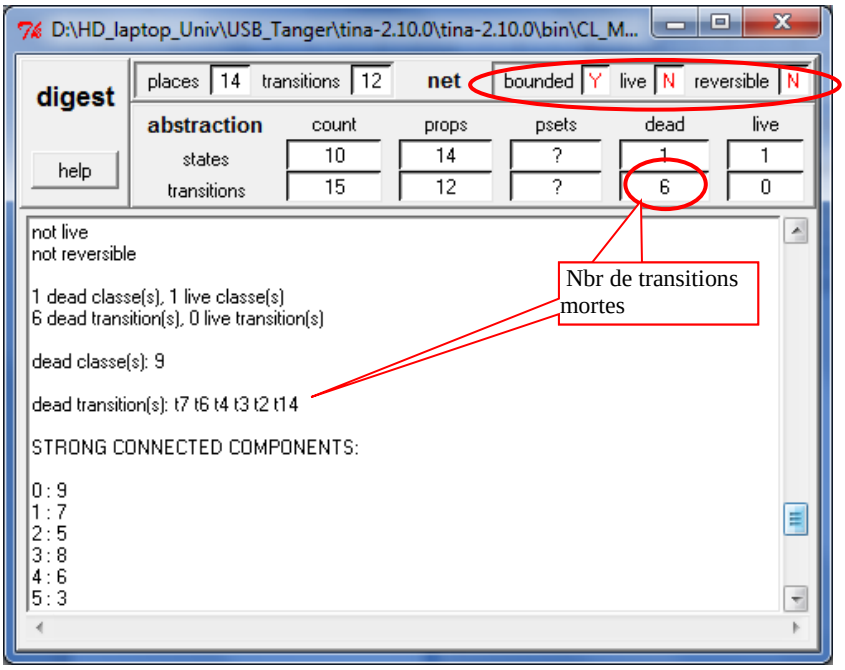


Figure 5-4 : *Résultat d'analyse par TiNA du RDP-T modèle*

5.4. Simulation et résultats

Pour simuler notre solution pour la sécurisation du protocole CL-MAC face aux attaques de type Sinkhole, nous avons opté pour le simulateur OMNET++/Castalia, comme outil de similation, ce dernier propose une grande variété de protocoles prédéfinis pour toutes les couches de la pile protocolaire [76]. Dans cette section, nous présenterons la simulation des performances de notre algorithme proposé. Nous avons, aléatroirement, déployé les noeuds du réseau dans un espace carré variable d'une expérience à l'autre. Le tableau 5-5 résume les paramètres de simulation sous castalia.

Tableau 5-5: *Paramètres de Simulation*

Paramètres	Valeurs
Temps de Simulation	100 sec,500 sec
Espace de déploiement	180X180m
Nombre de noeuds	6,10,20,30,40,50,60,70,80,90, 180
Le noeud Sink	Node 0
Le noeud Sinkhole	Node 4
Paramètre Radio	CC2420
Énergie de transmission (TX Power)	-5dbm
Protocole MAC	CL-MAC

Pour bien illustrer l'effet de l'attaque sinkhole, nous avons comparé le comportement du CL-MAC avec la présence et l'absence de l'attaque (mode safe et mode weak) pour évaluer les performances du protocole en fonction de l'érnegie consommée et du taux de réception et de perte de données échangées.

Comme le montre la figure 5-5, nous observons que chaque fois que le nombre de noeuds augmente, l'énergie consommée augmente quelque soit le mode de fonctionnement (safe ou weak modes). Nous remarquons aussi la différence d'énergie consommée dans ces deux modes. En présence d'une attaque, l'énergie consommée dépasse celle consommée en l'absence de l'attaque suivant cette équation (cette équation est valide sous nos conditions et dans notre environnement de simulation):

$$E_{weak-mode} = \alpha \times E_{safe-mode} + \beta \qquad (4)$$

Avec **α**= 2, 0975 et β = - 0,214, selon les résultats extraits de la simulation. La sur-consommation dans le mode weak par rapport à celle du mode safe est due au phénomène de retransmission des paquets de contrôle. Au départ et avant qu'un nœud ne puisse conclure qu'il s'agit d'une attaque, il essaiera de retransmettre le paquet plus d'une fois tout en ayant l'air d'une perte du paquet et non pas de son absorption par un sinkhole.

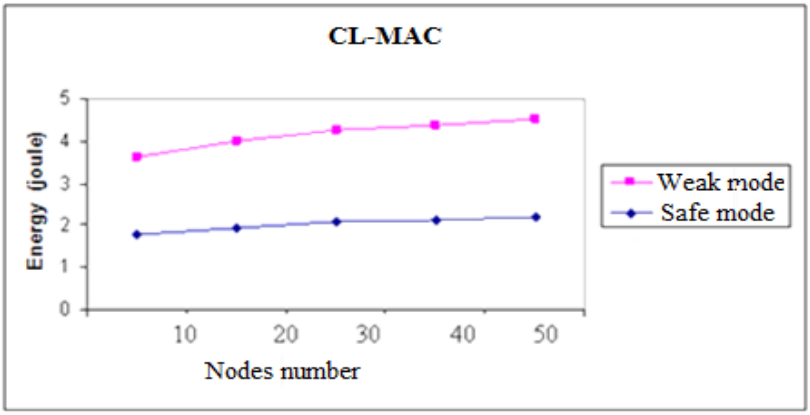


Figure 5-5: *Energie Consommée*

Dans les modes opérationnels safe ou weak, le rapport de réception des données communiquées du protocole CL-MAC sécurisé est tracé par la figure 5-6. Nous constatons que lorsque le réseau devient de plus en plus dense, le nombre d'échecs de paquets transmis augmente et contrairement en ce qui concerne le nombre de succès de paquets communiqués qui se voit décroitre. Cela est la conséquence de plusieurs facteurs environnementaux de la zone de déploiement du RSCF.

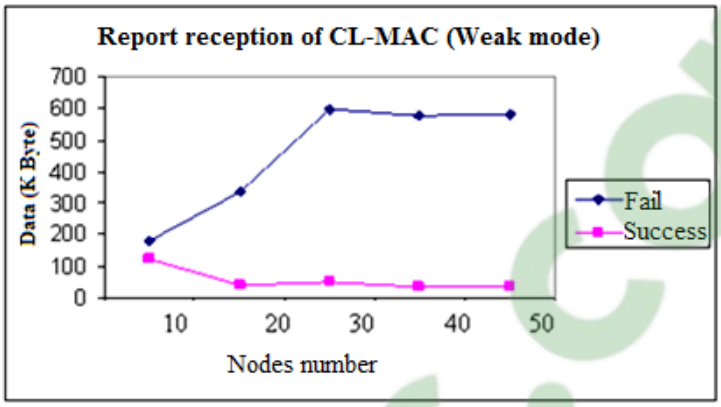


Figure 5-6: *Report de réception des données du protocole SCL-MAC en mode Weak*

Dans la figure 5-7, la comparaison du taux de perte entre les deux modes safe et weak dans la réception des paquets du CL-MAC, nous confirmons, selon les résultats de simulation, que quelque soit la taille du réseau (le nombre de nœuds), le taux de perte des paquets dans le mode vulnérable est beaucoup plus grand que celui réalisé dans le mode de fonctionnement dans un evironnement saint. Ceci est une évidence que la cause est due à la presence d'un noeud malicieux exerçant une attaque de type sinkhole. Les paquets transistant par ce noeud sont absorbés.

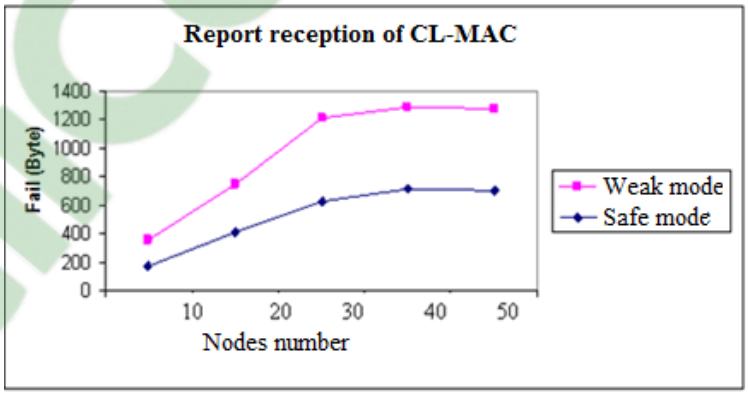


Figure 5-7: *Rapport d'échec de réception pour CL-MAC dans les deux modes de fonctionnement*

La figure 5-8 montre l'avantage de notre solution proposée. Il ressort de ces courbes que malgré la presence d'un noeud malicieux *sinkhole* dans le voisinage des chemins de routge conduisant vers le sink, les paquets de données générés par les nœuds ont atteint leurs destinations (la base station). Nous remarquons aussi que le deux courbes, représentant le mode de fonctionnement saint et vulnérable, se convergent lorsque le réseau devient plus dense (la taille du réseau dépasse 50 nœuds : cette remarque est valide sous les conraintes de notre simulation). La présence d'un seul noeud malicieux exerçant l'attaque *sinkhole* dans un

réseau de taille égale à 50 ou plus est moins dangereuse ou presque sans effet. C'est ce qui traduit la convergence des deux courbes avec l'augmentation de la taille du réseau.

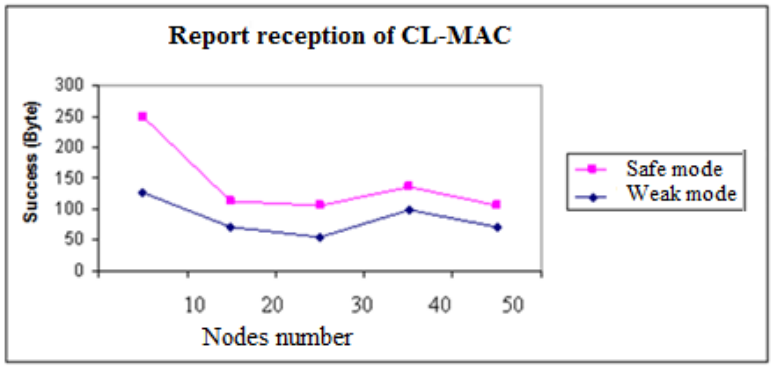


Figure 5-8: *Réception des données dans le mode saint et hostile*

Dans la figure 5-9, nous pouvons constater que dès que l'ataquant s'approche de la station de base (le sink reel), le taux de perte des paquets augmente et vice versa. Cela se justifie comme suit : les chemins de routage convergent tous vers la station de base, et en s'aprochant du sink, le nœud malicieux prendra le contrôle de plus de chemins. Sa zone de portée radio couvrira un nombre important de chemins de routage, et si l'attaquant s'indentifie comme un voisin en ce moment, il aura le contrôle total du trafic réseau.

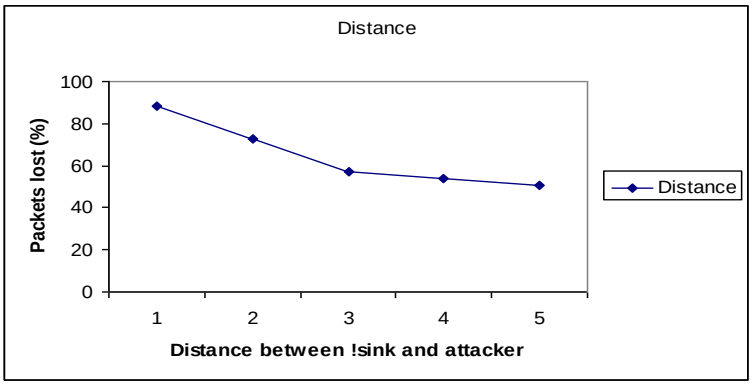


Figure 5-9: *Distance entre le Sink et Sinkhole.*

5.5. Conclusion.

Dans ce chapitre, nous avons traité le problème de la sécurité du protocole CL-MAC face aux attaques sinkhole. La philosophie du CL-MAC nous a imposé des contraintes supplémentaires et beaucoup plus sévères additionnées à celles des RCSFs. Les méthodes de cryptographie (asymetrique et symétrique, sténanographie, watermaking, etc.) se révèlent impraticables pour le CL-MAC. Il nous a fallu une méthode légére qui ne consomme presque pas d'énergie. La technique RTT figurait comme un choix idéal.

L'adaptation du RTT pour le protocole CL-MAC a été modélisée UML, et formellement validée par les réseaux de Petri temporels. L'outil TiNA nous a permis de vérifier quelques propriétés formelles du protocole sécurisé. Nous avons, aussi, effectué des simulations de l'algorithme sous OMNET++ et CASTALIA. Les résultats obtenus, que se soient à partir de TiNA ou ceux issus de la simulation, montrent l'efficacité de la solution face à l'attaque sinkhole.

6. SECURISATION DU PROTOCOLE CL-MAC CONTRE L'ATTAQUE WORMHOLE

6.1. Introduction.

Dans ce chapitre, nous présenterons le fonctionnement de base de notre version sécurisée SCL-MAC pour pallier les insuffisances ainsi que les failles de sécurité présentes dans le protocole CL-MAC. La solution est une adaptation de la technique RTT. Le protocole résultat a été doublement vérifié: en premier lieu, nous avons proposé un modèle formel de réseau de Petri temporel pour vérifier les propriétés de notre nouveau protocole. En deuxième lieu, nous avons utilisé le simulateur NS-2 et effectué une série de tests pour attester du bon fonctionnement du protocole.

Dans la seconde partie d'évaluation réservée au test NS-2, nous avons proposé trois scénarios: (i) le protocole est testé dans un environnement saint, (ii) deux nœuds malicieux matérialisant l'attaque de trou de ver sont activés sauf que l'attaque et passive où ces nœuds intrus sont installés hors des chemins de routage ou que le nœud initiateur de la communication est près du sink, (iii) identique au second scénario avec un bon emplacement des nœuds malicieux sur le même chemin de routage.

Les résultats formels sont beaucoup plus expressifs que les résultats expérimentaux produits par NS-2. Formellement, nous avons prouvé qu'en présence d'une attaque, le nœud détecteur de l'attaque sera écarté des chemins de routage passant par le voisinage de ce nœud.

6.2. Solution proposée

Dans [66], un mécanisme de détection des attaques de trou de ver basé sur le Round Trip Time (*RTT)* pour la sécurisation du protocole de routage AODV dans les RCSFs a été présenté. La détection se fait sur la base du calcul du temps qu'un message prend pour circuler entre les nœuds successifs et leurs voisins membres. Les considérations faites ici est que l'adversaire augmente le nombre de voisins d'un nœud dans sa portée radio, réduit le chemin de routage, et augmente la valeur RTT entre les nœuds successifs. Ce mécanisme suit trois phases: la construction de la liste de voisinage pour chaque nœud (*neigbors list*), la recherche des chemins entre les nœuds sources et la destination (*sink*) et enfin la localisation du lien wormhole (identification des positions des deux nœuds malicieux extrémités de l'attaque trou de ver) pour permettre de lancer l'action nécessaire contre cette attaque.

Dans une attaque typique de type *wormhole*, l'attaquant reçoit les paquets dans un point du réseau, et les transmet via une voie filaire ou sans fil avec beaucoup moins de latence comparée à celle d'un lien ordinaire utilisé par le réseau, puis il les injecte dans un autre endroit distant du réseau. Ici, nous supposons que l'attaque *wormhole* est bidirectionnelle montée avec deux nœuds malicieux quoiqu'en théorie l'attaque *wormhole* avec multi-extrémités existe. L'attaque peut être passive, quand les paquets sont reproduits dans un endroit loin du voisinage, ou active lorsque les paquets ne sont pas seulement reproduits mais plutôt modifiés (les champs de contrôle qui sont généralement touchés) ou de nouveaux paquets sont générés [67].

