REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou
Faculté de Génie Electrique et Informatique
Département Informatique

Mémoire

En vue de l'obtention du diplôme de Master en Informatique

Option : Systèmes Informatiques

Thème :

DHD-LEACH

Nouveau protocole de routage hiérarchique sécurisé pour les réseaux de capteurs sans fil

Proposé et dirigé par :

M^me^ Samia BENSAID.

Réalisé par :

M^elle^ Kathia ZIDANE.

Promotion : 2014 /2015

Remerciements

Je remercie tout d'abord le bon dieu qui m'a aidé et donné la volonté pour la réalisation de ce travail, mes vifs remerciements s'adressent à ma promotrice Mme S.BENSAID pour son suivi et son engagement lors de l'élaboration de ce travail, je remercie beaucoup Monsieur M.RAMDANE pour ses orientations et ses conseils qui m'ont été très efficaces, je remercie infiniment A.SADI qui m'a beaucoup aidé et encouragé, je remercie vivement les membres du jury qui m'ont fait l'honneur d'accepter d'évaluer mon travail. Finalement, je remercie tout ceux qui m'ont aidé et ont contribué de prés ou de loin à la réalisation de ce travail.

Merci.

Dédicaces

À mes chers parents,

À la mémoire de mon cher oncle Mohamed,

À la mémoire de mes grands parents,

À ma chère sœur Hayat ainsi que son mari Rachid,

À mes chers frères : Jugurtha et notre petit Karim,

Et à tous ceux qui m'aiment.

Chapitre I : Généralités sur les RCSFs

Chapitre II : La sécurité dans les RCSFs

Chapitre III : Le protocole de routage hiérarchique LEACH

Chapitre IV : Sécurisation du protocole de routage hiérarchique LEACH

Chapitre V : Réalisation

Chapitre I : Généralités sur les RCSFs

Chapitre II : La sécurité dans les RCSFs

Chapitre IV : Sécurisation du protocole de routage hiérarchique LEACH

Chapitre V : Réalisation

ADC : Analog to Digital Converter.

APTEEN : AdaPtive Threshold sensitive Energy Efficient sensor Network protocol.

BS : Base Station.

CH : Cluster-Head.

CHs : Cluster-Heads.

CSMA : Carrier Sense Multiple Access.

CDMA : Code Division Multiple Access.

DD : Directed Diffusion.

GEAR : Geographic and Energy Aware Routing.

GloMoSim : Global Mobil Simulator

GPS : Global Positioning System.

IDE : Integrated Development Environment

LEACH : Lew Energy Adaptive Clustering Hierarchy.

LEAP : Localized Encryption and Authentication Protocol.

LQI : Link Quality Indicator

MANET : Mobile Ad hoc NETwork.

MAC : Message Authentication Code.

MAC : Media Access Control.

NAM : Network Animator.

NS : Network Simulator.

OSI : Open System Interconnexion.

OTCL : Object Tools Command Langage.

OMNeT++ : Objective Modular Network Test-bed in C++

PEGASIS : Power-Efficient GAthering in Sensor Information Systems.

PCL : Parallel Computing Labaratory

RCSF : un Réseau de capteurs sans fil.

RCSFs : les Réseaux de capteurs sans fil.

RSSI : Received Signal Strength Indication

RSA : Rivest, Shamir et Adleman

SNEP : Secure Network Encryption Protocol.

SB : Station de Base.

SRPBCG : Secure Routing Protocol Cluster-Gene-Based for WSNs.

SPIN : Sensor Protocols for Information via Négociation.

TDMA : Time Division Multiple Access.

TEEN : Threshold sensitive Energy Efficient sensor Network protocol.

UCLA : University of California, Los Angeles

WSN : Wireless Sensor Network.

µTESLA : the micro version of the Timed Efficient Stream Loss-Tolerant Authentication Protocol.

Résumé

Les progrès réalisés dans le domaine des communications sans fil et le besoin d'observer et de contrôler des phénomènes physiques, ont permis le développement des micro-capteurs, moins coûteux et multifonctionnels. Ces caractéristiques ont permis de se projeter dans la naissance des réseaux de capteurs sans fil (RCSFs), et de favoriser leur utilisation dans une multitude d'applications. Celles-ci nécessitent souvent un déploiement dans des environnements hostiles, où les nœuds capteurs et les liens de communication sont continuellement exposés à des menaces importantes.

Par conséquent, les services offerts par les RCSFs doivent garantir le niveau de sécurité requis par l'application. Cet objectif est compliqué d'avantage à cause des caractéristiques des réseaux de capteurs.

Dans cette optique, nous nous sommes intéressés à la sécurisation du protocole de routage LEACH, l'un des protocoles de routage hiérarchiques les plus répandus dans les RCSFs. Pour atteindre cet objectif, nous avons étudié ce protocole et les attaques les plus dangereuses pouvant le menacer. Cela nous a permis de proposer et mettre en place un nouveau protocole qu'on a nommé DHD-LEACH (Dynamic Hole Detection in LEACH) qui intègre des mécanismes de sécurité permettant de détecter les attaques de type trou noir et gris.

Mots clés : RCSFs, routage, sécurité, LEACH, DHD-LEACH.

Introduction générale

Introduction générale

L'intérêt croissant des réseaux de capteurs sans fil (RCSFs) peut se justifier par leurs caractéristiques intéressantes, à savoir : un grand nombre de noeuds capteurs de taille miniature, autonomes, à faible coût et multifonctionnels. Ils sont déployés dans des environnements parfois hostiles auxquels l'homme n'a pas toujours accès. Grâce aux caractéristiques intéressantes des RCSFs, ils peuvent être utilisés dans divers domaines, tels que la gestion des catastrophes, la protection des frontières, la santé, le domaine militaire,…etc.

Dans un RCSF, chaque nœud capteur a la possibilité d'accomplir trois tâches : détecter les informations du phénomène observé, les traiter puis les communiquer à la station de base (*sink*) via les ondes radio. La station de base est chargée de transmettre les informations remontées par les différents capteurs au gestionnaire des tâches via internet ou satellite. Pour cela, la communication entre les noeuds d'un RCSF doit être régie par un ensemble de règles « protocoles » afin de leurs permettre de fonctionner correctement, et les concepteurs de ces protocoles doivent prendre en considération l'architecture et les exigences de l'application ainsi que les caractéristiques des noeuds capteurs. De nombreux chercheurs se sont intéressés par ce sujet, ce qui leurs a permis de proposer différents protocoles de routage.

Dans ce mémoire, nous allons nous intéressé aux problèmes relatifs au routage des données sur les RCSFs et plus précisément le routage hiérarchique des données. Ce dernier est considéré comme un outil permettant plus de performance en ce qui concerne la consommation énergétique par rapport aux autres types de routage, à savoir, le routage à topologie plate et le routage géographique. Les protocoles de routage hiérarchiques supposent que tous les capteurs du réseau sont amicaux et coopératifs. Toutefois, cette hypothèse n'est pas toujours valable, car les RCSFs sont vulnérables à plusieurs types de menaces. Par conséquent, pour préserver les bonnes opérations du réseau, la sécurité doit être considérée comme une composante essentielle du mécanisme de routage. Notre projet consiste à sécuriser le protocole de routage LEACH, conçu pour les topologies des RCSFs hiérarchiques. Ce mémoire est organisé comme suit :

Le premier chapitre intitulé « Généralités sur les RCSFs » : est une introduction aux réseaux de capteurs sans fil. Nous y décrirons leur architecture et caractéristiques, et leurs domaines d'applications. Nous nous intéresserons par la suite au routage dans les RCSFs. Pour cela, nous allons expliquer les contraintes de conception d'un protocole de routage, ainsi qu'une classification de ces protocoles.

Dans le deuxième chapitre intitulé « La sécurité dans les RCSFs », nous exposons les différentes vulnérabilités de ces derniers et les types de menaces qui peuvent les viser. Nous définirons aussi les objectifs et les besoins de la sécurité et les mécanismes qui les vérifient. Nous nous intéresserons par la suite au routage sécurisé dans les RCSFs.

Dans le troisième chapitre intitulé « Le protocole de routage hiérarchique LEACH », nous expliquons l'architecture de communication du protocole de routage LEACH et les protocoles MAC utilisés par ce dernier. Nous expliquons en détail l'algorithme et les caractéristiques du protocole LEACH. Par la suite, nous étudions les attaques pouvant perturber son fonctionnement.

Dans le quatrième chapitre intitulé « Sécurisation du protocole de routage hiérarchique LEACH », nous allons proposer notre solution de sécurité pour le protocole LEACH et nous expliquons le schéma de sécurisation et le déroulement de notre nouveau protocole sécurisé nommé DHD-LEACH (Dynamic Hole Detection in LEACH).

Le cinquième et dernier chapitre intitulé « Réalisation », quant à lui, nous permettra d'exposer les résultats de simulation de notre solution, précédés par une présentation des outils nécessaires pour notre simulation, à savoir le simulateur OMNeT++ et la plateforme Castalia.

Finalement, nous clôturons par une conclusion générale et des perspectives.

CHAPITRE 1

Généralités sur les réseaux de capteurs sans fil

Introduction

Depuis quelques années, les progrès réalisés dans le domaine des communications sans fil et le besoin d'observer et de contrôler des phénomènes physiques tels que la température, la pression, la luminosité…etc, ont permis de produire des composants de quelques millimètres cubes de volume appelés capteurs. Ces derniers sont capables de capter des données environnementales, calculer des informations à l'aide des valeurs captées et les communiquer à un centre de contrôle via les ondes radio à travers un réseau de capteurs sans fil.

Dans ce chapitre, nous présentons les réseaux de capteurs sans fil, leur architecture ainsi que leurs caractéristiques et leurs domaines d'applications. Nous discutons, également, des protocoles de routage dans les RCSFs et leur classification ainsi que leurs contraintes de conception.

1. Les réseaux de capteurs sans fil

1.1 Architecture d'un micro capteur

Un capteur est un petit dispositif électronique capable de mesurer une valeur physique environnementale (température, lumière, pression, humidité, vibration), et de la communiquer à un centre de contrôle via une station de base en utilisant les ondes radio.
Chaque capteur assure les fonctions suivantes :

- Collecter des données environnementales.
- Calculer des informations à l'aide des valeurs collectées.
- Communiquer ces informations à travers le réseau.

Un nœud capteur contient quatre unités de base : l'unité de capture, l'unité de traitement, l'unité de communication et l'unité d'alimentation. Il peut contenir également, suivant son domaine d'application, des modules supplémentaires tels qu'un système de localisation (GPS : Global Positioning System), ou bien un système générateur d'énergie (cellules solaires). On peut même trouver des micro-capteurs, un peu plus volumineux, dotés d'un système mobilisateur, chargé de déplacer le micro-capteur en cas de nécessité **[BUN07]**.

1.1.1 L'unité de capture

L'unité de capture est composée de deux sous-unités : le capteur lui-même et un convertisseur analogique/numérique (ADC : Analog to Digital Converter). Le capteur est responsable de fournir des signaux analogiques basés sur le phénomène observé au convertisseur analogique/numérique. Ce dernier transforme ces signaux en un signal numérique compréhensible par l'unité de traitement.

1.1.2 L'unité de traitement

L'unité de traitement comprend un processeur associé à une petite unité de stockage et fonctionne à l'aide d'un système d'exploitation spécialement conçu pour les micro-capteurs (TinyOS par exemple). Cette unité est chargée d'analyser les données captées et d'exécuter les protocoles de communications qui permettent de faire collaborer le nœud capteur avec les autres nœuds du réseau.

1.1.3 L'unité de communication

L'unité de communication est équipée d'un couple émetteur/récepteur, appelé transceiver (transmitter et receiver). Elle est responsable d'effectuer toutes les émissions et réceptions des données via un support de transmission radio.

1.1.4 L'unité d'alimentation

Un micro-capteur est muni d'une ressource énergétique (la batterie) pour alimenter tous ses composants. Cependant, en conséquence de la taille réduite d'un micro-capteur, la ressource énergétique dont il dispose est limitée et généralement irremplaçable. Dès lors, l'énergie est la ressource la plus précieuse dans un réseau de capteurs, puisqu'elle influe directement sur la durée de vie des micro-capteurs et du réseau en entier. L'unité d'alimentation constitue donc l'un des systèmes les plus importants. Elle est responsable de répartir l'énergie disponible aux autres modules et de réduire la consommation énergétique en mettant en veille les composants inactifs. Cette unité peut aussi gérer des systèmes de rechargement d'énergie, telles que les cellules solaires, afin d'étendre la durée de vie totale du réseau **[BER04].** L'architecture d'un micro capteur est illustrée par la Figure I-1.

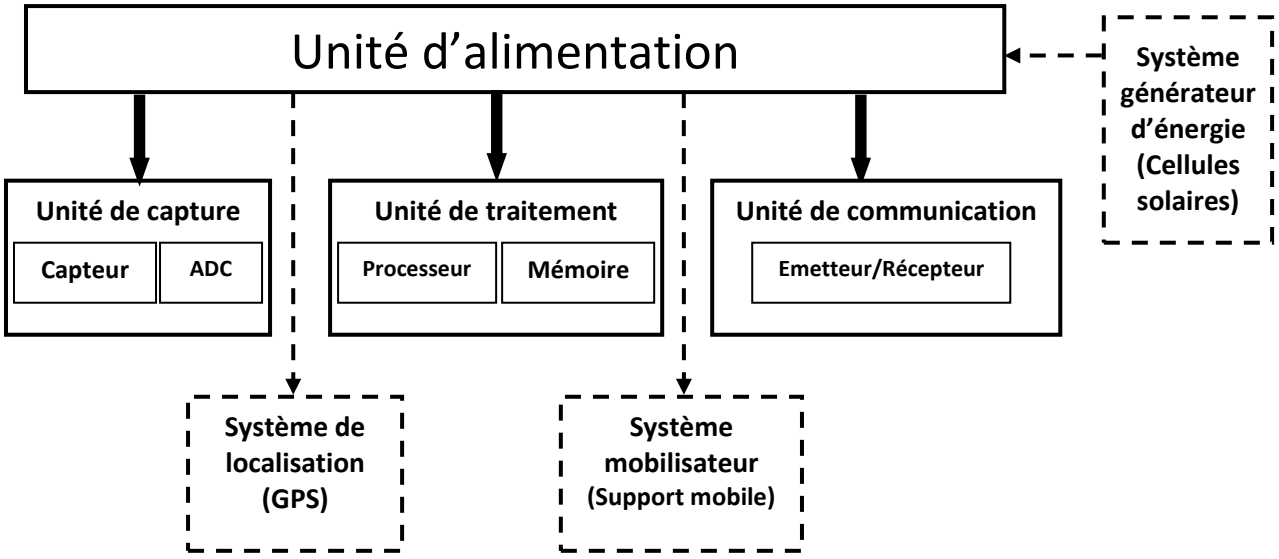

Figure I-1. Architecture d'un micro capteur.

1.2 Architecture d'un RCSF

Un réseau de capteurs sans fil est un ensemble de nœuds capteurs alimentés, dotés de capacités de calcul et de communication sans fils. Un RCSF est composé de deux types de nœuds : les capteurs et la station de base. Les capteurs sont organisés en champs de captage, ils sont disposés de manière aléatoire et ils forment la zone de couverture. Chacun de ces nœuds a la capacité de collecter des données et de les router à la station de base par l'intermédiaire d'une architecture multi-sauts. La station de base récupère les informations remontées par les différents capteurs et les transmet ensuite par internet ou par satellite à l'ordinateurs central « gestionnaire de tâches » pour analyser ces données et prendre des décisions **[CHA08]**. L'architecture d'un RCSF est illustrée par la Figure I-2.

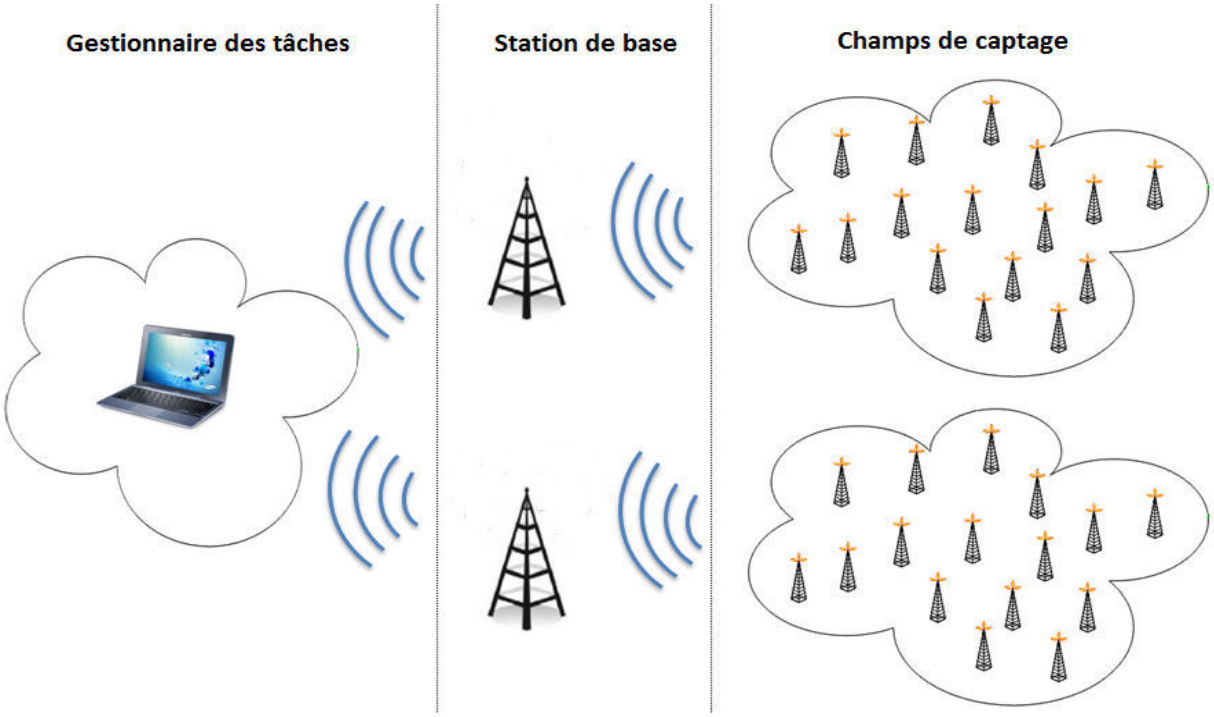

Figure I-2. Architecture d'un réseau de capteurs sans fil.

2. Fonctionnement d'un RCSF

Il y a deux méthodes pour collecter les informations d'un réseau de capteurs : à la demande ou bien suite à un événement **[BUN07]**.

2.1 A la demande

Lorsque l'on souhaite avoir l'état de la zone de couverture à un moment *T*, la station de base émet des requêtes vers toute la zone pour que les capteurs remontent leur dernier relevé vers la station de base, et les informations sont alors acheminées par le biais d'une communication multi-sauts. La collecte d'informations à la demande est illustrée par la Figure I-3.

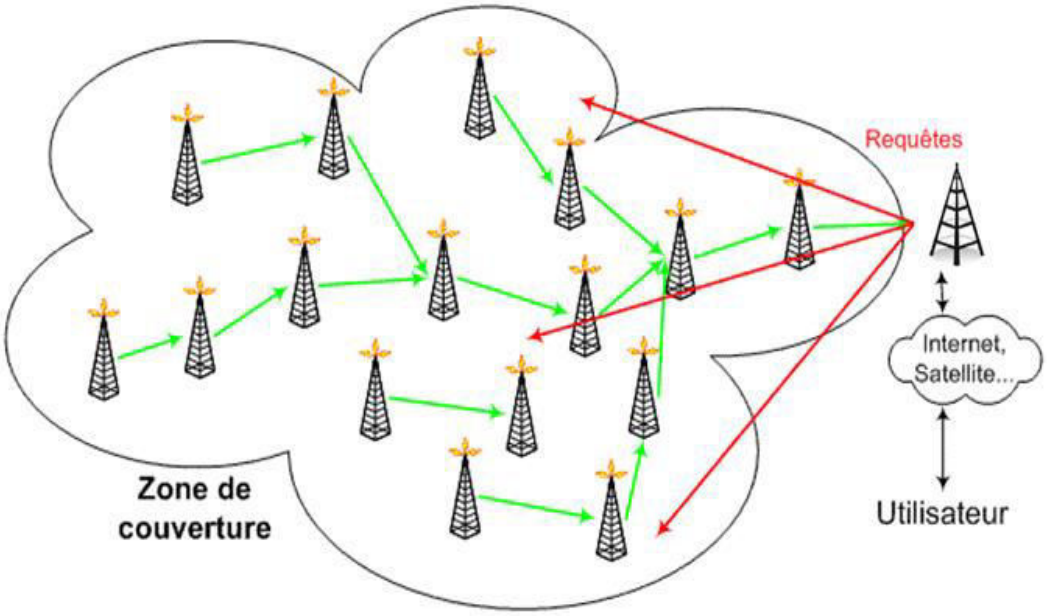

Figure I-3. Collecte d'informations à la demande.

2.2 Suite à un événement

Un événement se produit en un point de la zone de couverture (changement brusque de température, mouvement...). Par conséquent, les capteurs situés à proximité remontent alors les informations relevées et les acheminent jusqu'à la station de base. La collecte d'informations suite à un événement est illustrée par la Figure I-4.

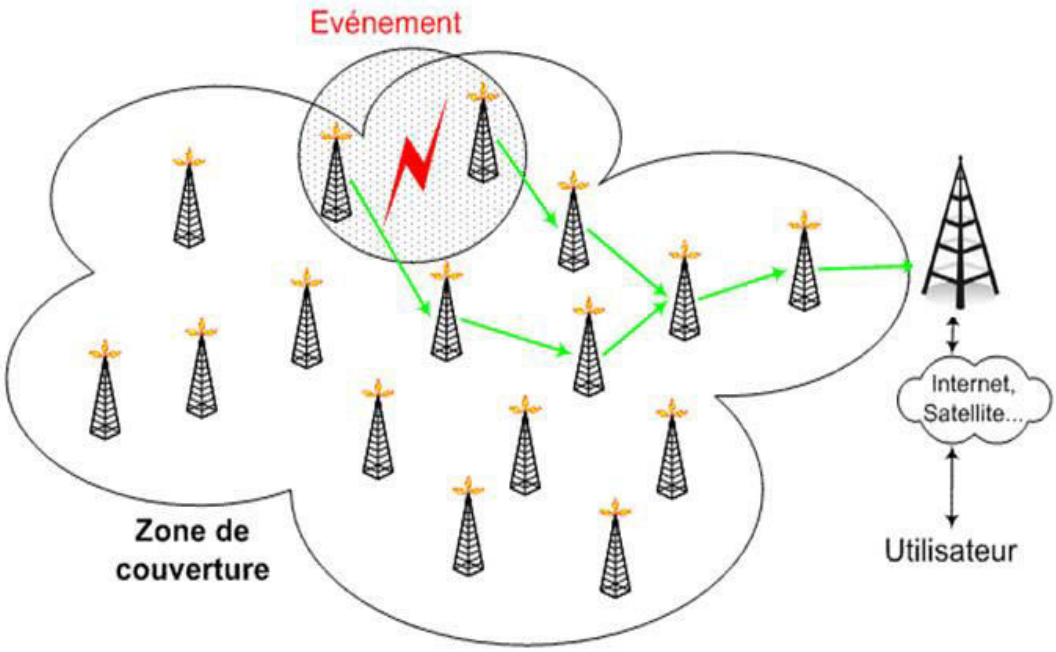

Figure I-4. Collecte d'informations suite à un événement.

3. Caractéristiques des RCSFs

Parmi les caractéristiques les plus importantes d'un réseau de capteurs sans fil, nous citons **[CHA08]** :

Durée de vie limitée

Les noeuds capteurs sont très limités par la contrainte d'énergie, ils fonctionnent généralement sans surveillance dans des régions géographiques éloignées. Par conséquent, recharger ou remplacer leurs batteries devient quasiment impossible.

Ressources limitées

Généralement les noeuds capteurs ont une taille très petite. Ce facteur de forme limite la quantité de ressources qui peuvent être mises dans ces noeuds. Par conséquent, la capacité de traitement et de mémoire sont très limitées.

Topologie dynamique

La topologie des réseaux de capteurs sans fil change d'une manière fréquente et rapide, car la défaillance d'un nœud capteur peut être très probable. En plus, les noeuds capteurs peuvent être mobiles.

Agrégation des données

Dans les réseaux de capteurs sans fil, les données captées par les noeuds capteurs situés dans la même zone de couverture sont très reliées, ce qui implique l'existence de redondances de données. Une approche répandue consiste à agréger les données au niveau des noeuds intermédiaires afin de réduire la consommation d'énergie lors de la transmission de ces données.

La scalabilité

Les réseaux de capteurs engendrent un très grand nombre de capteurs, ils peuvent atteindre des milliers, voir des millions de capteurs. Le défi à relever par les RCSFs est d'être capable de maintenir leurs performances avec ce grand nombre de capteurs.

Sécurité limitée

Les RCSFs sont plus touchés par le paramètre de sécurité par rapport aux réseaux filaires. Cela se justifie par les contraintes et limitations physiques qui font que le contrôle des données transférées doit être minimisé.

Média de transmission

Les capteurs d'un RCSF sont reliés entre eux par un canal sans fil. La qualité du signal de transmission peut être dégradée à cause des collisions et des interférences.

Auto-organisation

Un RCSF doit s'auto-organiser périodiquement, de sorte qu'il puisse s'adapter aux changements de la topologie tout en assurant son bon fonctionnement.

Tolérance aux pannes

La tolérance aux pannes est définit par la capacité de maintenir les fonctionnalités du réseau même en cas de présence de pannes. Les défaillances rencontrées dans le réseau peuvent être causées par un manque d'énergie, dommage physique du capteur ou bien des interférences environnementales. Des techniques plus tolérantes doivent être appliquées afin d'assurer le bon fonctionnement du réseau.

4. Domaines d'application des RCSFs

La taille de plus en plus réduite des micro-capteurs, le coût de plus en plus faible, la large gamme des types de capteurs disponibles (mouvement, température, vibrations,...) ainsi que l'évolution des supports de communication sans fil utilisés, ont élargi le champ d'applications des réseaux de capteurs sans fil. Dans ce qui suit, nous présentons quelques domaines d'application des réseaux de capteurs sans fil **[BER04]**.

Applications militaires

Comme dans le cas de plusieurs technologies, le domaine militaire a été un moteur initial pour le développement des réseaux de capteurs. Le déploiement rapide, le coût réduit, l'auto-organisation et la tolérance aux pannes des réseaux de capteurs sont des caractéristiques qui rendent ce type de réseaux un outil appréciable dans un tel domaine. Comme exemple d'application dans ce domaine, on peut penser à un réseau de capteurs déployé sur un endroit stratégique ou difficile d'accès, afin de surveiller toutes les activités des forces ennemies, ou d'analyser le terrain avant d'y envoyer des armées (détection d'agents chimiques, biologiques ou de radiations).

Applications liées à la sécurité

Les fissures dans la structure d'un bâtiment, suite à un séisme ou au vieillissement, pourraient être détectées par des capteurs intégrés dans les murs ou dans le béton sans alimentation électrique ou autres connexions filaires. Les capteurs doivent s'activer périodiquement et peuvent ainsi fonctionner durant des années. Un réseau de capteurs de

mouvements peut constituer un système d'alarme qui servira à détecter les intrusions sur un large secteur, on peut ainsi prévenir des cambriolages.

Applications environnementales

On peut créer un réseau autonome en dispersant les nœuds dans la nature. Des capteurs peuvent ainsi signaler des événements tels que feux de forêts, tempêtes ou inondations. Ceci permet une intervention beaucoup plus rapide et efficace des secours. On pourrait disperser des capteurs au-dessus d'un emplacement industriel pour détecter et contrôler des fuites de gaz ou de produits chimiques. Ces applications permettraient de donner l'alerte en un temps record et de pouvoir suivre l'évolution de la catastrophe. Comme dans l'agriculture, des nœuds peuvent être incorporés dans la terre. On peut ensuite questionner le réseau de capteurs sur l'état du champ (déterminer par exemple les secteurs les plus secs afin de les arroser en priorité). On peut aussi imaginer équiper des troupeaux de bétail de capteurs pour connaître en tout temps leurs positions, ce qui éviterait aux éleveurs d'avoir recours à des chiens berger.

Applications métier

On pourrait imaginer devoir stocker des aliments nécessitant un certain taux d'humidité et une certaine température (min ou max). Dans ces applications, le réseau doit pouvoir collecter ces différentes informations et alerter en temps réel si les seuils critiques sont dépassés.

Applications médicales

La surveillance des fonctions vitales d'un organisme vivant peut être facilitée par des micro-capteurs avalés ou implantés sous la peau. Des gélules sous forme de micro-capteurs ou de micro-caméras pouvant être avalées existent déjà et permettent, sans recours à la chirurgie, de transmettre des images depuis l'intérieur d'un corps humain. La surveillance de la glycémie, la surveillance des organes vitaux ou la détection précoce de certains cancers sont des applications biomédicales envisagées. Des réseaux de capteurs permettaient également une surveillance temps réel des maternités (détection de vol de bébés) ou des patients.

Applications commerciales

Il est possible d'intégrer des nœuds capteurs au processus de stockage et de livraison. Le réseau ainsi formé pourra être utilisé pour connaitre la position, l'état et la direction d'un paquet. Il devient alors possible pour un client qui attend la réception d'un paquet, d'avoir un avis de livraison en temps réel et de connaitre la position actuel du paquet. Pour les entreprises manufacturières, les réseaux de capteurs permettront de suivre le procédé de production à partir des matières premières jusqu'au produit final livré.

5. La pile protocolaire adoptée par les RCSFs

Le rôle de cette pile consiste à standardiser la communication entre les composants du réseau afin que différents constructeurs puissent mettre au point des produits (logiciels ou matériels) compatibles. Ce modèle comprend 5 couches qui ont les mêmes fonctions que celles du modèle OSI (Open System Interconnexion) [**ANU11]**. Comme la communication n'est pas le seul souci dans les réseaux de capteurs, il y a d'autres critères très importants qu'il faut en tenir compte. De ce fait, 3 couches supplémentaires sont ajoutées pour la gestion de la puissance d'énergie, la gestion de la mobilité ainsi que la gestion des tâches. Le but d'un système en couches est de séparer le problème en différentes parties (les couches) selon leurs niveaux d'abstraction. Chaque couche du modèle communique avec une couche adjacente (celle du dessus ou celle du dessous). Chaque couche utilise ainsi les services des couches inférieures et en fournit à celle de niveau supérieur. La pile de protocoles utilisée par la station de base ainsi que par tous les nœuds capteurs est donnée dans la Figure I-5.

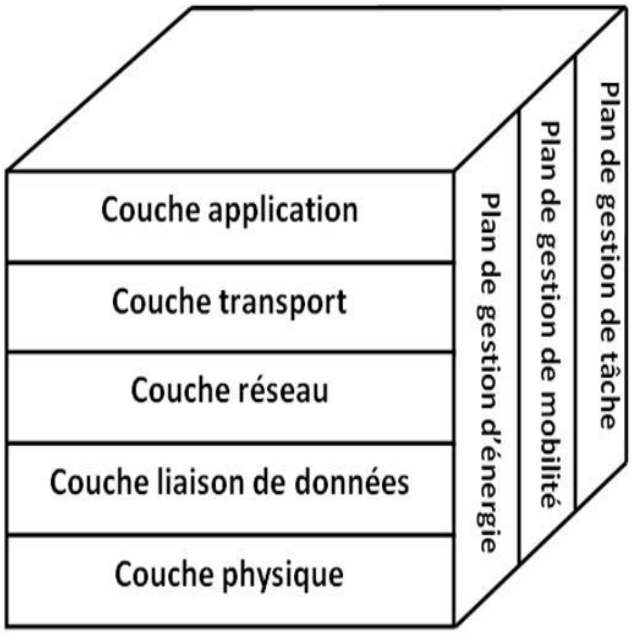

Figure I-5. La pile protocolaire des RCSFs.

Comme illustré dans la Figure I-5, la pile consiste en couche application, transport, réseau, liaison de données, physique et trois plans de gestion : d'énergie, de mobilité et de tâche [**ANU11]**. Nous décrivons dans ce qui suit les rôles et les fonctions des cinq couches et des trois plans de gestion.

La couche physique

Elle permet de moduler les données et les acheminer dans le media physique tout en choisissant les bonnes fréquences.

La couche liaison de données

Elle est responsable de l'accès au media physique et la détection et la correction d'erreurs intervenues sur la couche physique. De plus, elle établit une communication saut-par-saut entre les noeuds. C'est-à-dire, elle détermine les liens de communication entre eux dans une distance d'un seul saut.

La couche réseau

Dans la couche réseau, le but principal est de trouver une route et une transmission fiable des données captées par des nœuds capteurs vers la station de base en optimisant l'utilisation de l'énergie des capteurs.

La couche transport

Cette couche est chargée du transport des données, de leurs découpages en paquets, du contrôle de flux, de la conservation de l'ordre des paquets et de la gestion des éventuelles erreurs de transmission.

La couche application

Cette couche assure l'interface avec les applications. Il s'agit donc du niveau le plus proche des utilisateurs, géré directement par les logiciels.

Plan de gestion d'énergie

Ce plan contrôle l'utilisation de la batterie. Par exemple, après la réception d'un message, le capteur éteint son récepteur afin d'éviter la duplication des messages déjà reçus. En outre, si le niveau d'énergie devient bas, le nœud diffuse à ses voisins une alerte les informant qu'il ne peut pas participer au routage. L'énergie restante est réservée au captage.

Plan de gestion de la mobilité

Ce plan détecte et enregistre le mouvement du nœud capteur. Ainsi un nœud peut garder trace de ses nœuds voisins. En déterminant leurs voisins, les noeuds capteurs peuvent apaiser l'utilisation de leur énergie et la réalisation de tâche.

