REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Hadj Lakhder - Batna

Faculté des Sciences de l'Ingénieur

Département d'Informatique

Mémoire de Magistère

Thème :

Protocole de sécurité Pour les Réseaux de capteurs Sans Fil

Préparé par : **Samir ATHMANI**

Proposé et dirigé par : **Dr. Azeddine BILAMI**

Pour l'obtention du **Magistère en Informatique**

Option : **I**ngénierie des **S**ystèmes d'**I**nformations

Soutenue publiquement le : 15/07/2010 devant le jury composé de :

Pr. Mohammed BENMOHAMMED	Professeur	**Président**	Université de Constantine
Dr. Azeddine BILAMI	M.C.	**Rapporteur**	Université de Batna
Dr. Abdelmadjid ZIDANI	M.C.	**Examinateur**	Université de Batna
Dr. Okba KEZZAR	M.C.	**Examinateur**	Université de Biskra

Sommaire

Liste des Figures

Liste des tableaux

Résumé

Le domaine d'application des réseaux de capteurs ne cesse d'accroître avec le besoin d'un mécanisme de sécurité efficace. Le fait que les RCSF traitent des données très souvent sensibles, opérant dans des environnements hostiles et inattendus, la notion de sécurité est considérée comme indispensable. Cependant, à cause de la limitation des ressources et la faible capacité de calcul d'un nœud capteur, le développement d'un mécanisme garantissant une sécurité pose de vrais défis de conception.

Dans ce mémoire nous avons essayé de proposer un nouveau mécanisme de sécurité dédié aux RCSF. Notre objectif principal est de sécuriser le processus de transfert des données vers la station de base. Le protocole proposé protège les données transférées contre les attaques des nœuds intrus en utilisant un mécanisme de sécurité basé sur l'utilisation de message de contrôle MAC (Message Authentification Code) pour l'authentification. Les performances de notre protocole sont évaluées à l'aide du simulateur NS2.

Abstract

The sensor networks application domain continues to increase with the need for an effective security mechanism. The fact that WSN often deal with sensitive data operating in hostile and unexpected environments, makes the concept of security considered essential. However, because of limited resources and low computing capacity of a sensor node, the development of a mechanism that ensures security is a real design challenges.

In this report we have tried to propose a new security mechanism dedicated to WSN. Our main objective is to secure the process of transferring data to the base station. The proposed protocol protects transferred data against intruder nodes attacks, using a security mechanism based on the use of control message MAC (Message Authentication Code) for authentication. Our protocol performances are evaluated using the simulator NS2.

Introduction générale

L'essor des technologies sans fil offre aujourd'hui de nouvelles perspectives dans le domaine des télécommunications. En comparaison avec l'environnement filaire, l'environnement sans fil permet aux utilisateurs une souplesse d'accès et une facilité de manipulation des informations à travers des unités de calcul mobiles (PC portable, PDA, capteur…).

Les réseaux sans fil peuvent être classés en deux catégories : les réseaux avec infrastructure fixe préexistante, et les réseaux sans infrastructure. Dans la première catégorie, une importante infrastructure logistique et matérielle est nécessaire pour le déploiement du réseau ; le modèle de la communication utilisé est généralement le modèle cellulaire (les réseaux GSM par exemple). La deuxième catégorie est celle des réseaux ad hoc.

Un réseau ad hoc peut être défini comme un ensemble d'entités mobiles interconnectées par une technologie sans fil formant un réseau temporaire sans l'aide de toute administration ou de tout support fixe.

Avec les avancées techniques en terme de performances et de miniaturisation, réalisées dans les microsystèmes électromécaniques (MEMS: microcontrôleur, transceiver RF…) et les communications sans fil, une nouvelle variante de réseaux ad hoc s'est créée afin d'offrir des solutions économiquement intéressantes pour la surveillance à distance et le traitement des données dans les environnements complexes et distribués : Les réseaux de capteurs sans fil.

Chapitre 1 : Introduction au réseau de capteur sans fil

1. Introduction :

Les progrès réalisés ces dernières décennies dans les domaines de la microélectronique, de la micromécanique, et des technologies de communication sans fil, ont permis de produire avec un coût raisonnable des composants de quelques millimètres cubes de volume. Ces derniers, appelés micro-capteurs, intègrent : une unité de captage chargée de capter des grandeurs physiques (chaleur, humidité, vibrations) et de les transformer en grandeurs numériques, une unité de traitement informatique et de stockage de données et un module de transmission sans fil.

De ce fait, les micro-capteurs sont de véritables systèmes embarqués. Le déploiement de plusieurs d'entre eux, en vue de collecter et transmettre des données environnementales vers un ou plusieurs points de collecte, d'une manière autonome, forme un réseau de capteurs sans fil. Selon le magazine Technology Review du MIT, il s'agit de l'une des dix nouvelles technologies qui bouleverseront le monde et notre manière de vivre et de travailler.

L'optimisation des communications dans les réseaux de capteurs sans fil industriels se base sur une connaissance des technologies de réseaux d'une part et des communications sans fil d'autre part. En ce qui suit, une présentation des réseaux informatique sera faite, en passant par les réseaux sans fil et les réseaux de capteurs en montrant le lien avec la communication.

2. Réseau informatique :

On nomme réseau un ensemble d'acteurs, d'agents économiques, de nœuds, ou lieux de communication grâce auxquels les messages circulent. L'information se concentre et se redistribue ainsi. On parle d'un réseau.

Un réseau informatique est un ensemble d'équipements reliés entre eux pour échanger des informations. Par analogie avec un filet (un réseau est un « petit rets », c'est-à-dire un petit filet), on appelle nœud *(node)* l'extrémité d'une connexion, qui peut être

une intersection de plusieurs connexions (un ordinateur, un routeur, un concentrateur, un commutateur).

Les infrastructures ou supports peuvent être sur des câbles dans lesquels circulent des signaux électriques, l'atmosphère (ou le vide spatial) où circulent des ondes radio, ou des fibres optiques qui propagent des ondes lumineuses. Elles permettent de relier « physiquement » des équipements assurant l'interconnexion des moyens physiques qui sont définis par des protocoles. Les équipements d'un réseau sont connectés directement ou non entre eux, conformément à quelques organisations types connues sous le nom de topologie de réseau. Les contraintes liées à l'application exigent souvent l'utilisation de telle ou telle topologie.

Les protocoles de communication permettent de définir de façon standardisée la manière dont les informations sont échangées entre les équipements du réseau : il s'agit de procédures qui contrôlent le flux d'information entre deux équipements. Des logiciels spécifiques qui gèrent ces protocoles sont installés sur les équipements d'interconnexion comme les commutateurs réseau, les routeurs, les commutateurs téléphoniques, les antennes GSM, etc.

Les réseaux informatiques sont classés suivant leur portée :

- le réseau personnel (PAN) relie des appareils électroniques personnels ;

- le réseau local (LAN) relie les ordinateurs ou postes téléphoniques situés dans la même pièce ou dans le même bâtiment ;

- le réseau métropolitain (MAN) est un réseau à l'échelle d'une ville ;

- le réseau étendu (WAN) est un réseau à grande échelle qui relie plusieurs sites ou des ordinateurs du monde entier.