Comme l'illustre la figure 4-11 du chapitre 4 précédent, pour l'attaque *wormhole* passive, nous avons employé le mécanisme du RTT pour détecter la présence des intrus dans un voisinage et ceci dans les deux couches protocolaires, la couche réseau (lors des échanges des paquets route request et route replay) et la couche MAC lors de la mise à jour de la liste de voisinage (la diffusion du paquet Hello dans un voisinage). Et pour la seconde forme d'attaque wormhole, la forme active lorsque même l'entête du paquet est modifiée de telle sorte qu'il puisse atteindre un des nœuds distants voisin de la seconde extrémité du wormhole, nous avons simplement utilisé: (i) une variable booléenne (*Flag*) qui indique la réception du bon paquet comme étant une réponse à un paquet initial provenant d'un nœud figurant dans le chemin de la communication en cours (le nœud voisin d'un saut).(ii) Une variable Black-List,contenant les identificateurs des noeuds suspects, utilisée par le noeud routeur placé just avant le noeud suspect dans le chemin du routage pour éviter la partie infectée du chemin et voir un chemin alternatif. Chaque fois qu'un nœud détecte une attaque dans son voisinage, il diffuse un message d'alerte "AM" (la figure 6-1 montre la structure du message d'alerte). A la fois, cela permet aux nœuds, prédécesseurs et successeurs dans le chemin allant de la source au sink, de mettre à jour leurs variables black-list respectives (voir figure 6-1) [68].

Frame Control	ID (Fakednode)	Alert Message	CRC

Figure 6-1: *Structure du message d'alerte "AM"*

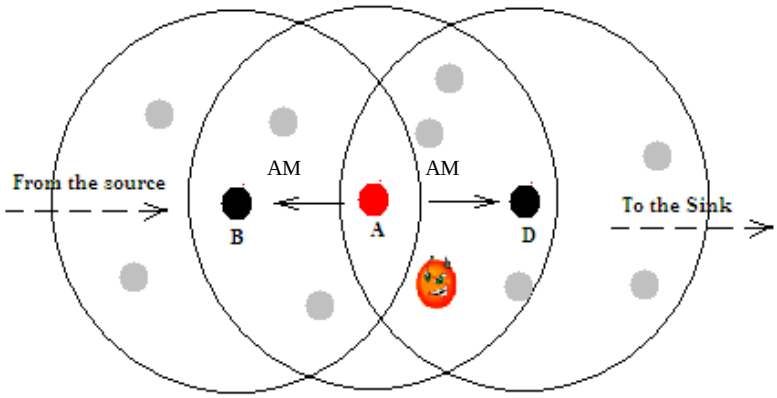


Figure 6-2: *Evitement de l'attaque à l'aide du message AM*

Algorithm: SCL-MAC

```
1: Algorithm Secured CL-MAC
2:   Input   : R-Table; // routing table
3:   Output  : AM;      // alert message
4: Var
5:   Black-List, neighbors_list: id*;
6:   Flag    : Boolean;
7:   Timer   : time type;
8: Start
9:   Built-Neighbors-List (Neighbors_list);
10:    Synchronize-the-Scheduler;
11:    Get-Routing-Table(R-Table);
12:  Repeat
13:       Flag ← False;
14:       M ← Construct-Message(R-Table);
15:       Send(M);
16:       Timer ← Get-Time();
17:       Wait;
18:       Receive(M');
19:       If ((Get-Time()- Timer)≤ threshold) Then
20:       Case Type-of-Message(M') Of
21:           Replay: If Flag Then
22:                     Goto X; //the second replay
23:       Else   // 1st message replay
24:                     Flag ← True;
25:       End-If
26:           Alert:  Update-Black-List;
27:          // Insert AM.ID into Black-List
28:       End-Case-Of
29:       Else
30:         X: There is an attack;
31:            AM ← Construct-Alert-Massage;
32:            Broadcast(AM);
33:            Halt-communication;
34:       End-If
35:  End repeat
36:  End-Algorithm
```

Algorithme 6-1: *Algorithme du protocole SCL-MAC*

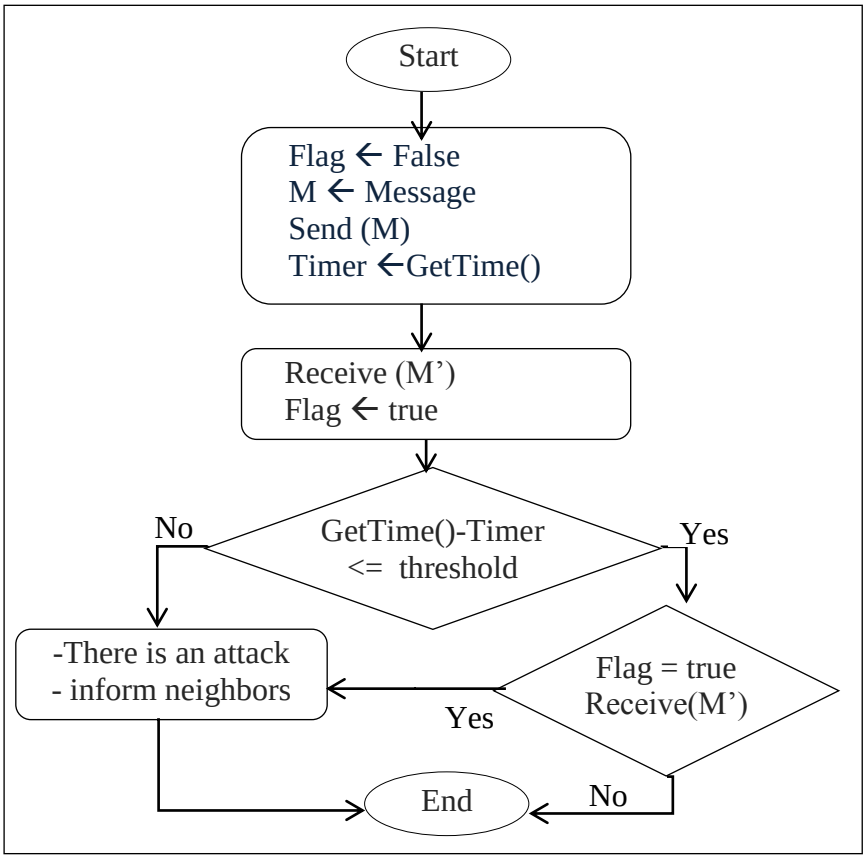


Figure 6-3: *Organigramme du fonctionnement de base de SCL-MAC*

6.2.1. Détection de l'attaque Wormhole lors du processus d'établissement des chemins de routage el de la découverte du voisinage

Dans cette phase, la détection de l'attaque wormhole est assurée par le mécanisme RTT discuté dans [66].

6.2.2. Détection de l'attaque Wormhole lors des communications

Après l'établissement des chemins de routage ainsi que les listes de voisinage pour chaque nœud du réseau, et pendant le lancement de la seconde phase du protocole CL-MAC (phase exploitation). Nous avons défini la variable booléenne *Flag* qui est associée au paquet RTS. Cette variable est initialisée par la valeur false, indiquant que le nœud émetteur "*A*" est en attente d'un paquet CTS comme une reponse provenant de son interlocuteur (le nœud dont l'identificateur est le contenu du deuxième champ de l'entête du paquet RTS), le nœud "*B*".

Quand le nœud "*A*" reçoit le paquet CTS du nœud "*B*"comme réponse à son paquet RTS, à ce moment, il affecte la valeur *true* à la variable *Flag*et commence à envoyer le/les paquet(s) DATA à destination du nœud "*B*".

Dans un mode de fonctionnement saint sans aucune attaque, tous les nœuds dans le voisinage de ceux communicant, entrent en mode veille permettant ainsi aux nœuds "*A*" et "*B*" de communiquer clairement, et quatre paquets seront échangés (RTS, CTS, DATA and ACK). Supposons maintenant que l'on soit en présence de deux nœuds malicieux garantissant l'attaque trou de ver (voir les figures 4-9 et 4-11). S'il s'agit d'une attaque passive, la

réplication du paquet RTS dans l'autre partie du réseau, comme l'illustre la figure 4-9, n'aura aucun effet sur la communication en cours sur le chemin 1, ceci est garanti par le fait que les paquets RTS, CTS sont des paquets unicast. Par contre, toute communication, dont le chemin de routage passe par un des nœuds se trouvant dans la zone2, sera altérée. La réplication du paquet RTS dans cette zone par la seconde extrémité du trou de ver force les nœuds de cette zone 2 à éteindre leurs transceivers et passer en mode veille, ainsi brisant momentanément le chemin 2.

Si les attaquants sont bien placés comme dans le schéma de la figure 4-9 (a), la première extrémité du trou de ver prend le paquet RTS généré par le nœud "*A*" dans la zone1, puis la seconde extrémité le réplique dans la zone2. Alors, tous les nœuds du voisinage entrent en mode veille conduisant ainsi à un cas de dysfonctionnement du protocole CL-MAC.

Si l'attaque est active de telle sorte que le champ destination de l'entête des paquets RTS et CTS sont modifiés par les nœuds malicieux extrémité du trou de ver de manière à ce que le nœud "*C*" qui est loin du nœud "*A*" puisse recevoir le paquet RTS et produire une réponse CTS qui sera reçue par "*A*" après qu'il soit injecté dans son voisinage. Dans cette situation, le nœud "*C*" doit répondre à son faux voisin le nœud "*A*" (voir la figure 4-11). Le temps écoulé entre l'émission du paquet RTS et la réception du paquet CTS provenant du nœud "*C*" est plus grand que le temps nécessaire pour un allé et retour entre les nœuds "*A*" et"*B*". Quand le paquet CTS du nœud "*C*"atteint le nœud "*A*", ce dernier vérifie sa variable *Flag* si elle est *true*. Dans cette situation, la présence d'une attaque trou de ver est détectée. Le nœud "*A*" altèrera la communication en cours et lance un message d'alerte "AM" en mode diffusion à ses voisins. Les nœuds qui reçoivent le message "AM"retirent le nœud "*A*" de leurs liste de voisinage et l'insèrent dans la liste noire, etarrêtent de faire suivre les paquets au nœud "*A*". Sur la figure 6.6, les nœuds "*B*" et "*D*" ajoutent l'identificateur du nœud"*A*" à leurs listes noires respectives. Puis le nœud "*B*" essaiera de maintenir la communication entre la source et le sink par la recherche d'un nouveau chemin de "*B*" vers le sink en détournant la zone du nœud "*A*".

6.2.3. Evitement de l'attaque Wormhole dans CL-MAC

Dans la version SCL-MAC, le mécanisme d'évitement de l'attaque proposé pour cette version est basé sur les messages d'alerte initiés par le nœud détecteur de l'attaque et diffusés à ses voisins. Le nœud dans le chemin de routage qui est situé juste avant le nœud détecteur de l'attaque (dans les figures 6-2 et 6-5, il s'agit du nœud "*A*"). L'identifiant du nœud *A* sera supprimé de la liste des voisins et ajouté à la liste noire. Le processus de mise à jour de la table de routage se lance automatiquement après cette mis-à-jour du voisinage sans prendre en compte le nœud"*A*".

Par un organigramme, la figure 6-4 illustre le fonctionnement du mécanisme d'évitement des attaques trou de ver tandis que la figure 6-5 montre l'effet du fonctionnement de cette procédure.

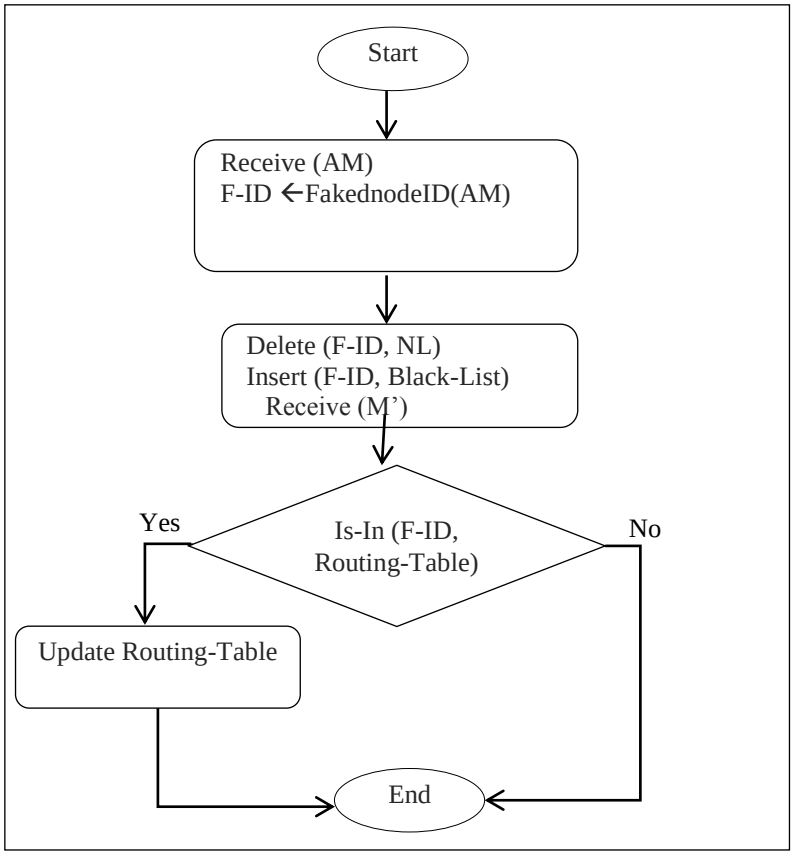


Figure 6-4:*Organigramme du mécanisme d'évitement de l'attaque trou de ver*

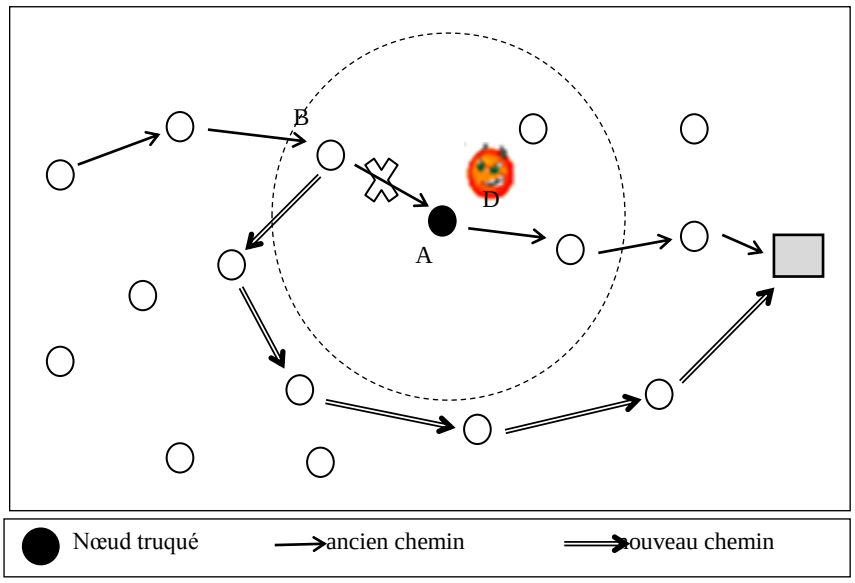


Figure 6-5: *Découverte d'un nouveau chemin en utilisant le mécanisme d'évitement*

6.3. Modélisation formelle

Afin de produire un modèle formel qui nous permet de vérifier d'une manière formelle le comportement de la version sécurisée SCL-MAC, nous avons choisi de le faire en utilisant un modèle mathématique convenable selon les spécifications du protocole SCL-MAC. Les réseaux de Petri temporels (*Time Petri Nets*) ou (RDP-T) [69, 70], sont beaucoup plus attractifs à la fois par leurs capacité de modéliser facilement les contraintes temporelles du protocole, et l'existence de l'outil d'analyse TiNA (*Time Net Analyzer*). TiNA est un environnement logiciel pour l'édition des réseaux de Petri et des automates [71]. TiNA nous permet de vérifier plusieurs propriétés d'un réseau de Petri comme la vivacité, le caractère bornétude, la réinitialisabilité, etc. La figure 6-6 illustrele modèle TPN de notre solution sécurisée du protocole CL-MAC.