Plan de gestion de tâches

Ce plan balance et ordonnance les différentes tâches de captage de données dans une région spécifique. Il n'est pas nécessaire que tous les noeuds de cette région effectuent la tâche de captage au même temps, certains noeuds exécutent cette tâche plus que d'autres selon leur niveau de batterie.

6. Le routage dans les réseaux de capteurs sans fil

Le routage consiste à trouver un chemin pour envoyer le message de la source vers la destination. La délivrance des données dans un RCSF représente la fonctionnalité la plus importante du réseau. Pour cela, nous allons définir les contraintes de conception des protocoles de routage, puis nous allons les classifier selon des critères **[SHI10]**.

6.1 Les contraintes de conception des protocoles de routage

Plusieurs contraintes sont décisives pour toute conception d'un protocole de routage pour les RCSFs **[SHI10]**. Parmi ces contraintes, nous citons :

Tolérance aux pannes

La propriété de tolérance aux pannes est définie par la capacité du protocole de routage à maintenir ses fonctionnalités, en cas de panne de quelques noeuds. Le but de la tolérance aux pannes est d'éviter la faille totale du système malgré la présence de fautes dans un sous ensemble de ses composants élémentaires.

Consommation d'énergie

Le facteur le plus important à prendre en considération est l'énergie consommée par un capteur lors de la détection et la transmission des données captées sur le réseau. Le routage est la fonction qui consomme le plus d'énergie. Pour préserver l'énergie et augmenter la durée de vie d'un réseau, les chercheurs ont opté pour des techniques qui favorisent le traitement local des données afin de réduire la taille du paquet. Ces techniques évitent la redondance des informations à transmettre et mettent le capteur en mode sommeil le plus longtemps possible**.**

Limitations des capacités des nœuds

Un capteur est très limité en ce qui concerne les traitements locaux à cause de sa taille minimale. Cela signifie que le protocole de routage doit être simple et peu exigeant en termes de capacité de calcul et de stockage.

Scalabilité

Les applications des RCSFs nécessitent en général un déploiement dense des noeuds. Les protocoles de routage ne devraient pas souffrir d'une dégradation de performances dans le cas d'endommagement de noeuds aussi bien qu'avec un nombre plus élevé de nœuds.

Hétérogénéité

Généralement, les noeuds d'un RCSF sont homogènes ayant les mêmes capacités de calcul, de mémoire et de ressources énergétiques. Ces noeuds pourront être rapidement

épuisés puisqu'ils réalisent plusieurs tâches à la fois comme le captage, le traitement et le routage de données. Pour y remédier, une solution envisagée par certaines applications consiste à intégrer des noeuds spéciaux plus puissants que les autres et qui seront chargés d'effectuer les tâches les plus coûteuses en termes de ressources énergétiques. Cependant, l'intégration d'un ensemble de noeuds hétérogènes dans un seul réseau, impose de nouvelles contraintes liées au routage de données. La conception des protocoles de routage doit prendre en compte les différents types de noeuds, et les contraintes qui en résultent.

La topologie dynamique

Des centaines de nœuds sont déployés à travers un champ, certains nœuds peuvent être attachés à des objets mobiles, certains nœuds peuvent être ajoutés au réseau et d'autres peuvent tomber en panne, ce qui rend la topologie instable (dynamique). Le réseau devra être capable de s'auto-organiser pour continuer ses applications.

L'environnement

Les capteurs sont souvent déployés en masse dans des endroits tels que des champs de bataille au delà des lignes ennemies, à l'intérieur de grandes machines, au fond d'un océan, dans des champs biologiquement ou chimiquement pollués. Par conséquent, ils doivent pouvoir fonctionner sans surveillance dans des régions géographiques éloignées.

6.2 Classification des protocoles de routage

Les protocoles de routage pour les RCSFs ont été largement étudiés **[HEI99]**. Les auteurs dans **[ALK04]** ont proposé une classification basée sur deux critères principaux : la topologie du réseau (plate, hiérarchique et géographique) et le type d'application (time-driven et event-driven). Selon **[RAJ09]**, les protocoles de routage peuvent aussi être classés selon le mode d'établissement et la maintenance des routes (les protocoles proactifs, les protocoles réactifs et les protocoles hybrides). La classification des protocoles de routage dans les RCSFs est donnée dans la Figure I-6.

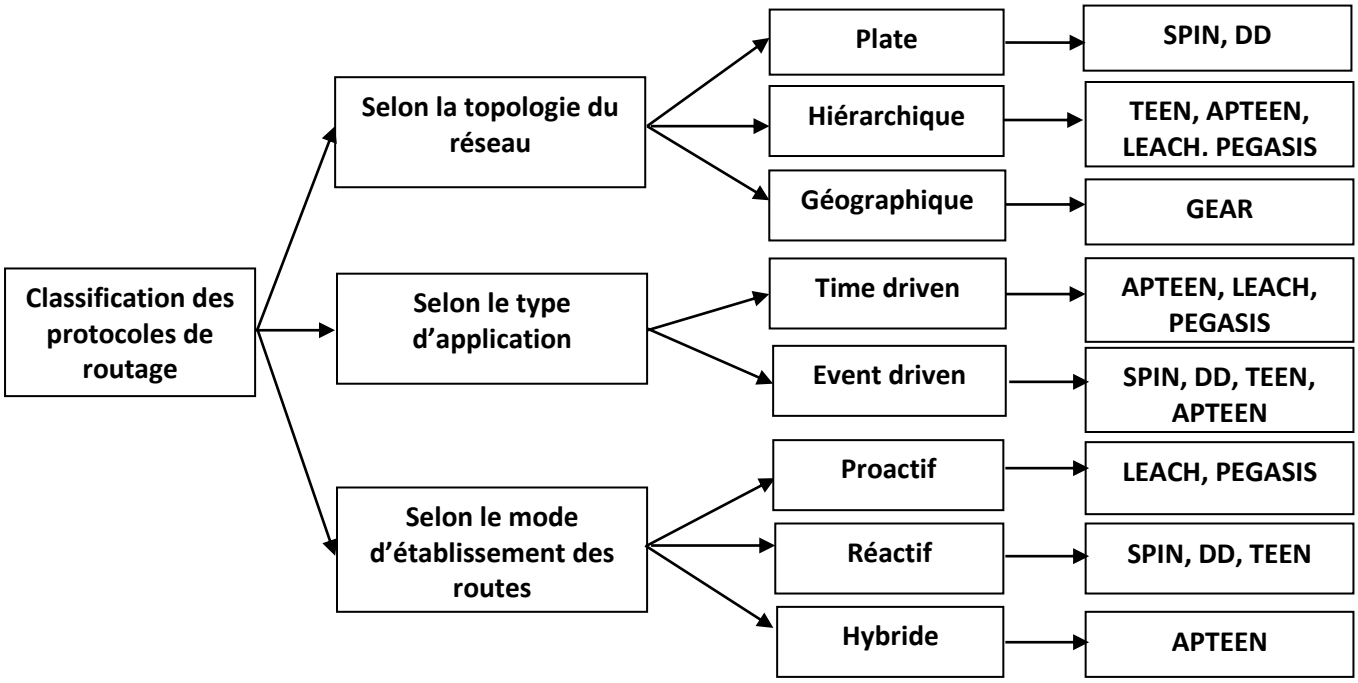

Figure I-6. Classification des protocoles de routage dans les RCSFs.

Dans ce qui suit, nous allons détailler chaque critère de classification.

6.2.1 La topologie du réseau

La topologie détermine l'organisation des capteurs dans le réseau. Il existe trois topologies dans les RCSFs : la topologie plate, la topologie hiérarchique et géographique **[MAL12]**.

Topologie plate

Un réseau de capteurs sans fil plat est un réseau homogène, où tous les noeuds sont identiques en termes de batterie et des fonctions, excepté la station de base. Dans ce type de topologie, les capteurs communiquent entre eux afin d'acheminer l'information au nœud centralisé (station de base). Ce processus d'acheminement d'information peut prendre deux formes: communiquer directement avec la station de base comme illustré dans la Figure I-7 (a), ou via un mode multi-sauts comme dans la Figure I-7 (b). Dans le type de topologie illustré par la Figure I-7 (a), tous les noeuds peuvent envoyer leurs données à la station de base en utilisant une forte puissance, ceci peut conduire à la diminution de la durée de vie du réseau.

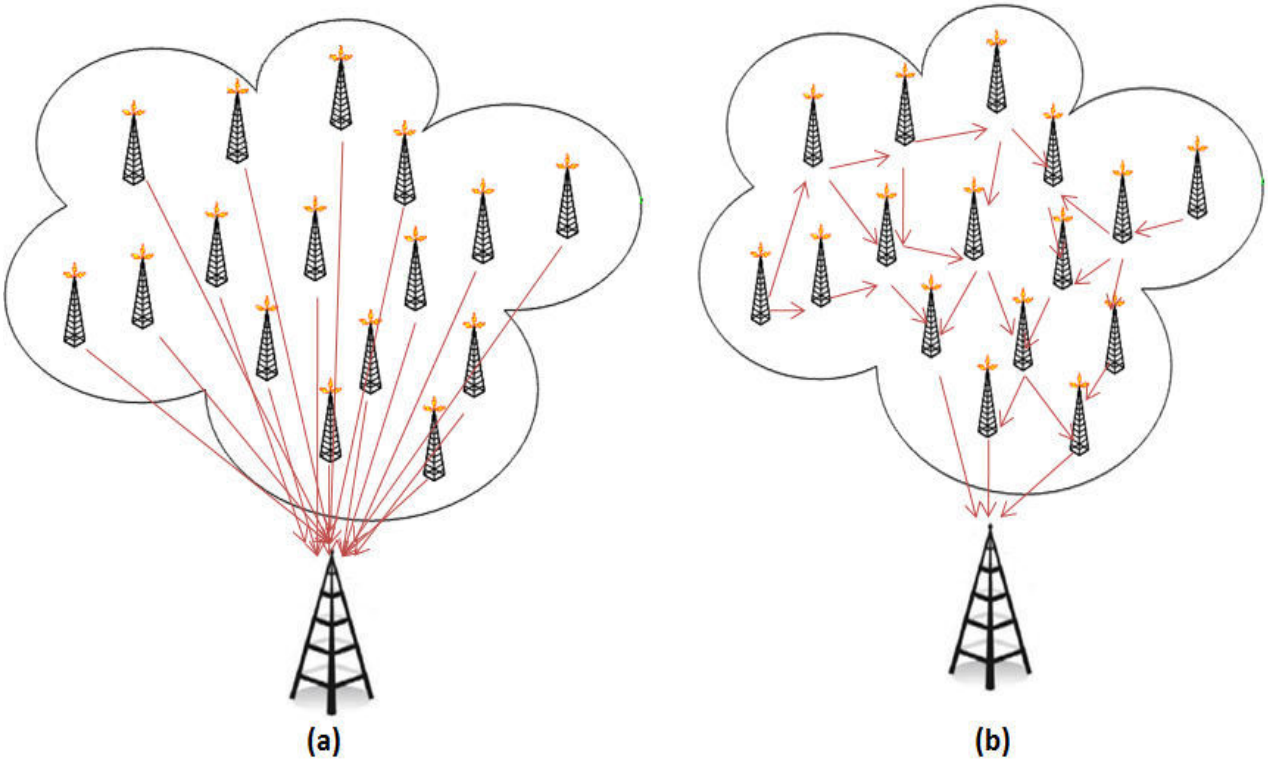

Figure I-7. Architecture de communication dans une topologie plate.

Topologie hiérarchique

Dans cette architecture, le réseau est partitionné en sous ensembles appelés clusters. Tel qu'il est illustré dans la Figure I-8, chaque cluster est constitué d'un ensemble de nœuds capteurs et d'un nœud Cluster-Head (chef de cluster). Tous les capteurs d'un cluster envoient les données à leur Cluster-Head. Ensuite, ce dernier s'en charge de les acheminer vers la station de base.

Le choix du chef de cluster se fait soit, selon le nombre de voisins en considérant comme chef de cluster le nœud ayant le maximum de voisins, soit selon le niveau d'énergie du nœud en considérant comme chef de cluster, le nœud ayant une forte batterie. Les membres du cluster se chargent de la tâche de capture, ensuite ils envoient les données captées à leur chef de cluster, ces derniers sont utilisés pour traiter, agréger et acheminer les informations vers la station de base. Avec cette méthode on assure une minimisation efficace d'énergie.

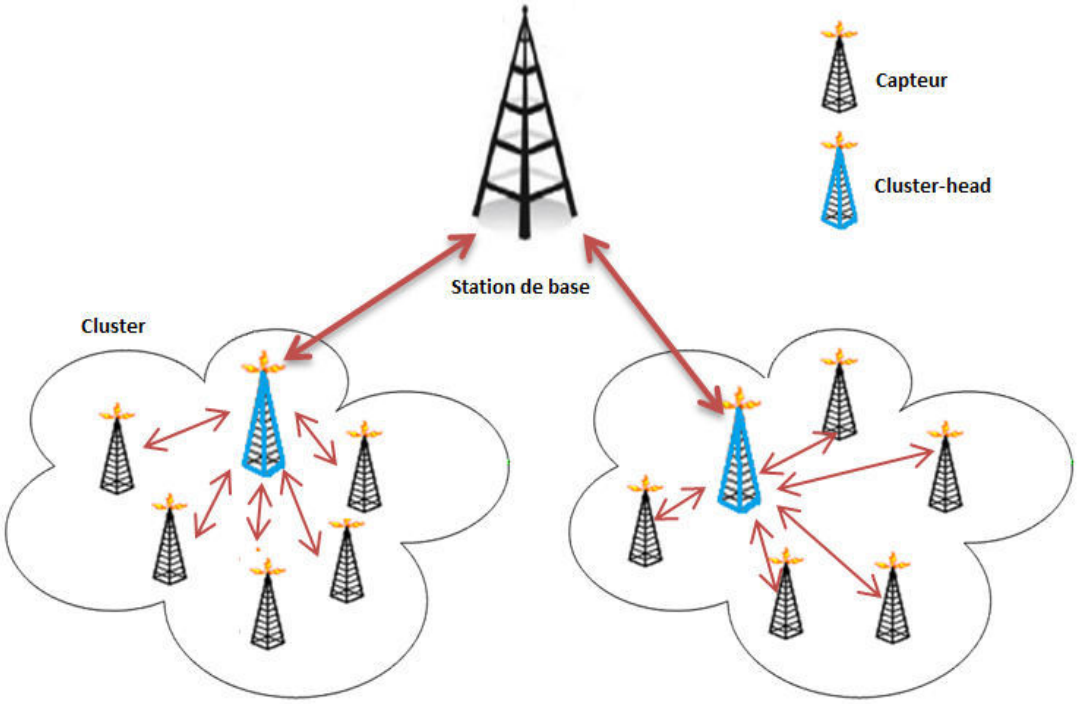

Figure I-8. Topologie à base de cluster.

Topologie géographique

Les protocoles de routage géographiques utilisent seulement les informations concernant les positions de leurs voisins directs (il n'exige pas des tables de routage), Ils n'exigent pas d'inondation. Seuls les noeuds qui se trouvent dans la zone d'acheminement désignée sont autorisés à transmettre le paquet de données. La région de transmission peut être définie par le nœud source ou par des noeuds intermédiaires pour exclure les noeuds qui peuvent provoquer la déviation des paquets de données **[RAM13]**. Par contre, chaque nœud doit connaitre sa position et chaque nœud source doit connaitre la position de la destination. Ces positions peuvent être obtenues en utilisant un dispositif dédié (GPS par exemple).

6.2.2 Le type d'application

Il est possible de distinguer deux modèles de livraison de données: time-driven et event-driven **[CHA08].**

Time-driven

Cette approche consiste à la livraison des données de façon périodique. Cet aspect permet aux capteurs de se mettre en veille pendant les périodes d'inactivité, et n'enclencher leur dispositif de capture qu'à des instants particuliers. Ainsi, la durée de vie du réseau va être allongée. Le modèle time-driven est approprié pour des applications qui nécessitent un

prélèvement périodique des données. Par exemple, cela est utile dans des applications de monitoring (feu, météo).

Event-driven

Ce modèle est généralement adopté dans les applications temps-réel où les capteurs doivent réagir immédiatement à des changements soudains des valeurs captées. Dans ce cas, le protocole de routage doit être réactif et doit donner des réponses rapides à l'occurrence d'un certain nombre d'évènements.

6.2.3 Le mode d'établissement des routes

Suivant la manière de création et le maintien des chemins pendant le routage, nous distinguons trois catégories de protocoles de routages : protocoles proactifs, réactifs et hybrides **[RAJ09]**.

Protocole proactif

Ces protocoles de routage essaient de maintenir les meilleurs chemins existants vers toutes les destinations possibles au niveau de chaque nœud du réseau. Les routes sont sauvegardées mêmes si elles ne sont pas utilisées. Chaque nœud du réseau maintient une table de routage pour toutes les destinations indépendamment de l'utilité des routes. Les protocoles proactifs sont adaptés aux applications qui nécessitent un prélèvement périodique des données. Et par conséquent, les capteurs peuvent se mettre en veille pendant les périodes d'inactivité, et n'enclencher leur dispositif de capture qu'à des instants particuliers.

Protocoles réactifs

Ces protocoles (dits aussi, les protocoles de routage à la demande) créent et maintiennent des routes selon les besoins. Lorsque le réseau a besoin d'une route, une procédure de découverte de route est lancée. Ce type de protocoles est pratique pour des applications temps réel où les capteurs doivent réagir immédiatement à des changements soudains des valeurs captées.

Protocoles hybrides

Ces protocoles combinent les deux idées des protocoles proactifs et réactifs. Ils utilisent un protocole proactif pour les nœuds voisins à deux ou à trois sauts. Au-delà de la zone du voisinage, le protocole hybride fait appel à un protocole réactif pour chercher des routes vers les autres nœuds du réseau.

6.3 Exemples de protocoles de routages dans les RCSFs

Dans ce qui suit, nous citons un ensemble de protocoles de routage répandus dans les applications des réseaux de capteurs sans fil. Nous donnons un bref aperçu sur le principe de chacun.

6.3.1 LEACH

LEACH (Low Energy Adaptive Clustering Hierarchy) est l'un des protocoles les plus populaires pour les réseaux de capteurs sans fil **[WEN00]**. Son principe est de former des zones communes de calcul et de traitements en se basant sur la puissance du signal et le niveau d'énergie des noeuds capteurs. Chaque zone est dirigée par un chef de zone, jouant le rôle de routeur vers la station de base, en effectuant des traitements sur les données reçues de son cluster et leur expédition vers la prochaine destination. Ce rôle de chef de zone est échangé entre les noeuds d'un cluster afin de répartir équitablement la consommation d'énergie entre eux.

Le protocole LEACH se déroule en rounds. Chaque round se compose de deux phases : phase d'initialisation suivie d'une phase de transmission. La phase d'initialisation consiste à élire les Clusters-Head (CHs) et définir les clusters. La phase de transmission est responsable de la transmission des données captées. Après une période de temps passée, une requête est diffusée sur le réseau par la station de base afin d'évaluer les performances des noeuds chefs de zones en terme d'énergie et procéder, le cas échéant, à son remplacement. Le protocole LEACH est conçu pour des applications time-driven et un mode de transmission proactif. **[WEN00]**. Le protocole LEACH est favorable en termes d'efficacité énergétique, puisqu'il permet la minimisation du nombre de messages circulant sur le réseau.

6.3.2 TEEN et APTEEN

TEEN (Threshold sensitive Energy Efficient sensor Network protocol) et son extension APTEEN (AdaPtive Threshold sensitive Energy Efficient sensor Network protocol) sont des protocoles de routage hiérarchiques qui ont une réactivité importante à des changements soudains de la zone à surveiller. Ils conviennent pour les applications critiques. Dans les deux protocoles, le facteur clé est la valeur de l'attribut mesuré. La caractéristique supplémentaire d'APTEEN est la capacité de changer la périodicité et les paramètres de TEEN en fonction des besoins des utilisateurs et des applications. TEEN est conçu pour être sensible à des changements soudains des attributs tels que la température. La réactivité est importante pour les applications critiques dont le réseau fonctionne dans un mode réactif. L'architecture du réseau de capteurs est basée sur un groupement hiérarchique où les nœuds forment des clusters, et ce processus va se répéter jusqu'à ce que la station de base soit atteinte **[MAN01] [MAN02].** APTEEN est une extension de TEEN qui fait à la fois la collection des captures périodique de données et qui réagit aux événements critiques. Quand la station de base forme des clusters, les Clusters-Head diffusent les attributs, les valeurs des seuils, ainsi

que le calendrier de transmission à tous les nœuds. Le Cluster-Head effectue également l'agrégation de données afin d'économiser l'énergie. Les protocoles TEEN et APTEEN répondent aux applications event-driven et time-driven en introduisant un mode de transmission réactif nécessaire aux applications critiques et un mode de transmission proactif favorable aux applications périodiques.

6.3.3 PEGASIS

PEGASIS (Power-Efficient GAthering in Sensor Information Systems) est une version améliorées du protocole LEACH. Le principe de ce protocole est d'organiser le réseau sous forme d'arbre hiérarchique où les noeuds collecteurs sont considérés comme des feuilles et la station de base comme la racine. Les données captées transitent d'une feuille à la racine par des noeuds intermédiaires formant une chaine. A la réception d'un paquet de données, le nœud intermédiaire procède à son traitement avant son expédition vers son voisin direct de la chaine. Le dernier nœud de la chaine (appelé leader) transmet les données fusionnées à la station de base. Les noeuds leader sont choisis, pour un intervalle de temps bien défini dans le but de répartir équitablement l'énergie consommée durant un round de transmission **[LIN02].** Le protocole PEGASIS est favorable en termes d'efficacité énergétique, il permet de répartir équitablement les tâches et les ressources.

6.3.4 SPIN

SPIN (Sensor Protocol for Information via Négociation) est un protocole de routage qui se base sur le principe de méta-données (une description de la donnée) lors du processus de diffusion de requêtes sur le réseau par la station de base. Le récepteur de la requête a la possibilité de choisir d'accepter la donnée ou non. Tous les noeuds du réseau sont traités comme des noeuds de destination dont la tâche principale est de recueillir un point de vue globale sur leur environnement. Egalement, SPIN s'attaque à la redondance de données transmises sur un réseau de capteurs sans fil. SPIN se fonde sur un modèle de négociation inter noeuds pour le traitement des messages reçus contenant la nature de la tâche à réaliser. Le principe est d'émettre un message ADV(ADVertise), contenant la description de la donnée à transmettre à tous les noeuds du réseau. Chaque nœud récepteur du message ADV décidera de la réception ou non de la donnée à transmettre selon sa description. Ainsi, si le nœud est intéressé par l'information, il envoi une réponse sous forme de requête REQ (REQuest) au nœud émetteur du message ADV. Par la suite, et à la réception de toutes les réponses, le nœud émetteur commence à envoyer les données aux noeuds intéressés **[KUL02].** Le protocole SPIN est favorable en termes d'efficacité énergétique, puisqu'il permet d'éviter les problèmes de l'inondation et de l'implosion.

6.3.5 Directed Diffusion (DD)

Directed Diffusion ou diffusion dirigée est un protocole de propagation de données. Son utilité réside dans sa capacité à réduire considérablement le nombre d'opérations de la

couche réseau par la création de plusieurs chemins pour le routage d'informations et ainsi permettre une économie d'énergie conséquente aux capteurs **[INT00]**. Cette réduction d'opération passe par une mise en place de quatre éléments : i) la nomination des données (décrire les intérêts), ii) propagation des intérêts, iii) propagation des données, iv) renforcement des chemins de transport de données. Le protocole DD à travers ces quatre éléments définit au préalable la nature de la donnée à capter, décrit cette donnée sous forme d'intérêt et diffuse cette requête sur le réseau. Les noeuds concernés commencent à capter et propager les données sur des chemins différents. A la réception des premières données par la station de base, elle procède au choix puis au renforcement des meilleurs chemins. La station de base définit une requête d'intérêt sur une information particulière à vouloir récupérer sur l'environnement de déploiement des capteurs. Ensuite, la requête sera diffusée par une inondation globale du réseau et les noeuds concernés commencent à récolter les informations décrites par cette requête. Les données récoltées seront transmises vers la station de base. La station de base, et après avoir reçue les premières bonnes informations, renforce les chemins vers les noeuds émetteurs. Le protocole DD se caractérise par une réduction du nombre d'opérations de la couche réseau, il est donc économe en termes d'énergie.

6.3.6 GEAR

Le protocole GEAR (Geographic and Energy Aware Routing) est un protocole de routage basé sur les informations géographiques des noeuds capteurs. Les informations géographiques sont utiles pour restreindre le champ de diffusion des requêtes d'intérêts et ne prendre en considération que les régions ciblées **[YAN01]**.

Le Tableau I-1 est un tableau récapitulatif des protocoles de routage cités précédemment **[DEV13].**

Protocole de routage	Classification	Auteurs et années	Caractéristiques
SPIN (*Sensor Protocols for Information via Negotiation*)	- Protocole plat - Réactif - Event-driven	J. Kulik et al ,2002 [KUL02]	Le protocole SPIN est favorable en termes d'efficacité énergétique, puisqu'il permet d'éviter les problèmes de l'inondation et de l'implosion.
DD (*Directed Diffusion*)	- Protocole plat - Réactif - Event-driven	Intanagonwiwat., C et al. 2000 [INT00]	Le protocole DD se caractérise par une réduction du nombre d'opérations de la couche réseau, il est donc économe en termes d'énergie.
TEEN(*Thresh old sensitive Energy Efficient sensor Network*)	- Protocole hiérarchique - Réactif - Event-driven	A.Manjeshwar and D.P Agrawal,2001 [MAN01]	TEEN est conçu pour être sensible à des changements soudains des attributs dans la zone de couverture des capteurs.

APTEEN (*Adaptive Periodic Thresholdsensitive Energy Efficient sensor Network Protocol*)	- Protocole hiérarchique - Hybride - Time-driven & event-driven	A. Manjeshwar et D. P. Agrawal, 2002 [MAN02]	APTEEN est une extension de TEEN qui fait à la fois la collection des captures périodique de données et qui réagit aux événements critiques
LEACH (*Low Energy Adaptive Clustering Hierarchy*)	- Protocole hiérarchique - Proactif - Time-driven	Wendy, 2000 [WEN00], [WEN02]	Le protocole LEACH est favorable en termes d'efficacité énergétique, puisqu'il permet la minimisation du nombre de messages circulant sur le réseau.
PEGASIS (*Power Efficient Gathering In Sensor Information System*)	- Protocole hiérarchique - Proactif - Time-driven	Lindsey and Raghavendra 2002. [LIN02]	Le protocole PEGASIS est favorable en termes d'efficacité énergétique, il permet de répartir équitablement les tâches et les ressources.
GEAR (*Geographic and Energy Aware Routing*)	- Protocole géographique -Basé localisation	Yan Yu, Ramesh Govindan, Deborah Estrin , 2001 [YAN01]	GEAR découpe le réseau en région. Il sélectionne le nœud voisin pour l'acheminement de donnée en tenant compte de son énergie résiduelle. Il utilise un algorithme récursif pour la diffusion des paquets vers les régions destinatrices.

Tableau I-1. Tableau récapitulatif de quelques protocoles de routage.

Conclusion

Dans ce chapitre, nous avons présenté les réseaux de capteurs sans fil, leur architecture, leurs caractéristiques ainsi que leurs domaines d'applications. Nous avons vu qu'ils présentent un intérêt considérable et réalisent une nouvelle étape dans l'évolution des technologies de l'information et de la communication. La conception de réseaux de capteurs autonomes, reliés par des liens sans fil, est un domaine de recherche très actif qui suscite un intérêt croissant, vu la diversité de ses applications : santé, environnement, industrie, militaire. La plupart de ces applications ont besoin d'un niveau élevé de sécurité. Dans le prochain chapitre nous aborderons l'aspect de sécurité dans les réseaux de capteurs sans fil.

CHAPITRE 2

La sécurité dans les réseaux de capteurs sans fil

Introduction

Les différents domaines d'applications des réseaux de capteurs sans fil (militaires, médical,…) ont souvent besoin d'un niveau de sécurité élevé. Or, de part des caractéristiques des RCSFs (contrainte d'énergie, communication sans fil, topologie dynamique, nombre important de capteurs, sécurité physique limitée, capacités réduites des nœuds), la sécurisation des réseaux de capteurs est à la source, aujourd'hui, de beaucoup de défis scientifiques et techniques.

Dans ce chapitre, nous donnons un aperçu sur la sécurité dans les réseaux de capteurs sans fil. Premièrement, nous présentons les vulnérabilités de la sécurité dans les RCSFs. Ensuite, nous présentons les menaces qui peuvent les contrer, listons après les objectifs et les besoins de la sécurité requis par les protocoles, et les mécanismes utilisés pour combattre ces menaces. Nous discutons à la fin du chapitre de la sécurité du routage dans de tels réseaux.

1. Vulnérabilités de la sécurité dans les RCSFs

Un réseau de capteurs sans fil est un réseau qui a de nombreuses contraintes par rapport à un réseau informatique traditionnel. En raison de ces contraintes, il est difficile d'employer directement les approches de sécurité existantes sur les RCSFs. Par conséquent, pour développer des mécanismes de sécurité utiles tout en empruntant les idées des techniques de sécurité actuelles, il est tout d'abord nécessaire de connaître et de comprendre ces contraintes en premier **[WAL06].**

1.1 Ressources limitées

Toutes les approches de sécurité nécessitent une certaine quantité de ressources pour la mise en œuvre. Toutefois, la ressource énergétique, la mémoire et l'espace de stockage sont limitées dans un micro capteur.

- **Energie limitée :** dans la plupart des cas, une fois que les nœuds capteurs sont déployés, ils ne peuvent pas être facilement remplacés. Par conséquent, le niveau de la batterie pris avec eux sur le terrain doit être conservé pour prolonger la durée de vie du nœud capteur ainsi que le réseau de capteurs en entier, et l'impact de l'énergie du code de sécurité ajoutée doit être considéré. Lors de l'ajout d'un mécanisme de sécurité à un nœud capteur, on doit s'intéresser à l'impact que la sécurité a sur la durée de vie d'un capteur (la batterie). La puissance supplémentaire consommée par les nœuds capteurs en raison de la sécurité est liée au traitement des fonctions de sécurité (par exemple, le cryptage, le décryptage, la signature de données), et l'énergie requise pour stocker les paramètres de sécurité de manière sécurisée (par exemple, le stockage des clés cryptographiques) **[YOU03]**.

- **Mémoire et espace de stockage limités :** un capteur est un petit dispositif doté d'une petite mémoire et d'un peu d'espace de stockage pour le code. Afin de construire un mécanisme de sécurité efficace, il est nécessaire de limiter la taille du code de l'algorithme de sécurité **[WAL06].**

Le Tableau II-1 résume les caractéristiques physiques limitées de la majorité des noeuds capteurs disponibles sur le marché **[GIL08]**.

Capteur	Circuit radio	CPU	RAM	Mémoire de stockage
MICA2	CC1000	ATMega128	4 KB	128 à 512 KB
MICAZ	CC2420	ATMega128	4 KB	128 à 512 KB
TelosA	CC2420	TI MSP 430	2 KB	60 -512 KB
TelosB	CC2420	TI MSP 430	10 KB	48 KB - 1 MB
BTnode3	CC1000/Bluetooth	ATMega128	64 KB	128 -180 KB
XYZ	CC2420	ARM 7	32 KB	256 KB
Shimmer	CC2420/Bluetooth	TI MSP 430	10 KB	48 KB - Up to 2 GB
Cricket	CC1000	AT Mega128	4 KB	128 -512 KB
Imot2	CC2420	Intel PXA271	256 KB	32 MB

Tableau II-1. Caractéristiques physiques des nœuds capteurs disponibles sur le marché.

1.2 Communication sans fil

Le milieu sans fil est ouvert et accessible à tout le monde. Par conséquent, toute transmission peut facilement être altérée, ou retransmise par un attaquant. Le transfert de données peut être non fiable à cause de la nature du canal sans fil. Même si le transfert de données est fiable, on peut avoir des conflits de collisions dû à la nature de diffusion des RCSFs **[WAL06].**

- **Transfert non fiable :** les paquets routés dans un réseau de capteurs sans fils peuvent être endommagés en raison d'erreurs du canal de communication sans fil. Par conséquent, paquets manquants ou bien, le résultat sera perdu. Un taux très élevé d'erreur du canal force également le développeur du logiciel à consacrer des ressources pour la gestion des erreurs. Si le protocole de communication néglige la manipulation d'erreur appropriée, il est possible de perdre la sécurité critique des paquets.

- **Conflits :** même si le canal est fiable, la communication peut encore ne pas être fiable. Cela est dû à la nature de la diffusion du réseau de capteurs sans fils. Si des paquets se rencontrent au milieu du transfert, des conflits vont se produire et le transfert lui-

même échouera. Dans un réseau de capteurs à haute densité, cela peut être un problème majeur.

1.3 Fonctionnement sans surveillance

Dans la majorité des domaines d'application des réseaux de capteurs, les noeuds capteurs peuvent être laissés sans surveillance pendant de longues périodes de temps. Par conséquent, les RCSFs sont gérés à distance et sans surveillance, ce qui les exposent aux agressions physiques.

- **L'exposition à des agressions physiques :** le capteur peut être déployé dans un environnement ouvert à des adversaires, mauvais temps. La probabilité qu'un capteur subit une attaque physique dans un tel environnement est donc beaucoup plus élevée. Ainsi, les capteurs peuvent être capturés, compromis ou détruits par des attaquants. Un attaquant peut prendre le contrôle d'un nœud dans le réseau après le déploiement, ce qui permet de l'endommager physiquement, le rendant ainsi non fonctionnel. L'attaquant est capable de modifier les informations captées par le nœud capteur, ce qui lui permet d'effectuer une variété d'attaques. De plus, des informations vitales pour la sécurité du réseau (tel que la table de routage, des données et des clés cryptographiques), peuvent être extraites du nœud capturé.