3. Réseaux sans fil :

3.1 Définition :

Un réseau sans fil est un réseau informatique ou numérisé qui connecte différents postes ou systèmes entre eux par ondes radio.

Il peut-être associé à un réseau de télécommunications pour réaliser des interconnexions entre nœuds. La norme la plus utilisée actuellement pour les réseaux sans fil est la norme IEEE802.11, mieux connue sous le nom de Wi-Fi.

Le rayonnement géographique des ondes est relativement limité étant donné la faible puissance d'émission des solutions matérielles actuelles. Pour cette raison, les réseaux sans fil se sont avant tout développés comme réseaux internes, propres à un bâtiment, soit comme réseau d'entreprise, soit comme réseau domestique.

Néanmoins, des projets de réalisation de réseaux à grande échelle ont vu le jour, notamment le WiMAX.

3.2 Les catégories des réseaux sans fil :

Il existe plusieurs catégories de réseaux sans fil qui diffèrent par le périmètre géographique qu'ils couvrent ainsi que par les types d'applications supportées. Le schéma suivant illustre les catégories des réseaux sans fil.

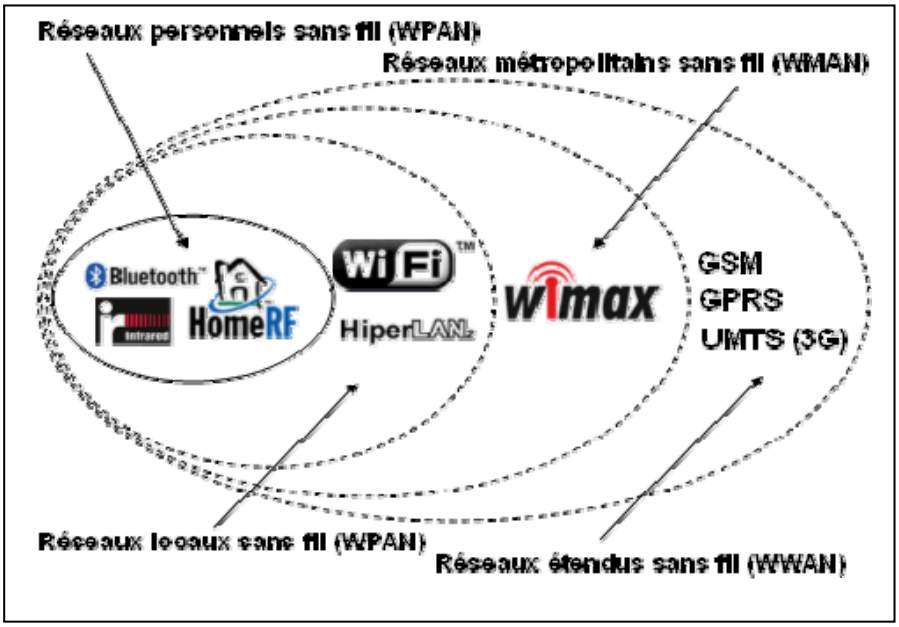

Figure 1 : Les catégories des réseaux sans fil [1]

3.2.1 Le réseau personnel sans fil (WPAN) :

Il concerne les réseaux sans fil d'une faible portée : de l'ordre de quelques dizaines de mètres. Ce type de réseau sert généralement à relier des périphériques (imprimante, téléphone

portable, appareils domestiques, PDA…). Il existe plusieurs technologies utilisées pour les WPAN :

La technologie Bluetooth : Elle est connue aussi sous le nom de la norme IEEE 802.15.1, elle a été lancée par Ericsson en 1994, proposant un débit théorique de 1 Mbps lui permettant une transmissions de la voix, des données et des images [2], d'une portée maximale d'une trentaine de mètres. Bluetooth est une technologie peu onéreuse, grâce à sa forte intégration sur une puce unique de 9 mm sur 9 mm [03] ; Elle présente également l'avantage de fonctionner sur des appareils à faible puissance d'où une faible consommation d'énergie [04].

La technologie ZigBee : Elle est connue aussi sous le nom de la norme IEEE 802.15.4 et permet d'obtenir des liaisons sans fil à bas prix et avec une très faible consommation d'énergie, ce qui la rend particulièrement adaptée pour être directement intégrée dans de petits appareils électroniques (capteurs, appareils électroménagers...). Les réseaux ZigBee permettent d'offrir des débits jusqu'à 250 Kbits/s dans la bande classique des 2,4GHz. Les RCSF constituent une des applications que cette norme peut couvrir [05].

Les liaisons infrarouges : Elles permettent de créer des liaisons sans fil de quelques mètres avec des débits pouvant monter à quelques mégabits par seconde. Cette technologie est largement utilisée dans la domotique (télécommandes), elle souffre toutefois des perturbations dues aux interférences lumineuses.

3.2.2 Le réseau local sans fil (WLAN) :

C'est un réseau permettant de couvrir une portée d'environ une centaine de mètres. Il permet de relier entre-eux les terminaux présents dans la zone de couverture. Il existe deux technologies concurrentes :

Les réseaux Wi-Fi (Wireless-Fidelity) : Ils proviennent de la norme IEEE 802.11, qui définit une architecture cellulaire. On y trouve principalement deux types de réseaux sans fil : Ceux qui travaillent à la vitesse de 11 Mbits/s à 2.4 GHz (IEEE 802.11b) et ceux qui montent à 54 Mbits/s à 5 GHz (IEEE 802.11 a/g) [03].

Les réseaux HiperLAN 2 (High Performance LAN 2.0) : Ils découlent de la norme européenne élaborée par l'ETSI (European Telecommunications Standards Institute). HiperLAN 2 permet d'obtenir un débit théorique de 54 Mbps sur une zone d'une centaine de

mètres dans la gamme de fréquence comprise entre 5 150 et 5 300 MHz [01]. Ce type de réseau n'a pas reçu autant de succès que la technologie Wi-fi.

3.2.3 Le réseau métropolitain sans fil (WMAN) :

Il est connu aussi sous le nom de Boucle Locale Radio (BLR). Il convient de rappeler que la BLR permet, en plaçant une antenne parabolique sur le toit d'un bâtiment, de transmettre par voie hertézienne de la voix et des données à haut débit pour l'accès à l'internet et la téléphonie. Il existe plusieurs types de réseaux WMAN dont le plus connu est :

les réseaux Wimax (Worldwide interoperability for Microwave Access) : ils émanent de la norme IEEE 802.16 et ont pour but de développer des liaisons hertéziennes concurrentes aux techniques xDSL terrestres et offrent un débit utile de 1 à 10 Mbit/s dans la bande 10-66 GHz pour une portée de 4 à 10 kilomètres, ce qui destine principalement cette technologie aux opérateurs de télécommunication [03].