6.3.1. Hypothèses du modèle

Les hypothèses imposées sur le comportement du protocole SCL-MAC sont les suivantes: nous avons supposé que les durées de propagation des paquets et des périodes séparatrices sont définies comme suit : La durée *DIFS* = *SIFS* = 1 unité de temps, les paquets de contrôle *RTS*, *CTS* and ACK consomment, chacun, 3 unités de temps, le paquet de données *DATA* requiert 10 unités de temps pour être transmis. Les deux extrémités de l'attaque trou de ver prennent une unité de temps pour pouvoir absorber un signal d'un endroit et le reproduire dans un autre endroit éloigné en utilisant un canal réservé (bande ultra fréquence ou lien filaire). Initialement, seulement les places $p1$, $p8$, et $p17$ sont marquées, chacune, par un jeton indiquant que le système (le protocole) est prêt à initialiser une nouvelle communication. Le marquage initial M_0 représente l'état après déploiement du réseau RCFS (L'état prêt pour établir la première communication).

$$M_0 = \begin{bmatrix} p1 \\ p8 \\ p17 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad (5)$$

6.3.2. Modèle TPN du protocole SCL-MAC

Les significations des transitions du modèle RDP-T de notre proposition sont données par le tableau 6-1 suivant :

Tableau 6-1: *Description des transitions du RDP-T du protocole SCL-MAC*

Transitions	Description
*t*1	*génération du paquet RTS* (*émetteur*)
*t*2	*émission du paquet DATA* (*émetteur*)
*t*3	*Entrée en mode veille* (*émetteur*)
*t*4	*Entrée en mode réveil* (*émetteur*)
*t*5	*émission du paquet CTS* (*récepteur*)
*t*6	*émission du paquet ACK* (*récepteur*)
*t*7	*entrée en mode veille* (*récepteur*)
*t*8	*entrée en mode réveil* (*récepteur*)
*t*9	1^{er}*extrémité du wormhole envoie un paquet RTS à la* $2^{ème}$*extrémité.*
*t*10	*la* $2^{ème}$ *extrémité du wormhole injecte le paquet RTS dans son voisinage*
*t*11	*Un voisin de la* $2^{ème}$ *extrémité du wormhole génère un paquet CTS*
*t*12	*la* $2^{ème}$ *extrémité du wormhole route le paquet CTS vers la* 1^{er}*extrémité*
*t*13	*la* 1^{er}*extrémité du wormhole injecte le CTS dans la zone A*(le 2^{nd} *CTS injecté dans le voisinage de A*)
*t*14	*Le* 1^{et} *CTS reçu par l'émetteur*
*t*15	*Le* 2^{e} *CTS reçu par l'émetteur* (*CTS routé depuis la zone B par un tunnel)*
*t*16	*L'émetteur s'arrête de fonctionner* (l'attaque wormhole est détectée)
*t*17	*Le nœud dans la zone B, attend infiniment le paquet DATA*
*t*18	*Le nœud dans la zone B, entre en mode veille*
*t*19	*Le nœud dans la zone B, entre en mode réveil*

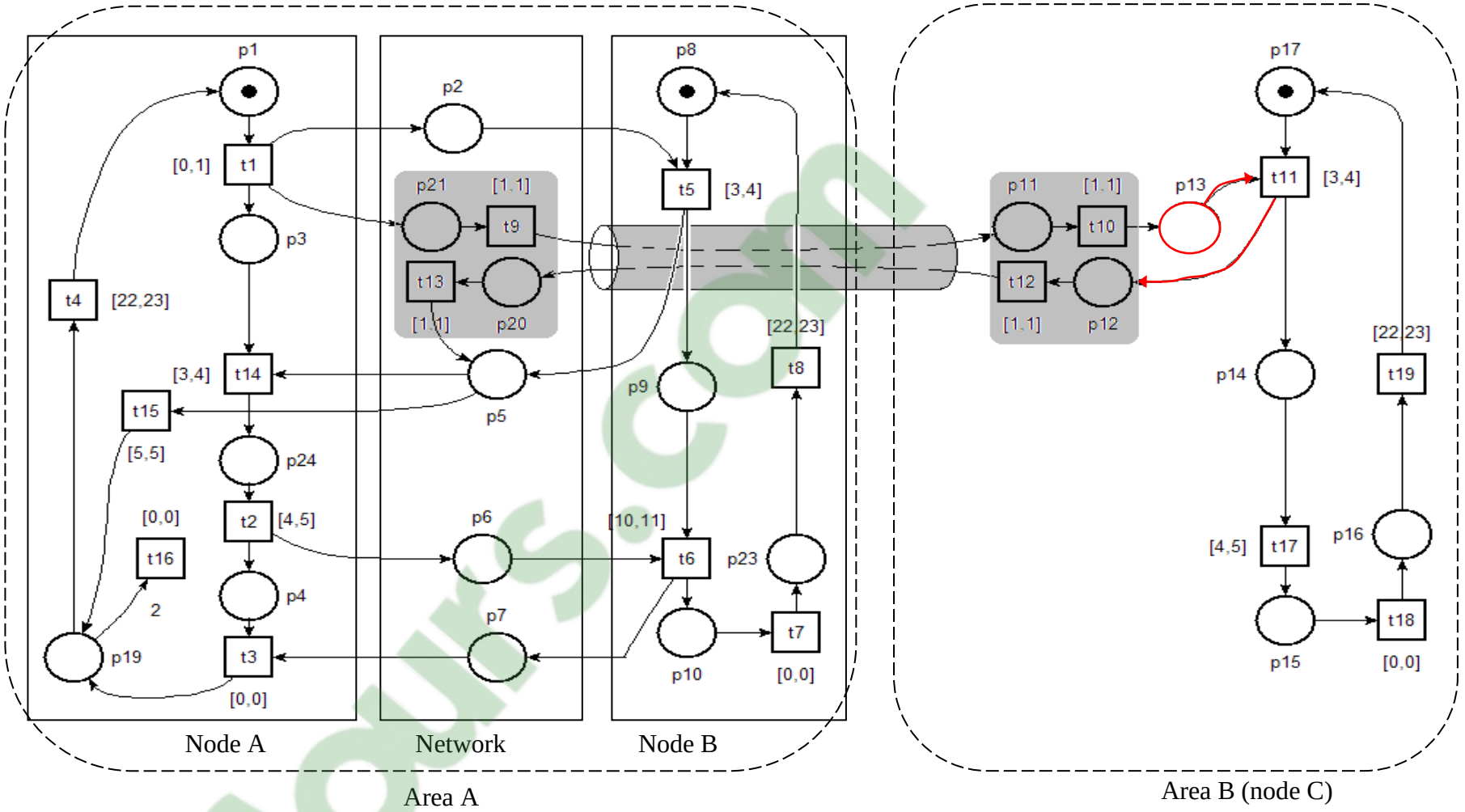


Figure 6-6: *RDP-T avec l'attaque wormhole (passive et active)*

La figure 6-6 illustre le modèle RDP-T de notre protocole SCL-MAC, chaque nœud est représenté comme suit:

Émetteur:représenté par les transitions *t*1, *t*2, *t*3, *t*4, *t*14, *t*15 et *t*16. Les transitions telles que *t*1 et *t*2 interprètent les actions d'envoi des paquets RTS et DATA respectivement à des moments bien précis. Ces dernières sont indiquées par les intervalles de temps associés aux transitions (un paquetRTS est envoyé après un DIFS et celui du DATA est envoyé après la reception du paquet CTS dans une fourchette temporelle entre 3 et 4 unités de temps). Ceci est modélisé par l'intervalle de temps associé à la transition *t*2. La transition *t*3 permet au nœud de basculer en mode veille après avoir reçu l'ACK alors que la transition *t*4 permet au jeton de rester dans la palce *p*19 (état endormi, sleep state), le temps nécessaire avant de débuter une nouvelle session de communication.

Les transitions *t*14 et *t*15 modélisent la variable *flag* associée au paquet RTS, *t*14 permet au nœud émetteur d'injecter son paquet DATA (flag = true) après avoir reçu le premier paquet CTS, celui en provenance de son réel voisin, le nœud*B* (voir la figure 6-6). La transition *t*15 image le fait d'initialiser la variable *flag* par la valeur *false* (instruction de la ligne 13 dans l'Algorithme) lorsqu'un autre paquet CTS est reçu (ce paquet est généré par le nœud intrus extrémité du wormhole). La dernière transition *t*16 représente le comportement du nœud "*A*" face à la détection de l'attaque wormhole (le nœud"*A*" diffuse un message d'alerte et éteint son transceiver).

La première extrémité du wormhole (le 1er nœud intrus), selon la figure 6-6 se trouve dans la zone A, est modélisée comme suit: les transitions *t*9 et *t*13. La transition *t*9 décrit le phénomène de tunnelement du paquet RTS depuis la zone A vers la zone B. La transition *t*13 décrit le même processus dans le sens inverse du paquet CTS depuis la zone B vers la zone A. la transition *t*13 injecte, dans le voisinage du nœud "*A*", le second paquet CTS en reponse au paquet RTS qui a été dérouté par l'attaque vers la zone B, et qui a permis à un nœud de se manifester et répondre à cette demande. La réception de ce paquet truqué par le nœud "*A*" force la variable *flag* associée au paquet RTS à *false* et de là signalant la présence d'une attaque wormhole.

Seconde extrémité du wormhole: sur la figure 6-6, ce nœud se trouve dans la zone B: il est modélisé par les transitions *t*10 et *t*12. *t*10 introduit dans la zone B une copie truquée du paquet RTS généré par le nœud "*A*". La transition *t*12 transmet le paquet CTS généré par un des nœuds de la zone B en destination du nœud "*A*" se trouvant dans la zone A.

La place *p*19 recevra le premier jeton lorsque le premier nœud intrus injecte le faux paquet CTS, en provenance de la zone lointaineB (2^{ième} CTS), dans le voisinage du nœud "*A*". Le deuxième jeton sera immédiatement introduit dans la place *p*19 après que le nœud "*A,*"aura reçu le paquet ACK. Cette place (*p*19) décritl'état endormi du nœud "*A*". Alors, avant que le temps de sommeil ne s'écoule, la transition *t*16 est tirée et la place *p*19 est vidée de son contenu (zéro jeton). Alors la transition *t*4 ne sera plus sensibilisée et ne pourra donc jamais être tirée. Cette situation décrit bien l'acte d'isolation du nœud "*A*" du reste du réseau lors des prochaines communications. Le nœud "*A*" sera surpassé par la création d'un nouveau chemin de routage (l'attaque de trou de ver n'aura aucun effet du fait que les prochaines communications ne passeront pas par la zone A). Le nœud "*A*" éteint son transceiver, dans le réseau RDP-T, le nœud "*A*"ne sera plus initialisé.

Pour distinguer entre les deux formes d'attaque du trou de ver, nous avons conçu deux modèles de RDP-T. L'attaque active est présentée par la figure 6-6, tandis que l'attaque passive est représentée aussi sur la même figure sans la portion rouge du graphe. Pour obtenir le modèle RDP-T de l'attaque passive, nous avons enlevé la *P*13, l'arrête *P*13→*t*11 et l'arrête *t*11→*P*12.

La détection de l'attaque se fait soit au moment des échanges des paquets RTS et CTS ou lors des échanges des paquets DATA et ACK (le principe est le même).

Figure (a)

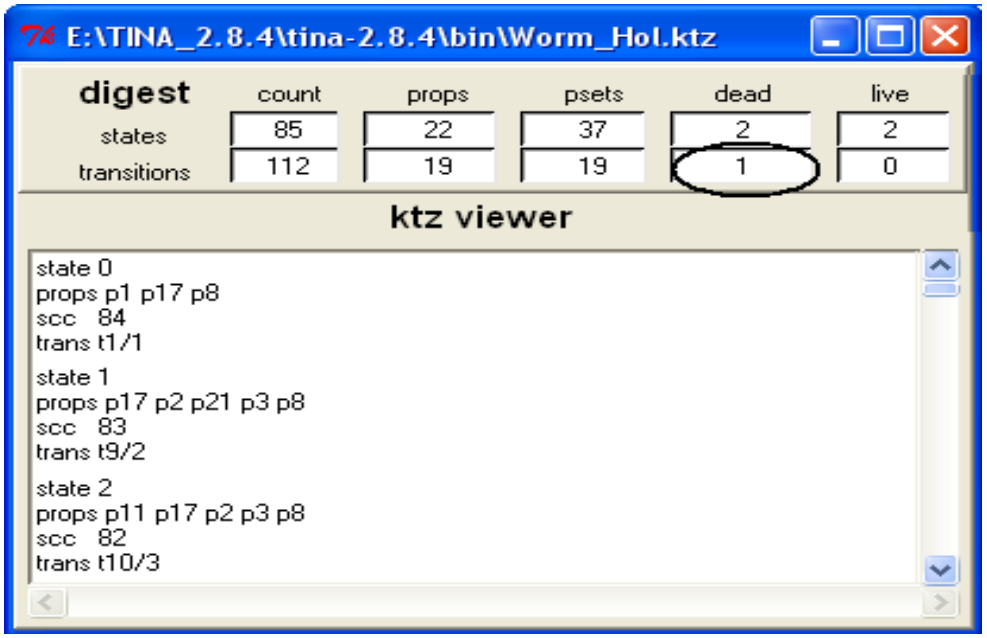
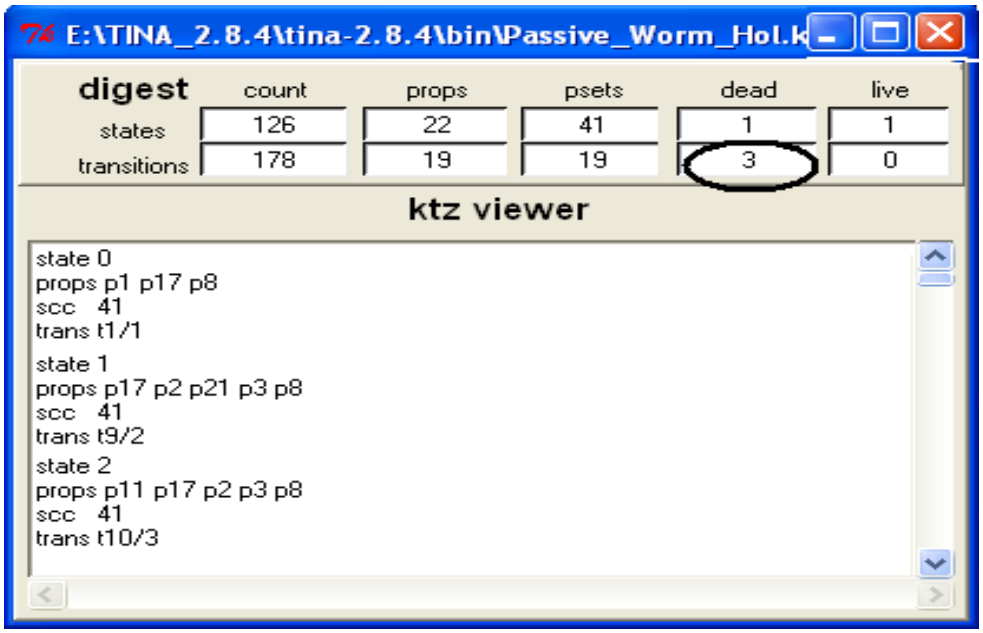