- **Gestion à distance :** la gestion à distance d'un réseau de capteurs rend pratiquement impossible de détecter les intrusions et les problèmes d'entretien physique (par exemple, le remplacement de la batterie, panne des capteurs). Peut-être l'exemple le plus extrême est un nœud capteur utilisé pour des missions de reconnaissance à distance derrière les lignes ennemies. Dans ce cas, le nœud n'aura aucun contact physique avec le gestionnaire du réseau une fois déployé **[WAL06].**

1.4 Absence d'infrastructure fixe

L'absence de toute infrastructure fixe augmente la vulnérabilité des réseaux de capteurs. En effet, il n'existe pas de contrôleur central pour surveiller le fonctionnement du réseau et identifier les tentatives d'intrusions. La plupart des réseaux de capteurs ont une station de base désignée, néanmoins son rôle est limité à la collecte des données et la distribution de requêtes, et ne comprend aucune forme de surveillance.

1.5 Agrégation des données

La technique d'agrégation des données permet de réduire la consommation énergétique en réduisant la quantité de données transférées vers la station de base en éliminant les données redondantes est inutiles. L'agrégation des données est l'une des techniques d'optimisation de la durée de vie des RCSFs. Toutefois, cela exige que les noeuds

intermédiaires accèdent aux données échangées pour effectuer le traitement d'agrégation de données. Conséquemment, la confidentialité des données est non respectée, ce qui pose un autre défi pour les mécanismes de sécurité **[CAS06]**.

1.6 Le déploiement aléatoire et l'utilisation à grande échelle

La capacité d'être déployé dans des grandes surfaces avec un nombre important de noeuds capteurs est l'une des caractéristiques les plus intéressantes des RCSFs. Le déploiement est fait sans aucune connaissance préalable de la position des noeuds. L'environnement du déploiement est généralement dynamique avec des topologies qui changent fréquemment. Par conséquent, les réseaux de capteurs nécessitent des mécanismes de sécurité plus robustes pour faire face à l'instabilité de l'environnement. De plus, ces mécanismes devraient être adaptés, de telle sorte que le grand nombre de noeuds n'affectera pas leur efficacité.

2. Les menaces contre les RCSFs

Une menace est définie comme l'arrivée potentielle d'événements qui peuvent causer des pertes. Les menaces qui peuvent affecter la sécurité des RCSFs sont divisées en deux catégories : les mauvais comportements et les attaques **[KAR03]**.

2.1 Les mauvais comportements

On définit un mauvais comportement comme un acte non autorisé d'un nœud interne qui peut entraîner involontairement des dommages à d'autres noeuds. C'est-à-dire, ce nœud a d'autres objectifs que de lancer une attaque**.** Par exemple, un nœud refuse de transférer les paquets vers les autres noeuds pour préserver ses ressources **[ZIO02]**.

2.2 Les attaques

Une attaque est un ensemble de techniques informatiques, visant à causer des dommages à un réseau, en exploitant les failles de celui-ci. Les attaques peuvent aggraver les problèmes de sécurité. En effet, les conséquences liées à ces attaques peuvent varier d'une simple écoute du trafic jusqu'à l'arrêt total du réseau selon les capacités des attaquants. Pour les combattre, il est nécessaire de connaître les classes d'attaques afin de mettre en œuvre des solutions optimales. Les attaques sont classifiées selon les deux catégories ci-dessous **[KAR03]**.

2.2.1 Attaque passive

Elle est déclenchée lorsqu'un nœud non-autorisé, obtient un accès à une ressource sans modifier les données ou perturber le fonctionnement du réseau. Cette attaque consiste

seulement à écouter les informations circulant sur le médium. Elle est facilement réalisable si les messages circulant sur le réseau sont en clair. Par ailleurs, elle est difficile à détecter, car comme elle est passive, elle ne modifie pas l'activité du réseau. Mais une fois l'attaquant ayant acquis suffisamment d'informations, il peut produire un attentat contre le réseau, ce qui transforme l'attaque passive en une attaque active.

2.2.2 Attaque active

Elle est déclenchée lorsqu'un nœud non autorisé obtient un accès à une ressource en apportant des modifications aux données ou en perturbant le bon fonctionnement du réseau. Nous présentons dans ce qui suit les attaques actives les plus connues dans les RCSFs **[MAR04].**

- **Jamming (le brouillage radio) :** un attaquant va envoyer des ondes sur la même fréquence que le réseau de capteurs sans fil. Ainsi les noeuds ne pourront plus communiquer car le médium est saturé par ce brouillage radio.

- **Wormhole attack (l'attaque du trou de ver) :** l'attaque du trou de ver nécessite l'insertion d'au moins deux noeuds malicieux. Ces deux noeuds sont reliés entre eux par une connexion puissante. 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. 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 va tromper les autres noeuds sur les distances réelles, 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 II-1, où deux noeuds malicieux X_1 et X_2, 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.

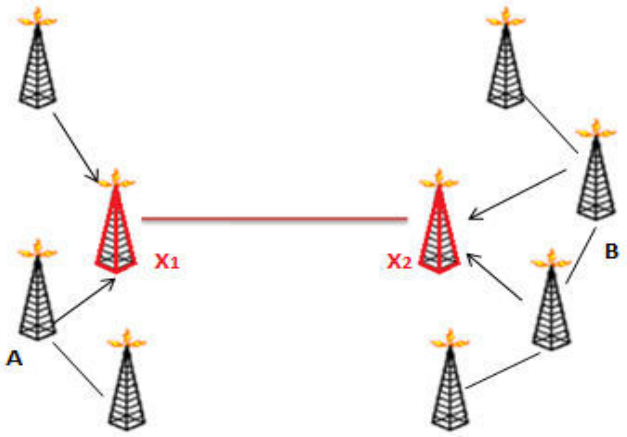

Figure II-1. Exemple d'une attaque de type trou de ver.

- **Sinkhole attack (l'attaque du trou de la base) :** dans cette attaque, un nœud malicieux va s'attaquer directement à l'information circulant par la station de base, qui est le plus souvent le point qui recueille le plus d'informations de l'intégralité du réseau. Pour cela, le nœud malicieux va proposer aux noeuds le chemin le plus rapide pour atteindre la station de base, en utilisant une connexion plus puissante. Ainsi l'ensemble de ces noeuds vont s'adresser en particulier à ce nœud malicieux pour transmettre l'information à la base. Toutes les informations qui transitent de ces nœuds vers la base pourront être récupérées par l'attaquant. Pour générer une attaque encore plus puissante, un attaquant peut utiliser des stratégies de type trou de ver associées à une attaque de type trou de la base. Le but sera avec ces trous de ver de couvrir tous les noeuds du réseau. Cette situation est représentée dans la Figure II-2, où les noeuds malicieux X_1, X_2 et X_3 sont reliés par des connexions puissantes et forment des trous de ver. X3 est relié à la station de base par une connexion puissante pour réaliser une attaque du trou de la base. On parle alors d'une sphère d'influence exercée par l'attaquant sur le réseau, car il est ainsi capable de récupérer l'intégralité des informations qui circulent dans le réseau de capteurs sans fil **[KAR03]**.

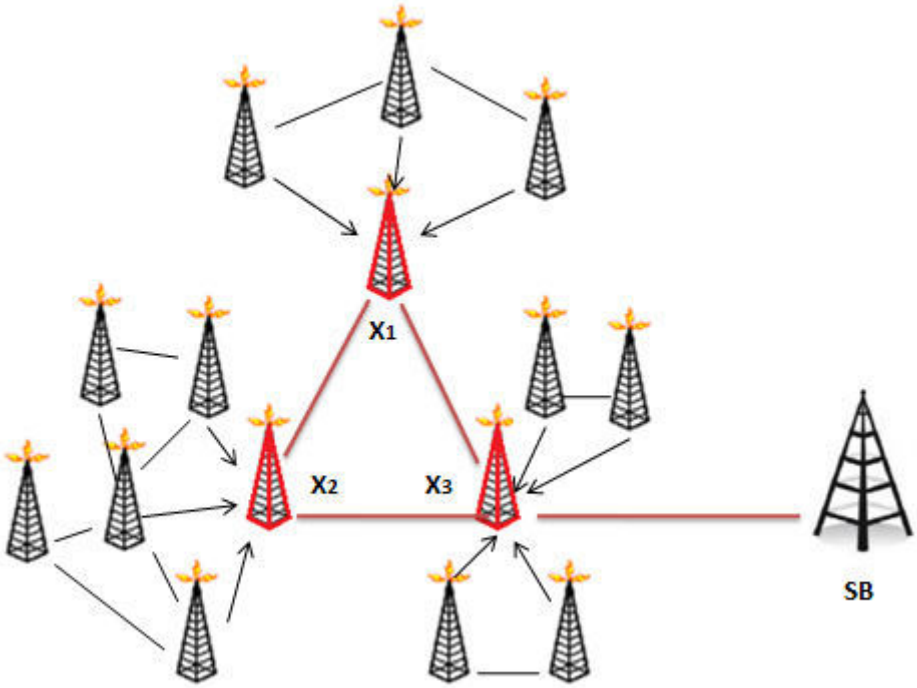

Figure II-2. Exemple d'utilisation d'attaques de type trou de ver pour réaliser une attaque de type trou de la base.

- **Hello Flooding :** les protocoles de découvertes sur les réseaux ad-hoc utilisent ce qu'on appelle des messages de type HELLO pour s'insérer dans un réseau et pour découvrir ses noeuds voisins. Dans une attaque dite HELLO Flooding, un attaquant va utiliser ce mécanisme pour saturer le réseau et consommer son énergie. Un exemple, représenté par la Figure II-3, d'un nœud malicieux avec une connexion puissante qui

lui permet d'envoyer à un grand nombre de noeuds des messages de type HELLO, de manière continue. Les noeuds voisins V vont alors essayer de lui répondre, même s'ils sont situés à des distances qui ne permettent pas d'atteindre le nœud malicieux. A force de tenter de répondre à ces messages ils vont petit à petit consommer l'intégralité de leur énergie **[KAR03].**

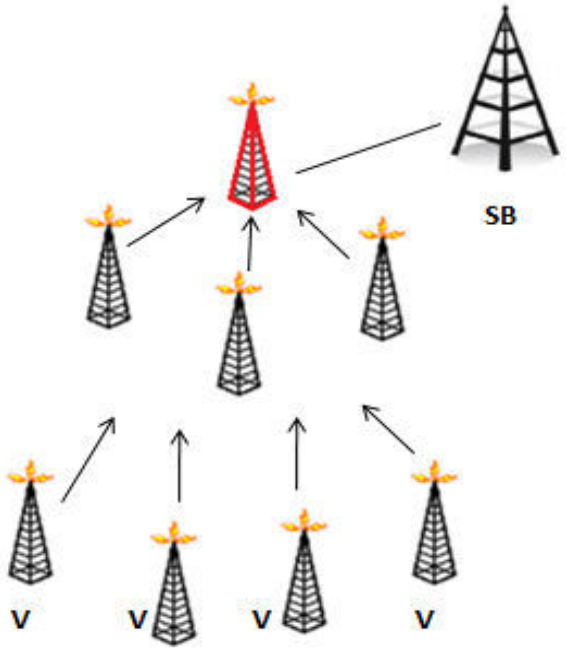

Figure II-3. Attaque de type HELLO Flooding.

- **Sybil attack (l'attaque Sybille) :** dans certains algorithmes, la fiabilité du routage est implémentée par l'établissement d'une redondance de chemins. Un attaquant peut altérer ce genre de systèmes en "reconnaîtrant" plusieurs identités, ce qui permet de créer plusieurs routes passant par le nœud malicieux, qui ne sont en réalité qu'un seul chemin.

- **Blackhole attack (l'attaque du trou noir) :** l'attaque du trou noir consiste tout d'abord à insérer un nœud malicieux dans le réseau. Ce nœud, par divers moyens, va obliger le maximum de noeuds voisins à faire passer l'information par lui. Ensuite comme un trou noir dans l'espace, toutes les informations qui vont passé en son sein ne seront jamais retransmises. La figure II-4 représente un exemple de trou noir, mis en place par un nœud malicieux X qui a modifié le routage pour que les clusters 1, 2, 3 fassent passer l'information par lui pour communiquer entre clusters. Dans ce cas de figure, le trou noir X ne retransmettra aucune information, empêchant toute communication entre les différents clusters.

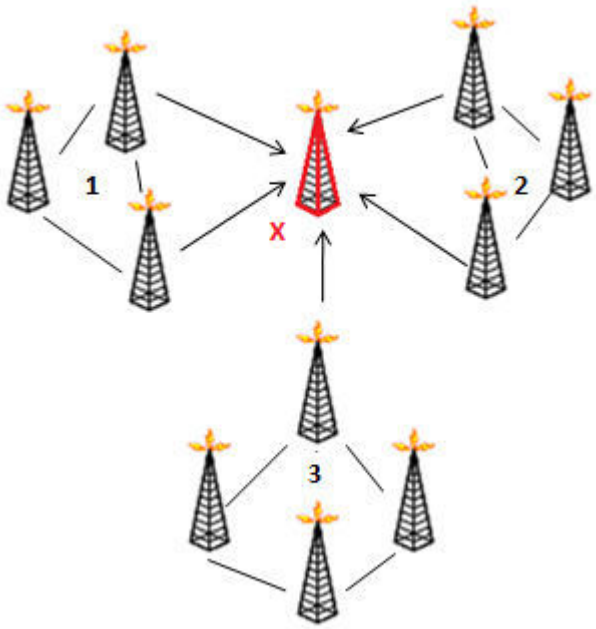

Figure II-4. Exemple de trou noir dans un réseau clustérisé.

3. Objectifs et exigences de sécurité

Comme évoqué précédemment, les RCSFs nécessitent dans de nombreuses applications des solutions qui assurent la sécurité des informations circulant sur le réseau. Ce type de réseau pose également des conditions uniques de ses propres caractéristiques. Par conséquent, un protocole de sécurité pour un RCSF, doit satisfaire une ou plusieurs conditions de sécurité **[WAL06]**, à savoir :

Confidentialité des données

Ce service désigne la garantie que l'information n'a pas été divulguée, et que les données ne sont compréhensibles que par les entités qui partagent le même secret. Le mécanisme utilisé pour sécuriser le transfert des données est le chiffrement.

Intégrité des données

L'intégrité de données garantit que les données reçues n'ont pas été altérées durant leur transit dans le réseau de manière volontaire ou accidentelle. En fait, on veut éviter qu'un intrus puisse modifier cette information pour en tirer certains avantages.

L'authentification

L'authentification consiste à vérifier l'identité authentique des noeuds. En effet, on ne peut pas assurer la confidentialité et l'intégrité des messages échangés si, dès le départ, on

n'est pas sûr de communiquer avec le bon nœud. Si l'authentification est mal gérée, un attaquant peut se joindre au réseau et injecter des messages erronés en raison de la nature du support sans fil et la nature des réseaux de capteurs (déployés dans des zones hostiles et sans surveillance) **[KRA97].**

Fraîcheur des données

Même si la confidentialité et l'intégrité des données sont assurées, nous devons également assurer la fraîcheur de chaque message. La fraîcheur de données suggère que les données soient récentes, et elles s'assurent qu'aucun vieux message n'a été répété. Sans mécanisme de sécurité vérifiant que les données transmises sont récentes, un attaquant pourrait capturer des données circulant sur le réseau à un temps *T*, puis les retransmettre à un temps *T+1* pour tromper le réseau et faire circuler de fausses informations. On peut prendre l'exemple d'un réseau de capteurs censé détecter les incendies, qui détecterait une première fois un incendie réel. L'attaquant enregistrait les informations envoyées lors de cet événement. Il pourrait alors plus tard renvoyer ces mêmes données pour déclencher une fausse alerte. Pour résoudre ce problème, un numéro de séquence peut être ajouté aux paquets de données pour filtrer les anciens messages.

Le contrôle d'accès

C'est la capacité des noeuds du réseau (ou bien de la station de base) d'empêcher des éléments externes d'accéder au réseau, et cela en attribuant aux participants légaux des droits d'accès afin de distinguer les messages provenant des sources internes du réseau de ceux externes **[GAR06].**

La non répudiation

La répudiation est signalée dans deux cas différents: dans le premier cas, le nœud récepteur affirme que les données n'ont jamais été reçues, même si elles ont été correctement reçues. Par contre, dans le deuxième cas c'est le nœud expéditeur qui affirme qu'il n'a jamais envoyé les données, même si le message a été correctement délivré au destinataire. Un système de sécurité doit interdire la répudiation afin d'améliorer la traçabilité des messages dans le réseau **[WAL06].**

La disponibilité

Ce service représente la propriété d'un système d'être accessible par une entité autorisée dans les limites spécifiées. La disponibilité reste difficile à assurer dans les RCSFs. En effet, un nœud peut ne pas servir des informations afin de ne pas épuiser ses ressources d'énergie, de mémoire et de calcul **[BAL02]**.

4. Mécanismes de sécurité

Pour contrer les attaques qui menacent les réseaux de capteurs sans fil, plusieurs équipes de recherche tentent de trouver des solutions appropriées. Ces solutions doivent bien sûr prendre en compte les spécificités des réseaux de capteurs sans fil. Il faut donc trouver des solutions simples qui permettent de sécuriser le réseau tout en consommant le moins d'énergie possible et adapter ces solutions à une puissance de calcul faible. Dans ce qui suit, nous présentons quelques mécanismes de sécurité **[MAR04].**

4.1 Génération de clés

Une solution proposée par **[MAK07]** consiste à utiliser une clé de génération. A chaque période ou génération, la station de base envoie une nouvelle clé à l'ensemble du réseau. Cette clé sert de certificat à chacun des noeuds, pour prouver son appartenance au réseau. Si un nœud non identifié tente de rentrer dans le réseau de capteurs sans fil et qu'il ne possède pas cette clé de génération, il ne pourra être accepté en son sein. Un autre intérêt de cette technique est qu'elle permet de limiter les attaques de substitution d'un capteur et de sa reprogrammation pour être réinjecté dans le réseau. Si ce nœud est subtilisé à l'instant *0* avec la clé de génération *K(0)*, le temps qu'un attaquant le reprogramme pour le remettre dans le réseau il se sera écoulé un temps *"x"*. Quand le capteur sera repositionné dans le réseau, la nouvelle clé de génération sera alors *K(x).* Le nœud malicieux demandera à ses noeuds voisins de rentrer dans le réseau avec la clé *K(0)* et non pas *K(x),* car il n'a pas pu recevoir la nouvelle clé. Comme *K(0) != K(x)*, les noeuds voisins n'accepterons pas sa requête et le nœud malicieux ne pourra pas rentrer dans le réseau.

4.2 La cryptographie

Le mot *« cryptographie »* est composé des mots grecs: *« crypto »* qui signifie caché, et *« graphy »* qui signifie écrire. C'est donc l'art de l'écriture secrète **[BET08].** La cryptographie est l'étude des techniques mathématiques qui permettent d'assurer certains services de sécurité. Elle est définie comme étant une science permettant de convertir des informations *« en clair »* en informations cryptées *« codées »,* c'est à dire non compréhensibles, et puis, à partir de ces informations cryptées, de restituer les informations originales. Afin de protéger un message, on lui applique une transformation qui le rend incompréhensible, c'est ce qu'on appelle le chiffrement, qui, à partir d'un texte en clair, donne un texte chiffré ou cryptogramme. Inversement, le déchiffrement est l'action qui permet de reconstruire le texte en clair à partir du texte chiffré. Les solutions de la cryptographie sont réparties en deux types : cryptographie symétrique et cryptographie asymétrique, appelée couramment cryptographie à clé publique **[AND04].**

4.2.1 Cryptographie à clé symétrique

Le principe de la cryptographie symétrique se base sur le partage d'une même clé *K* de chiffrement entre deux entités qui veulent échanger un message. Si un nœud émetteur souhaite communiquer un message à un nœud destinataire, l'émetteur va chiffrer son message *M* avec la clé *K*, puis transmettra son message crypté M_K sur le réseau au destinataire. Ce message ne peut pas être déchiffré sans la clé *K*, ce qui garantit sa confidentialité. Quand le destinataire reçoit le message M_K, il déchiffre le message *M* avec la même clé *K* de chiffrement. La cryptographie à clé symétrique est illustrée par la Figure II-5.

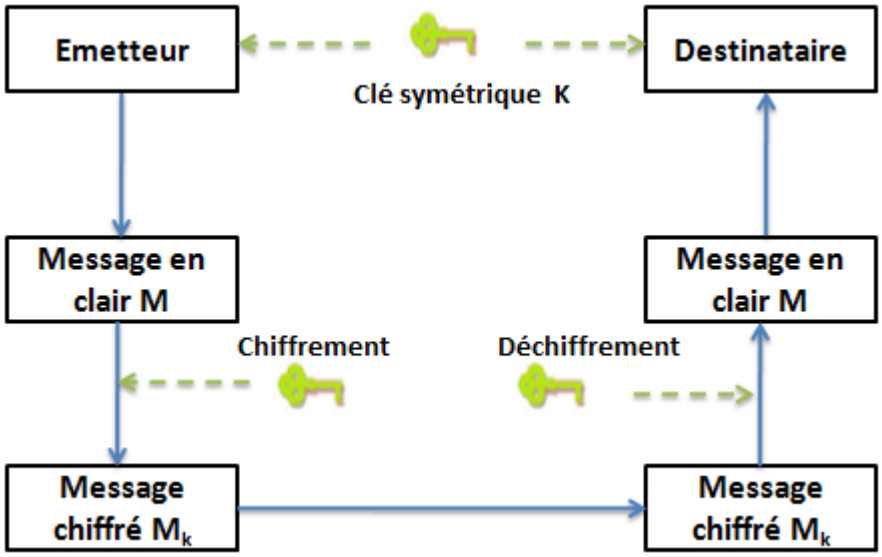

Figure II-5. Chiffrement symétrique.

4.2.2 Cryptographie à clé publique

La cryptographie à clé publique se base sur l'utilisation de deux clés, ainsi, Deux clés sont générées par le récepteur du message : une clé privé gardée sécrète, et une clé publique diffusée à tous les noeuds émetteurs pour permettre aux noeuds souhaitant lui communiquer des informations de chiffrer leur message avec la clé publique. La cryptographie à clé asymétrique est illustrée par la Figure II-6. La clé publique sert au chiffrement d'un message et la clé privée sert au déchiffrement du même message. La puissance de ce mode de chiffrement repose sur l'impossibilité de déduire la clé privée à partir de la clé publique. La cryptographie à clé publique est fondée sur le principe de fonction à sens unique. Ainsi un message crypté avec la clé publique, par l'émetteur ne peut être déchiffré avec cette même clé. Elle n'est déchiffrable que par la clé privée du récepteur, qui garantit ainsi à l'émetteur du message chiffré avec la clé publique du récepteur, que seul le récepteur qui a généré cette clé publique peut déchiffrer son message.

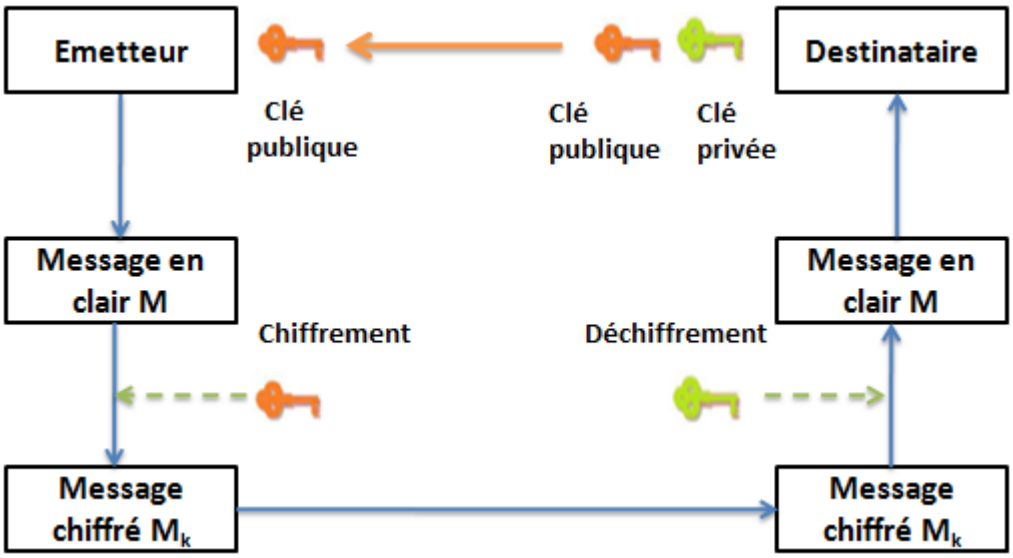

Figure II-6. Chiffrement asymétrique.

4.3 Le partitionnement des données

Les auteurs dans **[ABD07]** et **[DEN05]** offrent une solution pour empêcher la récupération d'information dans les réseaux de capteurs sans fil par le partitionnement des données. Comme son nom l'indique le but est de découper l'information en plusieurs parties. Si un capteur cherche à envoyer une information, celui-ci va la découper en plusieurs paquets de taille fixe. Chaque paquet sera ensuite envoyé sur des chemins différents. Ces paquets seront finalement reçus par la station de base, qui pourra ensuite les rassembler pour pouvoir reproduire l'information. Ce mécanisme oblige un attaquant à récupérer l'ensemble des paquets s'il veut pouvoir lire l'information. Il doit aussi être capable d'écouter l'ensemble du réseau, pour récupérer les différents paquets qui circulent sur des chemins différents, ce qui est difficile voire impossible, vu la taille des réseaux et le nombre important de chemins possibles entre chaque paire de nœuds. Un exemple de cette solution est représentée par la Figure II-7, où un capteur A divise un message en 3 paquets qui vont suivre respectivement 3 chemins différents. Cependant cette solution augmente considérablement la consommation d'énergie (avec un risque de surcharge de traitement), car elle sollicite un nombre important de nœuds.

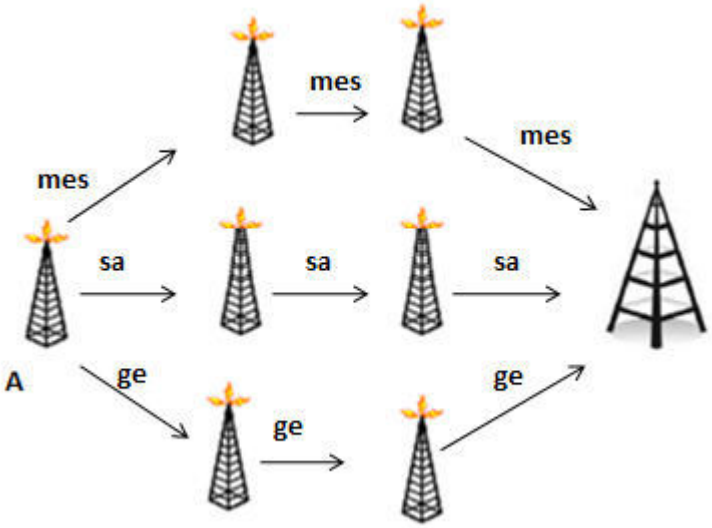

Figure II-7. Routage par partitionnement des données.

4.4 L'indice de confiance et la réputation

Une solution proposée par **[ZHA03]**, **[KIM04]** consiste à utiliser les mécanismes de confiance et de réputation. Dans les réseaux de capteurs sans fil, il est difficile de savoir, au vu du nombre de noeuds, quel nœud peut être un nœud malicieux. Pour le détecter et conserver l'intégrité du réseau, chaque nœud du réseau va surveiller ses noeuds voisins et leurs actions au cours du temps. En fonction des actions réalisées par ses noeuds voisins, un nœud va augmenter une note de l'indice de confiance de ces noeuds, basée sur sa réputation. Si un nœud ne répond jamais à une requête, son indice de confiance va diminuer, de la même manière que si ce nœud retransmet toujours correctement l'information qu'on lui a demandé de transmettre, son indice de confiance va augmenter. A l'aide de ces indices de confiance, un nœud va alors choisir le routage le plus adapté pour transmettre son information. Contrairement à des protocoles classiques de routage où le nœud chercherait le chemin le plus rapide en nombre de sauts ou de distance géographique, il va choisir ici de transmettre son information via les noeuds avec les indices de confiance les plus élevés, en d'autre terme, la route qui lui semble la plus sûre.

Ce mécanisme est représenté par la Figure II-8, où un nœud A doit transmettre une information à un nœud D. Au lieu de passer par le chemin le plus court qui passe par X, qui est un nœud avec un indice de confiance faible de 3 (sur une note de 10), et donc est potentiellement un nœud à risque, le nœud A va transmettre l'information par les nœuds B et C qui sont avec des indices de confiance de 8 et 9, et donc proposent le chemin le plus sûr. Les solutions basées sur l'indice de confiance sont peu coûteuses en termes d'énergie et permettent, selon le type de sécurité voulu, de ne pas avoir recours à la cryptographie. Cependant pour des réseaux qui demandent une sécurité maximale, elles ne sont pas toujours adaptées. Ainsi un nœud malicieux qui enregistrerait des informations sur le réseau et, par ailleurs, se comporterait de manière normale, est difficilement détectable.

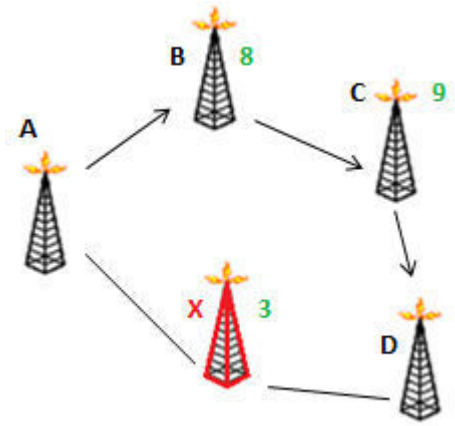

Figure II-8. Choix de routage par réputation.

4.5 Chien de garde

Le mécanisme du chien de garde ou watchdog introduit par **[GAR06]**, consiste à déterminer au sein du réseau de capteurs, des capteurs spécifiques chargés de vérifier le bon transit de l'information. La Figure II-9 présente le fonctionnement du mécanisme du chien de garde. Un capteur A envoie une information à un capteur B. Le capteur C désigné comme chien de garde écoute la communication, il peut ainsi vérifier la nature de l'information envoyée et vérifier qu'elle correspond aux informations censées circuler sur le réseau, ou bien si le capteur B n'est qu'un capteur chargé de retransmettre l'information à un autre capteur, pour par exemple avertir la base d'un événement détecté. Le capteur C vérifiera alors que le capteur B effectue cette tâche. Dans le cas contraire, il pourra par exemple déterminer que le capteur B est un capteur malicieux, et détecter une attaque de type trou noir. Si cette technique apporte une réelle solution pour la détection des capteurs malicieux, elle nécessite une consommation supplémentaire du réseau, le capteur chien de garde devant écouter chaque communication, l'écoute étant une des fonctions les plus coûteuses pour un capteur.

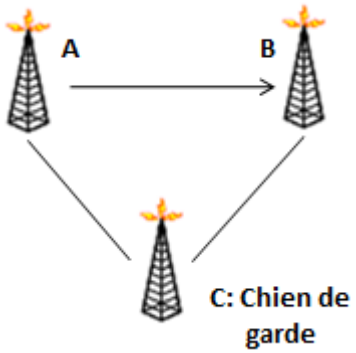

Figure II-9. Exemple de chien de garde.

4.6 La stéganographie

C'est une méthode proche de la cryptographie. Mais là où la cryptographie a pour mission de rendre illisible un message **[MAR10]**, la stéganographie consiste à en cacher son existence. Dans cette méthode, l'information est cachée dans les interférences du signal radio. Cette implémentation permet alors la création d'un canal secret où l'information peut être échangée sans être détectée par un attaquant qui écouterait le réseau en analysant les paquets transitant sur le réseau.

5. Le routage sécurisé dans les RCSFs

Le routage sécurisé est une méthode d'acheminement des informations vers une destination donnée d'une manière sécurisée dans un réseau de connexion. Le développement des protocoles de routage sécurisés spécifiques aux réseaux de capteurs a attiré un grand nombre de chercheurs. Mais les limites imposées par l'architecture matérielle des capteurs, en particulier celle de l'énergie et l'hostilité des environnements, figurent parmi les facteurs primordiaux à prendre en compte lors de la conception d'un protocole de routage sécurisé. De ce fait, ils doivent assurer une consommation minimale d'énergie tout en maintenant le bon fonctionnement du réseau et sans dégrader ses performances.

5.1 Classification des protocoles de routage sécurisés dans les RCSFs

Nous illustrons dans la Figure II-10, la classification de quelques protocoles de routage sécurisés selon leur topologie.

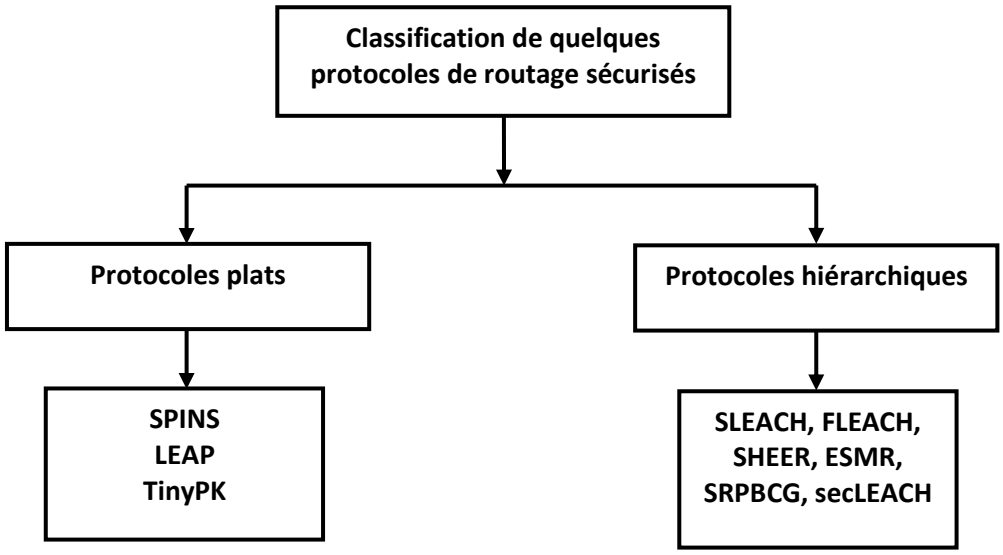

Figure II-10. Classification de quelques protocoles de routage sécurisés dans les RCSFs.