3.2.4 Le réseau étendu sans fil (WWAN) :

Il est connu sous le nom de réseau cellulaire mobile et il est le plus répandu de tous puisque tous les téléphones mobiles sont connectés à un réseau étendu sans fil. Les principales technologies sont les suivantes : GSM (Global System for Mobile Communication), GPRS (General Packet Radio Service), UMTS (Universal Mobile Télécommunication System) [01].

4. Réseaux de capteurs sans-fil

4.1 Les capteurs « traditionnels »

Les réseaux de capteurs sans-fil visent à étendre les domaines d'application des capteurs. En effet, des capteurs sont déjà utilisés pour la création de systèmes d'acquisition de données car ils sont capables de détecter des phénomènes dans un environnement proche, de les quantifier et de transmettre les données ainsi obtenues à d'autres éléments du système en vue de leur traitement. Les éléments mesurés sont des grandeurs physiques telles que la pression, l'humidité, les vibrations, etc. [06] Pour réaliser de telles mesures, ces capteurs sont constitués de différents éléments hardwares.

Afin de les mettre en évidence, considérons par exemple, un système d'acquisition de données permettant de relever la température ambiante d'un milieu. Ce système est constitué d'un thermomètre électronique, lui-même composé d'un capteur et d'un afficheur digital. Une fois la température établie, la mesure est transformée en signal analogique grâce à l'élément de détection. Pour qu'elle puisse être affichée, il est nécessaire que ce dernier soit digitalisé. On utilise pour cela un convertisseur.

De plus, la présence du dispositif électronique précédemment cité impose une source d'énergie qui peut être soit le secteur, soit des batteries, etc. Enfin, si le destinataire des données (dans notre exemple, l'afficheur) est éloigné du capteur, il faut que ce dernier puisse transmettre la donnée numérisée, il peut donc y avoir également un module de communication comme un réseau câblé par exemple.

Si l'on ne considère que les éléments impliqués dans l'exemple précédent alors le capteur ainsi schématisé serait incomplet. En effet, deux autres composants peuvent être intégrés dans un tel instrument : il s'agit du conditionneur de signaux et d'un processeur de signal numérique.

Le conditionneur de signaux intervient afin d'améliorer la qualité des mesures. En effet, sous certaines conditions, notamment environnementales, celles-ci peuvent être faussées c'est-à-dire que le signal émis peut présenter du bruit, que son amplitude peut-être trop faible, qu'il peut être biaisé ou dépendre de paramètre secondaire. Le processeur de signal numérique permet quant à lui de traiter le signal c'est-à-dire d'en extraire des données, de le modifier ou de l'adapter en vue de la transmission ou du stockage.

De plus, l'élément de détection n'est pas toujours capable de mesurer la quantité intéressante mais une quantité reliée. Le conditionneur de signaux et le processeur de signal numérique permettent de résoudre ces problèmes respectivement par l'utilisation de filtre passe-bas, de mécanisme de Compensation, d'amplificateur opérationnel, etc.[07] et par des mécanismes de traitement du signal. La Figure 2 ci-dessous est une modélisation de ces différents éléments inspirés à la fois par [06] et la représentation des capteurs proposée à la Figure 3.

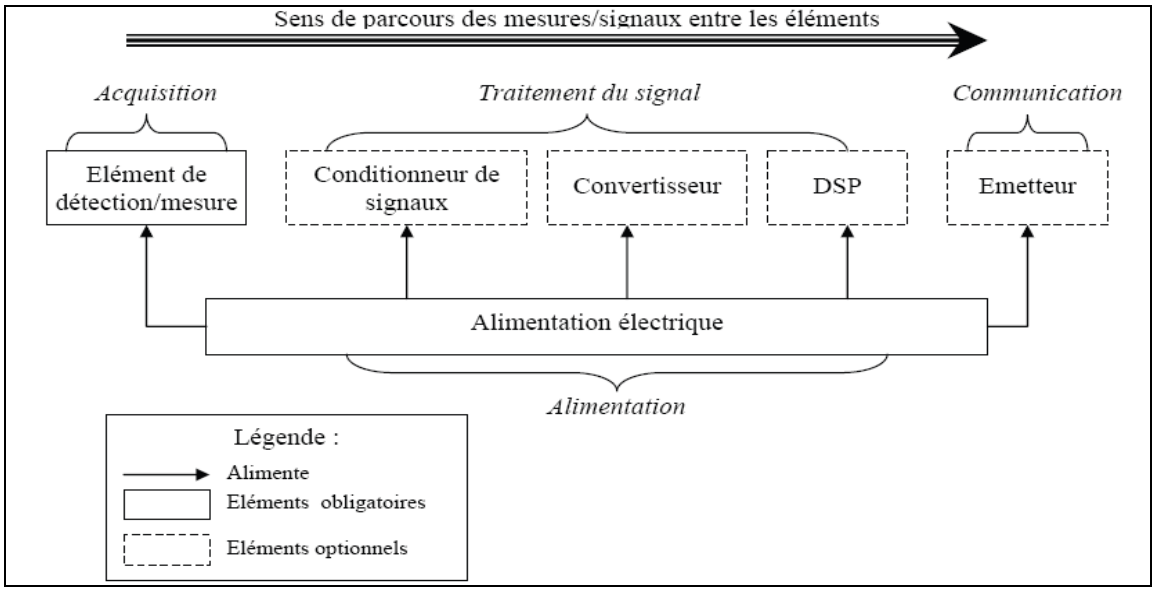

Figure 2 : Schématisation d'un capteur "traditionnel"

Après cette brève description des capteurs « traditionnels », nous en arrivons maintenant à la présentation des *motes* i.e. les capteurs utilisés dans les ***RCSFs***.

4.2 Les capteurs dans les réseaux de capteur sans fil

Les capteurs présents dans les réseaux de capteur sans fil se différencient de ceux précédemment décrits par plusieurs points. Le premier élément de distinction est révélé par le terme sans-fil. En effet, les *motes* ne sont plus câblés mais communiquent avec les autres éléments via un module de communication sans-fil.

De plus, alors que les capteurs précédents se contentaient d'envoyer des données à des éléments capables de les traiter, les *motes* peuvent également en recevoir. En effet, ils ont été conçus afin de fonctionner en réseau et afin de servir de relais pour que les données collectées et les autres informations essentielles au maintien du réseau puissent être propagées à travers celui-ci. Cette transmission est rendue possible grâce à un émetteur-récepteur. Celui-ci communique selon trois modes : grâce à des lasers, des infrarouges ou des radiofréquences. Un des facteurs limitant dans l'utilisation du *transceiver* est la portée. En effet, il est à noter que la distance maximale de transmission entre deux nœuds du réseau est limitée à quelques dizaines de mètres. Un autre facteur limitatif est la consommation d'énergie du transceiver qui est relativement importante dans le cas d'une émission de données.

La seconde distinction est due non seulement à une diminution des coûts de production mais également à une miniaturisation de la taille des composants. Bien que ces progrès s'appliquent également aux capteurs décrits précédemment. En effet, les capteurs ne sont désormais pas plus grands qu'une pièce de deux euros. Tous ces facteurs permettent d'envisager le déploiement d'un nombre beaucoup plus conséquent de senseurs à savoir des centaines voire des milliers. Cela permet de créer un réseau beaucoup plus dense où la fiabilité de chaque nœud n'est plus primordiale. En effet, en imaginant un déploiement aléatoire, plusieurs nœuds pourraient se retrouver au même endroit et couvrir la même région. Ce qui permettrait à un nœud défaillant d'être remplacé par un autre.