Figure (b)



Figure(c)

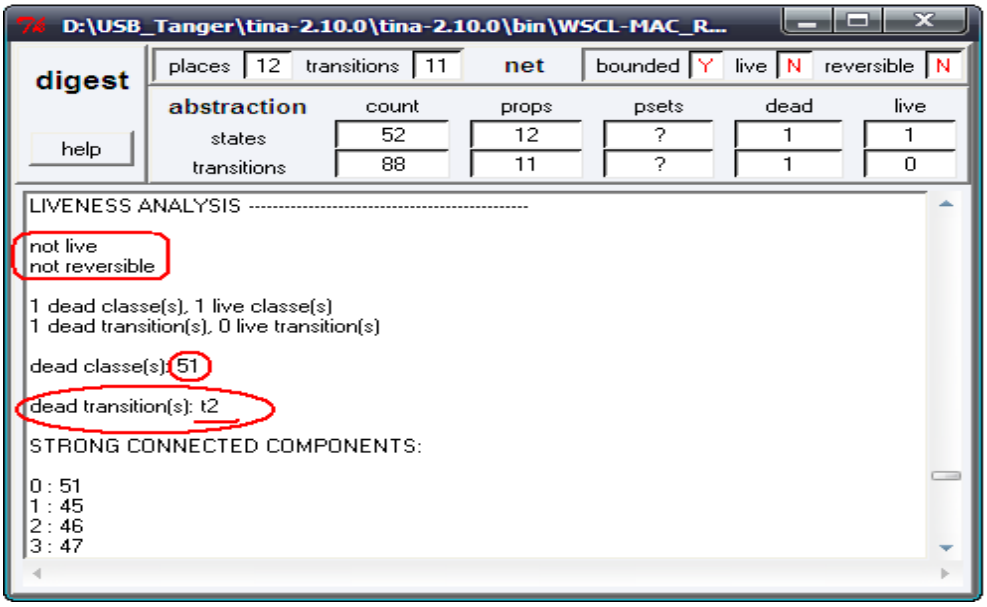


Figure 6-7: *Analyse des propriétés du RDP-T du protocole SCL-MAC*
(A) *forme active*
(B) *forme passive*
(C) *Analyse de l'accessibilité du RDP-T réduit*

6.3.3. Résultats d'analyse du RDP-T du SCL-MAC

L'étude comparative des résultats générés par l'outil TiNA est faite sur le même modèle (même réseau RDP-T) en présence (absence) del'attaque de trou de ver. Afin d'exécuter le modèle sans la présence de l'attaque de trou de ver, nous avons supprimé l'arc sortant de la transition *t*1 vers la place *p*21.

Avec l'outil TiNA, l'analyse d'accessibilité montre que la transition *t*4, dans le cas de la présence de l'attaque, est une transition morte empêchant ainsi la réinitialisation du nœud "*A*" (figure 6-7 (A)). Ce comportement reflète, dans le monde reel, l'isolation du nœud des futures communications. Les nœuds voisins du nœud "*A*" doivent éviter de router les paquets via le nœud "*A*" par le déclenchement de la procédure de recherché de nouveaux chemins vers le sink sans prendre en compte le nœud "*A*". La figure 6-7 (B) montre cette situation, dans le cas d'une attaque passive, les transitions *t*12, *t*13 et *t*16 sont des transitions mortes (non sensibilisées). Les transitions *t*12 et *t*13 indiquent que le nœud "*C*", se trouvant dans la zone B, ne doit pas réagir à la réception du paquet truqué RTS du nœud "*A*" injecté dans son voisinage par la seconde extrémité du trou de ver, et rentre en mode veille. Quand le nœud "*C*" (faisant partie d'un autre chemin passant par la zone B) entre en mode endormi, il brise ce chemin altérant, du coup, les communications sur ce lien (voir figure 6-6). La transition *t*16 est aussi morte selon les résultats des simulations faites par TiNA. Cela indique que le nœud "*A*" et ses voisins ne sont pas affectés par la présence de cette attaque.

6.4. Simulations

Dans cette section, nous validerons notre solution à l'aide d'une série de simulations avec le simulateur NS-2 sous l'environnement Ubuntu 11.2.

6.4.1. Paramètres réseau.

Un nœud mobile est formé des composants réseaux telles que la couche liaison (link layer ou LL) et l'interface file d'attente (interface Queue ou IFQ), le canal sans fil reçoit et transmet les signaux. Au départ, les types de chaque composante réseau doiventt être définis. Il faut également définir d'autres paramètres tels que le type d'antenne, le modèle de propagation radio, et le protocole MAC utilisé.

Nous avons effectué des simulations afin d'évaluer les performances de notre mécanisme de sécurité face à l'attaque trou de ver (*wormhole*). Nous avons pris un réseau de 20 nœuds déployé aléatoirement dans un champ de 800x800 m^2. La station de base (Sink) est identifiée dans la partie gauche du champ. Les nœuds sont identiques et ont la même portée radio de leurs transceivers. Nous avons supposé que la mobilité des nœuds est presque nulle. Le CBR est de 4 paquets/ seconde et la taille de chaque paquet est de 512 octets.

Le réseau est construit à partir de ses couches protocolaires telles que la couche physique, la couche liaison et la sous couche MAC. Ces composants sont définis avant de créer et configurer un nœud réseau. Dans notre simulation, nous avons défini le RCSF comme le montre le code suivant de la figure 6-8 :

```
# ======================================================================
# Define options
# ======================================================================

set val(chan)           Channel/WirelessChannel     ;# channel type
set val(prop)           Propagation/TwoRayGround   ;# radio-propagation model
set val(ant)            Antenna/OmniAntenna         ;# Antenna type
set val(ll)             LL                          ;# Link layer type
set val(ifq)            Queue/DropTail/PriQueue     ;# Interface queue type
set val(ifqlen)         50                          ;# max packet in ifq
set val(netif)          Phy/WirelessPhy             ;# network interface type
set val(mac)            Mac/SMAC                    ;# MAC type
set val(nn)             20                          ;# number of mobilenodes
set val(rp)             AODV                       ;# routing protocol
set val(x)              800                    ;X dimension of the topography
set val(y)              800                    ;Y dimension of the topography
set opt(energymodel)    EnergyModel                 ;
#set opt(energymodel)   RadioModel                  ;
set opt(radiomodel)     RadioModel                  ;
set opt(initialenergy) 1000                   ;# Initial energy in Joules


#Mac/SMAC set syncFlag_ 1
Mac/SMAC set dutyCycle_ 10

# ======================================================================
# ----------------------------------------------------------------------
```

Figure 6-8: *Paramètres réseau implémentés sur le simulateur NS-2*

Nous avons créé 20 nœuds formant notre réseau. La classe nœud (node), dans notre code NS-2 a été définie selon la figure6-9. Un nœud se voit former de: protocole de routage, couche liaison, couche MAC, couche physique et IFQ (interface queue type).

```
#=========================================================
# CONFIGURE AND CREATE NODES
# =========================================================

$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)      \
                 #-channelType $val(chan)  \
                 -topoInstance $topo       \
                 -agentTrace ON            \
                 -routerTrace ON           \
                 -macTrace ON              \
                 -movementTrace OFF        \
                  -channel $chan_1         \
                 -channel $chan_2          \
                 -channel $chan_3          \
                 -channel $chan_4          \
                 -energyModel $opt(energymodel) \
                 -idlePower 1.0            \
              -rxPower 1.0              \
              -txPower 1.0              \
       -sleepPower 0.001         \
       -transitionPower 0.2      \
       -transitionTime 0.005     \
              -initialEnergy $opt(initialenergy)

$ns set WirelessNewTrace_ ON
```

Figure 6-9: *Configuration et instantiation d'un objet noeud*

6.4.2. Simulation de l'attaque wormhole

Dans notre code de simulation NS-2, nous avons introduit deux nœuds malveillants représentant l'attaque trou de ver (wormhole). Un tunnel entre ses deux nœuds a été établi en utilisant l'en capsulation.

L'implémentation et la simulation de l'attaque trou de ver ont été réalisées en introduisant des modifications sur le protocole MAC de base, à savoir S-MAC. Exactement, au niveau des procédures **HandleRTS** (receive) et celle de **HandleCTS,** et cela juste afin de tester notre mécanisme.

Lorsqu'un nœud "*A*" extrémité d'un lien de trou de ver reçoit un paquet RTS, il le transmet immédiatement vers l'autre extrémité "*B*" distante de ce lien. Une fois ce dernier reçoit le paquet RTS, il l'injecte dans son voisinage. Un des voisins du nœud malveillant "*B*" reçoit le faux paquet RTS, il génère un paquet CTS en réponse adressée à la source réelle génératrice du paquet RTS et l'injecte dans le réseau. Le processus de parcours est maintenant en sens inverse. Le nœud "*B*" reçoit le paquet CTS et le lien trou de ver le régénère dans la zone du nœud "*A*" 1^{ere} extrémité du trou de ver. Le nœud initial émetteur du paquet RTS, reçoit une réponse d'un nœud distant voisin de "*B*". La valeur du RTT est trop élevée comparée à un RTT dans un mode safe. Voir la figure6-10 suivante.

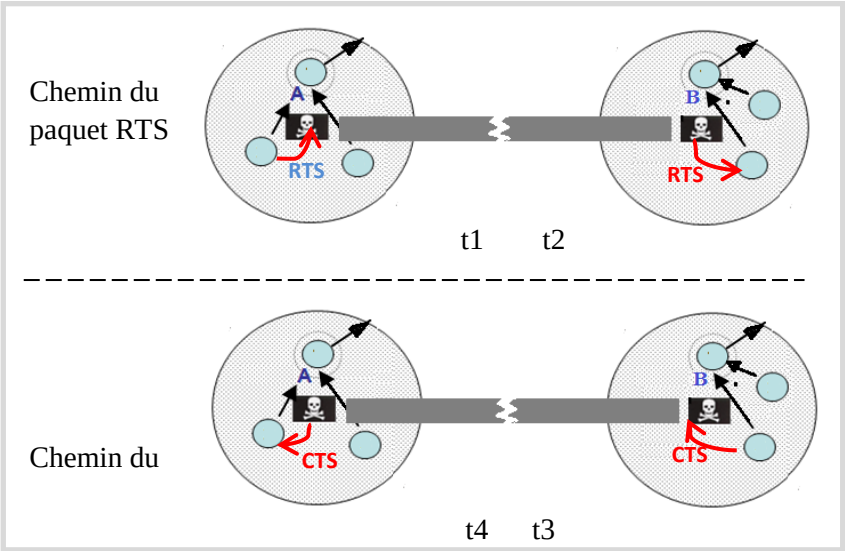


Figure 6-10: *Schéma du scenario de l'attaque active du wormhole*

6.4.3. Simulation du mécanisme de détection proposé.

Le mécanisme de détection des attaques wormhole est basé sur la technique du Round Tripe Time. L'échange des paquets RTS et CTS entre deux nœuds voisins est temporisé par des variables temporelles afin de calculer la valeur du RTT. Nous avons implémenté cette technique au niveau de la couche MAC du protocole CL-MAC (ce dernier est basé sur les protocoles S-MAC et Cross-MAC). Nous avons donc pris le protocole S-MAC existant sous NS-2, et effectué les adaptations nécessaires pour présenter à la fois le protocole de base CL-MAC et les nœuds malveillants modélisant l'attaque wormhole. Les changements opérés résident au niveau des fonctions **sendRTS ( )** et **handleCTS ( )** du S-MAC de NS-2.

6.4.4. Résultats expérimentaux

Les résultats de simulation sont portés sur réseau de taille 20. Dans NS-2, le package NAM allinone est un a build-in programme. NAM permet de nous tracer le parcours de paquets communiqués dans le réseau. Il permet aussi de voir si les paquets ont attient leurs propre destination ou ils sont perdu. Lorsque le fichier TCL est généré, le programme NAM est invoqué dans ce fichier.

L'implémentation utilise le trafic CBR sur UDP. Nous avons simulé une session CBR dans chaque expérience. Chaque fois, 500 paquets de données, de 512 octets chacun, sont générés à un taux de 4 paquets par seconde.

Dans notre simulation, nous avons créé deux scenarios: le premier en l'absence d'une attaque et l'autre dans la présence d'une attaque trou de ver. Le tour est joué en activant ou désactivant les nœuds malicieux. Trois expérimentations sont faites afin de vérifier l'efficacité de notre solution proposée pour la sécurisation du CL-MAC face à ce type d'attaque.

Dans la première expérimentation, nous avons testé le protocole CL-MAC dans un environnement saint, par inhibition des deux nœuds malicieux, modèle de l'attaque trou de ver. Une légère modification a été apportée sur l'algorithme du CL-MAC pour permettre le calcul de la valeur RTT entre deux nœuds successifs. La figure 6-11 illustre une capture écran d'une animation d'un réseau de 20 nœuds. Le nœud "Node1" est la source initiatrice de

la communication générant des paquets DATA vers le nœud sink destination “Node2”. Dans cette expérience, la source et la destination sont voisines. Les données arrivent à destination sans passer par des nœuds intermédiaires (routeurs). Après plusieurs exécutions, les valeurs du RTT correspondantes étaient presque les mêmes comme le montre la figure 6-12.