5.2 Exemples de protocoles de routage sécurisés

5.2.1 SPINS

L'une des premières solutions de sécurité pour les RCSFs est appelée SPINS (Sensor Protocols for Information via Negotiation) **[MAL97]**. Il se base sur un modèle de négociation et sur deux protocoles de sécurité : *SNEP* (Secure Network Encryption Protocol) et *µTESLA* (the micro version of the Timed Efficient Stream Loss-Tolerant Authentication Protocol). Le protocole *SNEP* utilise deux mécanismes de sécurité. Le premier consiste à chiffrer les données pour assurer leur confidentialité et le second de calculer un code *MAC* (Message Authentication Code) afin d'assurer l'authentification et l'intégrité des données entre les entités. A chaque premier échange de données entre deux noeuds avec le protocole *SNEP*, le nœud émetteur précède le message d'une chaine de bits aléatoires, aussi appelé vecteur initial. Cette technique empêche un intrus d'entrer en écoute dans le réseau. L'utilisation d'un vecteur aléatoire et d'un compteur empêchent l'écoute et l'interception du message, puisque le message est envoyé en clair suivi soit, d'une chaine de bits, ou d'un compteur incrémenté qui est différent à chaque bloc échangé. L'utilisation de ce compteur permet d'éviter des attaques de rejeu par paquet dont chaque message est numéroté, et cela permettra de garantir la fraicheur des données.

Le protocole *µTESLA* utilise une authentification symétrique liée à une méthode asymétrique, où les clés symétriques sont divulguées au cours du temps. Pour permettre cette authentification, il est nécessaire que la station de base et les différents noeuds soient vaguement synchronisés. La station de base a pour rôle d'ajouter au paquet à envoyer, le code *MAC* calculé à partir d'une clé secrète. Un nœud recevant ce paquet peut vérifier que la clé de déchiffrement du code *MAC* n'a pas encore été divulguée et ce grâce à son horloge de synchronisation. Si la clé n'est pas encore divulguée dans ce cas, il peut en déduire que seul la station de base qui a une connaissance sur la clé *MAC* et qu'aucun attaquant n'a pu altérer le message pendant sa transition. Pour cela, il peut stocker le paquet dans son cache en attendant la prochaine divulgation de la clé. Quand la clé sera divulguée il déchiffrera le message et vérifiera son authenticité. Chaque clé K est une clé issue d'une chaine de clés générée par une fonction à sens unique F, de telle manière que $K_i = F(K_{i+1})$. Cette clé de chiffrement utilisée pour le code *MAC* est générée dans un intervalle régulier de telle sorte que si un capteur ne reçoit pas tous les paquets de clés, il est capable de retrouver les anciennes à partir de la dernière clé reçue. Si un nœud possède la clé initiale K_0 et la clé K_2, mais ne reçoit pas la clé K_1, dans ce cas il vérifiera que la clé K_2 est bien celle envoyée par la SB, d'où $K_0 = F(F(K_2))$ et d'autre part il peut retrouver K_1 car $K_1 = F(K_2)$.

5.2.2 LEAP

Zhu et al. **[ZHU04]** ont proposé un protocole de gestion de clés pour les réseaux de capteurs appelé LEAP (Localized Encryption and Authentication Protocol). LEAP est basé sur une clé initiale transitoire K_i qui est générée par la station de base. Cette clé est chargée

dans chacun des noeuds du réseau pour dériver une clé principale. Ce protocole suppose que pour compromettre un nœud, l'adversaire nécessite un temps minimal T_{min} : c'est le temps de copier le contenu de la mémoire du nœud compromis. Il exploite ce temps pour permettre à deux noeuds voisins d'établir d'une manière sécurisée une clé de session symétrique à partir de la clé initiale K_i. Après un temps T_{min}, la clé K_i est supprimée de la mémoire du nœud. Ce protocole passe par quatre phases :

- Chargement de la clé initiale : la station de base génère une clé initiale K_i et cette dernière sera chargée au niveau de chaque nœud. Chaque nœud u dérive une clé principale $K_u = f(K_i(u))$, f étant une fonction pseudo-aléatoire.

- Découverte de voisins : après le déploiement, le nœud u découvre ses voisins en diffusant un message HELLO qui contient son ID. Aussi, il initie un temps qui sera déclenché après le temps T_{min}. Le nœud u attend un acquittement ACK de chacun de ses voisins v. L'ACK est authentifié en utilisant la clé principale K_v, qui est dérivée comme suit : $K_v = f(K_i(v))$.

- Etablissement de la clé par-paire : le nœud u calcule sa clé par paire K_{uv} avec le nœud v, comme suit : $K_{uv} = f(K_v(u))$. Le nœud v peut de même calculer K_{uv} de la même manière, K_{uv} sert comme clé entre u et v.

- Suppression des clés : lorsque le temps expire après T_{min}, le nœud u efface la clé initiale K_i et toutes les clés principales K_v de ses voisins. Il est à noter que le nœud u ne supprime pas sa clé principale K_u

A la fin de ces quatre étapes, le nœud u aura établi une clé par paire partagée avec chacun de ses voisins. Cette clé sera utilisée pour sécuriser les données échangées entre eux. Un adversaire peut écouter clandestinement tout le trafic dans cette phase, mais sans la clé K_i il ne peut pas injecter des informations incorrectes ou déchiffrer les messages. Un nœud malveillant compromettant un nœud après T_{min}, et il obtient seulement les clés du nœud compromis. Quand un nœud compromis est détecté, ses voisins suppriment simplement les clés qui ont été partagées avec lui. Cependant, le message HELLO est non-authentifié, et donc un adversaire peut exploiter ceci pour lancer des attaques en injectant un grand nombre de ce type de message. Le protocole LEAP emploie µTESLA afin d'assurer l'authentification d'émission avec la station de base.

5.2.3 TinyPK

TinyPK est un protocole de routage sécurisé proposé par Watro et al. **[MAL97]**. Il assure l'authentification à l'aide d'une paire de clé publique et privée. Les opérations cryptographiques se basent sur l'algorithme de chiffrement RSA (de ses concepteurs *Rivest*, *Shamir* et *Adleman*). Ces opérations sont exécutées par une entité externe du réseau. Le protocole nécessite une autorité de certification possédant une paire de clé

publique et privée. Toute partie externe qui souhaite communiquer avec les capteurs doivent avoir une clé publique, cette dernière doit être signée par l'autorité de certification afin de lui établir un identifiant unique et une clé privée correspondante à la clé publique. Chaque capteur est également pré-chargé d'une clé publique correspondante à l'autorité de certification. Les noeuds capteurs peuvent se communiquer les uns avec les autres par l'intermédiaire de la partie externe, mais ils doivent établir leurs identités à une partie interne qui s'en charge de toutes les opérations cryptographiques.

5.2.4 SLEACH

Le protocole SLEACH est la première version sécurisée du protocole LEACH **[FER05].** SLEACH assure la sécurité dans LEACH en utilisant SPINS (Security Protocol for Sensor Network) et les méthodes à clés symétriques et MAC (Message Authentication Code). SLEACH protège contre l'attaque sinkhole et HELLO flooding. Il empêche l'intrus d'envoyer des données erronées au Chef de cluster et au Chef de cluster de transmettre ces faux messages à la station de base. La solution est destinée à protéger seulement le réseau contre les attaques externes.

5.2.5 F-LEACH

L.B. Oliveria et al. **[OLI06]** ont proposé le protocole sécurisé FLEACH, c'est un protocole qui permet de sécuriser la communication entre deux noeuds dans le réseau de capteurs. Il utilise un système de prédistribution de clé aléatoire avec la cryptographie à clé symétrique pour améliorer la sécurité dans le protocole de base LEACH. FLEACH permet d'assurer les services de sécurité les plus importants : l'authentification, l'intégrité, la confidentialité et fraîcheur de données entre deux nœuds qui communiquent. Mais il est vulnérable à l'attaque de capture des nœuds capteurs.

5.2.6 SHEER

J. Ibriq et al. **[IBR06]** ont proposés un protocole de routage hiérarchique sécurisé (SHEER) qui fournit une communication sécurisée à la couche réseau, Il utilise un mécanisme de gestion d'énergie pour améliorer la performance énergétique du réseau et augmenter sa durée de vie. Pour sécuriser le routage, SHEER utilise un protocole de transmission de clé cryptographique sécurisée et symétrique (HIKES).

5.2.7 SRPBCG

Z. Quan et al. **[QUA09]** ont proposé un protocole de routage appelé Secure Routing Protocol Cluster-Gene-Based for WSNs (SRPBCG). La sélection des Chefs de clusters se fait de la même manière que LEACH, l'objectif est de gérer la confiance et la réputation au niveau local et d'authentifier l'identité de nœud. Le mécanisme d'authentification qui a été utilisé est la clé de cryptage et la distribution des clés qui est faite d'une manière très sûr et

efficace qui ne nécessitent que peu de mémoire, ce protocole ne traite que les attaques des adversaires et les nœuds compromis. La sécurité du protocole n'est pas considérée au moment de la création des clusters et la transmission des messages.

5.2.8 ESMR

C'est une solution de sécurité appelée modèle de sécurité efficace de protocole de routage (ESMR) **[CHE08]**, qui utilise la technique de la cryptographie à clé publique seulement. Le résultat de la simulation montre que le rendement du protocole ESMR n'est pas aussi bon que LEACH dans un environnement sans attaque, mais il devient de mieux en mieux quand le nombre d'attaquants augmente. Ce protocole ne traite que l'attaque externe.

5.2.9 Sec-LEACH

Sec-LEACH fournit une solution efficace pour assurer la sécurité des communications dans LEACH, il utilise la pré distribution de clés et µTESLA pour sécuriser les réseaux de capteurs hiérarchiques avec la formation de clusters dynamique, Sec-LEACH applique la distribution de clé aléatoire sur LEACH et introduit une clé symétrique et une chaîne de hachage à sens unique de manière à assurer la confidentialité et la fraîcheur des données. Sec-LEACH assure l'authentification, l'intégrité, la confidentialité et la fraîcheur de données aux communications **[OLI07].**

5.3 Comparaison entre les protocoles de routages sécurisés

Le tableau II-2 illustre une comparaison entre les protocoles présentés précédemment **[SHA11]**.

Protocole	Protocole de base	Prévention contre l'attaque	Objectifs de sécurité	Mécanisme de sécurité utilisé	Efficacité énergétique
SPINS	SPIN	Modification, rejeu	La confidentialité, l'intégrité, la fraicheur et l'authentification	Chiffrement et MAC	Moyenne
LEAP	-	Modification, rejeu, Sybil, hello flooding	La confidentialité, l'intégrité et l'authentification	Chiffrement et MAC	Moyenne
TinyPK	-	Modification, rejeu, sinkhole, hello flooding	L'authentification	Chiffrement et MAC	Moyenne
SLEACH	LEACH	Modification, rejeu, sinkhole, hello flooding.	L'intégrité des données et l'authentification	Chiffrement et MAC	Moyenne

F-LEACH	LEACH	Modification, rejeu, Sybil, hello flooding	Confidentialité, intégrité, fraicheur et authentification	Chiffrement, prédistribution de clé.	Moyenne
SHEER	-	Modification, rejeu, Sybil, hello flooding.	La confidentialité, l'intégrité et la fraicheur et l'authentification	Chiffrement, prédistribution de clé	Bonne
ESMR	-	Modification, rejeu	La confidentialité	Chiffrement, prédistribution de clé	Moyenne
Sec-LEACH	LEACH	Modification, rejeu, Sybil, hello flooding	La confidentialité, l'intégrité et la fraicheur et l'authentification	Chiffrement, prédistribution de clé, fonction de hachage.	Moyenne
SRPBCG	-	Modification, rejeu, noeuds compromis	La confidentialité, l'intégrité et l'authentification	-	Moyenne

Tableau II-2. Comparaison entre les différents protocoles de routage sécurisé dans les RCSFs.

Conclusion

Dans ce chapitre, nous avons donné un aperçu sur la sécurité dans les réseaux de capteurs sans fil. Nous avons présenté les vulnérabilités de la sécurité dans les RCSFs et les menaces qui peuvent les contrer. Nous avons aussi listé les objectifs et les besoins de la sécurité requis par les protocoles de routage dans les RCSFs et les mécanismes utilisés pour combattre ces menaces.

Comme évoqué précédemment, la communication entre les noeuds d'un RCSF doit être régie par un ensemble de règles « protocole ». Les protocoles de routage pour les RCSFs sont nombreux. On va s'intéresser particulièrement à la classe des protocoles hiérarchiques. Parmi cette classe de protocoles, on distingue le protocole LEACH qui constitue l'objet principal de notre étude. Ainsi, le prochain chapitre sera consacré pour son étude détaillée.

CHAPITRE 3

Le protocole de routage hiérarchique LEACH

Introduction

La propagation des données est une fonctionnalité très importante dans un RCSF. Plusieurs protocoles de routage ont été proposés pour les RCSFs grâce aux avantages qu'ils présentent. Parmi ces protocoles de routage, on distingue les protocoles de routage hiérarchiques. Deux grandes approches sont dérivées de ce type de protocoles : l'approche basée sur les chaînes (chaine-based approach) dont l'idée de formation de chaînes a été proposée pour la première fois dans l'algorithme PEGASIS, et l'approche basée sur les clusters (cluster-based approach) **[LIN02].**

LEACH est considéré comme étant le premier protocole de routage hiérarchique basé sur la seconde approche. Il est aussi l'un des algorithmes de routage hiérarchiques les plus populaires pour les RCSFs. Il combine l'efficacité en consommation d'énergie et la qualité de l'accès au média. L'idée est de former des clusters de nœuds capteurs basés sur les zones où il y a un fort signal reçu, puis utiliser des cluster-heads comme passerelles pour atteindre la destination.

Dans ce chapitre, nous expliquons l'architecture de communication du protocole LEACH et les protocoles MAC utilisés par ce dernier. Ensuite, nous expliquons en détail l'algorithme et les caractéristiques du protocole LEACH. Enfin, nous étudions les attaques pouvant perturber son fonctionnement.

1. Architecture de communication de LEACH

Le protocole LEACH (Low Energy Adaptive Clustering Hierarchy) proposé par Heinzelman et al **[HEI99]**, est un protocole de routage hiérarchique bien connu, appliqué dans les RCSFs. L'architecture de communication de LEACH consiste à former des cellules basées sur l'amplitude du signal, et utiliser les têtes de cellules comme routeurs vers la station de base. Ces cellules sont appelées groupes (clusters), quant aux têtes : chefs de groupes (Cluster-Heads). Les chefs de groupes sont choisis de façon aléatoire selon un algorithme spécifique d'élection basé sur une fonction de probabilité qui prend en compte différents critères, comme l'énergie disponible des noeuds **[BOU08].**

Comme la Figure III-1 l'indique, les nœuds capteurs sont chargés de collecter des données, les envoyer à leurs CH. Le CH à son tour agrège et transmet les résultats d'agrégation à la station de base selon une communication unicast (à un seul saut). Les CHs ont pour mission d'assurer les fonctions les plus coûteuses en termes d'énergie, à savoir la communication avec la station de base qui est supposée éloignée, ainsi que tous les traitements de données (agrégation, fusion et transmission des données), afin de réduire la quantité des données transmises. Ce dispositif permet d'économiser l'énergie puisque les transmissions vers la station de base sont uniquement assurées par les CHs plutôt que par tous

les noeuds du réseau. Par conséquent, LEACH réalise une réduction significative de la consommation d'énergie.

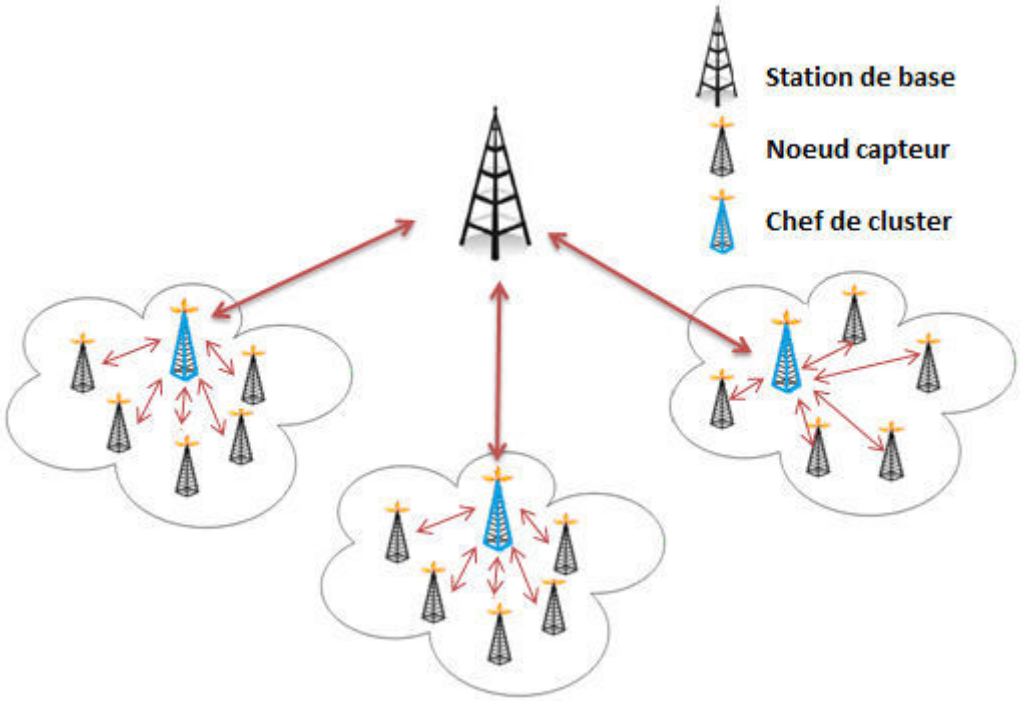

Figure III-1. Architecture de communication du protocole de routage LEACH.

2. Protocoles MAC utilisés par LEACH

Pendant le fonctionnement du protocole LEACH, ce dernier appelle certains protocoles MAC pour l'accès au media. On va les détailler dans cette section pour mieux comprendre le déroulement de son algorithme qui sera détaillé dans la section suivante. Le protocole LEACH utilise les schémas à allocation fixe. Ces derniers permettent d'allouer le media de transmission pour chaque nœud suivant des intervalles de temps (schéma TDMA) ou un schéma de codage particulier (schéma CDMA). Étant donné que chaque nœud est attribué en exclusivité à un intervalle, il n'y a presque pas de collisions entre les données. Toutefois, les schémas à allocation fixe s'avèrent inefficaces lorsque les noeuds n'ont pas de données à transmettre. En effet, ces intervalles sont affectés à des noeuds qui n'ont pas besoin de les utiliser **[TIX04][LAW01].**

2.1 TDMA

Le schéma d'accès multiple à répartition de temps ou TDMA (Time Division Multiple Access) permet de diviser le temps en intervalles (time-slot) attribués à chaque nœud. Ainsi, un seul nœud a le droit d'accéder au canal (il utilise toute la plage de la bande passante du canal), chaque nœud doit émettre ses données pendant les intervalles de temps qui lui sont accordés **[TIX04].**

2.2 CDMA

Le schéma d'accès multiple par répartition en code ou CDMA (Code Division Multiple Access) ne divise ni la plage de fréquences, ni l'intervalle de temps. Ainsi, des noeuds peuvent émettre leurs données continuellement et selon une large plage de fréquence. Le protocole CDMA utilise des techniques afin d'éviter les collisions entre les transmissions simultanées des noeuds **[LAW01].**

3. Algorithme de LEACH

Dans **[HEI02]**, Heinzelman et al, ont proposé un des algorithmes de clustering, connu sous le nom de Low Energy Adaptive Clustering Hierarchy (LEACH), où l'élection des Cluster-Heads se fonde sur une génération d'un nombre aléatoire. L'objectif de LEACH est de permettre une faible consommation énergétique lors de la phase de clustering ainsi que dans le routage des informations. Pour cela, il procède par rounds. Ces derniers ont approximativement le même intervalle de temps (durée) déterminé au préalable. Comme illustré dans la Figure III-2, chaque round est composé de deux phases successives. La première est la phase d'initialisation, nommée « Set-Up Phase », durant laquelle le réseau s'auto-organise en clusters. Et la seconde représente la phase de transmission, nommée « Study-State Phase », lors de laquelle les données sont collectées et routées dans le réseau hiérarchisé.

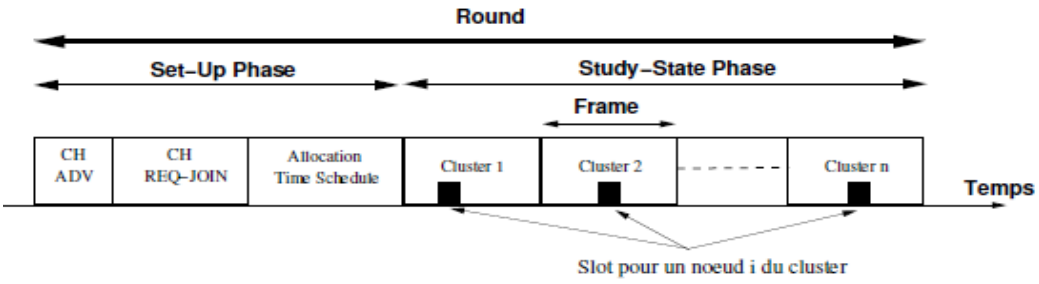

Figure III-2. Détails d'un round dans LEACH.

3.1 Phase d'initialisation

Comme illustré par la Figure III-3, la « Set-Up Phase » est composée de trois sous phases : une phase d'annonce, une phase d'organisation des clusters et une phase d'ordonnancement. Dans ce qui suit, nous allons expliquer en détail, le fonctionnement et le rôle de chaque sous phase.

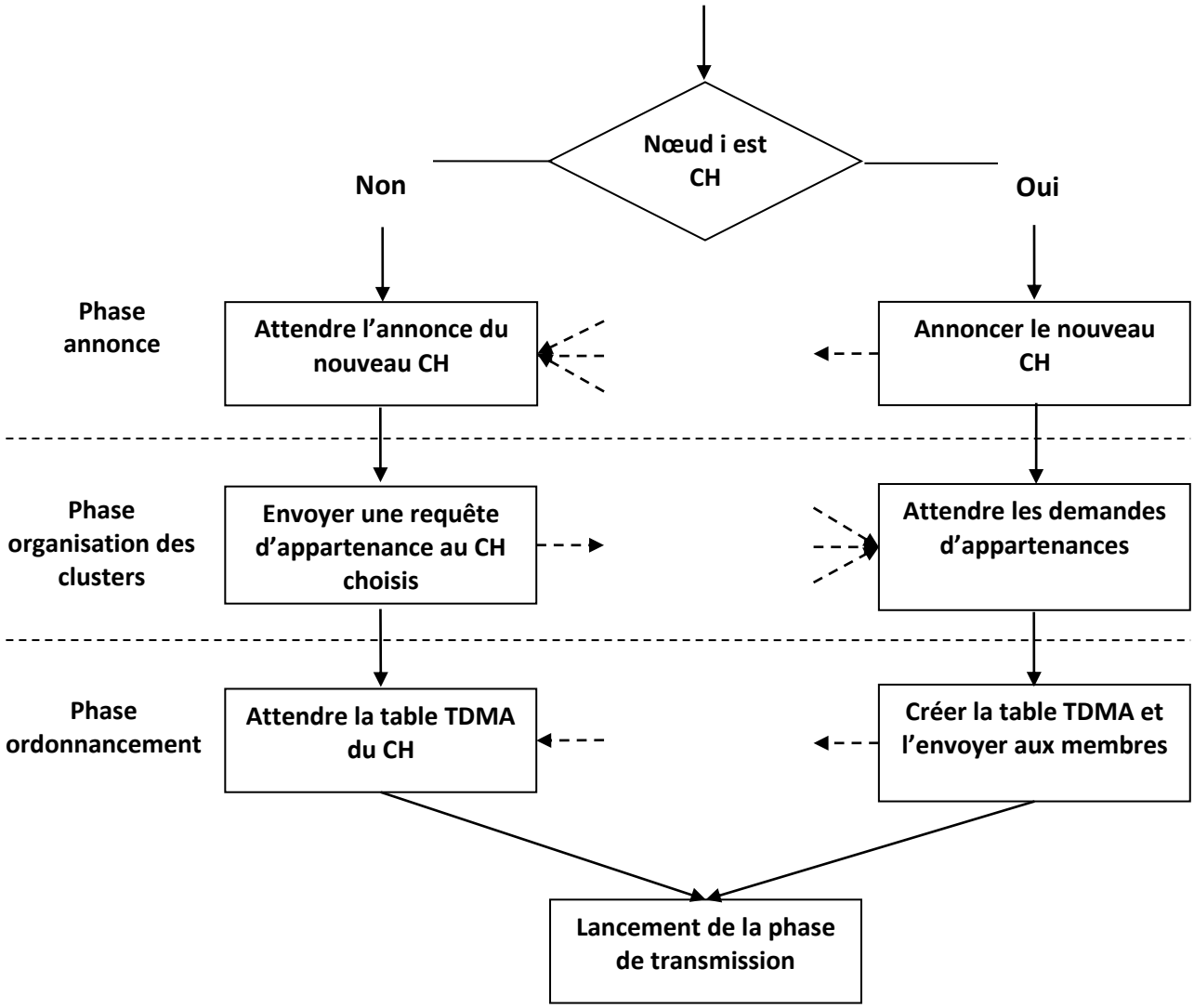

Figure III-3. Opérations de la phase initialisation de LEACH.

3.1.1 Phase d'annonce

Avant de lancer cette phase, on désire avoir un certain nombre de CHs. Ce nombre, que l'on note *P*, est fixe durant chaque round. Le pourcentage optimal du nombre de CHs désirés devrait être préférentiellement choisie ente 5% et 15% du nombre total de nœuds **[HEI02]**. Si ce pourcentage n'est pas respecté, cela mènera à une grande dissipation d'énergie dans le réseau. En effet, si le nombre de CHs est très élevé, on aura un nombre important de nœuds CHs qui se consacrent aux tâches les plus couteuses en termes de ressources énergétiques. Ainsi, on aura une dissipation d'énergie considérable dans le réseau. De plus, si le nombre de CHs est très petit, ces derniers vont gérer des groupes de grandes tailles. Ainsi, ces CHs s'épuiseront rapidement à cause du travail important qui leurs a été demandés. La phase d'annonce est déclenchée par la station de base en donnant à tous les nœuds du réseau le pourcentage *P* de cluster-heads désirés durant un round *r*. Les nœuds capteurs s'auto-élisent

pour être des CHs au début du round *r* (qui commence à l'instant *t*). Ils se basent sur le pourcentage désiré de CHs et le nombre de rounds au cours duquel un nœud a pris le rôle de CH. Ainsi, Tout nœud *i* génère un nombre aléatoire *X(i)* entre 0 et 1. Puis compare *X(i)* avec un seuil *T(i)* calculé comme illustré dans l'équation (1). Si le nombre aléatoire est inférieur au seuil *T(i)*, le nœud se déclare CH dans le round courant.

$$T(i) = \begin{cases} \frac{P}{1-P*(r \bmod (1/P))} & si\ i \in G \\ 0 & sinon \end{cases} \qquad (1)$$

Où :

P : Le pourcentage désiré de Cluster Heads.
r : Le numéro du round courant.
G : Ensemble des nœuds qui n'ont pas encore été élus comme cluster head lors des *1/P* rounds précédents. En d'autres termes, seuls les noeuds qui n'ont pas encore été élus CH lors des *1/P* rounds précédents peuvent prétendre l'être dans le round courant.

3.1.2 Phase d'organisation des clusters

Lors de la phase d'organisation des clusters, si *X(i)* < *T(i)*, le nœud *i* se déclare cluster-head et envoie un message ADV (ADVertise) contenant son identifiant pour avertir son voisinage de son nouveau statut. Sinon, il attend les messages ADV venant des autres cluster-heads. Pour éviter les collisions avec les autres messages ADV, la méthode d'accès CSMA est utilisée. Les nœuds qui ne sont pas élus cluster-heads vont s'appuyer sur la force du signal des messages ADV reçus (le CH le plus proche) pour adopter un cluster-head. Un nœud *v* va ainsi adhérer au cluster head ayant le plus fort signal (en cas d'égalité des signaux, *v* choisit aléatoirement un des deux). Puis, *v* diffuse un message REQ-JOIN à son cluster-head pour lui notifier son adhésion à son cluster.

Les messages ADV et REQ-JOIN sont envoyés en broadcast dans le réseau sous le format de paquet illustré par la Figure III-4. Tous les messages circulant dans le réseau sont encapsulés avec l'entête de niveau Mac (H-Mac). Nous définissons dans ce qui suit, la signification de chaque champ.

Src : adresse de l'émetteur du paquet.

Dst : adresse du destinataire du paquet.

SeqNum : numéro de séquence du paquet au niveau MAC.

RSSI : Received Signal Strength Indication, c'est une mesure de la puissance d'un signal reçu d'une antenne (un signal radio). Son utilité est de fournir une indication sur l'intensité du signal reçu.

LQI : Link Quality Indicator, c'est une mesure de la qualité actuelle du signal reçu.

NextHop : L'adresse du prochain routeur qui recevra le paquet.

LastHop : L'adresse du destinataire final qui recevra le paquet.

Leach Header : Entête ou type de commande LEACH (ADV, REQ-JOIN, DATA, TDMA…).

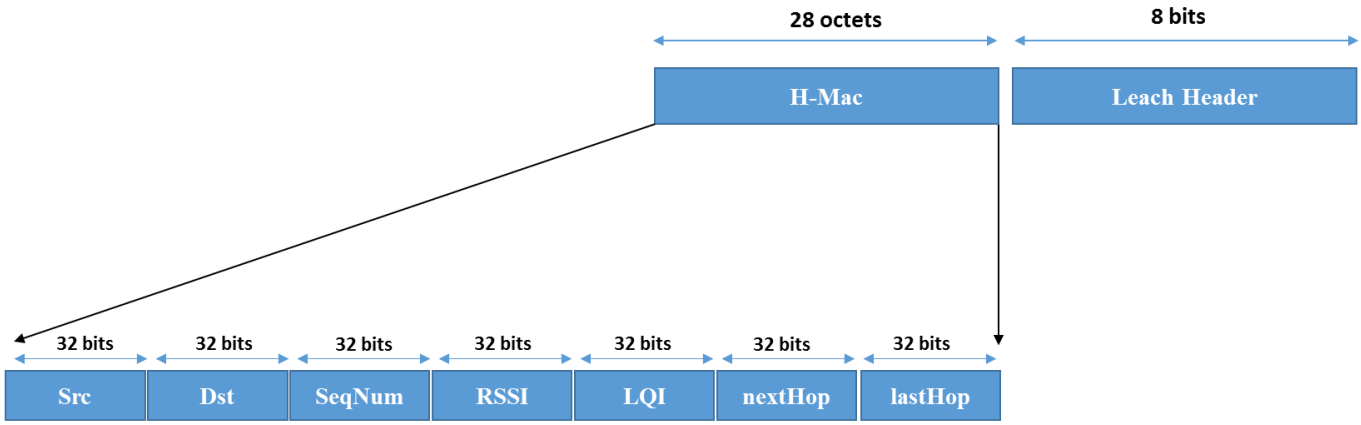

Figure III-4. Format des paquets ADV et REQ-JOIN.

3.1.3 Phase d'ordonnancement

Après la formation des clusters (groupes), chaque CH agit comme un centre de commande local pour coordonner les transmissions des données au sein de son cluster. Pour cela, Ils créent un ordonnanceur (schedule) TDMA, et attribuent à chaque nœud de leurs clusters un slot de temps durant lequel ils peuvent envoyer leurs données collectées. L'ensemble des slots assignés aux nœuds d'un cluster est appelé frame. La durée de chaque frame diffère selon le nombre de nœuds du cluster. Pour éviter les interférences entre clusters adjacents, les cluster-heads choisissent aussi aléatoirement un code à l'aide du protocole CDMA utilisé lors des communications des clusters heads vers la station de base. La « Set-Up phase » se termine ainsi puis démarre la « Study-State phase ». Le message d'ordonnancement TDMA est envoyé en broadcast dans le réseau, sous le format de paquet illustré par la Figure III-5.

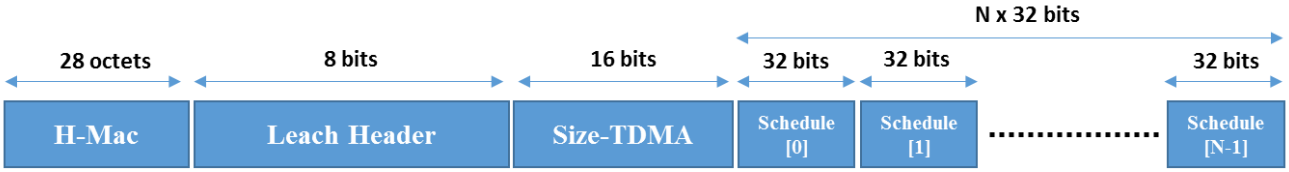

Figure III-5. Format d'un paquet d'ordonnancement.