La troisième différenciation se produit au niveau de la source d'énergie. En effet, dans le cas des *motes*, il s'agit d'une alimentation embarquée contrairement aux précédents où la source d'énergie pouvait être fournie par une source extérieure. Cela représente une contrainte importante pour la durée de vie des capteurs car la capacité des batteries est limitée et que, de plus, il n'est pas toujours possible de les remplacer lorsqu'elles sont déchargées.

Cela se conçoit facilement si l'on envisage un très grand nombre de capteurs et/ou un environnement « hostile » de déploiement. Enfin, toute opération effectuée par les *motes* consomme de l'énergie entraînant ainsi une décharge plus ou moins importante des batteries. Parmi les opérations nécessitant énormément d'énergie, nous pouvons citer toutes celles impliquant l'émetteur-récepteur afin d'émettre des données [08].

Bien qu'il existe des batteries rechargeables par énergie solaire, par vibrations, etc., et des politiques de gestion de l'énergie, celles-ci ne suffisent pas, à elles seules, à permettre une utilisation prolongée des senseurs. L'économie d'énergie reste, par conséquent, un des points cruciaux dont il faut tenir compte lorsqu'on parle de *RCSF*.

La dernière différence se situe entre la phase d'acquisition des données et la phase de communication. En effet, le principal avantage d'un *mote* est de disposer en plus d'une unité de traitement permettant non seulement de rapprocher le traitement des données de leur lieu de détection mais également de les agréger.

Cela s'intègre dans le cadre des politiques d'économie d'énergie essentielles aux *RCSF s*. En effet, cela permet de diminuer la taille des messages à transmettre aux autres éléments du réseau et ainsi de diminuer le temps d'utilisation de l'émetteur-récepteur qui, pour rappel, est l'élément consommant le plus d'énergie dans les *motes*. Pour pouvoir réaliser

cette tâche, le capteur doit donc disposer également d'une unité de stockage nécessaire à l'implantation et à l'exécution d'un programme logiciel.

Celle-ci est généralement de moins de 10ko de RAM et de moins de 100ko de ROM [03] et est donc de très faible capacité par rapport à ce que nous pouvons trouver sur les ordinateurs personnels actuels. Afin d'augmenter la taille de la mémoire disponible, il est également possible d'utiliser des mémoires additionnelles de type Flash ou Eprom par exemple.

Un autre type de mémoire supplémentaire pourrait bientôt être envisagé : il s'agit de la *Magnetoresistive Random Access Memory* ou *MRAM*. En effet, celle-ci est en cours de développement mais ses atouts pour une utilisation dans les réseaux de capteurs pourraient être multiples car elle est très rapide à l'écriture et à la lecture, non volatile, elle consomme peu d'énergie, peut être effacée et réécrite un nombre de fois illimité (ce qui lui donne une espérance de vie aussi longue que celle de l'appareil qu'elle équipe), son coût de fabrication est faible et elle ne subit pas les influences des radiations [09].

La Figure 3 présente les différents changements qui ont été apportés aux capteurs utilisés traditionnellement et que nous venons de décrire. Celle-ci est à mettre en parallèle avec la Figure 2 précédente.

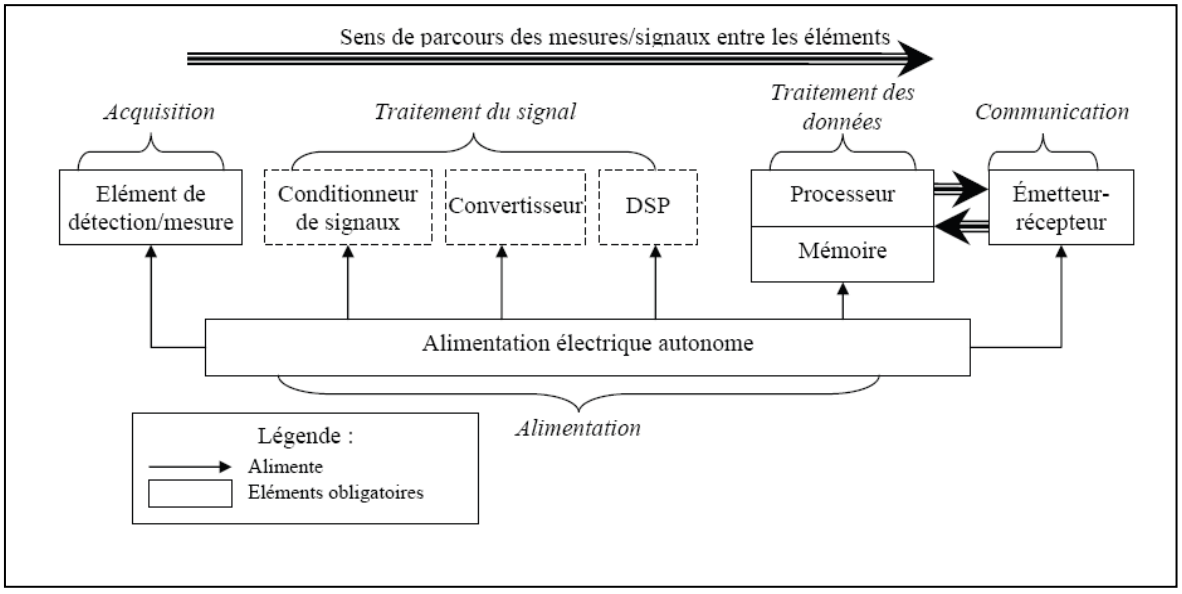

Figure 3 : Schéma d'un composant d'un réseau de capteurs, inspiré de [10]

En outre, ces appareils peuvent être également appelés « capteurs intelligents » tel que la norme présentée dans [11] les définis. En effet, celle-ci standardise leurs caractéristiques à savoir la définition des interfaces pour qu'ils puissent se connecter à des réseaux divers, les fonctionnalités additionnelles qu'ils doivent posséder en plus de celles nécessaires à une représentation correcte du phénomène quantifié, etc. Parmi ces fonctions, nous pouvons citer : la facilité d'installation, l'auto-identification, l'auto-diagnostic, la fiabilité, le temps d'éveil pour la coordination avec d'autres nœuds, quelques fonctions logicielles, le traitement du signal, des protocoles de contrôles standards et des interfaces réseaux [11].

De plus, les motes tout comme cette norme visent les mêmes objectifs à savoir rapprocher l'intelligence du point de mesure et minimiser leur coût d'intégration ou de maintenance dans des réseaux distribués [07].

Enfin cette norme définit également le terme de « capteur virtuel » car sa principale caractéristique est d'être indépendante du réseau dans lequel il se trouve. Pour réaliser cette condition, le capteur n'est en réalité constitué que par l'élément de détection/mesure, le conditionneur de signaux, le convertisseur analogique-numérique et le processeur du signal numérique que l'on peut schématiser comme à la Figure 4 qui est une modification du schéma présent dans [07].