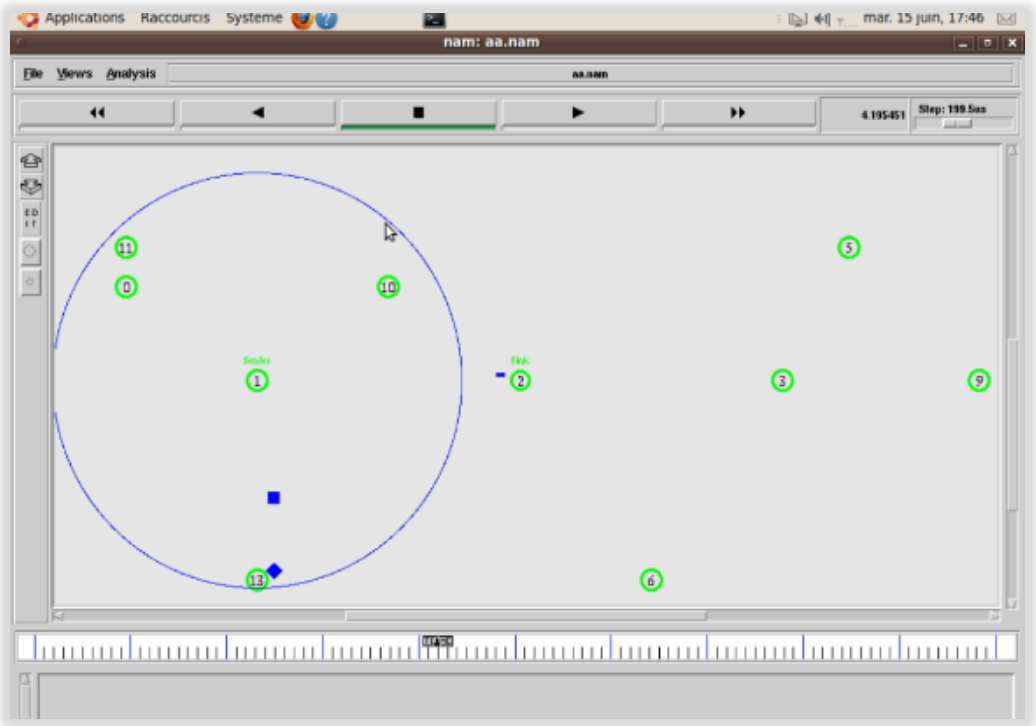


Figure 6-11: *Node1 envoie des paquets DATA vers Node2.*

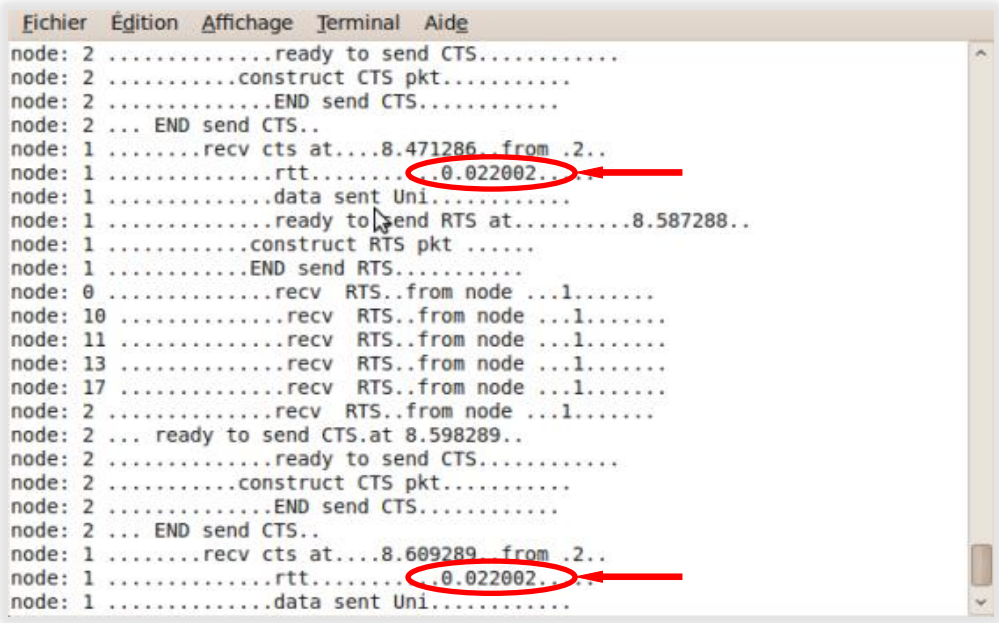


Figure 6-12 : *Valeur du RTT invariante*

Dans la deuxième, a été testé le protocole CL-MAC, en l’absence de l’attaque trou de ver. Seulement, les deux nœuds (Node1, Node2), source et destination (*sink*), sont déployés l’un loin de l’autre de telle sorte que les paquets DATA arrivent à destination après avoir transité

par des nœuds routeurs. Dans cette expérience, *Node*1 est la source tandis que *Node2* est le sink (destination), alors que *Node*9 et *Node3* sont des nœuds routeurs. La même observation a été faite, après une série d'exécutions de ce scenario. Les valeurs successives des RTT, entre chaque paire de nœuds et correspondant à chaque lecture, étaient presque identiques. Dans la figure 6-13, le nœud 9 achemine le paquet DATA vers le nœud 3 après l'avoir reçu du nœud 1. Dans la figure 6-14, c'est le nœud 3 qui route le paquet DATA vers le sink (noeud2) juste après sa réception à partir du nœud 9.

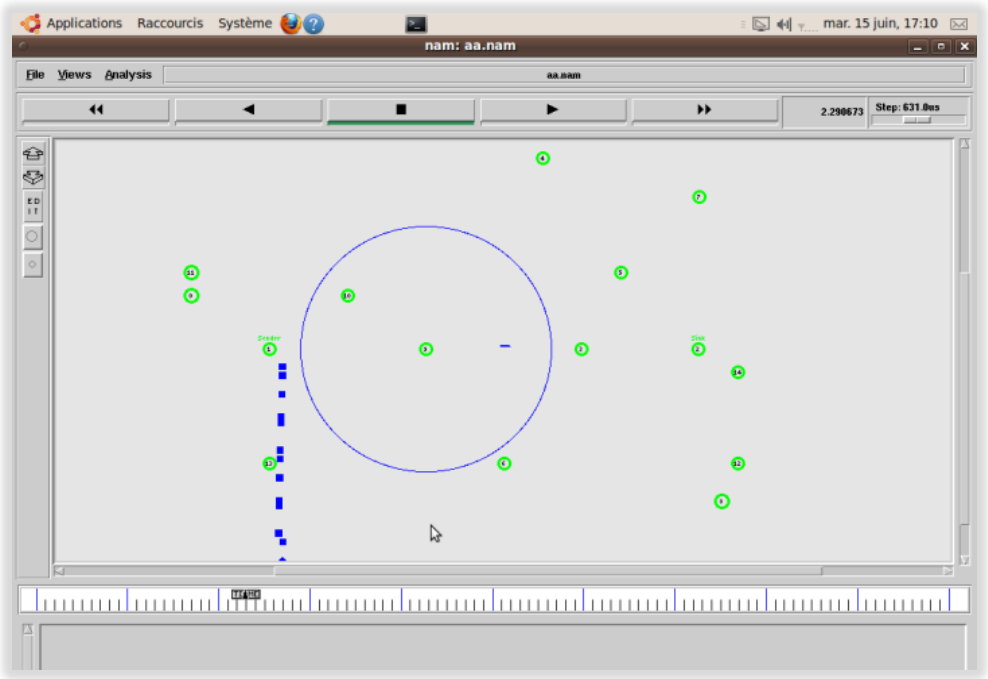


Figure 6-13: *Nœud 9 achemine le paquet DATA vers le nœud 3 (son voisin au prochain saut).*

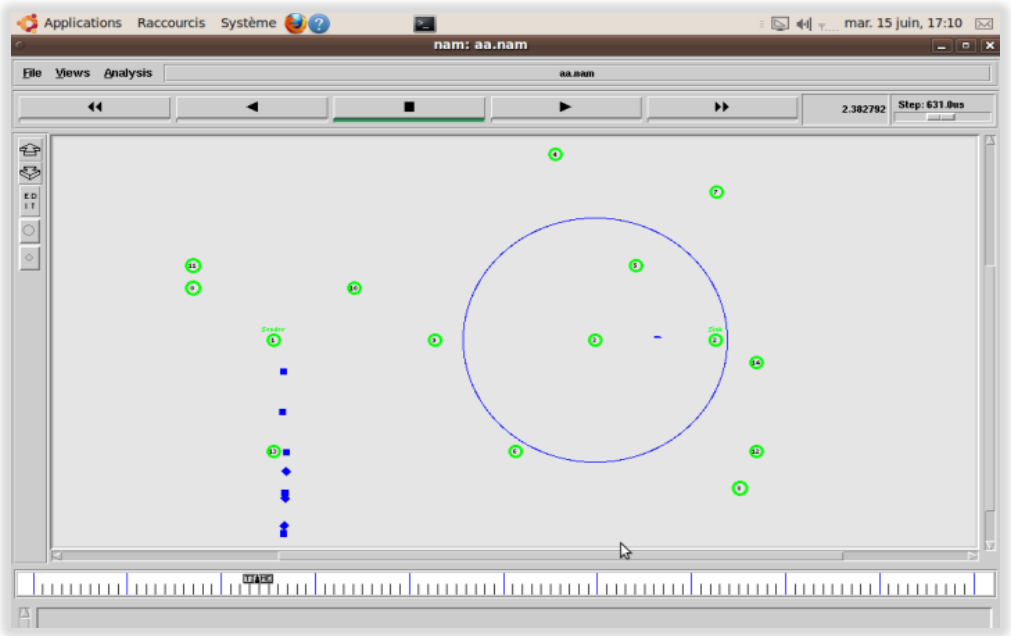


Figure 6-14: *Nœud 3 achemine le paquet DATA vers le sink (Noeud2).*

Dans la troisième expérimentation, nous avons activé le comportement malicieux des nœuds 9 et 3, formant ainsi un tunnel de type trou de ver. Le nœud 9 absorbe les paquets dans son voisinage, les transmet via un tunnel au nœud 3 modélisant par ce comportement l'attaque trou de ver. Le nœud 3 diffuse les paquets reçus dans son voisinage. Le calcul des valeurs de RTT, entre la source et la destination, était sensiblement plus élevé que celui des première et deuxième expérimentations. Les figures 6-15, 6-16 et 6-17 montrent le processus de tunneling entre les nœuds malicieux et les nœuds légitimes.

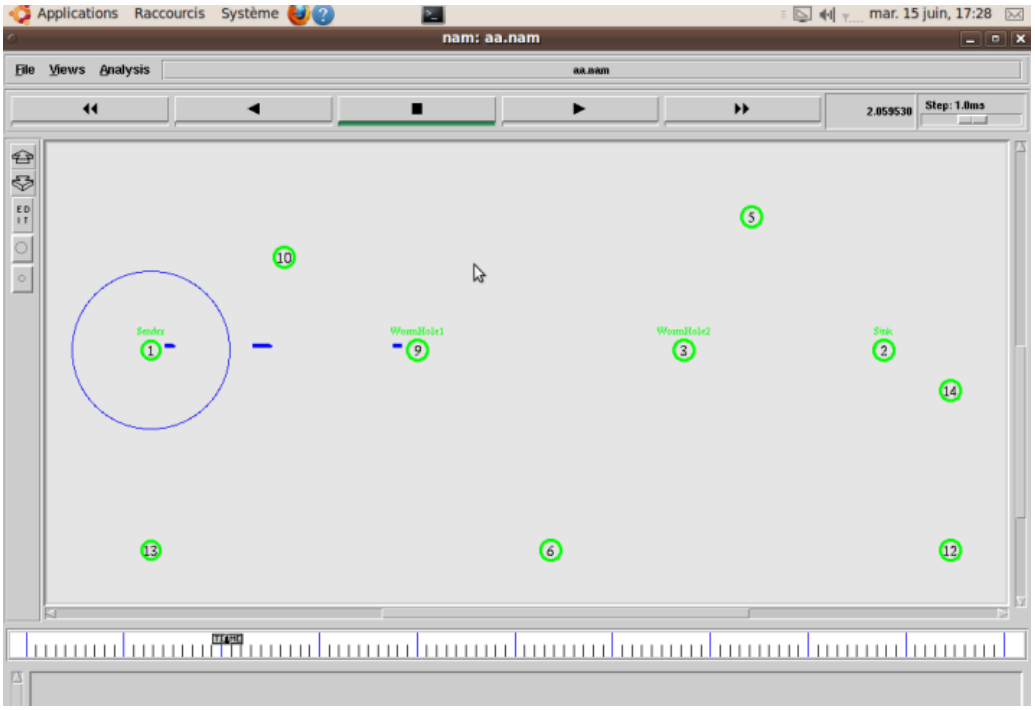


Figure 6-15: *Nœud 9 reçoit des paquets en provenance du nœud 1 (la source)*

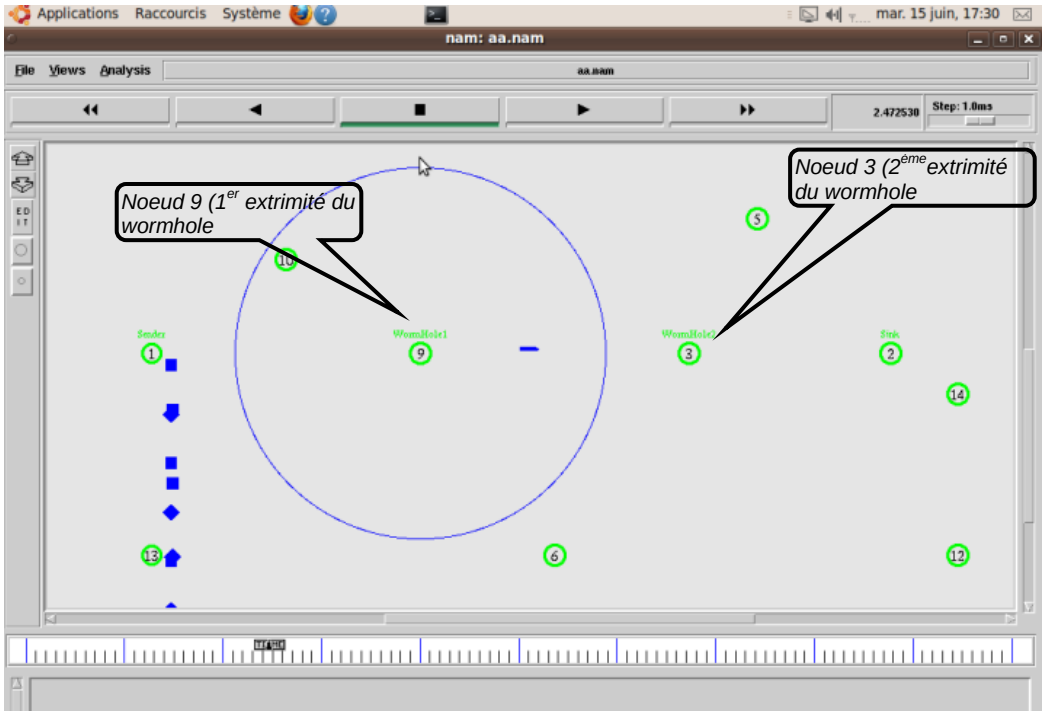


Figure 6-16: *Nœud transmet les paquets via un tunnel vers le nœud 3*

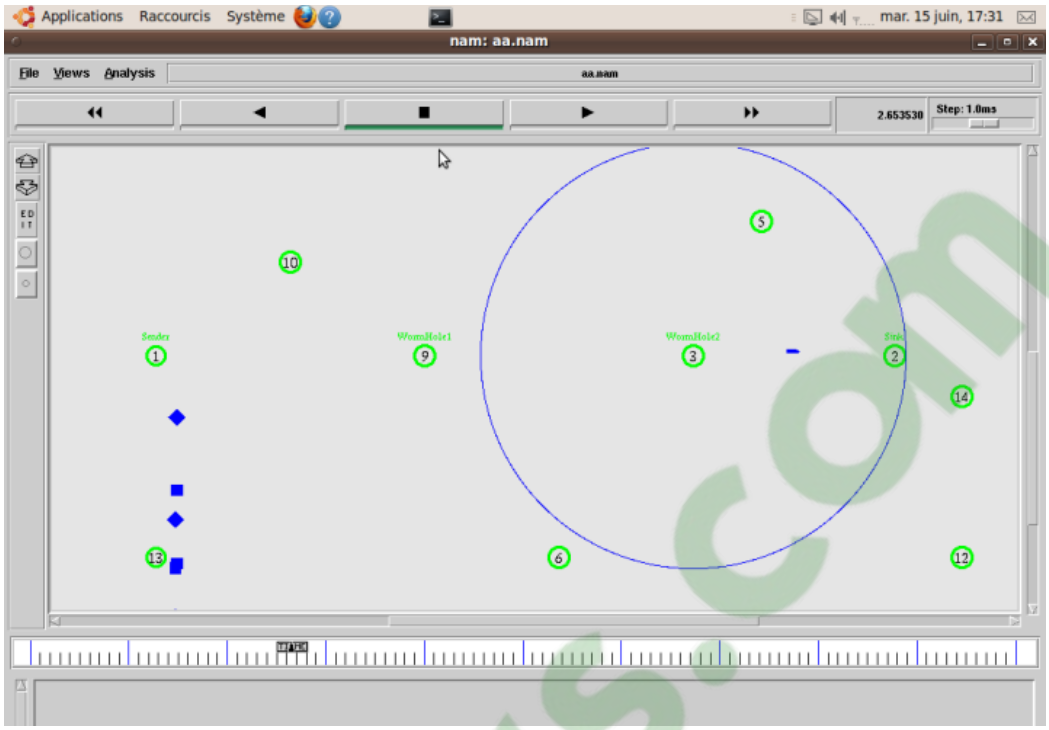


Figure 6-17: *Nœud 3(2ième extrémité du wormhole) injecte les paquets dans son voisinage (ici le paquet a été reçu par la destination)*