3.2 Phase de transmission

Lors de la phase de transmission, qui est la plus longue par rapport à la phase d'initialisation, les nœuds dans les clusters collectent leurs données puis les envoient à leurs cluster-heads durant leurs slots de temps obtenus précédemment à l'aide du protocole TDMA. Le format des paquets de données est illustré par la Figure III-6.

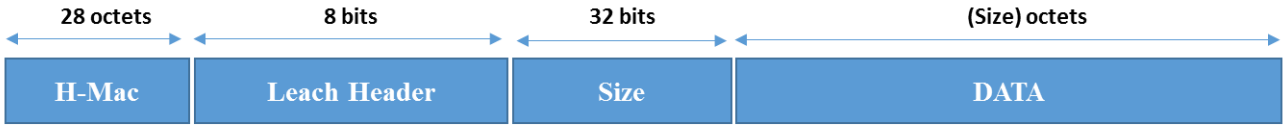

Figure III-6. Format d'un paquet de données.

Size : indique la taille en octet de chaque donnée transmise dans chaque paquet.

Chaque cluster-head agrège toutes les données issues de son cluster puis l'envoient à la station de base en utilisant son code obtenu précédemment à l'aide du protocole CDMA. Pour la retransmission de ces données à la station de base, le cluster-head encapsule l'ensemble des paquets reçus dans un seul paquet que nous avons appelé *Super-Packet*. Le format de ce dernier est illustré par la Figure III-7.

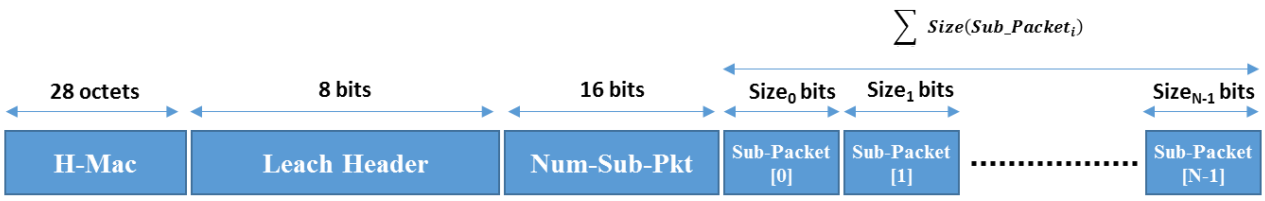

Figure III-7. Format d'un super paquet.

Num-Sub-Pkt : indique le nombre de sous paquets qui sont encapsulés dans le Super packet.

4. Caractéristiques de LEACH

Le protocole LEACH engendre beaucoup d'avantages pour ce qu'il offre comme bonne manipulation des ressources du réseau en respectant plusieurs contraintes telle que la consommation d'énergie. Bien que LEACH économise la consommation d'énergie, un nombre d'inconvénients restent plus ou moins apparents. Dans ce qui suit, on cite quelques avantages et inconvénients du protocole LEACH **[KHE04].**

4.1 Avantages de LEACH

Le protocole LEACH présente les avantages suivants :

- **Protocole auto-organisateur basé sur le groupement adaptatif :** LEACH est complètement distribué. Autrement dit, l'auto-configuration des clusters se fait indépendamment de la station de base (algorithme distribué). Les noeuds prennent leurs décisions de façon autonome et agissent de manière locale et n'ont pas besoin d'une information globale pour opérer de façon efficace. De plus, la collection de données est faite périodiquement (l'utilisateur n'a pas besoin de toutes les données immédiatement). Pour exploiter cette caractéristique, ce protocole introduit un groupement adaptatif, c'est-à-dire, il réorganise les clusters après un intervalle de temps aléatoire, en utilisant des contraintes énergétiques afin d'avoir une dissipation d'énergie uniforme à travers tout le réseau **[RAG03].**

- **Rotation des rôles de chefs de clusters :** la rotation des rôles de chefs de clusters s'avère un facteur important pour l'organisation des noeuds. Ce rôle est épuisant en termes d'énergie car les CHs sont actifs tout au long de leur élection. Puisque la station de base est généralement loin du champ de surveillance, les CHs diffusent une quantité plus importante d'énergie pour lui transmettre leurs données. Donc, si les CHs sont choisis d'une manière fixe, leur énergie s'épuisera rapidement, ce qui induit à leurs défaillances. Par conséquent, tous les autres noeuds seront sans CH et donc inutiles. C'est pourquoi, la plupart des algorithmes de groupement (clustering) adoptent la rotation du rôle de chefs de groupes **[ROM07].**

- **Faible énergie pour l'accès au média :** le mécanisme de groupes permet aux nœuds d'effectuer des communications sur des petites distances avec leurs CH afin d'optimiser l'utilisation du média de communication en la faisant gérer localement par un CH pour minimiser les interférences et les collisions.

- **Compression locale (agrégation) :** les CHs compressent les données arrivant de leurs membres, et envoient un paquet d'agrégation à la station de base afin de réduire la quantité d'informations qui doit lui être transmise. Cela permet de réduire la complexité des algorithmes de routage, de simplifier la gestion du réseau, d'optimiser les dépenses d'énergie et enfin de rendre le réseau plus évolutif (scalable).

4.2 Inconvénients de LEACH

En revanche, LEACH a les inconvénients suivants **[KHE04] :**

- On pourrait ne pas avoir des CHs durant un round si les nombres aléatoires *X(i)* générés par tous les noeuds du réseau sont supérieurs au seuil *T(i).*

- L'utilisation d'une communication à un seul saut entre les CHs et la station de base au lieu d'une communication multi-sauts diminue l'énergie des CHs.

- Les CHs les plus éloignés de la station de base meurent rapidement par rapport à ceux qui sont proches de la station de base.

- Le CH est toujours allumé et lorsqu'il meurt, le groupe deviendra inutile car les données recueillies par les noeuds du cluster ne sauront jamais atteindre la station de base.

- LEACH ne garantie pas une distribution homogène des CHs sur le réseau, car le seul critère d'élection du CH est une probabilité aléatoire. Donc, il est possible que les CHs puissent être concentrés dans une partie du réseau. Par conséquent, certains nœuds n'auront pas des CHs dans leurs voisinages.

- La rotation des CHs permet de ne pas épuiser les batteries. Cependant, cette méthode n'est pas efficace pour de grandes structures de réseaux, à cause de la surcharge d'annonces engendrées par le changement des CHs, et qui réduit le gain d'énergie initial.

- Le protocole LEACH n'est pas sécurisé. Aucun mécanisme de sécurité n'est intégré dans ce protocole. Ainsi, il est très vulnérable même aux simples attaques. Donc, un attaquant peut facilement monopoliser le réseau et induit à son disfonctionnement.

5. Attaques sur le protocole LEACH

Comme la plupart des protocoles de routage dans les RCSFs, LEACH est vulnérable à un certain nombre d'attaques de sécurité. Toutefois étant un protocole à base de clusters, nous comptons entièrement sur le CH pour la tâche d'agrégation des données et le routage vers la station de base. Par conséquent, les attaques qui visent les CHs sont fatales. Si un intrus parvient à devenir CH, il peut provoquer un disfonctionnement total du réseau. Un réseau de capteurs sans fil est un réseau qui a de nombreuses contraintes par rapport à un réseau informatique traditionnel. En raison de ces contraintes, ils peuvent facilement être attaqués par des noeuds malveillants. L'attaque est plus dangereuse si l'attaquant devient CH. Dans ce

cas, il peut affecter les données de l'ensemble du cluster attaché à lui. Diverses attaques ont été décrites dans la théorie des RCSFs hiérarchiques. Parmi ces attaques, nous citons :

5.1 Blackhole attack (trou noir)

Dans Blackhole attack, l'attaquant tente de recueillir la plupart des données du réseau et les supprime plus tard. Dans LEACH, les CHs sont élus en fonction de l'énergie résiduelle de divers nœuds. L'attaquant a une énergie initiale plus élevée, de sorte qu'il devient l'un des chefs de cluster dans le premier round, et même dans les prochains rounds, puisqu'il ne consomme aucune énergie pour la transmission de données. Par conséquent, il devient CH dans presque tous les rounds. Après être devenu CH, il reçoit les données de l'ensemble de ses membres de cluster, agrège ces données et ne transmet pas la suite des données à la station de base **[TRI13]**. Cette attaque est illustrée par la Figure III-8.

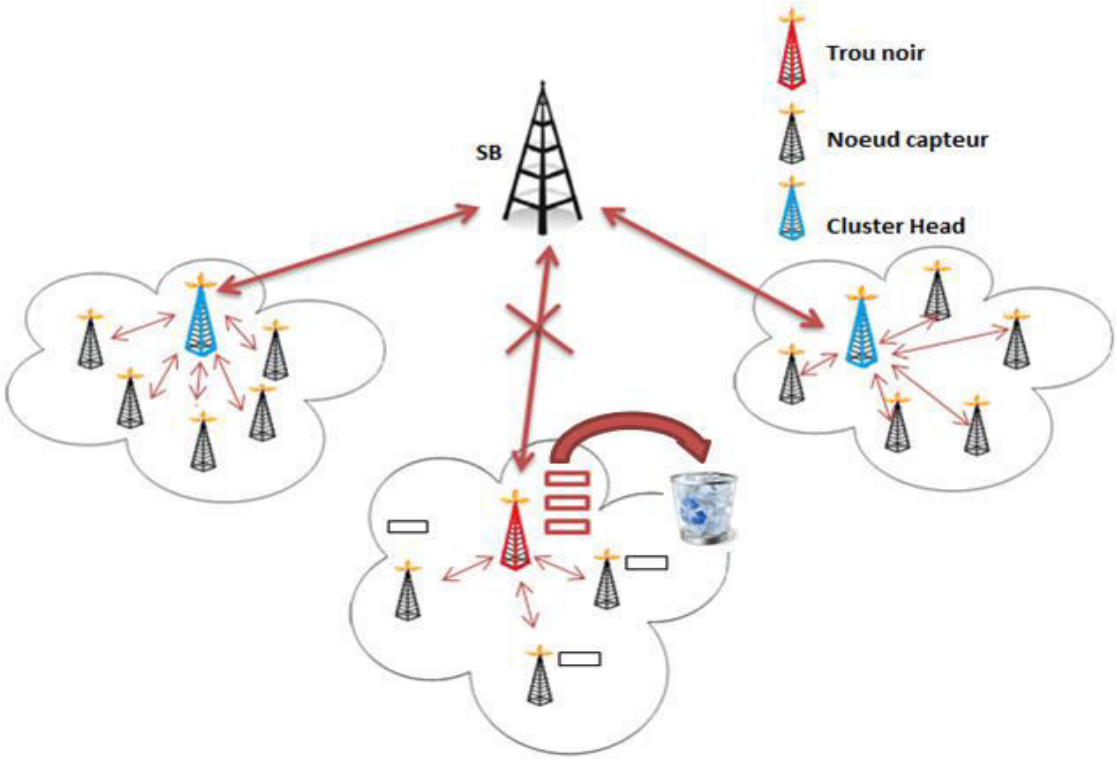

Figure III-8. Attaque trou noir dans LEACH.

5.2 Greyhole attack(trou gris)

Cette attaque est une variante du trou noir. Initialement, un nœud malveillant exploite le protocole LEACH pour s'annoncer comme ayant une forte puissance du signal pour devenir chef de clusters avec l'intention d'intercepter les paquets. Ensuite, un trou gris peut présenter son comportement malveillant dans de multiples façons. Il peut supprimer les paquets qu'il a reçus de son cluster, et qui sont en provenance de la station de base. Un autre type de cette attaque, peut se présenter comme un nœud qui se comporte méchamment pour

une certaine durée de temps pour supprimer les paquets reçus, mais peut passer à un comportement normal plus tard où il peut transmettre quelques paquets. Ce qui rend sa détection encore plus difficile **[TRI13].**

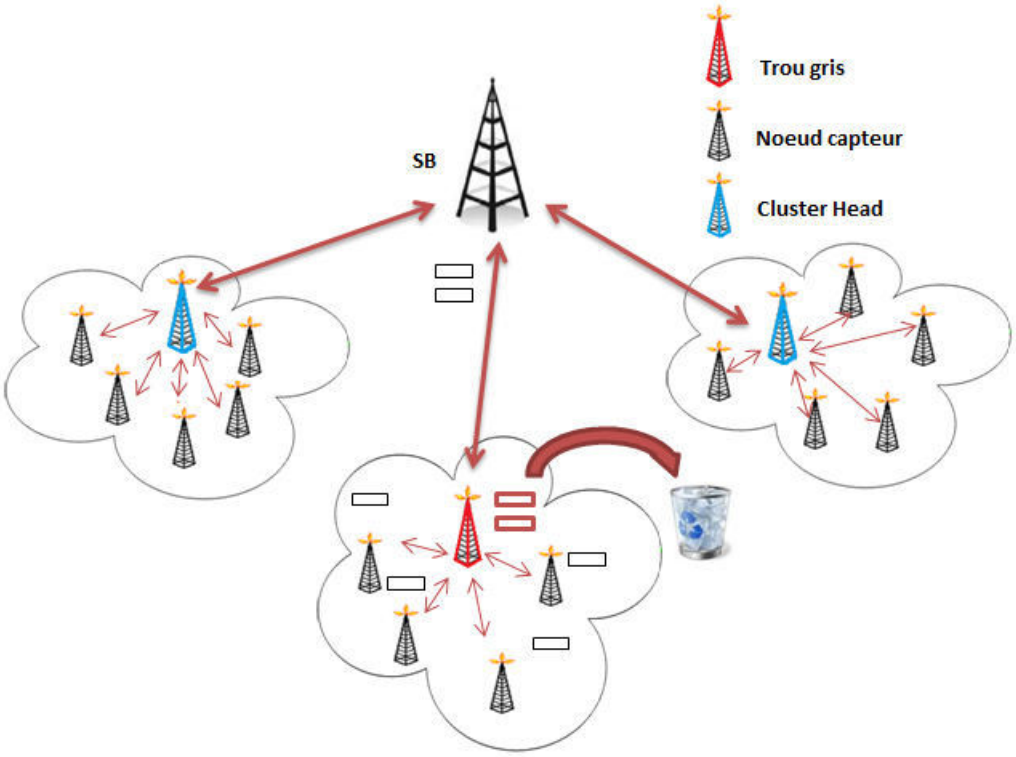

Figure III-9. Attaque trou gris dans LEACH.

5.3 Sinkhole (trou de la base)

Dans cette attaque, l'attaquant achemine tout le trafic vers lui afin de contrôler la plupart des données circulant dans le réseau. Ainsi il apparaît aux autres noeuds comme étant la station de base en émettant un signal plus fort que celui de la station de base originale. Donc, les CHs vont transmettre leurs données à la station de base avec une puissance plus faible. En conséquence, les données ne peuvent pas atteindre cette dernière. Dans le cas où elles peuvent l'atteindre, elle peut aussi les modifier ou les supprimer **[AGG03].**

5.4 Brute-force jamming attack(le brouillage radio)

Comme le protocole LEACH est basé sur le regroupement des noeuds, un attaquant suffit de brouiller une petite partie du réseau. En effet, il est plus intéressant de brouiller les canaux liant les CHs et la station de base. De plus, il ne sera pas possible de brouiller tout le réseau si l'attaquant n'est pas puissant. Le brouillage pourra être persistant si le réseau ne réagit pas à ce type d'attaques **[AGG03].**

Conclusion

Dans ce chapitre, nous avons présenté le protocole de routage hiérarchique LEACH dont l'objectif principal est le prolongement de la vie du réseau et la gestion efficace de la consommation énergétique. Nous avons vu que LEACH est un protocole à base de cluster, en s'appuyant fondamentalement sur les CHs pour l'agrégation des données et le routage vers la station de base. Cependant, puisqu'ils sont considérés comme les points qui recueillent toutes les données de leurs clusters, les attaques peuvent les viser en attaquant seulement les liens de communication entre ces CHs et la station de base.

Dans le prochain chapitre, nous allons nous intéresser particulièrement aux attaques de type trou noir et gris. Nous allons donc proposer une nouvelle alternative sécurisée du protocole LEACH qui permet de détecter ces attaques.

CHAPITRE 4

Sécurisation du protocole de routage hiérarchique LEACH

Introduction

Les réseaux de capteurs sans fil sont généralement déployés dans des zones non surveillées, ce qui les rend vulnérables à plusieurs types d'attaques, dans lesquelles, l'attaquant peut prendre le contrôle d'un ou plusieurs nœuds capteurs, dans le but de perturber le bon fonctionnement du réseau. C'est la raison pour laquelle, assurer la sécurité dans ce type de réseaux devient un enjeu très important, surtout en ce qui concerne le bon acheminement des données vers la station de base. Comme il a été déjà cité dans le chapitre précédent, le protocole LEACH est un protocole de routage hiérarchique non sécurisé. Ce protocole compte entièrement sur les Cluster-Heads pour l'acheminement des données vers la station de base. Pour cette raison, les Cluster-Heads font une cible de choix pour les attaquants dans le but d'empêcher le routage des données vers la destination finale qui est la station de base.

Nous allons dans ce chapitre proposer une solution de sécurisation pour ce protocole, qui permet de protéger le réseau contre les attaques de type trou noir et trou gris, en utilisant des mécanismes de détections de ces attaquants.

1. Problématique

Dans notre travail, nous nous sommes intéressés aux problèmes relatifs au routage des données dans les RCSFs, et plus précisément le routage hiérarchique des données. Ce dernier est considéré comme un outil permettant plus de performance en ce qui concerne la consommation énergétique par rapport aux autres types de routage, à savoir, le routage à topologie plate et le routage géographique. Les protocoles de routage hiérarchiques supposent que tous les capteurs du réseau sont amicaux et coopératifs. Toutefois, cette hypothèse n'est pas toujours valable, car les RCSFs sont vulnérables à plusieurs types de menaces.

LEACH est un protocole de routage hiérarchique à base de clusters. Tous les nœuds d'un cluster sont chargés de collecter des données environnementales et de les router à leurs Cluster-Head. Ces derniers recueillent ainsi toutes les données de leurs clusters, ensuite ils les agrègent et les routent vers la station de base. De ce fait, les Cluster-Heads font une cible de choix pour les attaquants, dans le but de créer un disfonctionnement sur les réseaux hiérarchiques. Plusieurs menaces contre les RCSFs ont été décrites dans les chapitres précédents, comme le brouillage radio, l'attaque du trou de ver, l'attaque du trou de la base, l'attaque Sybille et l'attaque du trou noir, etc. Dans les réseaux à topologie hiérarchique, l'attaque est plus dangereuse si l'attaquant devient Cluster-Head. Pour cela, nous nous sommes intéressés à deux attaques : le trou noir et le trou gris, qui sont des nœuds attaquants, qui essayent par divers moyens de se positionner dans le réseau comme étant des Cluster-Heads. Pour préserver les bonnes opérations du réseau, la sécurité doit être considérée comme une composante essentielle du mécanisme de routage. Nous allons donc, proposer des mécanismes pour la détection des attaques trous noirs et gris. Ainsi, nous obtenons un nouveau protocole LEACH sécurisé que nous appelons DHD-LEACH (Dynamic Hole Detection in LEACH).

2. Vue globale de la solution proposée

Il existe dans la littérature plusieurs variantes sécurisées du protocole LEACH (SLEACH, Sec-LEACH) **[OLI07][FER05]**. Ces variantes répondent principalement aux problèmes d'authentification, d'intégrité et la confidentialité des données. Mais très peu de ces algorithmes répondent à la problématique de la détection des nœuds attaquants afin de les écarter définitivement du réseau. Dans cette section, nous allons présenter une vue globale de la solution que nous avons proposé afin de sécuriser le protocole LEACH. On va donc intégrer à ce dernier des mécanismes simples et robustes pour la détection des nœuds attaquants. LEACH est un protocole de routage hiérarchique pour les RCSFs, il est donc vulnérable à plusieurs types d'attaques. Dans notre étude, nous allons nous intéresser aux attaques de type trou noir et trou gris que nous avons bien décrit dans le chapitre précédent. Dans ce type d'attaque, le nœud attaquant se positionne comme Cluster-Head dans le réseau, il reçoit ainsi toutes les données collectées par les nœuds capteurs de son cluster. Ensuite, au lieu de les transmettre vers la station de base, il supprime ces données reçues et ils les empêchent ainsi d'atteindre la station de base. Par conséquent, ces nœuds attaquants peuvent créer un disfonctionnement total du RCSF hiérarchique.

Lorsqu'on analyse les conséquences de ces deux attaques, la solution la plus intuitive est bien évidemment, mettre en œuvre un système de transmission de données avec demande d'accusé de réception. En effet, pour garantir que les paquets transmis par les nœuds capteurs ont bel et bien atteints leur destination finale qui est la station de base, cette dernière va leur fournir une preuve de la bonne réception de ces paquets sous forme d'un accusé de réception. Notre objectif est de trouver des mécanismes simples à mettre en œuvre, qui nous permettent de détecter ces attaquants le plus rapidement possible afin de les écarter. Quel que soit le mécanisme de sécurité intégré à n'importe quel protocole de routage engendre une perte en terme de performances. Le dilemme sécurité performante, dépend entièrement du type d'application qui aura recours à ces protocoles de routage.

3. Implémentation des attaques

Dans notre travail, nous nous somme intéressé aux attaques de type trou noir et trou gris. Ces attaquants comme expliqué dans la section précédente, s'attaquent au réseau en se positionnant comme étant des Cluster-Heads. Ensuite, selon leurs configuration, soit ils suppriment tous les messages à router (le cas du trou noir), soit ils suppriment qu'une partie de ces messages à router (le cas du trou gris). Pour implémenter ces attaques, nous partons sur l'hypothèse qu'un nœud attaquant dispose d'une ressource énergétique élevée, et d'une puissance de signal supérieure à celle d'un nœud légitime. Ce niveau d'énergie lui permettrait d'avoir une durée de vie maximale dans l'objectif de perturber le réseau de capteurs le plus longtemps possible. La puissance du signal lui permettrait de former des clusters avec le maximum de nœuds et avoir ainsi accès à un maximum de données. L'algorithme 1 décrit le comportement d'un nœud attaquant par rapport à un nœud légitime.

Algorithme 1: Modélisation des attaques : trou noir et trou gris

Entrées : *Mal - Id du nœud attaquant, CH - Cluster Head, BS - Base Station, E - Energie*
V - Nombre total de nœuds, CM - Membre d'un cluster, X (n_i) - Nombre aléatoire généré par n_i
TAtt - Type d'attaque à appliquer, n_i - Nœud i, T (n_i) - Seuil calculé par chaque nœud.

phase désignation CH :
$E_{init} = E_0$, $\forall i \in V$
if n_i== *mal* **then**
$CH = n_i$
else
$\forall i \in V$-{*Mal*} calcul $X(n_i)$ et $T(n_i)$
if $X(n_i) < T(n_i)$ **then**
$CH = n_i$
endif
endif
CH diffusent les Messages *ADV*
Tous les *CM* vont joindre leurs *CH*

phase agrégation :
CH crée *TDMA* Schedule
CM envoient les données au *CH*
if *CH != mal* **then**
agrégation des données et routage vers la BS
else
if *TAtt == Trou-Noir* ***then***
supprime tous les messages reçus
else
$\forall i \in CM$ et $P_{(i, T)}$ *paquet reçu du nœud i à l'instant T*
Ordre_Suppression= Random (True, False)
if *Ordre_Suppression == True* **then**
Supprime ($P_{(i, T)}$)
else
Agrégation ($P_{(i, T)}$)
endif
endif
endif

Contrairement au trou noir, le trou gris ne supprime pas la totalité des paquets de données reçus. Pour simuler ce cas de configuration le plus réellement possible, nous avons développé une fonction qui retourne une décision aléatoire sur l'opération (transmettre/supprimer) à appliquer sur un paquet de données reçu de la part d'un Nœud *i* à un instant donnée *T*. Ce comportement aléatoire des trous gris, rends leurs détection très

difficile par rapport aux trous noirs. Dans la section suivante, nous allons expliquer la stratégie que nous avons développée pour détecter ces attaques.

4. Mécanismes de détection des trous noirs et gris dans DHD-LEACH

Comme nous l'avons évoqué précédemment, la solution la plus intuitive pour la détection des trous noirs et gris consiste à mettre en place un mécanisme d'envoi de données avec demande d'accusés de réception (ACK). En effet, le seul moyen de savoir si la station de base à bien reçu les données qui lui ont été transmises, c'est que cette dernière acquitte la bonne réception de ces données avec un message de type accusé de réception. Notre objectif n'est pas de faire une demande d'accusé de réception pour tout paquet envoyé d'un nœud vers la station de base. Un tel comportement multiplierait au moins par deux le nombre de messages circulant sur le réseau, ce qui mène à la saturation de ces derniers, et surtout une réduction significative de la batterie des capteurs, qui est la ressource la plus précieuse dans un capteur, vu qu'elle est généralement irremplaçable. Nous allons expliquer dans ce qui suit, les mécanismes utilisés pour détecter les attaques trou noir et gris.

4.1 Mécanisme de détection des trous noirs

Dans le but de détecter les trous noirs qui exploitent le protocole LEACH pour devenir Cluster-Heads, nous avons ajouté à l'algorithme de LEACH, une nouvelle phase que nous avons nommé *phase validation Cluster-Heads*, qui s'effectue avant la phase de transmission de données. Cette phase a pour objectif de tester si le Cluster-Head auquel le nœud vient de s'adhérer, achemine correctement les données vers la station de base. Ainsi, le nœud capteur envoie une requête type « Hello », dans laquelle, il demande à la station de base de lui renvoyer un accusé de réception sur cette requête, confirmant ainsi sa bonne réception. Cette opération est effectuée une seule fois avant de passer à la phase de transmission des données.

La Figure IV-1 présente un organigramme sur l'ensemble des étapes de notre nouveau protocole DHD-LEACH, auquel nous avons ajouté la nouvelle phase *validation Cluster-Heads*, qui permet de détecter rapidement les trous noirs afin de les écarter.

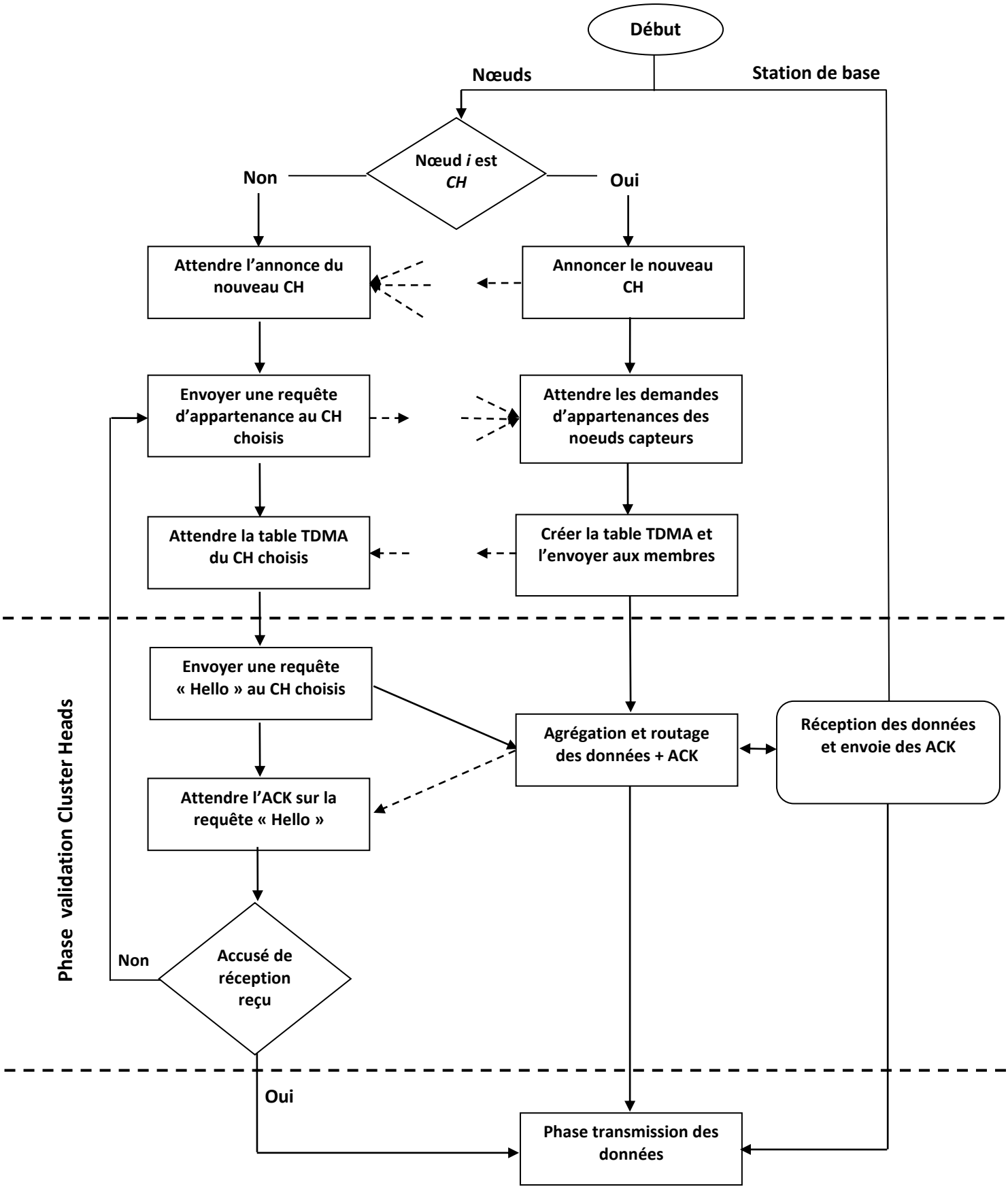

Figure IV-1. Organigramme de la phase validation Cluster-Heads.

Dans notre nouveau protocole DHD-LEACH, le nœud capteurs précède la phase de transmission de données par une nouvelle phase nommée *validation Cluster-Heads*. Avant qu'un nœud capteur envoie ses données collectées, il teste d'abord la validité du Cluster-Head auquel il vient s'adhérer, en envoyant une requête « Hello » qui contient une demande d'accusé de réception sur la requête elle-même. Après l'envoie de cette requête, deux cas peuvent se présenter :

Cas 1 : Accusé de réception reçu avec succès

Dans ce cas-là, le nœud capteurs reçoit l'ACK sur sa requête « Hello ». Il va constater que le Cluster-Head auquel il vient de s'attacher fonctionne correctement, donc le nœud peut passer à la phase de transmission des données.

Cas 2 : Accusé de réception non reçu

Dans ce cas, le nœud va constater que le Cluster-Head auquel il vient de s'attacher ne fonctionne pas correctement. Soit il est défectueux ou bien en panne (par exemple à cause d'un manque d'énergie), soit c'est un attaquant de type trou noir. Dans les deux cas, le nœud capteurs ajoute ce Cluster-Head suspect à une liste noire (*Blacklist)* qui va contenir l'ensemble des Cluster-Heads à éviter durant les prochains rounds. Ensuite, le nœud, refait une autre demande d'appartenance à un autre Cluster-Head.

Cette nouvelle phase permet de détecter rapidement les attaques de type trou noir et présente l'avantage d'être moins gourmande en termes de ressource. En effet, le contrôle de validité du Cluster-Head s'effectue par l'envoi d'un seul message, ce qui est négligeable par rapport à la totalité des messages envoyés par un nœud durant la phase de transmission.

Mais cette solution n'est pas suffisante pour détecter les trous gris. En effet, les trous gris laissent passer certains messages. Donc on peut avoir un cas de configuration, où le trou gris va transférer la requête « Hello » ainsi que son accusé de réception, mais supprime la transmission de données. Donc, la nouvelle phase *validation Cluster-Head* n'est pas suffisante pour détecter les trous gris. Pour cela, nous ajoutons un autre mécanisme pour pouvoir les détecter.

4.2 Mécanisme de détection des trous gris

A cause du comportement variable des trous gris, leurs détection est plus difficile par rapport aux trous noirs. Pour détecter ces trous gris, nous proposons d'effectuer des demandes d'accusés de réception sur les paquets de données envoyés de manière irrégulière. Cette irrégularité dans la demande d'accusés de réception a plusieurs objectifs. Le premier est d'éviter la saturation du réseau par la quantité de messages échangés. Le deuxième est d'empêcher les trous gris de déduire quand est ce que les nœuds capteurs demandent des accusés de réception. Cela peut permettre au trou gris de faire passer les messages pour éviter

sa détection par les nœuds de son cluster. La probabilité qu'un trou gris supprime les messages reçus n'est pas connue. Pour que notre solution soit efficace, les nœuds capteurs ne doivent seulement pas demander un accusé de réception sur le dernier message envoyé. Mais aussi, sur l'ensemble des messages envoyés sans demande d'accusé de réception. Ce mécanisme permet aux nœuds de vérifier que la station de base a bien reçu la totalité des messages qu'ils ont envoyés. La Figure IV-2 présente un organigramme sur le fonctionnement de la phase transmission de données dans DHD-LEACH.