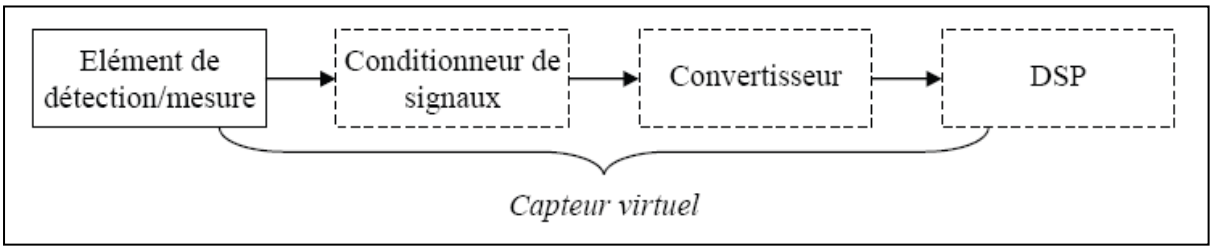

Figure 4 : Modèle de capteur virtuel, inspiré de [07]

4.3 La mise en réseau

Les types de capteurs que nous venons de décrire servent de pierre angulaire à l'édification de réseaux dits de capteurs sans-fil. D'autres éléments viennent compléter ce réseau. Il s'agit des *stations de base* et des *collecteurs*. Les collecteurs sont un autre type de nœud du réseau qui permet de rassembler des données provenant de plusieurs *motes* et qui possède également des fonctions de passerelle, reliant ainsi les capteurs avec un réseau externe – Internet par exemple.

Ces derniers possèdent beaucoup plus de capacités que les motes tant au niveau de la mémoire que de la vitesse de traitement ou des réserves en énergie. La station de base, quant à elle, est l'élément recueillant les données et/ou les analyses du réseau et servant également à son administration. Un réseau de capteurs sans-fil peut alors ressembler au schéma de la Figure 4 :

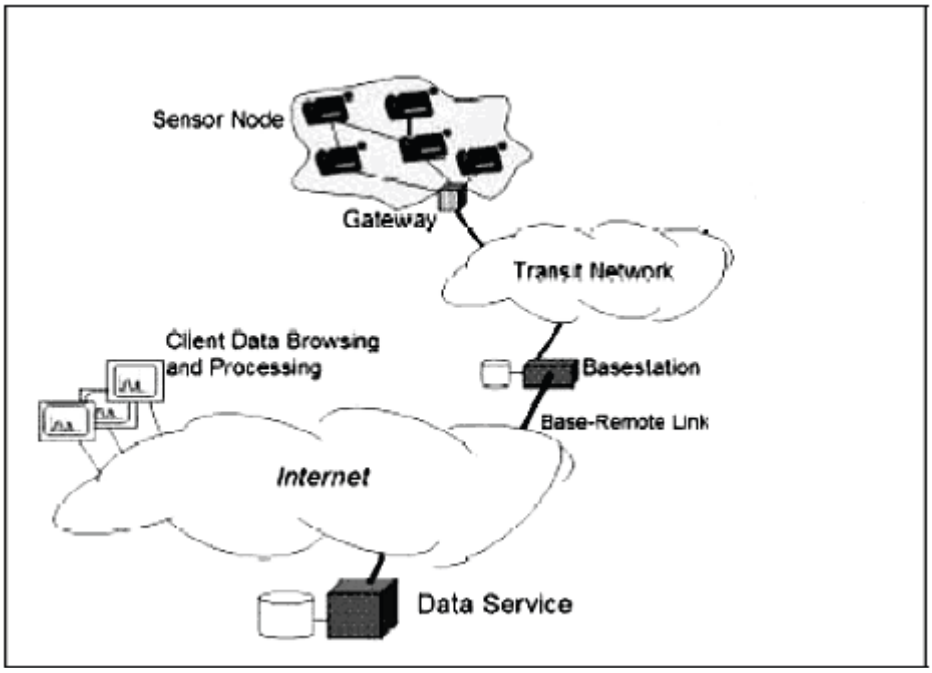

Figure 4 : Schématisation d'un réseau de capteurs sans-fil [16]

Ainsi les données détectées vont parcourir le réseau de capteurs en capteurs jusqu'à la station de base par exemple. En retour, celle-ci peut demander l'exécution d'une tâche particulière à un capteur donné.

Ces réseaux sont utilisés dans des domaines divers nécessitant l'acquisition d'informations concernant l'environnement. Le paragraphe suivant décrit les différentes utilisations possibles d'un *RCSF*.

5. Les principales caractéristiques des RCSF

- ***La consommation réduite d'énergie*** **:** Les nœuds capteurs utilisent des batteries de taille minuscule comme ressources en énergie, ce qui limite leur durée de vie. La spécificité des applications des RCSF (militaires, sismiques et autres) fait que la recharge ou le

remplacement de ces batteries est une tâche difficile ou presque impossible, ce qui nous mène à déduire que la durée de vie d'un nœud est essentiellement dépendante de la durée de vie de la batterie. Ainsi, la méthode de gestion de consommation d'énergie constitue une contrainte majeure dans ce type de réseau.

- ***L'auto-configuration des nœuds capteurs :*** Dans un RCSF, les nœuds sont déployés soit d'une manière aléatoire (missile, avion…), soit placés nœud par nœud par un humain ou un robot, et ceci à l'intérieur ou autour du phénomène observé (champ de guerre, surface volcanique, patient malade…) [12]. Ainsi, un nœud capteur doit avoir des capacités d'une part, pour s'auto-configurer dans le réseau, et d'autre part pour collaborer avec les autres nœuds dans le but de reconfigurer dynamiquement le réseau en cas de changement de topologie du réseau [13].

Dans un RCSF, chaque nœud X possède une unité émettrice/réceptrice qui lui permet de communiquer avec les nœuds qui lui sont proches; En échangeant des informations avec ces derniers, le nœud X pourra alors découvrir ses nœuds voisins et ainsi connaître la méthode de routage qu'il va adopter selon les besoins de l'application [14].

L'auto-configuration apparaît comme une caractéristique nécessaire dans le cas des RCSF étant donné que d'une part, leur déploiement s'effectue d'une manière aléatoire dans la majorité des applications, et d'autre part le nombre des nœuds capteurs est très grand.

- ***La scalabilité :*** Contrairement aux réseaux sans fil traditionnels (personnel, local ou étendu), un RCSF peut contenir un très grand nombre de nœuds capteurs (des centaines, des milliers…) [12]**.**

Un réseau de capteur est scalable parce qu'il a la faculté d'accepter un très grand nombre de nœuds qui collaborent ensemble afin d'atteindre un objectif commun.

- ***La tolérance aux pannes :*** Dans le cas de dysfonctionnement d'un nœud (manque d'énergie, interférences avec l'environnement d'observation…) ou aussi en cas d'ajout de nouveaux nœuds capteurs dans le réseau, ce nœud doit continuer à fonctionner normalement sans interruption [12]. Ceci explique le fait qu'un RCSF n'adopte pas de topologie fixe mais plutôt dynamique.