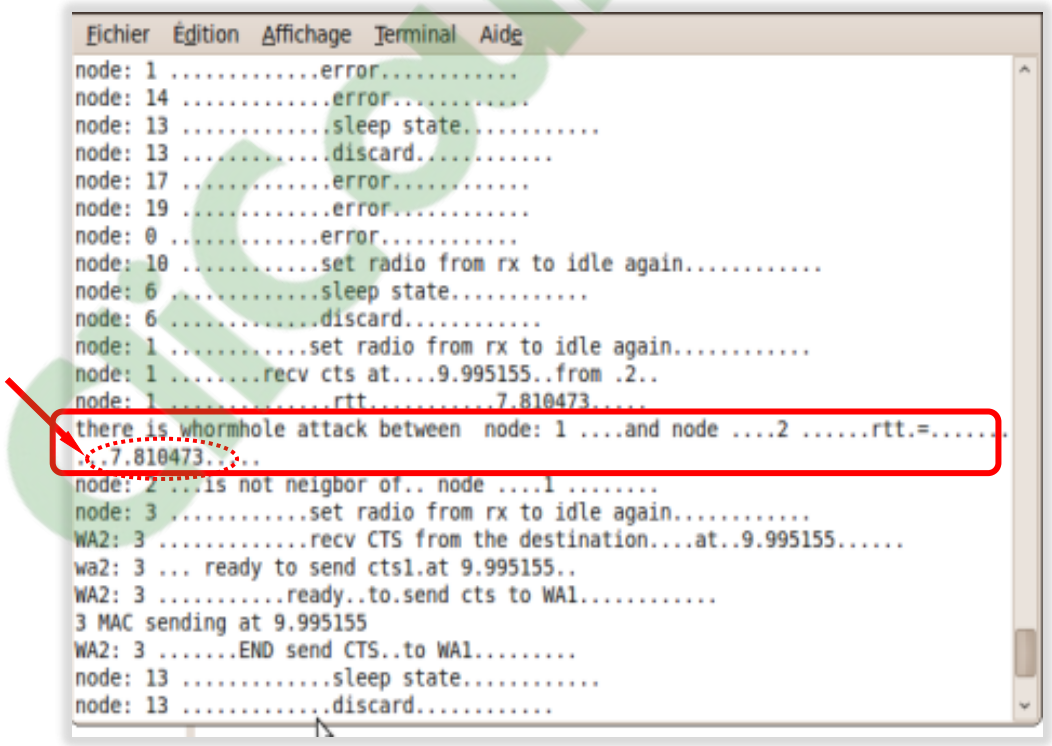


Figure 6-18: *Détection de l'attaque Wormhole*

Sur la figure 6-18, nous remarquons que la valeur du RTT entre les nœuds 1 et 2 a dépassé le seuil autorisé (7,810473 ms). Le protocole a declenché une alerte d'attaque par la diffusion du message AM, d'où la mention « *There is a wormhole attack between node 1 and node 2* » sur le fichier texte lors de l'exécution.

6.5. Conclusion.

Dans ce chapitre, nous avons présenté un protocole SCL-MAC, la version sécurisée du CL-MAC, afin de pallier ses vulnérabilités face à l'attaque wormhole. La solution proposée se base sur la technique RTT dont la valeur est calculée, à la fois au niveau de la couche réseau lors de la construction des chemins de routage et egalement durant la phase de découverte du voisinage au niveau de la couche MAC au moment où CL-MAC initialise les listes des voisins ou procède à leur mise à jour périodique.

Nous avons présenté un modèle de réseaux de Petri Temporels (*RDP-T*) afin de valider d'une manière formelle certaines propriétés comportementales de notre protocole SCL-MAC face à l'attaque trou de ver. Les résultats d'évaluation, du modèle proposé par l'outil TiNA, ont été fructueux. Lors de la détection d'une attaque de type trou de ver, le nœud proche de l'intrus bloque le chemin dont il fait partie et passe en mode hibernation. Une fois la place *p19* reçoit deux jetons (indication de la détection d'une attaque), la transition *t4*, modèle de ce fait, n'est plus franchissable.

En plus des résultats formels obtenus, l'algorithme du protocole SCL-MAC a été implémenté sous NS-2. Nous avons modélisé une seule attaque de type trou de ver (il existe dans la littérature le cas de multiples attaques). Trois scénarios ont été étudiés: le cas où les nœuds source et destination sont des voisins. Dans ce cas de figure, l'attaque n'aura aucun effet. Le second cas, lorsque ces nœuds sont éloignés et que le réseau opére dans un environnement saint. Nous avons mis en oeuvre le scénario plusieurs fois et nous avons calculé les valeurs RTT entre la source et la destination ; elles étaient trés proches. En troisième position, dans le scenario 2, nous avons intégré deux nœuds malicieux, à mi-chemin, entre la source et la destination. Suite à l'extraction des valeurs du RTT, comparés aux précédents, les résultats obtenus montrent belle et bien l'efficacité du mécanisme de détection de cette attaque.

De plus, nous pouvons affirmer que l'adaptation de la technique du RTT pour sécuriser le protocole CL-MAC face à l'attaque de trou de ver est très efficace pour la détection et l'évitement de ce genre d'attaque.De même, cette technique s'avère moins coûteuse en terme d'énergie ce qui est l'essence du protocole CL-MAC.

7. Conclusion générale et futures perspectives.

La fin du dernier siécle a été marquée par les progrès technologiques en matière de miniaturisation et nano-technologie des outils, des machines, et des moyens de communication. Ces moyens ont permis d'atteindre ce qui constituait un rève quelques années auparavant. Les RCSFs sont considérés comme une avancée dans ces domaines. Cependant, les RCSFs soulèvent d'importantes problématiques de recherche en termes d'exploitation des données récoltées, de localisation, de consommation d'énergie, d' l'auto-configurabilité, de compression des données et de sécurité des données échangées. Nos travaux présentés dans cette thèse s'inscrivent dans la trajectoire de la sécurité et de l'ergonomie énergétique.

Dans ce travail, nous nous sommes surtout intéressés au problème de la sécurité des RCSFs. La couche MAC a été notre champ d'investigation. Nous avons choisi le protocole MAC à couches croisées'CL-MAC' afin d'étudier sa vulnérabilité en matière de sécurité et, par la suite, tenter d'améliorer son comportement face aux attaques les plus dangeureuses. Deux types d'attaques ont fait l'objet de nos corpus, à savoir Sinkhole et Wormhole. Ces deux attaques opèrent et causent un dysfonctionnement du RCSF sans même avoir en possession la clé de la cryptographie, ce qui les rendent inévitablement plus devastatrices.

Ce manuscit a débuté par donner les notions de base des réseaux sans fil, réseaux Ad-hoc, et puis les réseaux de capteurs sans fil. L'accent a été mis sur les contraintes physiques des nœuds capteurs en matières de : (i) calcul qui doit être allégé afin d'être supporté par une unité CPU, réduite en espace mémoire, pour le stockage des données et le nano-processeur implanté sur ces nœuds miniaturisés. (ii) la source d'énergie limitée impose des contraintes sur la conception des protocoles de communication. Nous avons aussi abordé la notion de sécurité, classer les attaques sur les RCSFs et traité avec beaucoup de soins la technique de contre mesure basée sur le RTT. Cette contre mesure de détection du dysfonctionnement du protocole face aux attaques, nous a paru très efficace dans le sens où elle convient mieux aux contraintes physiques des nœuds capteurs dans la mesure où elle ne génére pas de trafic suplèmentaire.

La seconde partie de la thèse était consacrée aux travaux de recherche que nous avons menés dans le domaine de la sécurité des RCSFs et qui ont fait l'objet de plusieurs contributions. Pour l'attaque Sinkhole, les simulations ont été réalisées en utlisant les plateformes Castalia et Omnet++. Appuyés par des résultats formels, utilisant les réseaux de Petri temporels modélisant la solution proposée, les résultats de simulation ont prouvé l'efficacité de la solution adoptée pour notre nouveau protocole SCL-MAC.

Le second axe de recherche visait la sécurisation du protocole CL-MAC face à l'attaque Wormhole. La solution proposée a été testée sous le NS-2 permettant à un nœud de détecter un dysfonctionnement causé par un nœud malicieux présent dans le voisinage. Cette solution a été modèlisée par un réseau de Petri temporel. Les résultats obtenus en utilisant l'outil TiNA et ceux eus par le simulateur NS-2 étaient complèmentaires.

Le domaine des RCSFs constitue un axe de recherche très fertile depuis plus d'une décennie. Dans cette optique, à ce jour, de nombreux problèmes sont restés en suspend. La conception d'un RCSF est une tâche difficile et chaque domaine d'application exige ses propres cractéristiques et ses propres exigences. La conception devra prendre en considération des contraintes propres aux systèmes distribués et aux systèmes embarqués. Comme extension à nos travaux nous proposons :

- L'étude de la sécurité des RCSFs mobiles avec une forte mobilité; jusqu'à présent, la quasi-totalité des recherches se sont penchées sur l'étude des mécanismes de sécurité pour des réseaux statiques où les nœuds étaient supposés immobiles, ce qui n'est pas le cas en général. Dans un réseau mobile, un nouveau nœud rejoignant le réseau, est-il considéré comme un nœud à confiance ou comme étant un nœud malicieux ? Avec quel degré de confiance un nœud nouveau est-il accepté dans son voisinage?

- L'étude de réseaux de grande taille (le passage à l'échelle) ; pratiquement toutes les solutions de sécurité proposées ont été testées sur des réseaux de faible taille, contenant quelques centaines de nœuds capteurs. Cela est sûrement dû au surcoût induit par les primitives de sécurité.

- La prise en compte d'attaques multiples.

- L'étude des solutions dites '*bio-inspired*, comme s'inspirer des colonies de fourmis ou d'abeilles afin de doter les réseaux de capteurs de plus d'intelligence dans la gestion de la collecte d'informations. L'avenir est sûrement dans ce qu'on peut appeler SMART SENSORS [32].

8. Références bibliographiques

[1] B. Kechar, L. Sekhri, M. K. Rahmouni, « CL-MAC: energy efficient and low latency cross-layer MAC protocol for delay sensitive wireless sensor network applications », The Mediterranean Journal of Computers and Networks, Vol.6, N°1, pp. 1-14, January 2010.

[2] B. Kechar, **A. Louazani**, L. Sekhri, M. F. Khelfi, "Energy Efficient Cross-Layer MAC Protocol for Wireless Sensor Networks", 2nd International Workshop on Verification and Evaluation of Computer and Communication Systems (VECOS'08), Leeds, UK, July 2-3 2008, Publié dans The eWiC Series (ISSN : 1477-9358).

[3] **A. Louazani**, B. Kechar, M. F. Khelfi, "Wireless Sensor Networks: Basic notions, energy consumption and MAC protocol layer", DCCA 2007, Jordanie, pp. 1236-1246, 2007.

[4] B. Kechar, **A. Louazani**, "CL-MAC: An Energy Efficient Cross-Layer MAC protocol with latency improvement for wireless sensor networks", 17th International Conference on Computer Communications and Networks (ICCCN'08), St Thomas U.S. Virgin Islands, August 2008.

[5] B. Kechar, L. Sekhri, " Formal Modelling and Validation of a Novel Energy Efficiet Cross-Layer MAC Protocol in Wireless Multi hop Sensor Networks Using Time Petri Nets", New Technologies, Mobility and Security (NTMS'08) – Second IFIP International Conference on New Technologies,Mobility and Security, Tangier, Morocco, pp. 1-5, November 2008.

[6] B.Kechar, 'Problématique de la consommation d'énergie dans les réseaux de capteurs sans fil', Thèse de doctorat d'état, Juin 2010, Université d'Oran1, Algéria.

[7] S. Khan,A. P. Khan,N. A. Alraje, « Wireless sensor Networks : Current status and future Trends », CRC Press 2012, ISBN 978-1-4665-0606-0, Chapter 8, pp.189

[8] T. Halonen, J. Romero, J. Melero, "GSM, GPRS and EDGE Performance", John Wiley & Sons Ltd, ISBN: 0-470-86694-2, 2003.

[9] M. Ilyas, "The Handbook of Ad hoc Wireless Networks", CRC Press, ISBN: 0-8493-1332-5, 2003.

[10] Y. D. Lin, Y. C. Hsu, "Multihop cellular: A new architecture for wireless communications", Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings of IEEE INFOCOM 2000, Vol. 3, pp. 1273-1282, 2000.

[11] J. Zhang, I. Stojmenovic, "Cellular Networks", Book chapter, Handbook of Wireless Networks and Mobile Computing, ISBN: 0-471-41902-8, John Wiley & Sons, pp.654-662, 2005.

[12] H. Liu, X. Jia, P. Wan, "On energy efficiency in wireless ad hoc networks", Book chapter on: Ad hoc and Sensor Networks, ISBN: 1-59454-396-8, Nova Sciences Publishers, Inc, pp.31-54, 2006.

[13] R. Poovendran, L. Lazos**, "**A graph theoretic framework for preventing the wormhole attack in wirelessad hoc networks", Wireless Networks (Kluwer Academic Publishers), Vol. 13, No. 27, pp. 27-59, 2006.

[14] M. Heissenbuttel, "Routing and Broadcasting in Ad-hoc Networks", PHD thesis, University of Bern, Juin 2005.

[15] IEEE 802.15.4, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), Standard, IEEE, Dec. 2003.

[16] Feliz Kristianto Karnadi, Zhi Hai Mo et Kun-chan Lan, "Rapid Generation of Realistic Mobility Models for VANET", Wireless Communications and Networking Conference, WCNC 2007, IEEE, pp. 2506-2511. 2007

[17] L. Campelli, M. Cesana, R. Fracchia, "Evaluation of Integrated Routing/MAC Solutions for the Diffusion of Warning Messages in VANETs", Journal of Networks (Academy Publisher), Vol. 2, No. 6, pp. 13-23, December 2007.