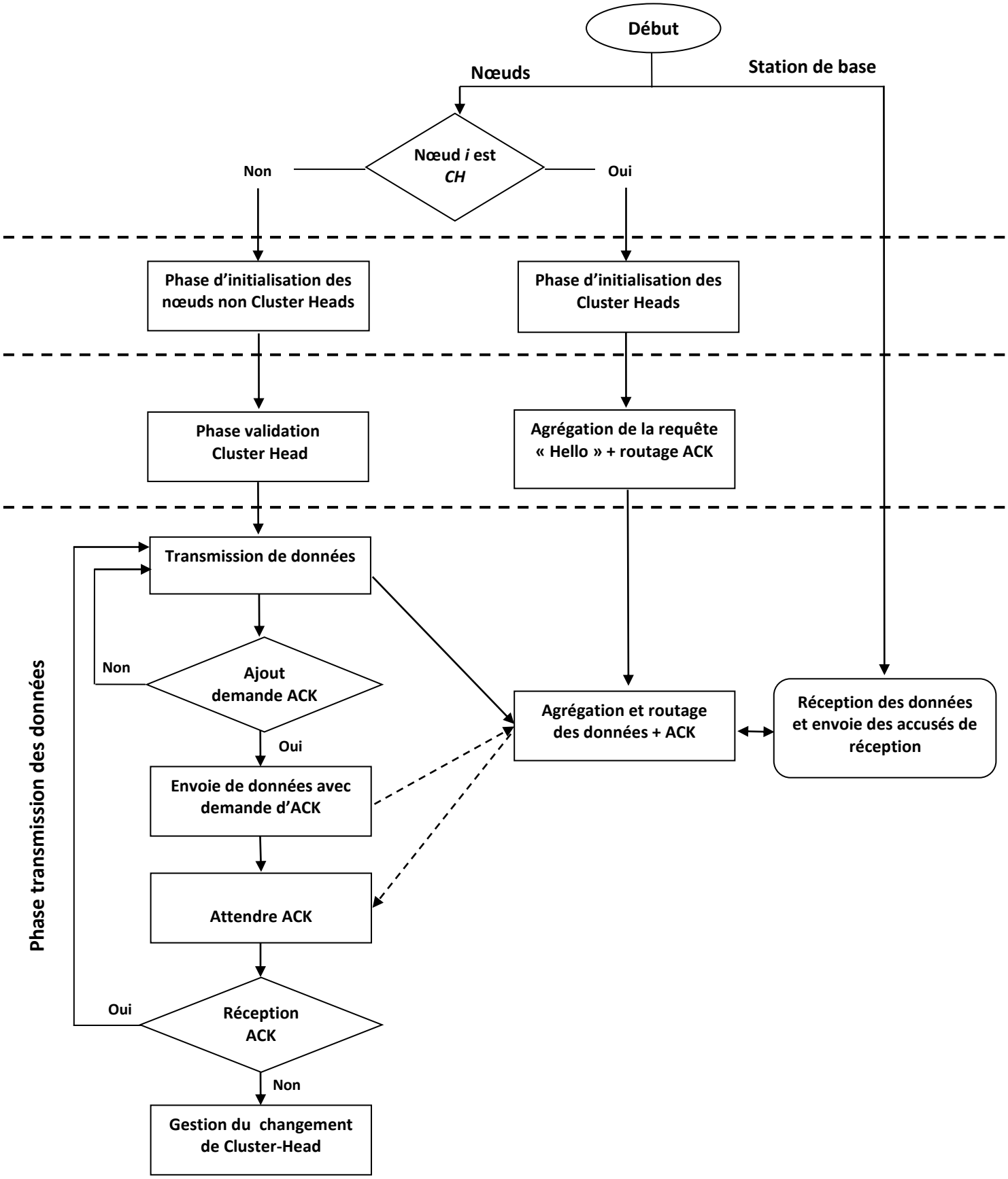

Figure IV-2. Organigramme de la phase transmission dans DHD-LEACH.

En précisant ainsi à la station de base d'acquitter l'ensemble des paquets qui lui ont été envoyés sans demande d'ACK. Le nœud capteur, pourra vérifier si l'ensemble des paquets qu'il a envoyés ont bien atteints leur destination. Lorsqu'un nœud envoi ses données vers la station de base à travers son Cluster-Head. Dans le cas où le Cluster-Head est un trou gris, on peut avoir les cas suivants :

Cas 1 : le trou gris supprime la requête « Hello »

Dans ce cas, le trou gris se comporte comme un trou noir. Le nœud capteur ne reçoit pas d'ACK sur sa requête, il ajoute donc ce Cluster-Head suspect à une liste noire (*Blacklist*) qui contient l'ensemble des Cluster-Head à éviter durant les prochains rounds. Ensuite, le nœud, refait une autre demande d'appartenance à un autre Cluster-Head.

Cas 2 : le trou gris supprime un paquet de donnée avec demande d'ACK

Dans ce cas, le nœud n'ayant pas de réponse sur sa demande d'accusé de réception, va constater que son Cluster-Head est soit malicieux ou en panne. Dans les deux cas, le nœud arrête la transmission des données avec lui, change de Cluster-Head et ajoute le Cluster-Head suspect à la liste noire (blacklist).

Cas 3 : le nœud reçoit l'accusé de réception demandé

Dans ce cas de configuration, deux cas peuvent se présenter. Soit, l'accusé de réception confirme la bonne réception de tous les paquets envoyés par le nœud, donc le nœud va constater qu'il n'y a aucun risque et continue à envoyer ses données jusqu'à la prochaine demande d'accusé de réception. Soit l'accusé de réception ne confirme qu'une partie des paquets envoyés, alors le nœud va constater que son Cluster-Head est suspect, soit il ne fonctionne pas correctement. Dans les deux cas, le nœud va cesser d'envoyer des données à travers ce Cluster-Head. Ensuite, il refait une autre demande d'appartenance à un autre Cluster Head après avoir ajouté l'ancien à la liste noire.

Pour la mise en place de la solution proposée avec ses différents mécanismes, nous allons présenter dans la section qui suit, les différents formats des paquets correspondants aux différents messages que nous avons introduits.

4.3 Formats des paquets dans DHD-LEACH

Dans cette section, nous allons présenter les formats des paquets de notre nouveau protocole DHD-LEACH après avoir modifié ceux de LEACH. Les paquets de la phase initialisation restent les mêmes (messages de type ADV, REQ-JOIN et TDMA). Nous allons définir le format d'un paquet de données, et le format d'un paquet de type accusé de réception dans DHD-LEACH.

4.3.1 Format d'un paquet de données dans DHD-LEACH

Comme la Figure IV-3 l'indique, un paquet de données dans notre nouveau protocole DHD-LEACH se présente comme l'ancien paquet de données sous LEACH, auquel nous avons ajouté deux champs : « *FLAG ACK* » et « *Id--Packet* ».

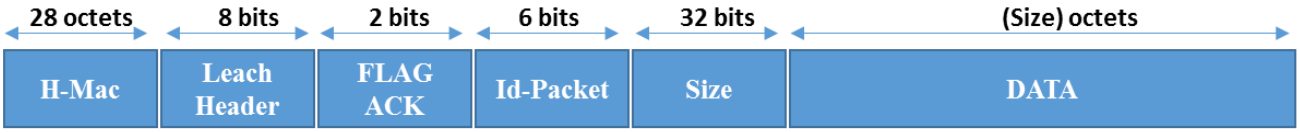

Figure IV-3. Format d'un paquet de données dans DHD-LEACH.

Id_Packet : indique l'identifiant du paquet au niveau applicatif. Chaque identifiant est unique à chaque paquet. Ces identifiants sont séquentiels. Ils sont utilisés pour identifier les messages dans les accusés de réception.

FLAG ACK : c'est avec cet octet, que le nœud émetteur d'un paquet de données, indique à la station de base, s'il faudrait envoyer un accusé de réception sur les paquets de données qu'il a envoyé. Comme illustré par le Tableau IV-1, ce champ peut avoir 4 valeurs :

FLAG ACK	Signification
0	Le nœud émetteur envoie un paquet de données sans demande d'accusé de réception.
1	Le nœud émetteur envoie un paquet de données, avec une demande d'accusé de réception sur le paquet lui-même.
2	Le nœud émetteur envoie un paquet de données, avec une demande d'accusé de réception sur le paquet lui-même. La station de base renvoie avec cet accusé, le nombre de paquets qu'elle a reçu de la part de ce nœud sans acquittement.
3	Le nœud émetteur envoie un paquet de données, avec une demande d'accusé de réception sur le paquet lui-même. La station de base renvoie avec cet accusé, le nombre de paquets qu'elle a reçu de la part de ce nœud sans acquittement ainsi que leurs identifiants.

Tableau IV-1. Signification des valeurs du champ *FLAG ACK*.

4.3.2 Format d'un paquet accusé de réception dans DHD-LEACH

Comme illustré par la Figure IV-4, le format d'un paquet de type accusé de réception, comporte une entête LEACH standard, auquel nous ajoutons un champ « *Nbr-ACK* » et un

vecteur des identifiants des paquets envoyés sans demande d'accusé de réception, que nous appelons « *IDs_PKT* ».

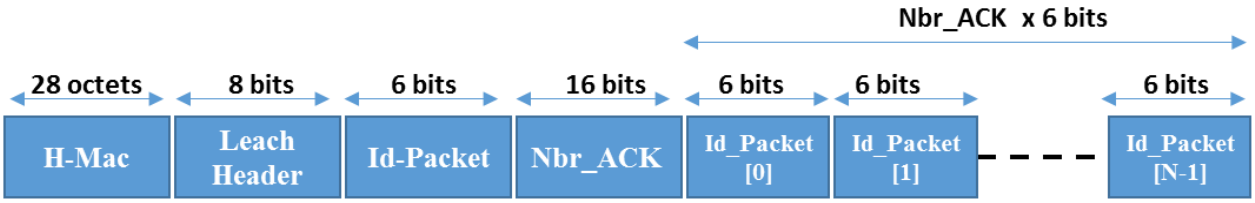

Figure IV-4. Format d'un accusé de réception dans DHD-LEACH.

Nbr_ACK : indique le nombre de paquets que la station de base a reçu de la part d'un nœud émetteur sans acquittement (c'est la réponse sur la demande d'accusé de réception, demandée avec le *FLAG ACK* = 2).

IDs_PKT : vecteur contenant les identifiants des paquets envoyés sans demande d'accusé de réception (c'est la réponse sur la demande d'accusé de réception, demandée avec le *FLAG ACK* = 3).

5. Tunnel sécurisé dans DHD-LEACH

La solution que nous avons proposée pour détecter les trous noirs et gris, répond parfaitement à nos besoins et à la problématique que nous avons fixée au départ. En effet, on pourrait aisément imaginer un attaquant intelligent qui se mettrait à renvoyer des faux accusés de réception dans le but de tromper les noeuds.

Pour pallier à ce problème, nous proposons d'ajouter à notre solution qui est la transmission de données avec demande d'accusé de réception, une méthode de transmission de paquets à travers un tunnel sécurisé. Un tunnel sécurisé, est un mécanisme qui encapsule tous les paquets que nous avons définis dans la section précédente dans des paquets chiffrés.

L'objectif d'un tel tunnel est de garantir que l'ensemble des paquets de données et accusés de réception circulant sur le réseau, reste confidentiel, et de détecter toute forme d'altération. Plusieurs algorithmes de chiffrement symétriques, asymétriques et des méthodologies de changement de clés de chiffrement ont été proposés **[ZHU04][MAL97] [OLI06][CHE08]**. Notre objectif n'est pas de mettre en œuvre de nouveaux algorithmes de chiffrement et de gestion de clés de chiffrement performants, mais juste de montrer qu'en appliquant un tel mécanisme nous augmentons de manière significative la robustesse de notre solution.

5.1 Format des paquets chiffrés dans DHD-LEACH

Pour la réalisation d'un tunnel de communication sécurisé entre les nœuds et la station de base, nous proposons d'encapsuler tous les paquets circulant dans le réseau dans des paquets chiffrés. Ces paquets se présentent sous le format illustré par la Figure IV-5.

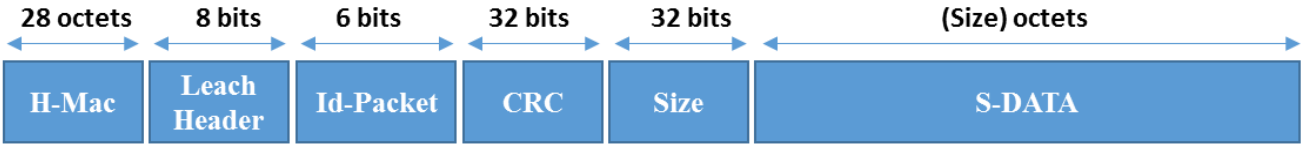

Figure IV-5. Format d'un paquet chiffré dans DHD-LEACH.

CRC : (Cyclic Redundancy Check) : signature du paquet à transmettre en clair avant le chiffrement.

Size : taille des données (paquet) chiffrées.

S-DATA : buffer qui contient les données chiffrés.

Pour encapsuler un paquet de données ou bien un paquet d'accusé de réception dans un paquet chiffré, nous devons d'abord procéder à la sérialisation. La sérialisation consiste à transformer une structure de donnée (en l'occurrence, nos paquets qui sont définis dans des structures de données) en un flux d'octets. L'algorithme 2 illustre les principales étapes de cette phase d'encapsulation.

Algorithme 2 : Encapsulation des paquets

Entrée : *Pkt_in - Paquet en clair,*
Key - clef de chiffrement,
Sortie : *SPkt_out - Paquet chiffré,*

```
byte [] Palin_Buffer = Serialize(Pkt_in) ;
Integer  CRC = Cyclic_Redundancy_Check(Palin_Buffer);
byte[] Encrypt_Buffer = Encrypte (Palin_Buffer, key);

SPkt_out.size = sizeof (Encrypt_Buffer);
SPkt_out.CRC = CRC;
SPkt_out.S-DATA = Encrypt_Buffer ;
```

À la réception du paquet chiffré par la station de base, ce dernier est déchiffré puis désérialisé et finalement analysé. L'algorithme 3 illustre les principales étapes de cette phase de décapsulation.

Algorithme 3 : décapsulation des paquets

Entrée : *SPkt_in - Paquet chiffré,*
Key - clef de déchiffrement,
Sortie : *Pkt_out - Paquet décapsulé,*

```
byte[] Encrypt_Buffer = SPkt_in.S-DATA ;
Integer CRC_out = SPkt_out.CRC;
byte[] Palin_Buffer =  Decrypte (Encrypt_Buffer, key);
Integer  CRC = Cyclic_Redundancy_Check (Palin_Buffer);

if CRC_out != CRC  then
Error_Manager(SPkt_in) ;
endif

Pkt_out = Deserialize(Palin_Buffer);
```

Les données transmises entre les nœuds et la station de base sont chiffrées à l'aide d'un algorithme de chiffrement symétrique à base d'une clé secrète. Nous avons utilisé l'algorithme de chiffrement AES (Advanced Encryption Standard) **[SEC11]**. Bien évidement il existe de nombreux algorithmes libres (Open source) qu'on pourrait utiliser pour le chiffrement des données **[SEC11]**. Le mécanisme de chiffrement à base de clé secrète, nécessite un mécanisme de gestion de ces clés. Pour se faire, nous allons proposer dans la section qui suit un protocole de gestion de clés.

5.2 Protocole d'échange de clés de chiffrement

Nous proposons dans cette section, un protocole pour garantir l'échange de clés de chiffrement entre la station de base et les nœuds. Nous supposons qu'à l'état initial du système $T=0$, chaque nœud i partage une clé secrète $K_{(i, 0)}$ avec la station de base. Pour éviter tout risque de casser la clé de chiffrement, nous proposons que cette dernière soit changée après une période de temps de durée bien déterminée T. Pour éviter que les nœuds attaquants puissent déduire la durée T, elle doit être déterminée de manière aléatoire.

Pour changer la clé de chiffrement $K_{(i, T)}$ utilisée durant le cycle T, le nœud génère une nouvelle clé notée $K_{(i, T+1)}$. Cette clé est transmise dans le tunnel sécurisé, dans un paquet chiffré avec l'ancienne clé $K_{(i, T)}$. A la réception de cette nouvelle clé par la station de base, elle met à jour cette nouvelle clé de chiffrement et renvoi un accusé de réception chiffré avec la nouvelle clé $K_{(i, T+1)}$. Lorsque le nœud reçoit un accusé de réception chiffré avec la nouvelle

clé, ce dernier saura que la station de base à mis à jour la nouvelle clé. Ce mécanisme garantit un changement de clé secrète à tout instant, sans risque que celle-ci ne soit dévoilée puisque toutes les transmissions sont chiffrées.

Conclusion

Dans ce chapitre, nous avons proposé un nouveau protocole de routage hiérarchique que nous avons appelé DHD-LEACH (Dynamic Hole Detection in LEACH). Ce protocole apporte de nombreux mécanismes pour sécuriser le protocole de base LEACH, en le rendant ainsi capable de détecter tout intrus appliquant des attaques de type trou noir et trou gris. Au-delà de ces deux types d'attaques qui ont été notre problématique initiale, notre nouveau protocole permet également de garantir la confidentialité des données et détecter toute forme d'altération des données, tout en respectant les contraintes des ressources limitées des RCSFs, qu'elles soient en termes d'énergie ou de puissance de calcul.

Dans le chapitre qui suit, nous allons présenter l'environnement de simulation que nous avons utilisé, ainsi que différentes illustrations présentant la mise en œuvre de notre approche.

CHAPITRE 5

Réalisation

Introduction

Tel qu'on l'a montré au cours du chapitre précédent, l'objectif principal de notre travail est la mise en œuvre d'une solution qui se charge de sécuriser le protocole de routage LEACH. De ce fait, nous avons établi un nouveau protocole DHD-LEACH qui est en mesure de détecter les trous noirs et gris visant le protocole de routage LEACH. L'objectif de ce chapitre est donc de démontrer l'efficacité du protocole DHD-LEACH par rapport au protocole LEACH en termes de sécurité ainsi que d'autres métriques de performances, via l'implémentation et la simulation des deux protocoles.

Pour cela, nous commençons par définir quelques environnements de simulation et les outils nécessaires pour notre simulation. Ensuite, nous décrirons la mise en œuvre des processus décrits dans le chapitre précédent. Nous terminons ce chapitre par une présentation des résultats relevés lors des tests de performances des deux protocoles LEACH et DHD-LEACH.

1. Environnements de simulation

Les RCSFs nécessitent une phase de test avant la mise en place. Pour cela, la solution la plus fiable et la moins coûteuse consiste en « la simulation ». La simulation des RCSFs consiste principalement à la reproduction du comportement des nœuds capteurs du monde réel dans le monde virtuel. Dans ce qui suit, nous citons quelques simulateurs de réseaux **[ABE12].**

1.1 Network simulator (NS2)

« NS2 » désigne la deuxième version du NS (Network Simulator). C'est un simulateur à événements discrets dans le temps (Une simulation à événements discrets est une modélisation informatique où l'état d'un système est représenté par une séquence chronologique d'événements discrets. Chaque événement arrive à un instant donné et modifie l'état du système). Il a été conçu pour être un simulateur générique. Par conséquent, il constitue un support important pour la simulation des protocoles standards tels que TCP, IP, et ce, aussi bien à travers les réseaux filaires qu'à travers les réseaux sans fil. Il supporte également les réseaux ad hoc et les réseaux de capteurs et comprend quelques protocoles implémentés tel que Directed Diffusion. NS est un simulateur multicouches et son développement suit une approche orienté-objet en utilisant deux langages de programmation : C++ et OTCL (Object Tools Command Langage dérivé de TCL). Le noyau, les modèles, les protocoles et les composants de base sont écrits en C++, tandis que les interfaces sont implémentées en OTCL. Des scénarios de simulation, écrits en OTCL, peuvent être introduits afin de décrire les conditions de la simulation, essentiellement, la topologie du réseau, les caractéristiques des liens physiques, les protocoles utilisés et les communications qui ont eu lieu. L'utilisation de l'OTCL permet aussi à l'utilisateur de créer ses propres procédures. Une

fois le scénario défini, NS entame la simulation et génère un fichier de trace « .log ». Ce fichier est interprété par l'outil NAM (Network Animator), associé au simulateur NS, afin de visualiser des animations de la simulation (transfert des paquets, taille des paquets, remplissage des files d'attentes).

1.2 GloMoSim

GloMoSim (Global Mobil Simulator) est un environnement de simulation pour les réseaux sans fil de grande taille. Il a été développé par PCL (Parallel Computing Labaratory) de l'UCLA (University of California, Los Angeles) dans le cadre du projet DARPA GLOMO (GLObal MObil information system). GloMoSim est construit suivant une approche basée sur les couches, qui est similaire au modèle à sept couches de l'OSI. Les couches constituant l'architecture de ce simulateur sont les suivantes : mobility, radio propagation, radio model, packet reception models, data link (MAC), network (routing), transport et application. Des fonctions standards (API) sont utilisées pour assurer la communication et l'échange des services entre ces couches. Avant de démarrer une simulation, l'utilisateur doit saisir les paramètres de configuration dans deux fichiers : app.conf et config.in. Le premier contient les paramètres relatifs à la couche applicative tandis que le second contient la description des couches physique, MAC, réseau et transport, et la description du canal radio, scénarios de mobilité, l'environnement (terrain, nombre de noeuds, etc.) et la durée de la simulation.

1.3 TOSSIM

TOSSIM est un simulateur à événements discrets basé sur la programmation par événements en utilisant un langage spécifique appelé NesC. Il a été conçu pour simuler les réseaux de capteurs en utilisant la plateforme TinyOS. Son principal but est de créer une simulation très proche de ce qui se passe dans ces réseaux dans le monde réel. Chaque interruption dans le système est capturée. Pour une compréhension moins complexe de l'activité d'un réseau, Tossim est équipé d'une interface graphique, TinyViz, qui donne un aperçu de notre réseau de capteurs à tout moment.

1.4 OMNeT ++

OMNeT ++ IDE (Integrated Development Environment) est basé sur la plateforme Eclipse. C'est un environnement open source qui fournit des outils pour la création et la configuration des modèles de réseaux (les fichiers NED et INI). Il peut être utilisé pour la modélisation des réseaux câblés, des communications sans fil, des protocoles, des multiprocesseurs de systèmes distribués et l'évaluation des aspects de performance des systèmes complexes ainsi la validation d'architectures matérielles. Actuellement, Ce simulateur est utilisé par des dizaines d'universités pour la validation de nouveaux matériels et logiciels, ainsi que pour l'analyse des performances et l'évaluation des protocoles de communications. OMNeT++ est devenu rapidement une plateforme de simulation populaire

que ce soit pour la communauté des scientifiques ou des industriels. Un modèle de simulation OMNeT++ consiste en un ensemble d'entités appelées « *Modules* » qui communiquent entre elles en échangeant des « *Messages* » qui seront émis à travers des « *Connections* » et des « *gates* ».

Cependant, quelque soit la qualité de la simulation, elle ne remplace pas totalement l'expérimentation. Le tableau V-1 décrit les avantages et les inconvénients des simulateurs décrits précédemment :

Simulateur	Avantages	Inconvénients
NS2	-NS2 est un logiciel de simulation multicouche. -Il est complètement libre et existe pour plusieurs plateformes. -Son développement orienté-objet permet l'ajout de composants à la demande. -Sa popularité a permis l'enrichissement de sa bibliothèque de protocoles.	-Il n'est pas destiné aux RCSFs, ce qui limite beaucoup son utilisation dans ce domaine. -Ses modèles standards contiennent peu de paramètres de configuration. -La dépendance entre les modèles rend difficile l'ajout de nouveaux modèles. -Inadapté aux réseaux de grande taille.
GloMoSim	-L'exécution parallèle augmente considérablement la sociabilité de la simulation : des réseaux de cent miles noeuds peuvent être simulés. -GloMoSim est disponible gratuitement et en open source pour Windows et Linux.	-L'utilisation de GloMoSim nécessite une maîtrise du langage PARSEC. -Les versions disponibles de GloMoSim n'exploitent pas totalement la puissance du parallélisme et ce, afin de favoriser QualNet (version commerciale de GlomoSim).
TOSSIM	-Tossim simule fidèlement le comportement d'un réseau en s'appuyant sur la plateforme TinyOs. En effet le code de simulation peut être exécuté directement dans un capteur utilisant le système TinyOS. -Il simule un réseau d'une manière simple et efficace. -Il supporte un grand nombre de noeuds qui peut aller jusqu'à un millier. -Grâce à l'interface graphique TinyViz, on peut visualiser les échanges radios pour avoir une vue globale du réseau.	-Tossim oblige les utilisateurs à exécuter le même code sur tous les capteurs. Pour corriger ce problème, les capteurs doivent être identifiés par des numéros et des conditions doivent être utilisées dans le code pour différencier le rôle de chaque capteur. -TOSSIM fournit seulement une abstraction de certains phénomènes du monde réel. -Manque de souplesse puisqu'il est lié à une interface de visualisation conçue séparément.
OMNeT++	-Son approche modulaire permet la combinaison et la réutilisation des modèles de simulation. -Le développement avec OMNeT++ se fait en C++, ce qui facilite son intégration dans	-OMNeT++ n'est pas prévu pour la simulation de RCSFs. -Plusieurs concepteurs travaillent sur OMNeT++ pour l'enrichir de modèles destinés aux réseaux sans fil. Toute

	d'autres environnements de développement. -Il offre des bibliothèques très riches qui comprennent des supports d'entrées/sorties, manipulation de données et représentations graphiques. -La description des modules se fait avec le langage NED qui est simple à utiliser. -L'interface graphique GNED d'OMNeT++ permet de construire des modules avec une génération automatique de la description NED.	fois ces travaux étant indépendants, il n'existe aucune plateforme pour les regrouper.

Tableau V-1. Avantages et inconvénients de quelques simulateurs.

2. Description des outils nécessaires pour notre simulation

Après avoir étudié quelques environnement de simulation des réseaux, on a choisis OMNeT ++ comme environnement de simulation grâce à son architecture modulaire. Comme OMNeT++ n'est pas prévu pour la simulation des RCSFs, on va utiliser la plateforme « Castalia » qui est basée sur OMNet ++ pour pouvoir simuler les réseaux de capteurs.

2.1 Présentation de OMNeT ++

OMNeT++ (Objective Modular Network Test-bed in C++) est un simulateur d'évènement basé sur le langage C++, destiné à simuler les protocoles réseau et les systèmes distribués **[MAL05]**. Il est totalement programmable, paramétrable et modulaire. C'est une application open-source développée par Andras VARGA, chercheur à l'Université de Budapest. Il est utilisé dans la modélisation et la simulation des systèmes utilisant des approches à événement discret. Parmi eux on peut citer :

- La modélisation des réseaux de communications.
- La modélisation de différents protocoles.
- La modélisation des réseaux de fils d'attente.
- Validation des architectures hardwares.
- Évaluation de performance d'aspects des systèmes complexes.

2.2 Architecture d'OMNeT ++

Le fonctionnement d'OMNeT ++ repose entièrement sur l'utilisation de *modules* qui communiquent entre eux par le biais de *messages*. Les modules de base sont appelés les *simple module*. Ceux-ci sont regroupés en *compound modules*. Les compounds modules peuvent eux mêmes être regroupés en compound modules**.** Ces modules sont organisés

hiérarchiquement et le nombre de niveaux hiérarchiques n'est pas limité **[OMN14]**. L'architecture modulaire du simulateur OMNeT++ est décrite par la Figure V-1.

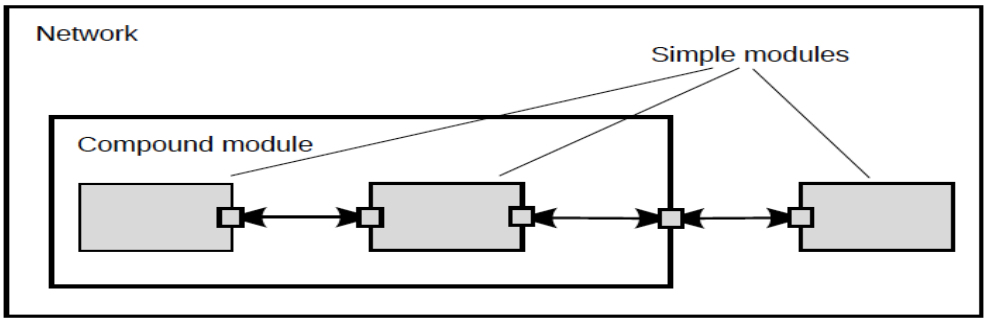

Figure V-1. Architecture modulaire du simulateur OMNeT++.

Les modules sont codés en C++ et sont des instances du type de base *module*. L'architecture est construite de telle sorte que les *simples modules* sont à la fois les émetteurs et destinataires des messages. Les *compounds modules* se contentent de relayer les messages aux simples modules de façon transparente. On peut attribuer différents paramètres aux connections reliant les modules: des délais de propagation, des débits de données, des taux d'erreur, etc. Les messages sont transmis par le biais de portes « *gates* ». Les portes sont les interfaces d'émission et de réception des modules et une connexion peut relier une porte d'entrée à une porte de sortie. On ne peut créer des connections que dans un seul niveau de l'hiérarchie des modules. Il est par exemple impossible de créer une connexion directe entre un module et un sous-module d'un autre module de même niveau dans la hiérarchie.

2.3 Le langage NED

La définition de la structure et la topologie du réseau se fait par le biais du langage NED. Une description NED est quasiment toujours constituée de la déclaration de simple module, de définitions de compound modules et une définition de réseau. Typiquement, la déclaration d'un simple module décrit les interfaces de ce module et ses paramètres. La définition d'un compound module comprend la déclaration des interfaces externes du module ainsi que la définition de ses sous modules et leur inter connexion. La définition d'un réseau définit l'ensemble d'un modèle comme une instance de type *Module* **[OMN14]**.

2.4 Les principaux fichiers d'OMNeT++

- **Fichier (.ned) :** utilise le langage de description du réseau NED, il peut être utilisé en 2 modes : mode graphique ou code. Ils permettent de décrire les paramètres et les ports du module.

- **Fichier (.ini) :** ce fichier est lié avec le fichier Ned, il permet à l'utilisateur d'initialiser les paramètres de configuration des différents modules ainsi la topologie du réseau.

- **Fichier (.msg) :** les modules communiquent en échangeant des messages. Ces derniers peuvent être déclarés dans un fichier dont l'extension est (.msg) où l'on peut ajouter des champs de données. OMNet++ traduira les définitions de messages en classe C++.

2.5 La plateforme Castalia

Le simulateur OMNeT ++ n'est pas spécialisé pour simuler les réseaux de capteurs sans fil. Pour cela, il existe plusieurs plateformes et simulateurs basés sur OMNeT++ qui essayent d'introduire ce manque comme : Mobility FrameWork, Mixim et Castalia. La plateforme « Castalia » est la plus appropriée pour notre projet. C'est un simulateur de très haut niveau, développé par *NICTA (National ICT Australia Ltd)* pour la simulation des réseaux de capteurs sans fil sur OMNeT++. Castalia est destinée aux chercheurs et développeurs qui souhaitent tester leurs algorithmes et/ou protocoles distribués dans les modèles de canaux et de radio sans fil réalistes. Le comportement des nœuds dans ce simulateur est réaliste **[CAS11].**

3. Implémentation et déroulement

Pour implémenter le protocole que nous avons proposé dans le chapitre précédent, nous avons procédé en trois étapes. Nous avons commencé par implémenter le protocole LEACH. Ensuite, nous avons implémenté les attaques *trou noir* et *trou gris*. Enfin, nous avons implémenté notre nouveau protocole DHD-LEACH. Pour illustrer le comportement des nœuds, nous avons utilisé l'interface graphique GNED d'OMNeT++. Nous allons illustrer les différentes phases du déroulement du protocole LEACH de base, le comportement des attaquants et finalement les phases de notre nouveau protocole DHD-LEACH.

3.1 Déroulement du protocole LEACH

Dans cette partie, nous expliquons le déroulement de l'algorithme LEACH. Pour cela, nous allons montrer le déroulement des phases initialisation et transmission de données. Pour se faire, nous avons réalisé une simulation dans un environnement comportant 21 nœuds.

3.1.1 Phase initialisation

Élection des Cluster-Heads

La station de base envoie un broadcast à tous les nœuds du réseau pour annoncer le nouveau round. Chaque nœud génère ainsi un nombre aléatoire *X(i)* et le compare au seuil

T(i). Si le nombre *X(i)* est inférieur au seuil *T(i)*, le nœud se déclare Cluster-Head durant le round courant. Comme illustré par la Figure V-2, nous pouvons voir que les nœuds 6, 7 et 19 sont élus Cluster-Heads. Cet évènement est marqué par le changement d'icône et couleur des Cluster-Heads par rapport aux autres nœuds. Ensuite, les Cluster-Heads 6, 7 et 19 diffusent des requêtes *ADV* sur le réseau pour annoncer leur nouveau statut.

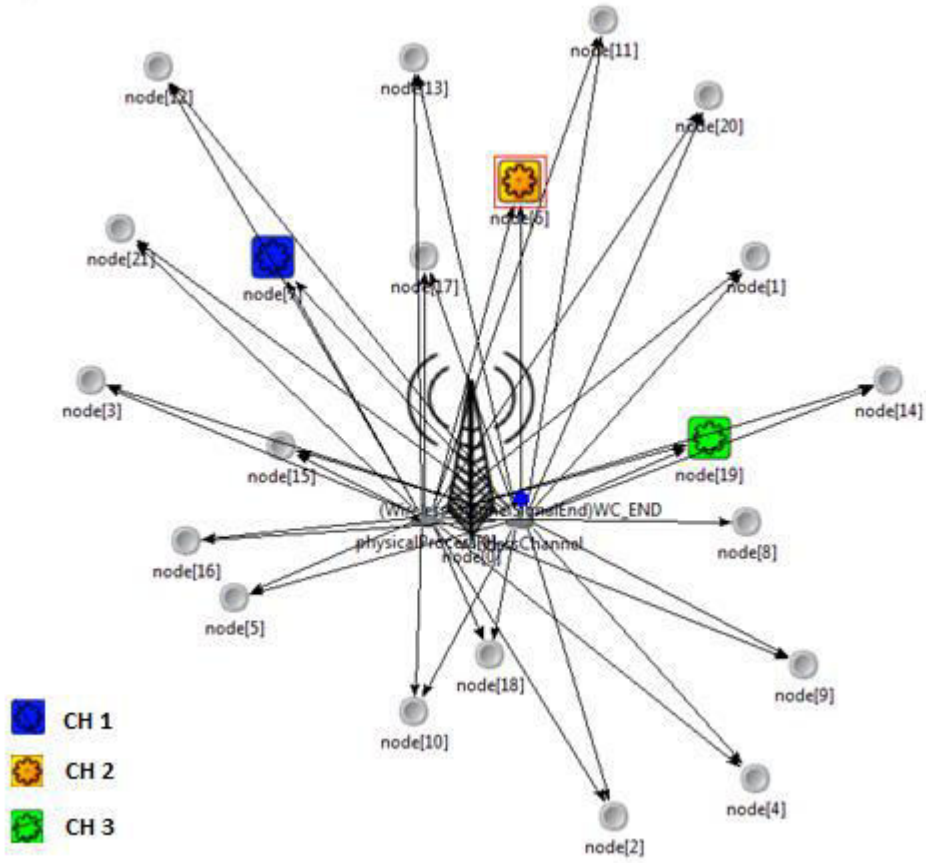

Figure V-2. Élection des Cluster-Heads.