- ***Une densité importante des nœuds*** *:* Les RCSF sont caractérisés par leur forte densité [15]. Cette densité peut atteindre, selon le type d'application, 20 nœuds/m^3 [12].

- ***La capacité de communication :*** Elle peut prendre deux aspects : Le multisaut ou à un seul saut [15]. Parce que le multisaut est moins énergivore, il reste le type de communication le plus sollicité par les applications de RCSF qui requièrent une faible consommation d'énergie.

- ***Les types de communication :*** Il existe différents types de communication utilisée dans les RCSF :

Unicast : ce type de communication est utilisé pour échanger des informations entre deux nœuds sur le réseau.

Broadcast : la station de base ou « Sink » transmet des informations vers tous les nœuds du réseau. Ces informations peuvent être des requêtes de données bien précises (ex : la température dans la région A), des mises à jour de programmes ou des paquets de contrôle…[12]

Local Gossip : ce type de communication est utilisé par des nœuds situés dans une région bien déterminée qui collaborent ensemble afin d'avoir une meilleure estimation de l'évènement observé et d'éviter l'émission du même message vers le nœud « Sink » ce qui contribue à consommer moins d'énergie.

Convergecast : il est utilisé dans les communications entre un groupe de nœuds et un nœud bien spécifique (qui peut être le « Sink »). L'avantage de ce type de communication est la diminution de contrôle d'entête des paquets (« control overhead ») ce qui économise l'énergie au niveau du nœud récepteur [13].

Multicast : il permet une communication entre un nœud et un groupe de nœuds. Ce type de communication est utilisé dans les protocoles qui incluent le « clustering » dans lesquels, le « Clusterhead » s'intéresse à communiquer avec un groupe de nœuds [12].

- ***Une architecture « data-centric » :*** Du fait que le remplacement ou la recharge des batteries des nœuds capteurs est une tâche non pratique et difficile à réaliser, alors il est d'usage normal qu'on trouve des nœuds capteurs redondants (effectuant la même tâche dans la même région) ;

L'importance d'un nœud particulier est, par conséquent, réduite par rapport à l'importance attribuée aux données observées par les nœuds.

Ce type d'architecture diffère des architectures « node-centric » adoptées par les réseaux traditionnels où les nœuds possèdent une place importante (Exemple : un utilisateur qui veut connecter *son laptop* au *serveur web X*) [13].

- ***Une collaboration entre les nœuds :*** Les contraintes strictes de consommation d'énergie mènent les nœuds capteurs à détecter et traiter les données d'une manière coopérative afin d'éviter le traitement redondant d'une même donnée observée, source de la perte d'énergie [6].

- ***La bande passante (ou capacité du canal) :*** c'est une caractéristique beaucoup plus importante dans les réseaux cellulaires (GSM) et les réseaux locaux sans fils (WLAN), que dans les RCSF ; le débit étant en effet un objectif secondaire pour les RCSF. [17]

6. Architecture des réseaux de capteurs

Puisque les RCSF se caractérisent par l'absence d'une infrastructure déterminée au préalable, les nœuds capteurs la construisent tout en permettant l'interaction avec l'environnement où ils appartiennent et en répondant aux différentes requêtes venant des utilisateurs ou des réseaux externes.

Par ailleurs, les nœuds capteurs comme tout autre composant de télécommunication adhèrent à une architecture protocolaire spécifique. La réalisation de cette dernière requiert la mise en œuvre de techniques développées pour les réseaux Ad Hoc. Cependant, de nouveaux problèmes apparaissent engendrés entre autre par la sévérité des contraintes dues aux limitations de ressources physiques des RCSF. C'est pourquoi, il est commode que la conception des protocoles de communication soit faite d'une manière optimale.

6.1 Architecture de communication

Après le déploiement des nœuds capteurs sur une certaine zone de captage, ceux-ci commencent par la découverte de leurs voisins afin de construire la topologie de communication. Ainsi, ils deviennent capables d'accomplir les tâches que leur sont affectées.

Selon une communication multi-sauts, les capteurs sont chargés de collecter des données, les router vers un nœud particulier appelé nœud puits. Ce dernier analyse ces données et transmet à son tour l'information collectée à l'utilisateur via internet ou bien satellite. Comme l'indique la figure 5, l'ensemble de nœuds construisant le RCSF est considéré comme étant un réseau d'acquisition de données. Par contre, le réseau de distribution de données est composé des utilisateurs, et du réseau de communication : l'internet, et les satellites.

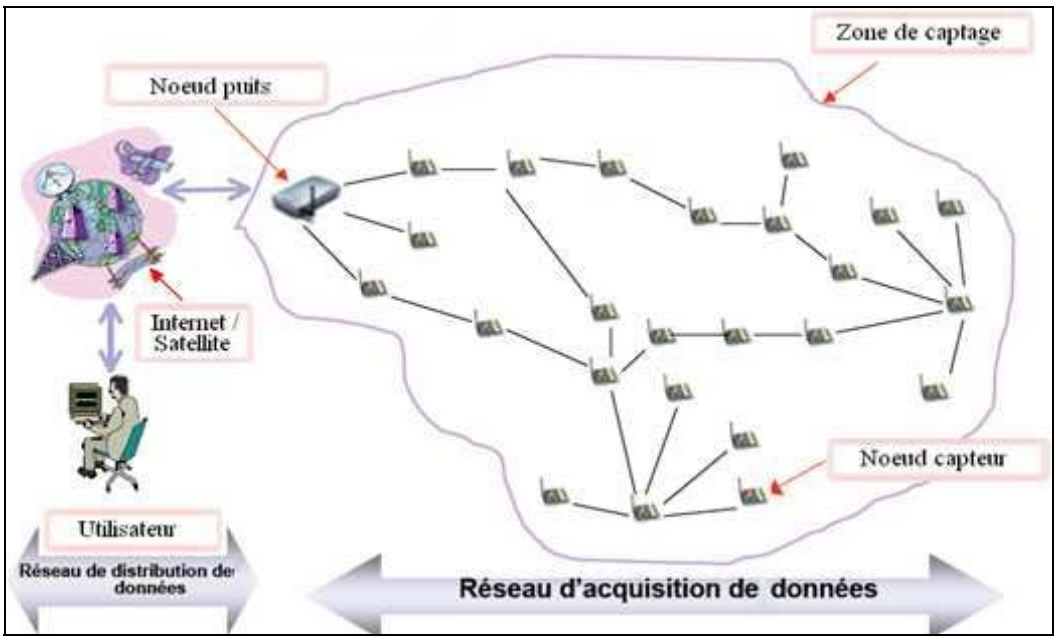

Figure 5 : Architecture de communication d'un réseau de capteurs. [18]

6.2 Architecture protocolaire

Dans le but d'un établissement efficace d'un RCSF, une architecture en couches est adoptée afin d'améliorer la robustesse du réseau. Une pile protocolaire de cinq couches est donc utilisée par les nœuds du réseau. Citons la couche application, la couche transport, la couche réseau, la couche liaison de données et la couche physique.