[18] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. «System Architecture Directions for Networked Sensors», In ACM Sigplan Notices, volume 35, pages 93–104, 2000.

[19] L. Vazquez, « Implementation and simulation of routing protocols for wireless sensor networks » Master thesis, University of Siegen, Mars 2006.

[20] A. Willig, "Wireless sensor networks: concept, challenges and approaches", Springer-Verlag, pp.224-231, 2006.

[21] K. Sakamura, "T-Engine, the Open Platform for the Ubiquitous Computing Age", Asian Solid-StateCircuits Conference, pp. 3-6, November 2005.

[22] K. M. Hou, "LiveNode: LIMOS versatile embedded wireless sensor node", WWSN 2007, NOTERE2007, Marrakech, Maroc, Juin 2007.

[23] M. A. M. Vieira, A. B. da Cunha, D. C. da Silva Jr."Designing Wireless Sensor Nodes", Bookchapter : Embedded Computer Systems: Architectures, Modeling, and Simulation, LNCS (Springer),Volume 4017/2006, pp. 99-108, 2006.

[24] H. Karl and W. Andreas, "Protocols and architectures for wireless sensor networks", John Wiley and Sons, 2005.

[25] A. Sobeih, W. Chen, J. C. Hou, L. Kung, N. Li, H. Lim, H. Tyan, H. Zhangt, "J-Sim: A Simulation andEmulation Environment for Wireless Sensor Networks", Wireless Communications, IEEE, Vol. 13, No.4, pp. 104-119, 2006.

[26] J. Yick, B. Mukherjee, D. Ghosal, "Wireless sensor network survey", Computer Networks, (Elsevier), Vol. 52, No. 12, pp. 2292-2330, 2008.

[27] C. Chong, S. P. Kumar, "Sensor Networks: Evolution, Opportunities, and Challenges", Proceedings of the IEEE, Vol. 91, No. 8, pp. 1247-1256, August 2003.

[28] D. Culler, D. Estrin, M. Srivastava, "Overview of Sensor Networks", Published by the IEEE Computer Society, Vol. 37, No.8, pp. 41-49, August 2004.

[29] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "Wireless sensor networks: A Survey",Computer Networks, Vol. 38, No.2, pp. 393-422, 2002.

[30] S. Tilak, N. B. Abu-Ghazaleh, W. Heinzelman, "A Taxonomy of Wireless Micro-Sensor NetworkModels", ACM SIGMOBILE Mobile Computing and Communications Review,Vol. 6 , No. 2, pp. 28-36, 2002.

[31] J. Kulik, W. R. Heinzelman, H. Balakrishnan, «Negotiation base protocols for Disseminating Information in Wireless Sensor Networks», Wireless Networks, vol. 8, pp.169-185, 2002.

[32] N. Labraoui, "La Sécurité dans les Réseaux sans fil AdHoc: Agrégation de données et Localisation ", Thèse de doctorat, Juin 2012, Université de Tlemcen.

[33] F. Z. Ouraghi, "Sinkhole attack detection in wireless sensor networks", mémoire de Master, juin 2012, Université de Chlef.

[34] S.Cheung Coleri,P. Varaiya, "Sensor networks for monitoring traffic", 42nd Allerton Conference on Communication, Control and Computing, 2004.

[35] L. Konrad et al., « Sensor networks for emergency response: Challenges and opportunities », IEEE Pervasive Computing, 2004, Vol. 3(4), pp.16-23.

[36] S. A. H. Sedjelmaci, "mise en œuvre de mécanismes de sécurité bases sur les IDS pour les réseaux de capteurs sans fil", Thèse de doctorat, université de Tlemcen, 2012.

[37] W. R. Heinzelman, A. Chandrakasan, H. Balakrishnan, "Energy efficient communication protocol for wireless microsensor networks", Proceeding of the 33rd Hawaii International Conference on System Sciences, IEEE, pp.1-10, 2000.

[38] O. Younis, S. Fahmy, "Heed: A hybrid energy-efficient distributed clustering approach for ad hoc sensor networks", IEEE Transactions on Mobile Computing, 3(4): 366-379, 2004.

[39] S. Lindsey, and C. Raghavendra, "PEGASIS: Power efficient gathering in sensor information system", In Proc.IEEE Aerospace Conference, vol.3, pp.1125-1130, 2002.

[40] A. Manjeshwar, D.P. Agarwal, TEEN: a routing protocol for enhanced efficiency in wireless sensor networks, 15th International Proceedings on Parallel and Distributed Processing, IEEE, pp. 2009-2015, 2000.

[41] L. Khelladi, N. Badache, « Improving directed diffusion with power-aware topology control for adaptation to high density », LOCALGOS'08 workshop, in conjunction with The 4th IEEE/ACM International Conference on Distributed Computing In Sensor System, Greece, 2008.

[42] K. Sohrabi, J. Gao, V. Ailawadhi, G.J. Pottie, « Protocols for self-organization of a wireless sensor network », IEEE Journal of Personal Communications, 7(5):16-27, 2000.

[43] A. Perrig, R. Szewczyk, J.D. Tygar, V.E. Wen, D. Culler, « SPINS: security protocols for sensor networks ». Wireless Networks Journal,8(5):521-534, September 2002.

[44] K. Sharma, M.K. Ghose, "Wireless Sensor Networks Security: A New Approach", In Proceedings of 16th International Conference on Advanced Computing and Communication (ADCOM 2008). Dec. 14-17, 2008, MIT Chennai.

[45] M. Meghdadi, S. Ozdemir, I. Güler, "A Survey of Wormhole-based Attacks and their Countermeasures in Wireless Sensor Networks"; The Institution of Electronics and Telecommunication Engineers (IETE), vol. 28, Issues. 2. pp 89-102, 2011.

[46] M. Yasir, "An Outline of Security in Wireless Sensor Networks Threats, Countermeasures and Implementations". Wireless Sensor Networks and Energy Efficiency: Protocols, Routing and Management Book, pp. 507-527, 2012.

[47] L. Hong, F. Hong, C. Fu, "Defending Against Wormhole Attack in OLSR". Geo-spatial Information Science (Quarterly), vol. 9, Issue. 3, pp. 229-233, Sep. 2012.

[48] L. Kyuho, J. Hyojin, K. Dongkyoo, "Wormhole Detection Method Based on Location in Wireless Ad-hoc Networks", Book chapter in New Technologies, Mobility and Security. pp. 361-372, 2007. Springer.

[49] K. Roshan, V. Bibhu, Preventive Aspect of Black hole Attack in Mobile AD Hoc Network, International Journal on Computer Network and Information Security (IJCNIS), June 2012 in MECS, 2012, 6, 49-55.

[50] C. Iwendi, A. Allen, K. Offor, "Smart Security Implementation for Wireless Sensor Network Nodes"; Journal of Wireless Sensor Networks (JWSN), vol.1. pp. 14-26, July 2013.

[51] K. Sharma, M. K. Ghose. "Wireless Sensor Networks: An Overview on its Security Threats". International Journal of Computer Applications (IJCA), Special Issue on "Mobile Ad-hoc Networks" MANETs, pp. 42-45, 2010.

[52] J. Perrig, J. Stankovich, D. Wagner, "Security in Wireless Sensors network", Communications of the ACM. vol.47, No.6, pp. 53-57. June 2004.

[53] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, F. Silva, «Directed Di usion for Wireless Sensor Networking», IEEE/ACM Transactions on Networking, vol.11, pp. 2-16, Feb. 2003.

[54] H.Jerome Saltzer, D. MichaelSchroeder, "The Protection of Information in Computer Systems"; Fourth ACM Symposium on Operating System Principles, October 1973.

[55] T. Stapko, "Practical embedded security: building secure resource-constrained systems", book from Newnes edition, Elsevier Inc, pp.2-12, 2008.

[56] W. Yong, A. Garhan, R.A. Byrav, survey of security issues in wireless sensor networks,IEEE Communications Surveys & Tutorials, 2006.

[57] S. M. Hedetniemi, S. H. Hedetniemi, A. Liestman,«A Survey of Gossiping and Broadcasting in Communication Networks», Networks, vol. 18, 1988.

[58] R. Anderson,H. Chan,A. Perrig, " Key Infection: Smart Trust for Smart Dust" ; 12th IEEE International Conference on Network Protocols (ICNP), 2004.

[59] A. D. Wood, J.A. Stankovic, Denial of service in sensor networks, IEEE Computer, 2002, pp. 48-56.

[60] C. Iwendi, A. Allen, K. Offor, "Smart Security Implementation for Wireless Sensor Network Nodes"; Journal of Wireless Sensor Networks (JWSN), vol.1. pp. 14-26, July 2013.

[61] V. Malathi, B. Sivakumar, "A Security Framework for Wireless Sensor Networks: IBE-Trust", Journal of Electronics and Communication Engineering, Volume 2, Issue 4 (Sep.-Oct. 2012), PP 05-09.

[62] R. Rosli, Y. M. Yusoff, H. Hashim, "Performance Analysis of ID-Bsed Authentification on Zigbee Transceiver", International Symposium on Wireless technology and applications (ISWTA'2012), 23-26 Sept 2012, pp 187 – 191.

[63] R. El Kaissi, A. Kayssi, A. Chehab, Z. Dawy," DAWWSEN: A Defense Mechanism against Wormhole ttack In Wireless Sensor Network", Proceedings of the Second International Conference on Innovations in Information Technology 2005 (IIT'05).

[64] K. W. Sandar, "Analysis of Detecting Wormhole Attack in Wireless Networks", World Academy of Science, Engineering and Technology 24, 2008. Pp. 422-428.

[65] P. Maidamwar, N. Chavhan, "A Survey On Security Issues To Detect Wormhole Attack In Wireless Sensor Network", International Journal on AdHoc Networking Systems (IJANS) Vol. 2, No. 4, October 2012, pp 37-50.

[66] Z. Tun, A. M. Htein, "Wormhole Attack Detection in Wireless Sensor Networks", Proceedings of world academy of science, engineering and technology, vol. 36. Dec. 2008. pp. 549-554.

[67] P. Hämäläinen, M. Kuorilehto,. M. Hännikäinen, "Security in Wireless Sensor Networks: Considerations and Experiments". SAMOS, Lecture Notes in Computer Science Vol. 4017. (pp. 167–177), 2006.

[68] **A. Louazani**, Sekhri Larbi, Kechar Bouabdellah, "A Security Scheme against Wormhole Attack in MAC Layer for Delay Sensitive Wireless Sensor Networks", International Journal of Information Technology and Computer Science(IJITCS), Vol. 6, No. 12, November 2014, pp.1-10.

[69] B. Berthomieu, M. Menasche, "An Enumerative Approach for Analyzing Time Petri Nets". IFIP Congress Series, Elsevier Science Publishers, Vol. 9, (pp. 41-46), 1983.

[70] B. Berthomieu, P.O. Ribet, F. Vernadat, "The tool TINA - Construction of abstract state spaces for Petri Nets and Time Petri Nets". International Journal of Production Research, Vol.42, No.14. pp. 2741-2756, 2004.

[71] R. Maheshwari, J.Gao, S. R. Das. "Detecting Wormhole Attacks in Wireless Networks Using Connectivity Information". INFOCOM 2007. 26th IEEE International Conference on Computer Communications, Anchorage, Alaska, USA, 6-12 May 2007, pp 107-115.

[72] **A. Louazani**, Sekhri Larbi, Kechar Bouabdellah," Sinkhole Attack Detection Approach in Wireless Sensor Networks", International Confetrence on Networking and Advanced Systems (ICNAS'2013), Jun 19-20, 2014, Annaba-Algeria.

[73] L. Eschenauer, D.Virgil, D. Gligor. "A key-management scheme for distributed sensor networks", In Proceedings of the 9th ACM conference on Computer and communications security, pages 41-47, 2002.

[74] C. Haowen, A. Perrig. Pike,"Peer intermediaries for key establishment in sensor networks", In Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005), volume 1, pages 524-535, 2005.

[75] C. Haowen, A. Perrig, D. Song,"Random key predistribution schemes for sensor networks", In Proceedings of the Symposium on Security and Privacy, pages 197-213, 2003.

[76] E.Egea-Lopez, J.Vales-Alonso, A.S. Martinez-Sala, P.Pavon-Marino, J.Garcia-Haro "Simulation Tools for Wireless Sensor Networks", Summer Simulation Multi conference – SPECTS- 2005.

[77] **A. Louazani**, "Protocole de contrôle d'accès au médium à efficacité énergétique pour les réseaux de capteurs sans fil", Mémoire de Magister en Informatique, Centre universitaire Mustapha Stambouli, Mascara, Algérie, juin 2008.

[78] H. Kumar Kalita, A. Kar; "Wireless sensor network security analysis", International Journal of Next-Generation Networks (IJNGN),Vol.1, No.1, Department of Computer Engineering, Jadavpur University, Kolkata, India December 2009.

[79] A. Varga, «OMNeT++ Installation Guide Version 4.2», www.omnetpp.com, copyright 2011.

[80] **A. Louazani**, Sekhri Larbi, Kechar Bouabdellah, "A Time Petri Net model for Wormhole Attack Detection in Wireless Sensor Networks", procceding of 4th International Confetrence on Smart Communication on Network Technologies (SaCoNeT'2013), Jun 17-19 ,2013, Paris-France, pp. 29-34.

[81] Z. Tun, A. Htein, A Maw, "Wormhole Attack Detection in Wireless Sensor Networks", Proceedings of world academy of science, engineering and technology, Vol. 2. pp. 481-486, 2008.

[82] Y. Hu, A. Perrig, B. D. Johnson, "Wormhole attacks in wireless networks", IEEE Journal on Selected Areas in Communications, 2006, 24(2), p.370–380.

[83] D. Martins, H. Guyenne, "Etat de l'art Sécurité dans les réseaux de capteurs sans fil", Sar SSI Book, 2008, pp. 167-181.

[84] M. Saxena; "Security in Wireless Sensor Networks: A Layer-based Classification", CERIAS Tech. Report 2007-04.

[85] M. Rout, B. Majhi, R. Majhi, G.Panda; "Novel Stock Market Prediction Using Hybrid Model of Adaptive Linear Combiner and Differential Evolution", Computer networks and information technologies: Proceeding of the 2nd International Conference on Advances in Communication, Network, and Computing CNC'2011, Banglore –India, March 2011, pp.187-196.

[86] S.Suresh kumer, T.V.P.Sundararajan,A.Shanmugam, "Performance Comparison of Three Types of Wormhole Attack in Mobile Adhoc Ntworks", proceeding of International Conference on Information Science and Applications (ICISA' 2010), Chennai- India-, 6 February 2010, pp. 443-447.

[87] R. Matam, S. Tripathy,'WRSR: wormhole-resistant secure routing for wireless mesh networks', EURASIP Journal on Wireless Communications and Networking 2013, pp.2-12.

[88] J. Kwok; "A Wireless Protocol to Prevent Wormhole Attacks"; A Thesis in TCC 402 (Requirements for the Degree Bachelor of Science in Computer Engineering), School of Engineering and Applied Science University of Virginia, March 23, 2004.

[89] C.C. Umunna, "Security and Performance Analysis of Topology-Based Intrusion Detection System in Ad Hoc Networks"; Master theses, Master of Science in Electrical Engineering; Blekinge Institute of Technology Karlskrona, Sweden, May 2009.