Organisation des clusters

Les noeuds non-Cluster-Heads répondent à l'annonce *ADV* du Cluster-Head le plus proche en termes de puissance du signal reçu. La Figure V-3 illustre la formation des groupes des Cluster-Heads 6, 7 et 19**.** Cet évènement est marqué par le changement de la couleur des noeuds non-Cluster-Heads. Ils prennent ainsi la même couleur que leurs Cluster-Head.

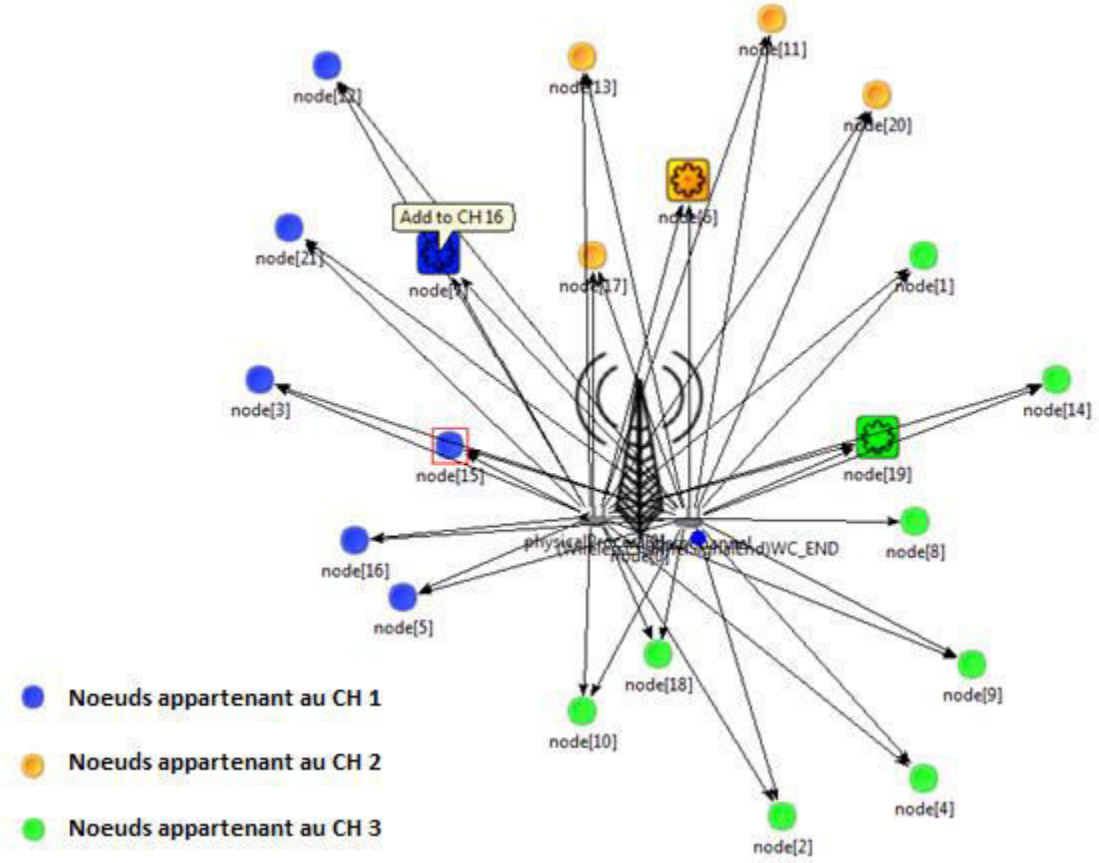

Figure V-3. Formation des clusters.

3.1.2 Phase transmission des données

Après la phase d'organisation des clusters, chaque membre de cluster, attend le début de son slot pour qu'il puisse envoyer ses données à son Cluster-Head. Chaque Cluster-Head agrège ainsi toutes les données issues de son cluster puis l'envoient à la station de base. Comme illustré par la Figure V-4, le Cluster-Head 19 agrège les données reçues de son cluster qui est composé des nœuds 1, 14, 8, 2, 9, 4, 18 et 10 et envoie son résultat d'agrégation à la station de base.

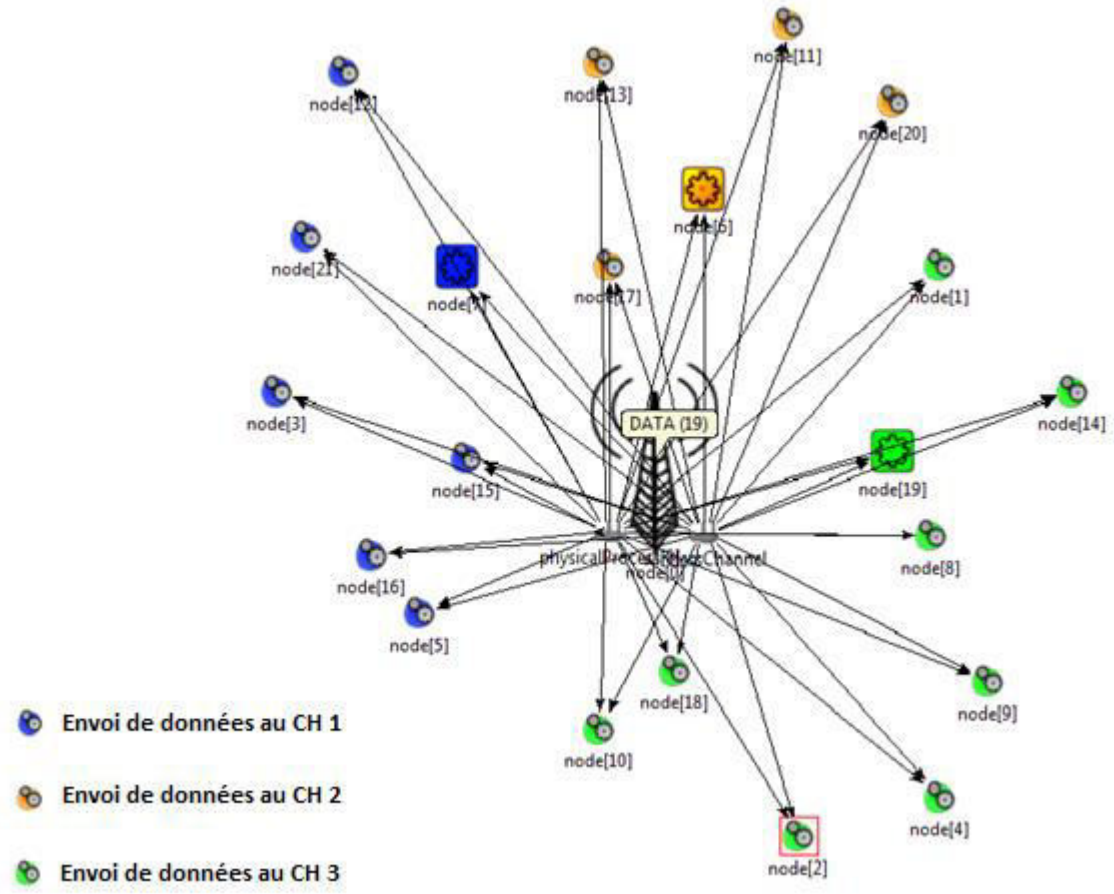

Figure V-4 : Envoi du résultat d'agrégation du Cluster-Head 19 à la station de base.

3.2 Implémentation des attaques

Les Cluster-Heads sont les points les plus vulnérables dans le réseau. Nous avons décidé de centraliser les attaques sur eux. Nous avons implémenté l'attaque *trou noir* et *trou gris* car elles sont considérées comme le type d'attaque le plus dangereux visant LEACH.

Comme illustré par la Figure V-5, nous pouvons voir que le nœud 18 est un nœud attaquant qui s'est positionné comme Cluster-Head dans le réseau. En diffusant son annonce *ADV*, on remarque, qu'il a trompé les nœuds 5, 10, 2 et 4. Ainsi, ils sont devenus membres du Cluster-Head attaquant.

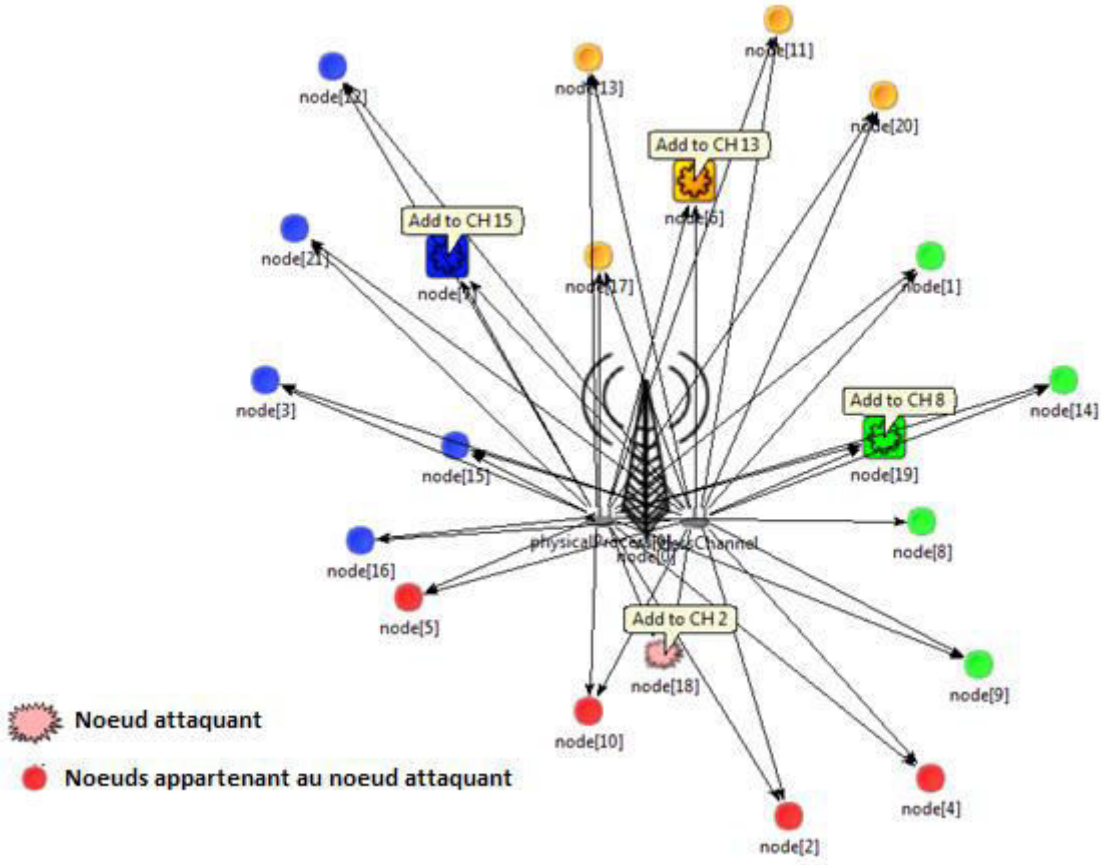

Figure V-5. Appartenance des nœuds capteurs au Cluster-Head attaquant.

3.3 Implémentation de DHD-LEACH

Dans cette partie, nous expliquons le déroulement de l'algorithme DHD-LEACH. Pour cela, nous allons montrer le déroulement de la nouvelle phase *validation cluster-Heads* et *transmission de données*.

3.3.1 Phase validation des cluster-heads

Dans le but de détecter les trous noirs qui exploitent le protocole LEACH pour devenir Cluster-Heads, nous avons ajouté à l'algorithme de LEACH, une nouvelle phase que nous avons nommé *phase validation Cluster-Heads.* Cette phase a pour objectif de tester si le Cluster-Head auquel le nœud vient de s'adhérer, achemine correctement les données vers la station de base. Ainsi, chaque nœud capteur envoie une requête type « Hello », dans laquelle, il demande à la station de base de lui renvoyer un accusé de réception sur cette requête, confirmant ainsi sa bonne réception. Comme illustré par la Figure V-6, tous les nœuds capteurs ont envoyés une requête « Hello » et ils ont reçus un ACK, sauf les nœuds qui appartiennent au Cluster-Head attaquant 18, qui sont toujours en attente d'un ACK.

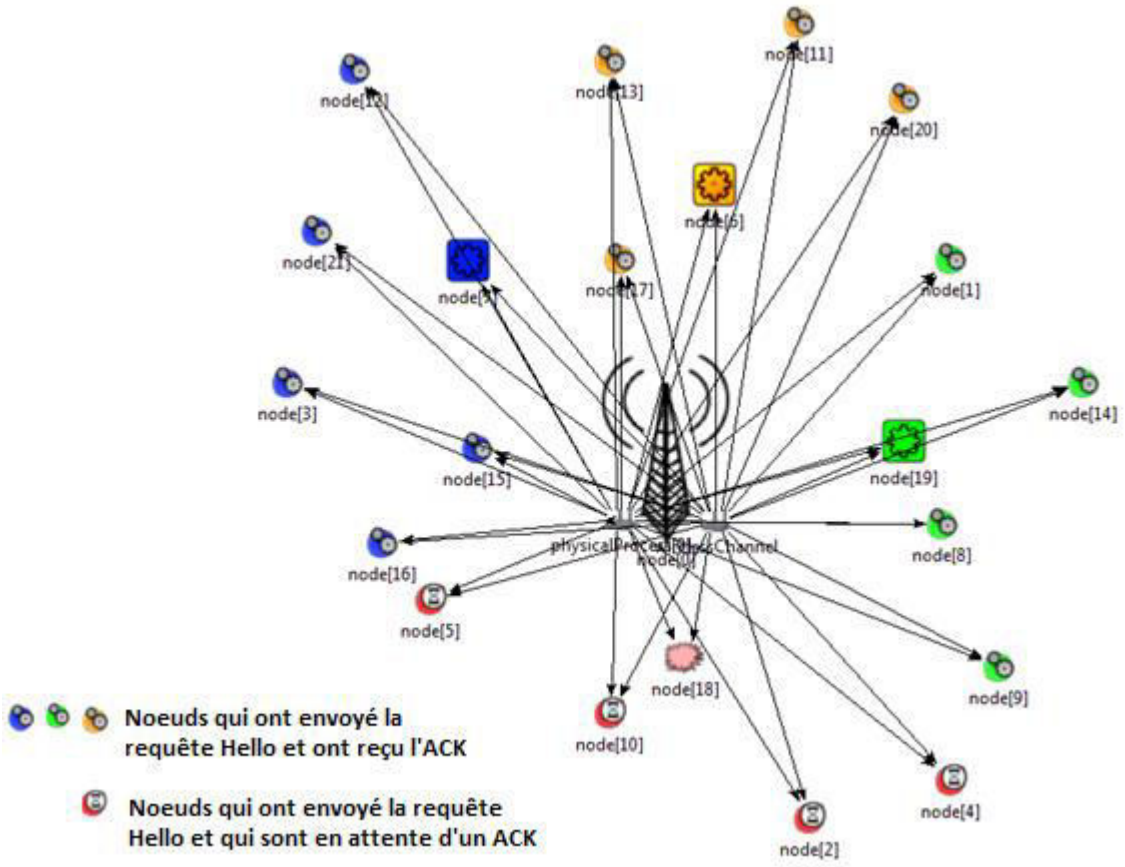

Figure V-6. Envoie de la requête « Hello » et attente des ACK.

Détection du *trou noir* et changement de Cluster-Head

Les nœuds qui appartiennent au Cluster-Heads 18, n'ont pas reçu un accusé de réception sur leurs requêtes « Hello ». Ils vont donc constater que le Cluster-Head auquel ils viennent de s'attacher ne fonctionne pas correctement. Soit il est défectueux (panne causée par un manque d'énergie), soit c'est un attaquant de type trou noir. Dans les deux cas, les nœuds capteurs ajoutent ce Cluster-Head suspect à une liste noire (*Blacklist)* qui va contenir l'ensemble des Cluster-Heads à éviter durant les prochains rounds. Ensuite, les nœuds, vont refaire une autre demande d'appartenance à un autre Cluster-Head. Comme illustré par la Figure V-7, les capteurs 5, 10, 2 et 4 ont détecté le trou noir 18 et ils se sont abonnés à d'autres Cluster-Heads.

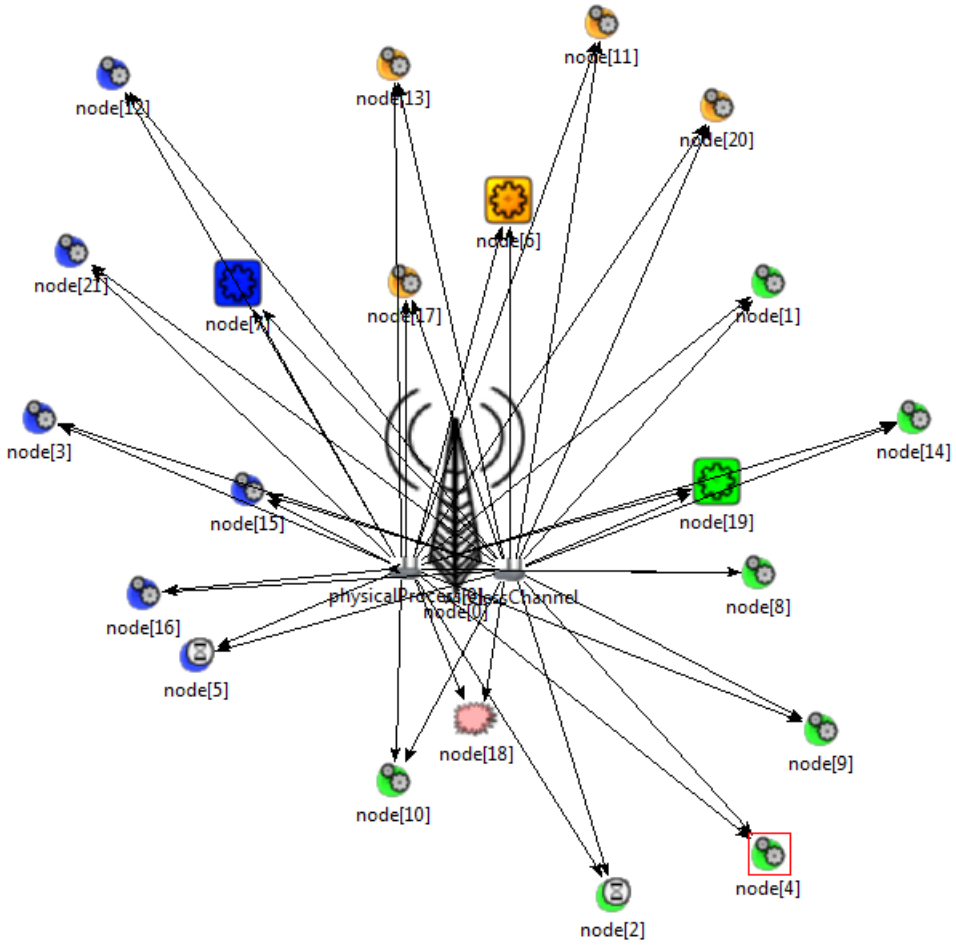

Figure V-7. Détection du trou noir et changement de Cluster-Head.

Le *trou gris* dans la phase *validation Cluster-Head*

Nous allons illustrer le comportement des nœuds qui appartiennent au *trou gris* 18 dans la phase validation cluster-Heads. Comme nous l'avons expliqué dans les chapitres précédents, les attaques type *trou gris* ne suppriment qu'une partie des paquets reçus de leur cluster. Comme illustré par la Figure V-8, l'attaquant a laissé passer la requête « Hello » en provenance des nœuds 4 et 5, et il a laissé passer aussi l'ACK. Par la suite, ils se sont mis en mode transmission de données, alors que les nœuds 2 et 10 n'ont pas reçu d'accusé de réception sur leur requête. Par conséquent, ils ont changé de Cluster-Head.

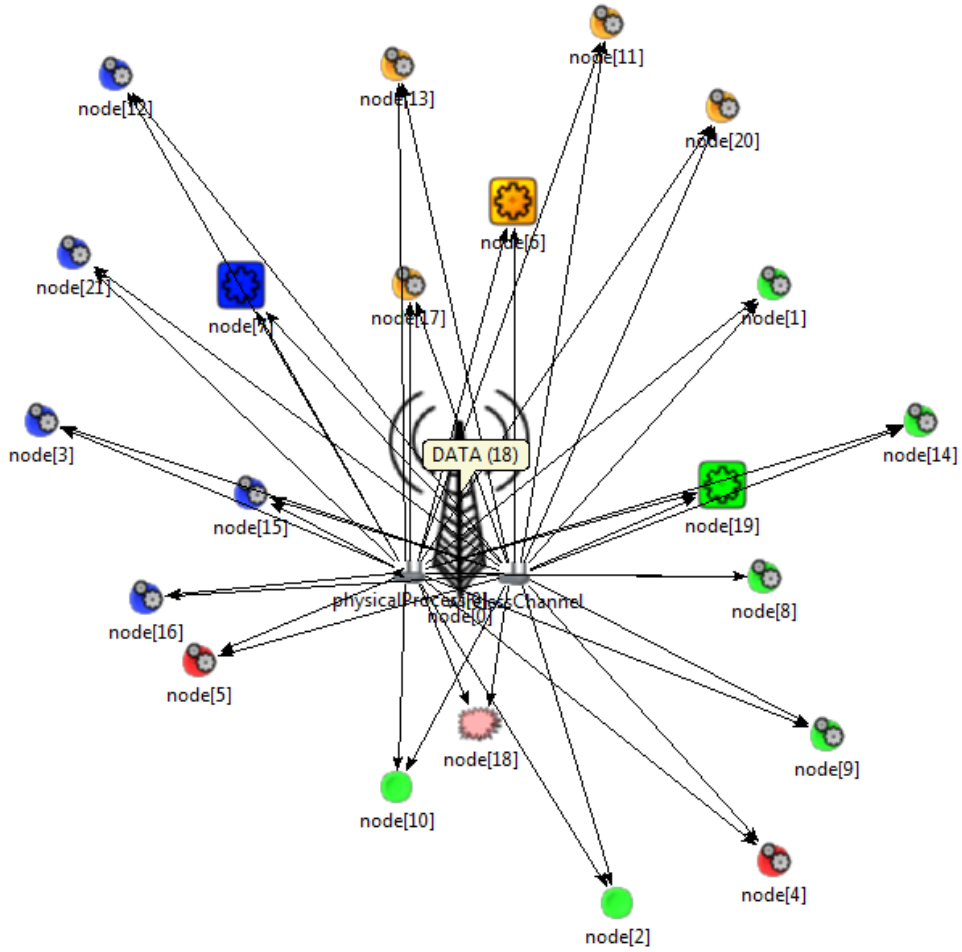

Figure V-8. Le trou gris dans la phase validation cluster-Heads.

3.3.2 Phase transmission de données avec demande d'ACK

Comme la phase *validation Cluster-Head* n'est pas suffisante pour la détection des *trous gris*. Nous avons introduit un mécanisme d'envoie des données avec demande d'accusés de réceptions d'une manière irrégulière. Comme illustré par la Figure V-9, ce mécanisme a permis à tous les nœuds trompés, de détecter le trou gris 18. Par conséquent ils ont tous changé de Cluster-Head.

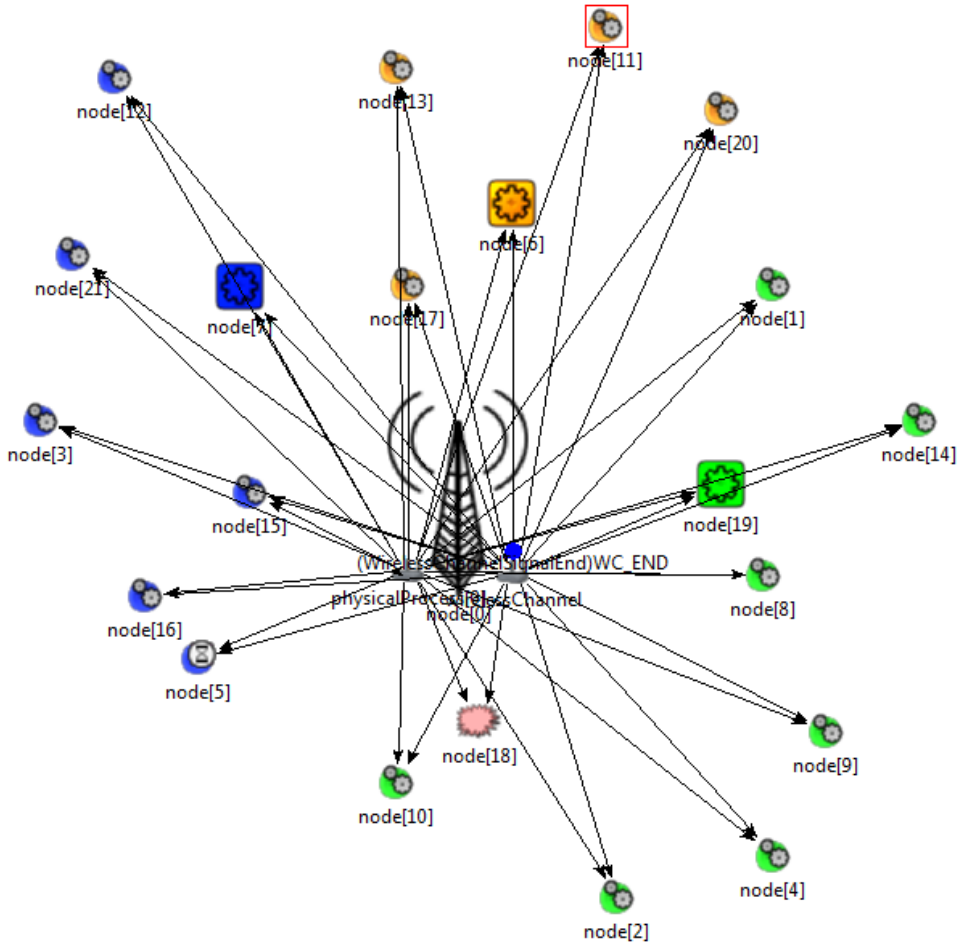

Figure V-9. Détection du trou gris et changement de Cluster-Head.

4. Tests et discussion des résultats

Dans cette section, nous évaluons les performances du nouveau protocole proposé DHD-LEACH. Pour cela, nous effectuons des tests sur les deux protocoles LEACH et DHD-LEACH que nous avons implémenté en présence des trous noirs et gris.

4.1 Critères de performances

Pour évaluer les performances du nouveau protocole DHD-LEACH, nous utiliserons les deux métriques suivantes :

Taux du bon acheminement des paquets de données

Un mécanisme de détection d'intrus vise à minimiser la perte des paquets de données en présence des attaques de type *trou noir* ou *trou gris*. Ce paramètre représente le taux de

messages bien acheminés vers la station de base. Il correspond au rapport entre le nombre de messages de données bien acheminés et le nombre de messages envoyés.

Energie consommée

L'énergie consommée est un facteur primordial dans les RCSFs. Pour notre comparaison, nous intéresserons à l'énergie consommée par chaque nœud dans différents cas de configuration.

4.2 Paramétrage de la simulation

Nous considérons un réseau de capteur sans fil déployé aléatoirement sur une surface 100m x 100m, avec une seule station de base. Les capteurs utilisés empruntent les caractéristiques des capteurs équipés de transceiver radio CC2420 et ils sont considérés homogènes (même quantité d'énergie initiale, même capacité de calcul et de stockage). Le Tableau V-2 résume quelques paramètres utilisés :

Paramètre	Valeur
Nombre de nœuds du réseau	21
Durée de simulation	40s
Protocoles de routages	LEACH, DHD-LEACH
Taux de paquet de données généré par un nœud	1/s
Surface de simulation (m x m)	100 x 100
Placement des noeuds	Aléatoire
Nombre de stations de base	1
Durée d'un round	40s
Energie initiale des nœuds capteurs	18720 Joules qui correspondent à deux batteries type AA.

Tableau V-2. Paramètres de la simulation.

4.3 Discussion des résultats

Dans ce qui suit, nous allons présenter et analyser les résultats de simulation obtenus, suivant les métriques de performances citées précédemment.

Cas 1 : le protocole LEACH en présence du *trou noir*

La Figure V-10 montre la courbe du taux de bon acheminement des paquets de données en fonction du temps au niveau du protocole LEACH en présence d'une attaque type *trou noir*. Cette simulation correspond au scénario représenté par la Figure V-5.

Durant les premières secondes, y'a pas eu d'envoi de paquets de données, car cette période correspond à la phase initialisation (durant laquelle, le réseau s'organise en clusters). On remarque, qu'à la fin de la simulation, le taux du bon acheminement des paquets pour les nœuds non attaqués est égal à 100%. Par contre, les nœuds 2, 4, 5 et 10 qui constituent les membres du cluster-head attaquant, ont un taux de bon acheminement qui est nul. Cela est dû au fait que le trou noir a supprimé tous les paquets de données en provenances de ces nœuds.

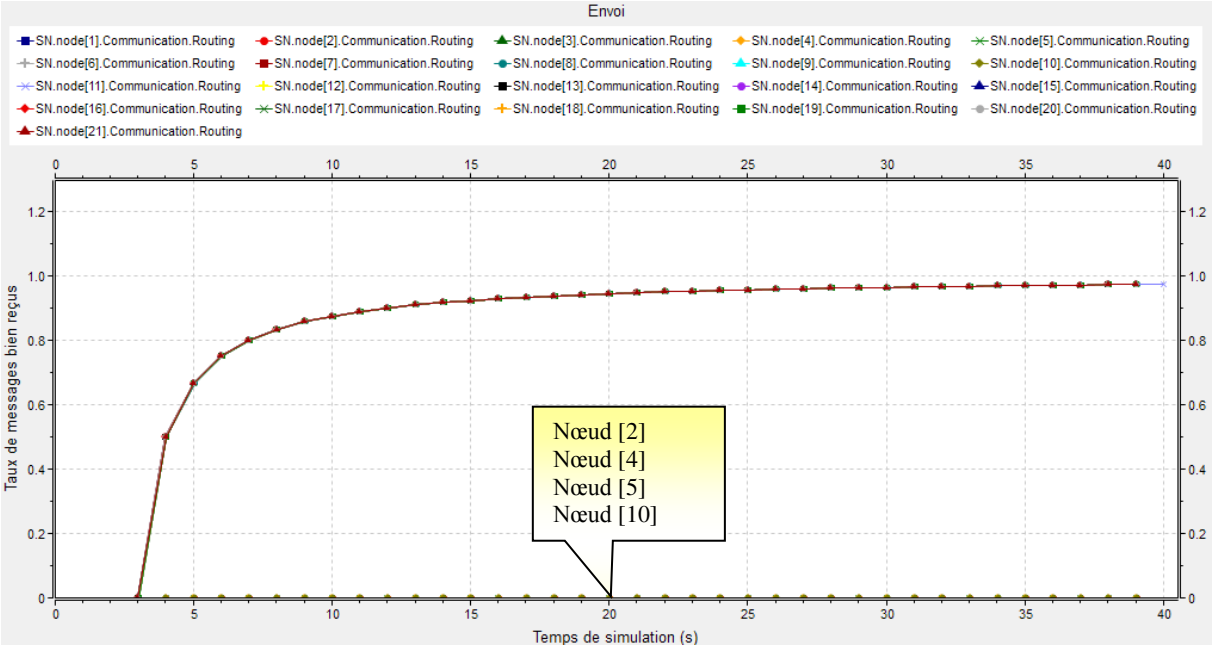

Figure V-10. Taux de bonne réception des paquets de données dans LEACH en présence du *trou noir*.

Cas 2 : DHD-LEACH en présence du *trou noir*

La Figure V-11 montre la courbe du taux de bon acheminement des paquets de données en fonction du temps, au niveau du protocole DHD-LEACH en présence d'une attaque type *trou noir*. Cette simulation correspond au scénario représenté par la Figure V-7.

Les nœuds qui n'appartienent pas au cluster head attaquant, se comportent exactement de la même manière que dans le cas LEACH de base en présence du *trou noir*. Par contre, on remarque que les nœuds 2, 4, 5 et 10 qui constituent les membres du cluster-head attaquant

n'envoient leurs données qu'à partir de la 7 ème seconde. Ce retard d'envoi de données par rapport aux nœuds non attaqués, correspond à la phase *validation du cluster-head* et au changement de cluster-head. Une fois que ces nœuds ont changé de cluster-head, on remarque qu'à la fin de la simulation, leur taux de bon acheminement des paquets n'atteint pas les 100%. Cela est dû à la perte des paquets « Hello ».