De plus, cette pile possède trois plans (niveaux) de gestion : le plan de gestion des tâches qui permet de bien affecter les tâches aux nœuds capteurs, le plan de gestion de mobilité qui permet de garder une image sur la localisation des nœuds pendant la phase de routage, et, le plan de gestion de l'énergie qui permet de conserver le maximum d'énergie.

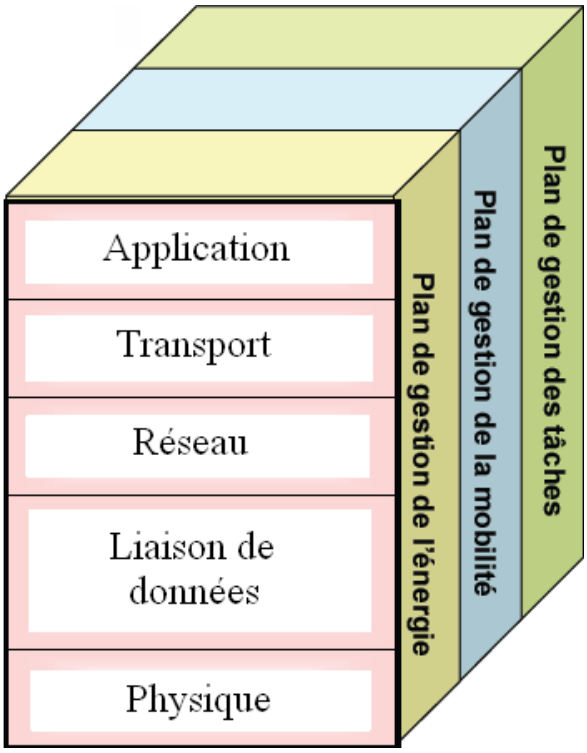

Figure 6 : La pile protocolaire dans les réseaux de capteurs. [18]

6.3 Couches de la pile protocolaire [18, 19]

· **Couche application :** Elle assure l'interface avec les applications. Il s'agit donc de la couche la plus proche des utilisateurs, gérée directement par les logiciels. Parmi les protocoles d'application, nous citons : SMP *(Sensor Management Protocol)* et TADAP *(Task Assignement and Data Advertisement Protocol)*.

· **Couche transport :** Elle vérifie le bon acheminement des données et la qualité de la transmission. Dans les RCSF, la fiabilité de transmission n'est pas majeure. Ainsi, les erreurs et les pertes sont tolérées. Par conséquent, un protocole de transport proche du protocole UDP et appelé UDP-Like *(User Datagram Protocol Like)* est utilisé.

Cependant, comme le protocole de transport universel est TCP *(Transmission Control Protocol*), les RCSF doivent donc posséder, lors d'une communication avec un réseau externe, une interface TCP-spliting pour vérifier la compatibilité entre ces deux réseaux communicants.

· **Couche réseau :** Elle s'occupe du routage de données fournies par la couche transport.

Elle établit les routes entre les nœuds capteurs et le nœud puits et sélectionne le meilleur chemin en termes d'énergie, délai de transmission, débit, etc.

Les protocoles de routage conçus pour les RCSF sont différents de ceux conçus pour les réseaux Ad Hoc puisque les RCSF sont différents selon plusieurs critères comme :

- l'absence d'adressage fixe des nœuds tout en utilisant un adressage basé-attribut.

- l'établissement des communications multi-sauts.

- l'établissement des routes liant plusieurs sources en une seule destination pour agréger des données similaires, etc.

Parmi ces protocoles, nous citons : LEACH *(Low-Energy Adaptive Clustering Hierarchy)* et SAR (*Sequential Assignement Routing*).

· **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 nœuds. C'est-à-dire, elle détermine les liens de communication entre eux dans une distance d'un seul saut.

Parmi les protocoles de liaison de données, nous citons: SMACS *(Self-organizing Medium*

Access Control for Sensor networks) et EAR *(Eavesdrop And Register).*

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

7. Comparaison entre les RCSF et les réseaux sans fil classiques

Le tableau suivant illustre une comparaison entre les RCSF et les réseaux sans fil classiques. Cette comparaison s'appuie sur cinq critères : le nombre de nœuds, l'importance de consommation d'énergie, l'importance de la QoS, l'identification des nœuds et les types de communication.

	RCSF	Réseau WLAN	Réseau cellulaire	Réseau WPAN
Nombre de nœuds	Très élevé	Diminué	Elevé	1 nœud maître avec 7 nœuds esclaves (Cas de Bluetooth)
L'importance de la consommation d'énergie	Très élevé.	Diminué du fait que les nœuds peuvent être rechargés facilement.	Diminué du fait que les nœuds peuvent être rechargés facilement.	Diminué
Importance de la qualité de service	Diminué	Elevée	Elevée	Elevée
Identification des nœuds	Dépend de l'application (généralement pas de mécanismes d'identification pour éviter l'overhead)	Existe	Existe	Existe
Type de communication	broadcast, multicast, Convergecast…	Point à point	Du mobile vers la station de base et vice versa.	Du nœud maître vers le nœud esclave.

Table1 : Comparaison entre les RCSF et les réseaux sans fil

8. Domaines d'application des réseaux de capteurs sans fil

Les RCSF peuvent avoir beaucoup d'applications (voir figure suivantes). Parmi elles, nous citons :

8.1 Applications militaires

Le déploiement rapide, l'auto-configuration et la tolérance aux pannes des réseaux de capteurs sont des caractéristiques qui font de ce type de réseaux un outil appréciable dans un

tel domaine. Déploiement 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 troupes (par la détection d'agents chimiques, biologiques ou de radiations, par exemple)[16].

8.2 Applications liées à la sécurité

Les altérations 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, voire des décennies. Un réseau de capteurs de mouvements peut constituer un système d'alarme distribué qui servira à détecter les intrusions sur un large secteur.

Déconnecter le système ne serait plus aussi simple, puisqu'il n'existe pas de point critique. La surveillance de voies ferrées pour prévenir des accidents avec des animaux et des êtres humains peut être une application intéressante des réseaux de capteurs. La protection des barrages pourrait être accomplie en y introduisant des capteurs. La détection prompte de fuites d'eau permettrait d'éviter des dégâts. Les êtres humains sont conscients des risques et attaques qui les menacent. Du coup, ils mettent à disposition toutes les ressources humaines et financières nécessaires pour leur sécurité.

Cependant, des failles sont toujours présentes dans les mécanismes de sécurisation appliqués aujourd'hui, sans oublier leur coût très élevé. L'application des réseaux de capteurs dans le domaine de la sécurité pourrait diminuer considérablement les dépenses financières consacrées à la sécurisation des lieux et à la protection des êtres humains tout en garantissant de meilleurs résultats[16][20].