[90] W. Wang, B. Bhargava, Yi Lu, W. Xiaoxin ; "Defending against Wormhole Attacks in Mobile Ad HocNetworks "; Wiley Journal of Wireless Communications and Mobile Computing (WCMC), vol. 6, no. 4, pp. 483–503, 2006.

[91] R. Shree, R.A. Khan;“ Wormhole Attack in Wireless Sensor Network”;International Journal of Computer Networks and Communications Security vol.2, No.1, january 2014, pp.22–26.

[92] J. H. Conway, J. A. Sloane, “Sphere Packings” , Lattices and Groups, 2nd ed. New York, NY: Springer-Verlag, 1993. pp.142-147.

[93] A.khan Pathan, H. Woo, L. Choong, S. Hong; ‘Security in wireless sensor networks: Issues and challenges’, The 8th International Conference on Advanced Communication Technology, 2006. ICACT 2006, Phoenix Park, Gangwon-Do, Korea. pp.1043-1048.

[94] C. Karlof, N. Sastry, D. Wagner, “TinySec: a link layer security architecture for wireless sensor networks”, Proceeding of the 2nd international conference on Embedded networked sensor systems, Baltimore, MD, USA, 2004, pp. 162 – 175.

[95] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J.D. Tygar, “SPINS: Security Protocols for Sensor Networks”, Wireless Networks, vol. 8, no.5, 2002, pp. 521-534.

[96] W. Du, Deng, Y.S. Han, P.K. Varshney, “A pairwise key pre-distribution scheme for wireless sensor networks”, Proc. of the 10th ACM conference on Computer and communications security, 2003, pp. 42-51.

[97] Z. Karakehayov, "Using REWARD to detect team black-hole attacks in wireless sensor networks", in Workshop on Real-World Wireless Sensor Networks (REALWSN'05), 20-21 June, 2005, Stockholm, Sweden.

[98] L. Eschenauer, V.D. Gligor, "A key-management scheme for distributed sensor networks", Proc. ACM CCS'02, 18-22 November 2002, pp. 41-47.

[99] S. Byungrak, H. Yong-sork, K. Jung-Gyu, “A design and implementation of forest-fires surveillance system based on wireless sensor networks for south Korea mountains”, International Journal of Computer Science and Network Security (IJCSNS), Vol. 6, No. 9B, pp. 124-130, 2006.

[100] R. Injong, W. Ajit, A. Mahesh, M. Jeongki, “Z-MAC: an Hybrid MAC for Wireless Sensor Networks”, Dept. of Computer Science, North Carolina State University, SenSys’05, November 2-4, 2005.

[101] R. Zurbuchen, «T-MAC, TRAMA», Universität Bern, November 24, 2005.

[102] A. Markus, “Comparison of TDMA and contention based MAC protocols on embedded sensor boards”, Masterarbeit, Universiťat Bern, 2006.

[103] G. Finn, “Routing and addressing in large metropolitan-scale internetworks”, ISI Research Report, ISU/RR-87–180, March 1987.

[104] J. Kuruvila, A. Nayak, I. Stojmenovie; “Progress Based Localized Power and Cost Aware Routing Algorithms for Ad Hoc and Sensor Wireless Networks”; Ad-Hoc, Mobile, and Wireless Networks, Proceedings of 3rd International conference ADHOC-NOW, Vancouver, Canada, July 2002, pp.294-299

[105] IEEE 802.15.4, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), Standard, IEEE, pp. 1-305, 2006.

[106] L. Audibert, «UML 2.0»,version 2006. Université Paris 13, http://www-lipn.univ-paris13.fr/ audibert/pages/enseignement/cours.htm.

[107] F. DI Gallo, « Méthodologie des systèmes d'information–UML »,. CNAM ANGOULEME 2000-2001.http://www.scribd.com/doc/2226078/cours-uml

[108] S. W. Ambler,"The Elements of UML (TM) 2.0 Style"; Cambridge University press. ISBN-13: 978-0521616782, May 9, 2005.

[109] J. Rumbaugh, I.Jacobson, G. Booch; "The Unified Modeling Language Reference Manual", Addition-Wesley; Second Edition, ISBN 0-321-24562-8, First printing, July 2004.

[110] F. Legendre, "Exploitation de la logique propositionnelle pour la résolution des problèmes cryptographiques", Thèse doctorat, Université de Reims Champagne-Ardenne, soutenue le 30 juin 2014.

4. Webographie

[Web 1]. http://www.ietf.org/rfc/rfc2501.txt Consulté le 07/10/2014

[Web 2]. http://www.ieee802.org/11/index.shtml Consulté le 09/10/2014

[Web 3]. http://wirelessman.org/ Consulté le 10/10/2014

[Web 4]. http://grouper.ieee.org/groups/802/20/ Consulté le 10/10/2014

[Web 5]. https://www.bluetooth.org/apps/content/ Consulté le 10/10/2014

[Web 6]. http://www.darpa.mil/ Consulté le 13/10/2014

[Web 7]. http://www.ti.com/ Consulté le 20/12/2014

[Web 8]. http://www.atmel.com/ Consulté le 13/10/2014

[Web 9]. http://www.btnode.ethz.ch/ Consulté le 13/10/2014

[Web 10]. http://www.eyes.eu.org/ Consulté le 01/01/2015

[Web 11]. http://www.hanback.com/english/sub1.htm Consulté le 01/01/2015

[Web 12]. http://www.scatterweb.com Consulté le 01/01/2015

[Web 13]. www.moteiv.com Consulté le 02/01/2015

[Web 14]. http://www.xbow.com/ Consulté le 02/01/2015

[Web 15]. http://www.sunspotworld.com Consulté le 02/01/2015

[Web 16]. http://www.xbow.com/ Consulté le 03/01/2015

[Web 17]. https://www.schneier.com/crypto-gram-0005.html Consulté le 18/11/2014

[Web 18]. http://www.albion.com/security/intro-4.html Consulté le 02/01/2015

[Web 19]. http://dictionary.reference.com/browse/computer+security Consulté le 02/01/2015

[Web 20]. *UML resource page*, http://www.uml.org/ Consulté le 04/01/2015 à 16h:30

9. LISTE DES PUBLICATIONS

Publication

- **Louazani Ahmed**, Sekhri Larbi and Kechar Bouabdellah,"A Security Scheme against Wormhole Attack in MAC Layer for Delay Sensitive Wireless Sensor Networks", International Journal of Information Technology and Computer Science(IJITCS), vol.6, no.12, pp.1-10, 2014. DOI: 10.5815/ijitcs.2014.12.01

Conférences internationales avec comité de lecture

- **Ahmed Louazani**, Sidi Mohammed Senouci, Bendaoud Mohammed Abderrahmane. *"Clustering-based Algorithm for Connectivity Maintenance in Vehicular Ad-Hoc Networks"*. 14th International Conference on Innovations for Community Services (I4CS), Reims, France June 4-6, 2014
- **Ahmed Louazani**, Larbi Sekhri and Bouabdellah Kechar, *"Sinkhole Attack Detection Approach in Wireless Sensors Network"*. Proceeding of the 1st International Conference on Networking and Advanced Systems (ICNAS) Annaba – ALGERIA- 19-20 June 2013, Pp. 81-85
- **Ahmed Louazani**, Larbi Sekhri and Bouabdellah Kechar, *"A Time Petri Net model for Wormhole Attack* Detection *in Wireless Sensor Networks".* 4th International Conference on Smart Communication in Network Technologies (*SACONET 2013*), Paris, France, 18-19 Juin 2013.
- Mounir Tahar Abes and **Ahmed Louazani**, *" Mobility Impact on Ad Hoc Routing Protocols"*, Procceding of the 1st International Conference on Technologies and Communication (ICNTC'2012), 5-6 december 2012, Chlef, Algeria, pp. 23-29.
- **Ahmed Louazani**, Larbi Sekhri and Bouabdellah Kechar, *"Detecting and Avoiding Wormhole Attack in a Cross-layer MAC Protocol for Wireless Sensor Networks",* Journées thématiques (RSACS-2011*)*, 22-23 juin, 2011, Oran, Algeria.
- Bouabdellah Kechar, sekhri Larbi and **Ahmed Louazani**, "*Etude formelle d'un Protocole MAC Inter-couches à* Economie *d'Energie Proposé Pour les Réseaux de Capteurs Sans Fil (RCSF)* " Conférence Internationale des Technologies de l'Information et de la Communication (*CITIC'09)* université Ferhat Abbes, SETIF, Algérie, 04 - 05 mai, 2009.
- Bouabdellah Kechar, Larbi Sekhri and **Ahmed Louazani,** "*Formal Modelling and Validation of a Novel Energy Efficient Cross-Layer MAC Protocol in Wireless Multi Hop Sensor* Networks *Using Time Petri Nets*", 2ndInternational Conference on New Technology, Mobility and Security (NTMS'2008), Tanger -Maroco, 5-7 November, 2008.

Conférences nationales avec comité de lecture

- **Ahmed louazani**, *"Wirless Sensor Network security design"*, 5ièmes Journées scientifiques Infodays'2011, Chlef, Algerie, 26-27 Avril, 2011.
- **Ahmed Louazani**, "*WSN security: WormHole Attack*", 4ièmes Journées scientifiques Infodays'2010, Chlef, Algerie, 13-14 Avril, 2010.
- **Ahmed Louazani**, "*Energy efficient cross-layer protocol design for Wireless Sensor Network*"; 3ièmesJournéesscientifiques Infodays'2009, Chlef, Algeria, 14-15 Avril, 2009.

10. ANNEXE

a. OMNET++ and Castalia Installation quidelines

Before starting the installation, refresh the database of available packages. Type in the terminal [79]:

```
$ sudo apt-get update
```

To install the required packages, type in the terminal:

```
$ sudo apt-get install build-essential gcc g++ bison flex perl \

tcl-dev tk-dev blt libxml2-dev zlib1g-dev openjdk-6-jre \

doxygen graphviz openmpi-bin libopenmpi-dev libpcap-dev
```

At the confirmation questions (Do you want to continue? [Y/N]), answer Y. This is a large file (148 MB) so it might take some time to download.

Get the source code. At the moment of writing the latest version is: OMNeT++ 4.1 (source + IDE, tgz)

An easy way to get the tgz file into your system is go in your home dir and type:

```
$ wget http://www.omnetpp.org/omnetpp/doc_download/2217-omnet-42-source--ide-tgz
```

Untar and unzip the source file:

```
$ tar xvfz omnetpp-4.2-src.tgz
```

A directory named omnetpp-4.1 will be created.

Set environment variables by typing (assuming you are using bash as your shell)

```
$ export PATH=$PATH:~/omnetpp-4.2/bin

$ export LD_LIBRARY_PATH=~/omnetpp-4.2/lib
```

Also add the above two export commands at the end of your .bash_profile file. You are now ready to build OMNeT:

```
$ cd omnetpp-4.2/

$ NO_TCL=1 ./configure

$ make
```

The last command will take a few minutes to complete. You are now done building OMNeT. Castalia does not use the Tcl functionality so we opt to build OMNeT without it. The installation process can be easier if Tcl in not required. If you wish, you can try build OMNeT with Tcl. Make sure that OMNeT++ is in the path .

b. Installing Castalia

Get the source code from http://castalia.npc.nicta.com.au/ . Assume the file you downloaded is named Castalia-3.2.tar.gz

Untar and unzip the source code:

```
$ tar –xvzf Castalia-3.2.tar.gz
```

A new directory will be created, named Castalia-3.0/.You are ready to build Castalia:

```
$ cd Castalia-3.2/

$ ./makemake
```

Wait for a few seconds till the script ends. This automatically generates a Makefile that you can use to build Castalia. Type:

```
$ make
```

Wait again for some time until everything is built. Check that the soft link CastaliaBin is created in Castalia-3.2/. You have now successfully built Castalia

c. Reachability analysis of T-PN

```
Tina version 2.10.0 -- 03/08/11 -- LAAS/CNRS
REACHABILITY ANALYSIS ----------------------------------------
bounded
10 classe(s), 15 transition(s)
CLASSES:

state 0
marking
        p1 p11 p9
firing domain
        0 <= t1 <= 1

state 1
marking
        p0 p11 p2 p5 p9*2
firing domain
        8 <= t13 <= 9
        3 <= t5 <= 4
        0 <= t8 <= 0

state 2
marking
        p0 p10 p11 p2 p5
firing domain
        8 <= t13 <= 9
        3 <= t5 <= 4
        19 <= t9 <= 20

state 3
marking
        p0 p10 p12 p2
firing domain
        5 <= t13 <= 6
        16 <= t9 <= 17

state 4
marking
        p0 p10 p12 p2
firing domain
        4 <= t13 <= 5
        15 <= t9 <= 16

state 5
marking
        p0*2 p10 p12 p2
firing domain
        0 <= t12 <= 0
        8 <= t13 <= 9
        11 <= t9 <= 12
```

state 6
marking
p0*2 p10 p12 p2
firing domain
0 <= t12 <= 0
8 <= t13 <= 9
10 <= t9 <= 11

state 7
marking
p10 p12
firing domain
11 <= t9 <= 12

state 8
marking
p10 p12
firing domain
10 <= t9 <= 11

state 9
marking
p12 p9
firing domain

REACHABILITY GRAPH:

0 -> t1@0/1, t1@1/1
1 -> t8@0/2
2 -> t5@3/3, t5@4/4
3 -> t13@5/5, t13@6/6
4 -> t13@4/5, t13@5/6
5 -> t12@0/7
6 -> t12@0/8
7 -> t9@11/9, t9@12/9
8 -> t9@10/9, t9@11/9
9 ->
0.000s

LIVENESS ANALYSIS ----------

not live
not reversible

1 dead classe(s), 1 live classe(s)
6 dead transition(s), 0 live transition(s)
dead classe(s): 9
dead transition(s): t7 t6 t4 t3 t2 t14

STRONG CONNECTED COMPONENTS:
ANALYSIS COMPLETED ----------

d. *Analyse* du RDP-Temporel Modèle de l'attaque Sinkhole

```
Struct version 2.10.0 -- 03/08/11 -- LAAS/CNRS

parsed net CL_MAC_Sinkhole_Attack_Detection

14 places, 12 transitions

net CL_MAC_Sinkhole_Attack_Detection
tr t1 [0,1] p1 -> p0 p2 p5 p9
tr t12 [0,0] p0*8 p2 ->
tr t13 [1,1] p0 -> p0*2
tr t14 p0 p3 -> p3
tr t2 [3,4] p2 p6 -> p3 p7
tr t3 [0,0] p3 p8 -> p4
tr t4 [19,20] p4 -> p1
tr t5 [3,4] p11 p5 -> p12 p6
tr t6 [10,11] p12 p7 -> p13 p8
tr t7 [19,20] p13 -> p11
tr t8 [0,0] p9*2 -> p10
tr t9 [19,20] p10 -> p9
pl p1 (1)
pl p11 (1)
pl p9 (1)

0.000s

P-FLOWS BASIS ----------------------------------------------------

p3 p7*-1 p8*-1
p11 p13 p3 p6 p8*-1
p12 p3*-1 p6*-1 p8
p1 p3 p4 p5 p6

0.000s

T-FLOWS BASIS ----------------------------------------------------

t1 t13*-1 t2 t3 t4 t5 t6 t7 t8 t9
t1 t14 t2 t3 t4 t5 t6 t7 t8 t9

0.000s

ANALYSIS COMPLETED  ----------------------------------------------
```