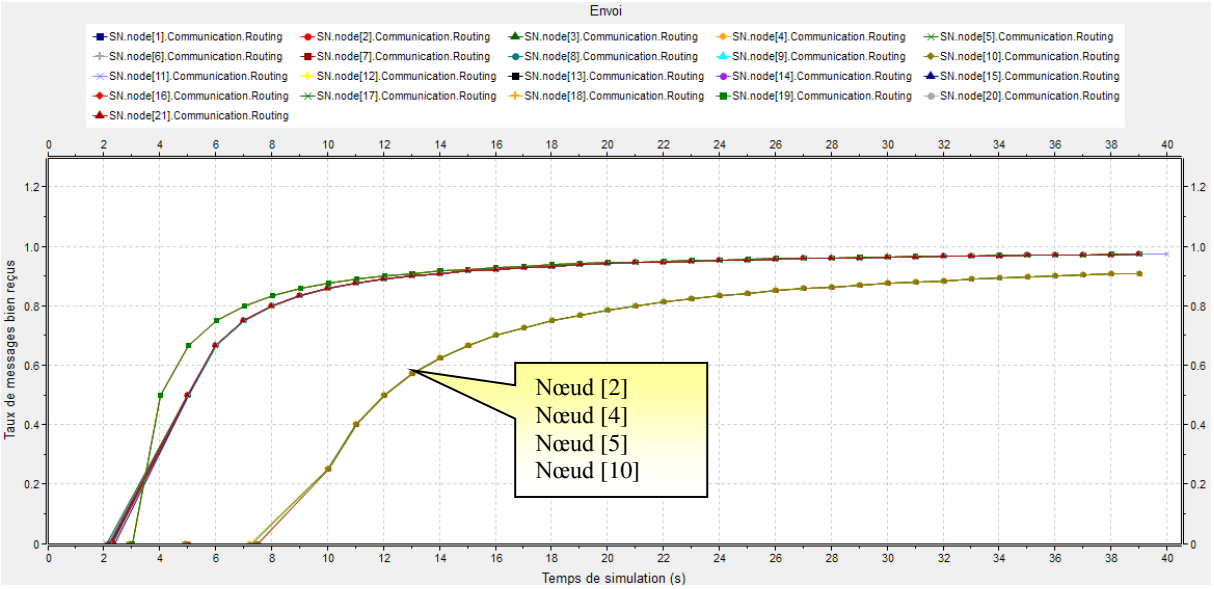

Figure V-11. Taux de bonne réception des paquets de données dans DHD-LEACH en présence du *trou noir*.

Cas 3 : LEACH en présence du *trou gris*

La Figure V-12 montre la courbe du taux de bon acheminement des paquets de données en fonction du temps au niveau du protocole LEACH en présence d'une attaque type *trou gris*. Cette simulation correspond au scénario représenté par la Figure V-5.

Durant les premières secondes, y'a pas eu d'envoi de paquets de données, car cette période correspond à la phase initialisation. On remarque qu'à la fin de la simulation le taux du bon acheminement des paquets pour les nœuds non attaqués est égal à 100%.

Par contre, en ce qui concerne les nœuds 2, 4, 5 et 10 qui constituent les membres du cluster-head attaquant, on peut constater qu'ils ont un taux de bon acheminement de paquets qui varie en fonction du temps. Cette variation correspond au comportement aléatoire du nœud attaquant lors de sa décision de suppression ou pas du paquet à transmettre vers la station de base.

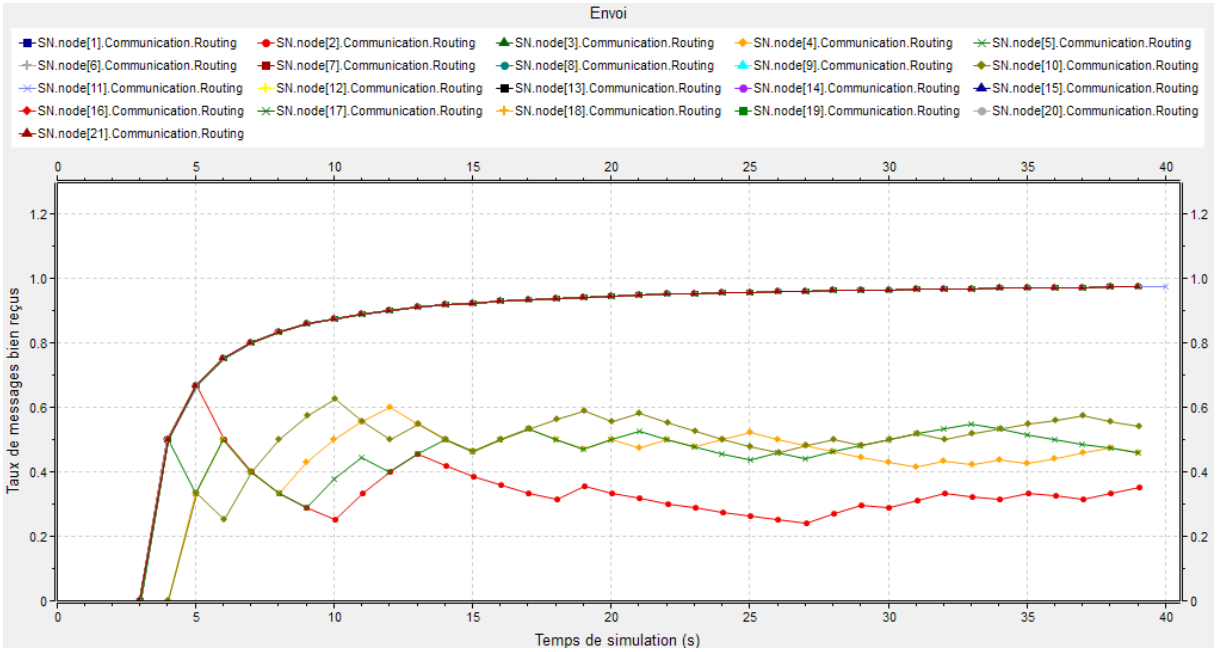

Figure V-12. Taux de bonne réception des paquets de données dans LEACH en présence du *trou gris*.

Cas 4 : DHD-LEACH en présence du *trou gris*

La Figure V-13 montre la courbe du taux de bon acheminement des paquets de données en fonction du temps au niveau du protocole DHD-LEACH en présence d'une attaque type *trou gris*. Cette simulation correspond au scénario représenté par la Figure V-8 et Figure V-9.

On constate que les nœuds 2 et 10 n'envoient leurs paquets de données qu'après le changement de cluster-head. Ce qui engendre un retard sur le début d'envoi des données.

Les nœuds 4 et 5 ne détectent le trou gris qu'après la demande d'accusé de réception des paquets déjà envoyés. On peut remarquer sur le graphe, que les nœuds 4 et 5 ont un taux de bon acheminement de données variable, ce qui correspond au comportement d'un trou gris. À partir de la 12 ème secondes, ces deux derniers ont changé de cluster-head et on le constate sur la courbe par le taux du bon acheminement qui est devenu stable et régulier.

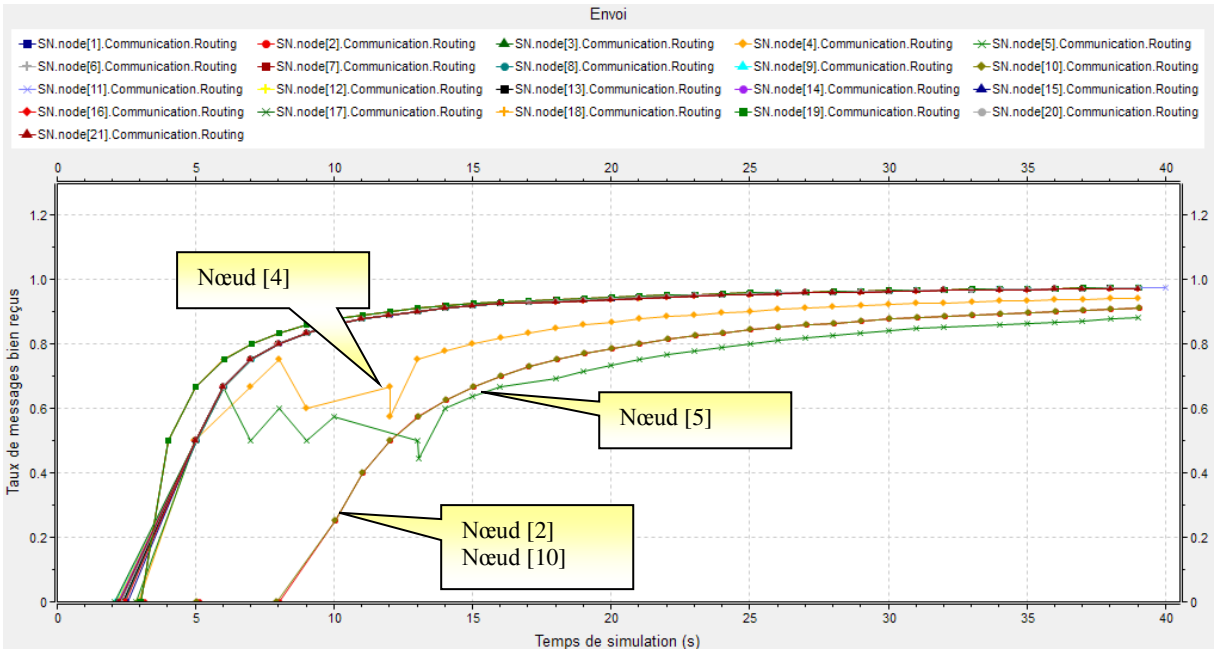

Figure V-13. Taux de bonne réception des paquets de données dans DHD-LEACH en présence du *trou gris*.

Consommation énergétique des nœuds

Pour étudier l'impact du protocole DHD-LEACH sur la consommation énergétique des nœuds, nous avons effectué des mesures de la consommation énergétique dans trois cas de configuration. Ensuite, nous avons comparé les résultats obtenus en fonction du protocole utilisé.

Le premier cas correspond aux nœuds qui utilisent le protocole LEACH de base. Les résultats obtenus dans ce cas de configuration vont nous servir de référence pour étudier le niveau de consommation des autres cas.

Le deuxième cas correspond au cas où les nœuds utilisent le protocole DHD-LEACH en présence d'une attaque type *trou noir*. Et le troisième cas, correspond aux nœuds qui utilisent le protocole DHD-LEACH en présence d'une attaque type *trou gris*.

Pour mesurer la consommation énergétique pour chaque cas de figure, nous avons mesuré la consommation énergétique de chaque nœud, puis nous avons calculé la moyenne de consommation énergétique de l'ensemble des nœuds.

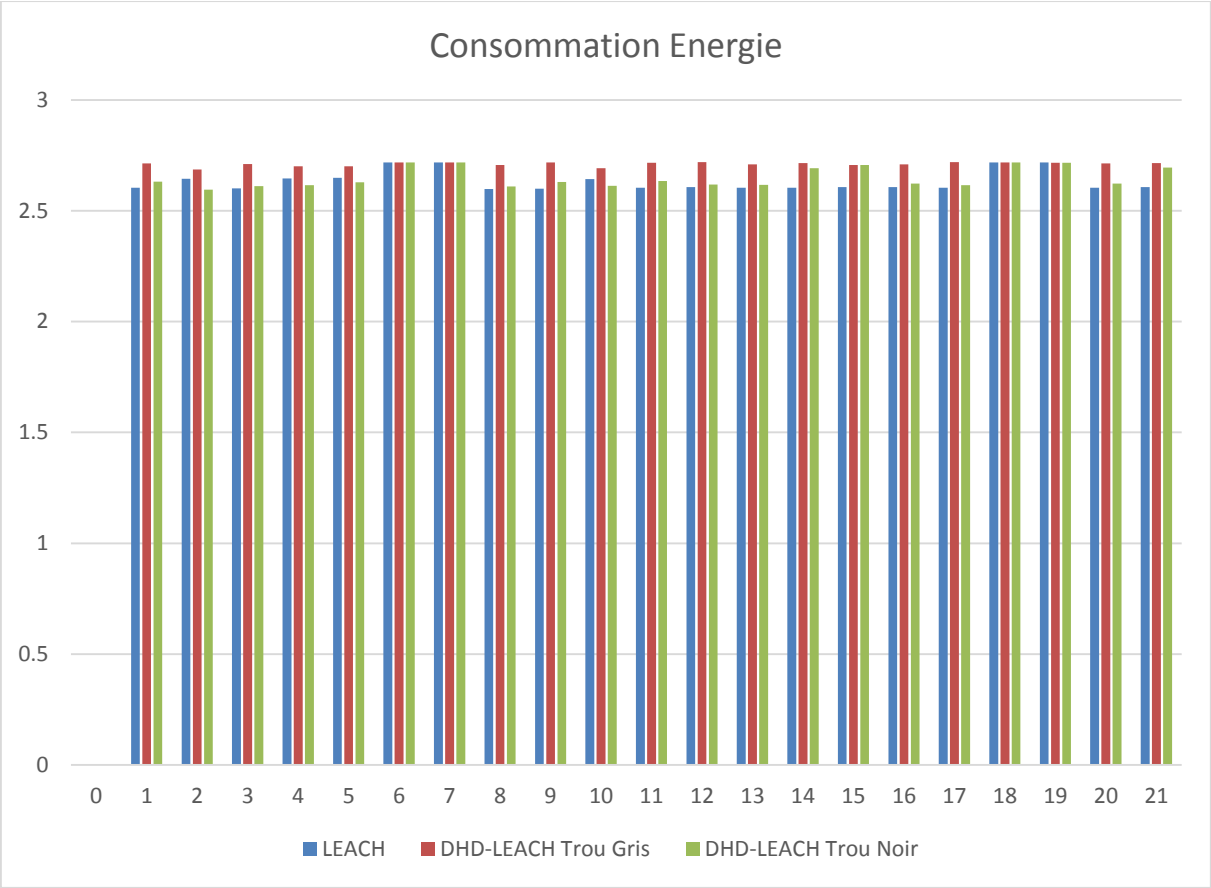

Figure V-14. Graphe de la consommation énergétique des nœuds.

La Figure V-14 montre un histogramme comparatif de la consommation énergétique des nœuds en fonction du protocole utilisé. La première chose qu'on peut constater en analysant l'histogramme, est que les cluster-heads 6, 7, 18 et 19 consomment un peu plus d'énergie par rapport aux autres nœuds du réseau, et cela quel que soit le protocole utilisé. Cela s'explique par la fonction d'agrégation de données assurée par les clusters-heads, qui est la plus coûteuse en termes de consommation énergétique.

On constate également que le protocole DHD-LEACH dans le cas d'attaque type *trou noir,* engendre une consommation moyenne correspondante à 0.58 % par rapport au protocole LEACH de base. Cela s'explique par l'ajout de la phase validation du cluster-head.

Et finalement, on constate que le protocole DHD-LEACH dans le cas d'attaque type *trou gris* engendre une consommation moyenne correspondante à 2.93 % par rapport au protocole LEACH de base. Cette consommation s'explique par l'ajout de la phase de validation du cluster-head et la gestion de la demande d'accusé de réception durant l'envoi des données.

Conclusion

Dans ce chapitre, nous avons commencé par présenter l'environnement de notre développement et simulation. Nous avons choisi de travailler sur *OMNeT++* grâce aux nombreux avantages qu'il propose. Nous avons utilisé *Castalia* comme plateforme de simulation des RCSFs grâce au comportement réaliste des nœuds capteurs. Durant tout ce chapitre, nous avons illustré le comportement du protocole LEACH de base que nous avons développé, puis le comportement des attaques type trou noir et trou gris que nous avons également développé. Nous avons également illustré le comportement du nouveau protocole que nous avons proposé, nommé DHD-LEACH face aux attaques type *trou gris* et *trou noir*.

Nous avons réalisé une étude comparative entre le protocole DHD-LEACH et le protocole LEACH de base. Cela nous a permis de montrer l'efficacité du protocole DHD-LEACH en présence des deux attaques. DHD-LEACH s'avère très efficace pour lutter contre les intrus de type *trou gris* et *trou noir*. De plus, la mise en place du protocole DHD-LEACH n'engendre pas une dissipation énergétique considérable par rapport aux avantages qu'il procure. En effet, DHD-LEACH engendre une consommation énergétique qui est en moyenne inférieur à 3% de la consommation globale.

Conclusion générale

Conclusion générale

Les RCSFs vont sans doute, dans les années à venir, constituer un développement technologique majeur, apportant des solutions aux différents problèmes dans plusieurs domaines d'applications liés à la sécurité, la santé, l'agronomie, la domotique, etc. Toutefois, tout au long de notre projet, nous avons constaté que la réalisation d'un protocole de routage pour les RCSFs pose de grands défis auxquels il faut répondre. La sécurité de ces réseaux est l'un des défis les plus importants à considérer. Beaucoup de chercheurs consacrent une grande importance à l'étude des menaces et des failles sécuritaires qui peuvent paralyser un RCSF. Ils développent par la suite des mécanismes qui permettent aux utilisateurs d'atteindre les besoins de sécurité requis.

Dans ce projet, nous nous sommes attelés à sécuriser le protocole de routage LEACH, conçu pour les topologies des RCSFs hiérarchiques. En effet, le protocole LEACH tel qu'il a été conçu ne comporte aucun mécanisme de sécurité. Il existe dans la littérature plusieurs variantes sécurisées du protocole LEACH (SLEACH, Sec-LEACH) **[OLI07] [FER05]**. Ces variantes répondent principalement aux problèmes d'authentification, d'intégrité et la confidentialité des données. Mais très peu de ces algorithmes répondent à la problématique de la détection des trous noirs et gris afin de les écarter définitivement du réseau. Nous avons vu que la manière la plus intuitive pour résoudre ce problème est de mettre en place un mécanisme d'envoi de données avec demande d'accusé de réception qui sera renvoyer par la station de base pour acquitter la bonne réception des paquets envoyés. Pour ce faire, nous avons proposé une nouvelle variante du protocole LEACH que nous avons appelé DHD-LEACH pour « *Dynamic Hole Detection in LEACH* ».

Contrairement au protocole LEACH de base, ce nouveau protocole comporte une phase supplémentaire que nous avons appelé « *phase validation* cluster-head» qui s'effectue après la phase initialisation. Cette phase a pour objectif de vérifier la validité du Cluster-Head auquel le nœud s'est abonné. Cette phase permet de détecter les intrus appliquant une attaque de type *trou noir*. Mais cette solution reste insuffisante pour la détection des *trous gris.* Pour remédier à ce manque, nous avons introduit au nouveau protocole, un mécanisme de demande d'accusé de réceptions de manière irrégulière (aléatoire) durant la phase transmission de données. Ce mécanisme permet à la station de base d'acquitter les paquets reçus et permettre

ainsi aux nœuds de savoir si la totalité des paquets transmis ont bel et bien atteint la station de base. Ce qui permet de déciller tout disfonctionnement dans l'acheminement des données, qu'il soit engendré par une panne ou une attaque de type *trou gris*.

Les expérimentations que nous avons réalisées montrent l'efficacité du nouveau protocole DHD-LEACH par rapport à LEACH de base, sans pour autant engendrer une consommation excessive de l'énergie.

Perspectives

Afin d'améliorer le nouveau protocole DHD-LEACH que nous avons proposé, nous envisageons trois pistes d'améliorations :

1. Intégrer un mécanisme de chiffrement plus adéquat pour les réseaux de capteurs sans fils : dans ce projet, nous nous sommes intéressés beaucoup plus à l'architecture globale du protocole DHD-LEACH. Nous avons utilisé un algorithme de chiffrement générique (AES) dans l'objectif d'illustrer le bon fonctionnement de notre solution. Ça serait judicieux d'intégrer à DHD-LEACH de nouveaux algorithmes de chiffrement plus adéquats aux ressources des réseaux de capteurs sans fils.

2. Intégrer un mécanisme d'échange de clés de chiffrement : dans ce projet nous avons proposé notre propre méthode de changement de clés. Ça serait intéressant d'intégrer au protocole DHD-LEACH d'autres méthodes d'échange de clés connu dans la littérature.

3. Intégrer un mécanisme de diffusion d'alerte dans le cas de détection d'un intrus dans le réseau : nous avons commencé à réfléchir sur une manière de diffuser une alerte sur le réseau par un nœud après la détection d'un intrus. A première vue un tel mécanisme peut être simple à intégrer, mais peut présenter en l'état, une grave faille de sécurité. En effet, on pourrait imaginer des attaquants intelligents qui se mettent à diffuser de fausses alertes dans le réseau afin de le perturber en écartant des Cluster-Head valides.

Références bibliographiques

[ANU11]: K. ANUPAMA, N. KAMDAR, D. VYAS, I. BOAKAR, S. SAHU et Ph. GEORGE, *Design and Implementation of a Cross Layered Protocol Stack for Sensor Networks in an Indoor Environment*, BITS PILANI K BIRLA GOA CAMPUS GOA INDIA ,2011.

[AGG03]: A. AGGARWAL et G. MARRO, *Vulnerability Analysis of Power-Aware Schemes in Sensor Networks: LEACH and PEGASIS*, Project paper, Department of Computer Science, University of California at Davis, 2003.

[ALK04]: N. ALKARAKI et A.E. KAMAL, *Routing Techniques in Wireless Sensor Networks: A Survey*, IEEE wireless communication, vol. 11, no. 6, pp 6-28, Dec 2004.

[ABD07]: M. ABDALLA, T. CLAVEIROLE, M. D AMORIM et Y. VINIOTIS, *Résistance contre les attaques par capture dans les réseaux de capteurs*, In *JDIR*, 2007.

[AND04]: R. ANDERSON, H. CHAN et A. PERRING, *Key Infection: Smart Trust for Smart Dust*, IEEE International Conference, 2004.

[ABE12]: M. ABED, M. BACHA, *Description des comportements d'un réseau de capteurs sans fils à l'aide de SMA*, mémoire de fin d'études, Ecole nationale Supérieure d'Informatique ESI, Algérie, 2012.

[BUN07]: A. BUNEL, E. DURIS et D. REVUZ, *Les réseaux de capteurs sans fil*, [En ligne], Page consultée le 09/02/2015, http://igm.univ-mlv.fr/~dr/XPOSE2006/Bunel/index.html

[BER04]: A. BERNARD, C. DEPOND, I. GARCIA et al, *Réseaux de capteurs sans fil*, [En ligne] (Page consultée le 12/02/2015) http://www.techno-science.net/?onglet=glossaire&definition=11711

[BAL02]: H. BALAKRISHNAN, A. CHANDRAKASAN et W.R. HEINZELMAN, *Energy-efficient communication protocol for Wireless micro sensor networks*, In IEEE Proc, Janvier 2002.

[BET08]: H. BETTAHAR et Y. CHALLAL, *Introduction à la sécurité informatique*, Supports de cours, Systèmes Intelligents pour le Transport, Université de Technologie de Compiègne, France, 15 Octobre 2008.

[BOU08]: DJ. BOUBICHE, *Protocole de routage pour les réseaux de capteurs sans fil,* Mémoire de magistère en Informatique, option : informatique industrielle, Université de l'Hadj Lakhdar-Batna, Faculté des sciences de l'ingénieur, 2008

[CHA08]: Y. CHALLAL, *Réseaux de capteurs sans fils,* [En ligne] (Page consultée le 15/02/2015)
http://moodle.utc.fr/pluginfile.php/16775/mod_resource/content/0/supportSIT60.pdf

[CAS06]: C. CASTELLUCIA et A. FRANCILLON. *Protéger les réseaux de capteurs sans fils*, INRIA, Grenoble-Rhône-Alpes, 2006.

[CHE08]: J. CHEN, H. ZHANG, et J. HU, *An efficiency security model of routing protocol in wireless sensor networks. In Proc. of the 2008 Second Asia International Conference on Modeling and Simulation,* pages 59–64, Washington, DC, USA, 2008, IEEE Computer Society.

[CAS11]: Castalia, *Wireless sensor network simulator*, [En ligne] (Page consultée le 07/05/2015) https://castalia.forge.nicta.com.au/index.php/en/

[DEV13]: R. DEVIKA, B. SANTHI et T. SIVASUBRAMANIAM, *Survey on Routing Protocol in Wireless Sensor Network*, International Journal of Engineering and Technology (IJET), Vol 5 No 1 Feb-Mar 2013.

[DEN05]: J. DENG, R. HAN, et Sh. MISHRA, *Countermeasures against traffic analysis attacks in wireless sensor networks: Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, Washington, DC, USA, 2005, IEEE Computer Society.

[FER05]: A. C. FERREIRA, M. A. VILACA, L. B. OLIVEIRA, E. HABIB, H. C.WONG, et A. A. LOUREIRO, On the security of cluster-based communication protocols for wireless sensor networks. In *Proc. 4th IEEE International Conference on Networking (ICNŠ05), volume 3420 of Lecture Notes in Computer Science*, pages 449–458, 2005.

[GIL08]: D. GIL, *Etude en vue de la réalisation de logiciels bas niveau dédiés aux réseaux de capteurs sans fil : microsystème de fichiers*, Thèse, Ecole Doctorale sciences pour l'ingénieur de Clermont-Ferrand, 2008.

[GAR06]: A. GARHAN, W. YONG et RA. BYRAV, *survey of security issues in wireless sensor networks*, IEEE Communications Surveys & Tutorials, 2006.

[HEI99]: W.R. HEINZELMAN, J. KULIK et H. BALAKRISHAN, *Adaptive protocols for information dissemination in wireless sensor networks*, In ACM MobiCom, Aout 1999.

[HEI02]: W. B. HEINZELMAN, A. P. CHANDRAKASAN et H. BALAKRISHNAN, *An Application-Specific Protocol Architecture for Wireless Microsensor Networks*, IEEE Transactions on Wireless Communications, 1(4):660{670, 2002.

[INT00]: G. INTANAGONWIWAT, R. GOVINDAN et D. ESTRIN, *Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks*, Proc. Sixth Ann. ACM/IEEE Int'lConf. Mobile Computing and Networking (MOBICOM), pp. 56-67, Aug. 2000.

[IBR06]: J. IBRIQ et I. MAHGOUB, A secure hierarchical routing protocol for wireless sensor networks. *In: Proc. 10th IEEE International Conference on Communication Systems*, pages 1–U″ 6, Singapore, october 2006.

[KUL02]: J. KULIK, W.R HEINEZLMAN et H BALAKRISHAN. *Negotiation based protocols for disseminating information in wireless sensor networks*. Wirel.2002, 8, 169–185.

[KAR03]: Ch. KARLOF et D. WAGNER, *Secure routing in wireless sensor networks: attacks and countermeasures*, *Ad Hoc Networks*, 293–315, 2003.

[KRA97]: H. KRAWCZYK, M. BELLARE et R. CANETTI, *HMAC: Keyed-Hashing for Message Authentication*, RFC 2104, February, 1997.

[KIM04]: K. KIM, H. ZHU, F. BAO et R. H. DENG, *computing of trust in wireless networks*. In *Proceedings of 60th IEEE Vehicular Technology Conference, Los Angles, California*, September 2004.

[KHE04]: L. KHELLADI, N. BADACHE, *Les réseaux de capteurs: état de l'art*, Rapport de recherche, Algérie, Février 2004.

[LIN02]: S.LINDSEY et C.S. RAGHAVENDRA, *PEGASIS: Power Efficient Gathering in Sensor Information Systems*, IEEE Transactions on Parallel and Distributed Systems, vol. 13, n°19, pp. 924-935, September 2002.

[LAW01]: E. LAWREY*, The suitability of OFDM as a modulation technique for wireless telecommunications, with a CDMA comparison»,* Rapport d'ingéniorat, Université James Cook, Australie, 2001.

[MAL12]: P. MALLANAGOUDA et C. BIRADAR, *A Survey on Routing Protocols in Wireless Sensor Networks*, IEEE, ICON 2012.

[MAN01]: A. MANJESHWAR et D. P. ARGARWAL, *TEEN: a routing protocol for enhanced efficiency in wireless sensor networks*, In 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing.2001.

[MAN02]: A. MANJESHWAR et D. P. ARGARWAL, *An Efficient Sensor Network Routing Protocol (APTEEN) with Comprehensive Information Retrieval*, Proc. Second Int'l Workshop Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, Apr. 2002.

[MAR04]: D. MARTINS et H. GUYENNET, *Sécurité dans les réseaux de capteurs sans fil,* Université de Franche comté, 2004

[MAK07]: M. L. MAKNAVICIUS et Ch. BEKARA, *A new resilient key management protocol for wireless sensor networks*, In, *WISTP*, Springer, 2007

[MAR10]: D. MARTINS, *Sécurité dans les réseaux de capteurs sans fil Stéganographie et réseaux de confiance,* thèse pour obtenir le grade de docteur en Informatique de l'université de Franche-Comté, UFR des sciences et techniques, 2010.

[MAL97]: D MALKHI et M. REITER, *Unreliable Intrusion Detection in Distributed Computations*. In Proc. 10 th Computer Security Foundations Workshop (CSFW97), June 1997.

[MAL05]: C. MALLANDA, A. SURI, V. KUNCHAKARRA, S.S. IYENGAR, R. KANNAN and A. DURRESI ,*Simulating Wireless Sensor Networks with OMNeT++*, Department of Computer Science, Louisiana State University, Baton Rouge, 2005 .

[OLI06]: L. B. OLIVERIA, H. C. WONG, M. BERN, R. DAHAB, et A. A. F. LOUREIRO. SecLEACH - a random key distribution solution for securing clustered sensor networks. In *Proc. of the Fifth IEEE International Symposium on Network Computing and Applications*, pages 145–154, Washington, DC, USA, 2006, IEEE Computer Society.

[OLI07]: L. B. OLIVEIRA, A. FERREIRA, M. A. VILACA, H. C. WONG, M. BERN, R. DAHAB, et A. A. F. LOUREIRO, *SecLEACH-on the security of clustered sensor networks. Signal Processing*, 87(12):2882–2895, December 2007.

[OMN14]: User Manuel, OMNeT++ version 4.6, [En ligne] (Page consultée le 07/05/2015) https://omnetpp.org/documentation

[QUA09]: Z. QUAN et J. LI, Secure *routing protocol cluster-gene-based for wireless sensor networks*, In *Proc.* The 1st International Conference on Information Science and Engineering (ICISE2009), pages 4098–4102, December 2009.

[RAJ09]: V.B RAJASHRE, V.C .PATIL, Dr. S.R. SAWANT et Dr. R.R. MUDHOLKAR, *classification and comparison of routing protocols in wireless sensor network*, Ubiquitous Computing and Communication Journal, Special Issue on Ubiquitous Computing Security Systems, Volume: Ubiquitous Computing Security Systems, 2009.

[RAM13]: S.B. RAMA et O.S. KHANNA, Geographic *Routing Protocols for Wireless Sensor Networks: A Review*, International Journal of Engineering and Innovative Technology (IJEIT) Volume 2, Issue 12, June 2013.

[RAG03]: S. RAGHUWANSHI, *An Energy Efficient Cross Layer Design Scheme for Wireless Sensor Networks,* Master's Thesis, Virginia Polytechnic Institute and State University, 29 Août 2003.

[ROM07]:Y. ROMDHANE, *Evaluation des performances des protocoles S-MAC et Directed Diffusion dans les réseaux de capteurs*, Projet de fin d'études, Ecole Supérieure des Communications de Tunis (Sup'Com), 2006 / 2007.

[SHI10]: S.K SHIO, M.P SINGH et D.K SINGH, *Routing Protocols in Wireless Sensor Networks – A Survey*, International Journal of Computer Science & Engineering Survey (IJCSES) Vol.1, No.2, November 2010.

[SHA11]: S. SHARMA et S. K JENA, *a Survey on Secure Hierarchical Routing Protocols in Wireless Sensor Networks*, Department of Computer Science and Engineering National Institute of Technology Rourkela, February 12-14, 2011, Rourkela, Odisha, India.

[SEC11]: *La sécurité informatique, L'AES (Advanced Encryption Standard,* Page consultée le 05/06/2015, https://fr.wikipedia.org/wiki/Advanced_Encryption_Standard.

[TIX04]: S. TIXSEUIL, T. HELMAN, *Un algorithme TDMA réparti pour les réseaux de capteurs***,** INRIA Projet Grand Large, Universités Iowa et Paris-Sud XI, 2004.

[TRI13]: M. TRIPATHI, M.S. GAUR et V. LAXMI, *Comparing the Impact of Black Hole and Gray Hole Attack on LEACH in WSN,* Malaviya National Institute of Technology, Jaipur, India, 2013

[WEN00]: R. WENDI, A. CHANDRAKASAN et B. HARI, *Energy-Efficient Communication Protocol for Wireless Microsensor Network*s, Massachusetts Institute of Technology, Proceedings of the Hawaii International Conference on System Sciences, January 4-7, 2000.

[WEN02]: R. WENDI, A. CHANDRAKASAN et B. HARI, *An Application-Specific Protocol Architecture for Wireless Microsensor Networks*, IEEE Transactions on Wireless Communications, vol. 1 no. 4, October 2002.

[WAL06]: J.P WALTERS, Z. LIANG, W. SHI, et V. CHAUDHARY. *Security in Distributed, Grid, and Pervasive Computing:* Wireless Sensor Network Security: A Survey. Auerbach Publications, CRC Press, 2006

[YAN01]: Y. YAN, R. GOVINDAN et D. ESTIN, *Geographical and Energy Aware Routing: a recursive data dissemination protocol for wireless sensor networks*, In Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking 2001, pp.70-84.

[YOU03]: M. YOUNIS, G. JOLLY, M.C. KURSU et P. KOKATE, *A Low-Energy Key Management Protocol for Wireless Sensor Networks*, IEEE Symposium on Computers and Communications (ISCC'03). 2003.

[ZIO02]: J. ZIOBRO, F. HU, J. TILLET et N. SHARMA, *Wireless Sensor Networks: Problems and Solutions*, Rochester Institute of Technology, Rochester, New York USA, 2002

[ZHA03]: Z. YAN, P. ZHANG, et T. VIRTANEM. *Trust evaluation based security solution in ad hoc networks*. In *NordSec 2003, Proceedings of the Seventh Nordic Workshop on Secure IT Systems*, 2003.

[ZHU04]: S. ZHU, S. SETIA et S. JAJODIA, *LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks,* Center for Secure Information Systems George Mason University, 2004