8.3 Applications environnementales

Des thermo-capteurs dispersés à partir d'un avion sur une forêt peuvent signaler un éventuel début d'incendie dans le champ de captage; ce qui permettra une meilleure efficacité pour la lutte contre les feux de forêt. Dans les champs agricoles, les capteurs peuvent être semés avec les graines. Ainsi, les zones sèches seront facilement identifiées et l'irrigation sera donc plus efficace. Sur les sites industriels, les centrales nucléaires ou dans les pétroliers, des capteurs peuvent être déployés pour détecter des fuites de produits toxiques (gaz, produits chimiques, éléments radioactifs, pétrole, etc.) et alerter les

utilisateurs dans un délai suffisamment court pour permettre une intervention efficace. Une grande quantité de capteurs peut être déployée en forêt ou dans un environnement de conservation de la faune afin de recueillir des informations diverses sur l'état du milieu naturel et sur les comportements de déplacement.

Par exemple, l'université de Pise en Italie a réalisé des réseaux de capteurs pour le contrôle des parcs naturels (feux, animaux,..). Il est ainsi possible "d'observer", sans déranger, des espèces animales difficiles à étudier dans leur environnement naturel et de proposer des solutions plus efficaces pour la conservation de la faune. Les éventuelles conséquences de la dispersion en masse des micro-capteurs dans l'environnement ont soulevé plusieurs inquiétudes. En effet, chaque micro-capteur est doté d'une batterie qui contient des métaux nocifs. Néanmoins, le déploiement d'un million de capteurs de 1 millimètre cube chacun ne représente qu'un volume total d'un litre. Même si tout ce volume était constitué de batteries, cela n'aurait pas des répercussions désastreuses sur l'environnement.

Dans les immeubles, le système de climatisation peut être conçu en intégrant plusieurs micro-capteurs dans les tuiles du plancher et les meubles. Ainsi, La climatisation pourra être déclenchée seulement aux endroits où il y a des personnes présentes et seulement si c'est nécessaire. Le système distribué pourra aussi maintenir une température homogène dans les pièces. Utilisée à grande échelle, une telle application permettrait de réduire la demande mondiale en énergie réduisant du même coup les gaz à effet de serre. Rien que pour les États-Unis, on estime cette économie à 55 milliards de dollars par an avec une diminution de 35 millions de tonnes des émissions de carbone dans l'air. Ainsi, dans un contexte mondial où le réchauffement de la planète devient une préoccupation grandissante, une telle conséquence environnementale serait un pas dans la bonne direction[21].

8.4 Applications médicales

Surveillance permanente des patients et une possibilité de collecter des informations physiologiques de meilleure qualité facilitant ainsi le diagnostic de maladies grâce à des micro-capteurs qui pourront être ingères ou implantes sous la peau.

(i) les micro-cameras qui peuvent être ingérées et sont capables, sans avoir recours `a la chirurgie, de transmettre des images de l'intérieur d'un corps humain,

(ii) la création d'une rétine artificielle composée d'une centaine de micro-capteurs pour améliorer la vision de l'œil.

8.5 Applications écologiques

L'intégration de plusieurs micro-capteurs dans le système de climatisation et de chauffage des immeubles. Ainsi, la climatisation ou le chauffage ne sont déclenches qu'aux endroits o`u il y a des personnes présentes et seulement si c'est nécessaire. Le système distribué peut aussi maintenir une température homogène dans les pièces. Utilisée `a grande échelle, une telle application permettrait probablement de réduire la demande mondiale en énergie [16].

8.6 Applications de traçabilité et de localisation

Suite `a une avalanche il est nécessaire de localiser les victimes enterrées sous la neige en équipant les personnes susceptibles de se trouver dans des zones à risque par des capteurs. Ainsi, les ´équipes de sauvetage peuvent localiser plus facilement les victimes. Contrairement aux solutions de traçabilité´e et de localisation basées sur le système de GPS (Global Positionning System), les réseaux de capteurs peuvent ˆêtre tr`es utiles dans des endroits clos comme les mines par exemple.

8.7 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 connaître la position, l'état et la direction d'un paquet ou d'une cargaison. 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 connaître la position actuelle 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[21].

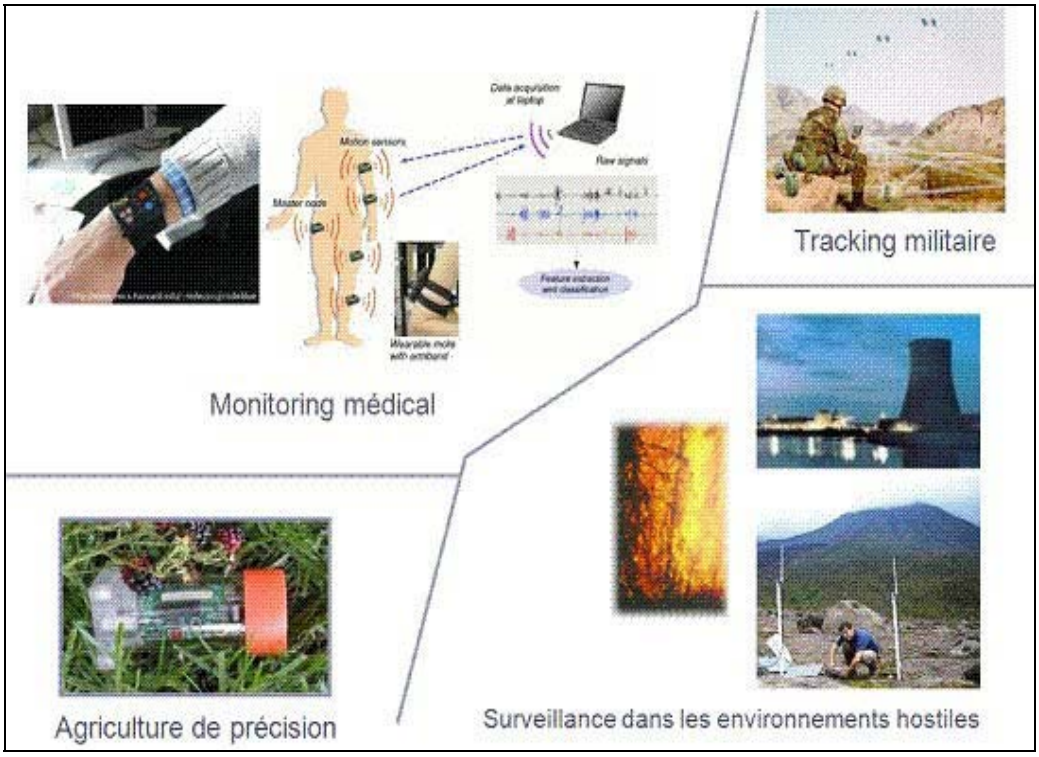

Figure 7 : Applications des RCSF [20]

9. Les challenges/Les besoins

L'utilisation d'un middleware apparaît comme un besoin fondamental pour faciliter l'évolution des réseaux de capteurs. En effet, la description non exhaustive des différentes utilisations possibles d'un tel réseau démontre la diversité des caractéristiques des applications. En effet, les applications des *RCSFs* sont fortement dépendantes du domaine et des objectifs envisagés.

A chaque application correspond une architecture du réseau et une implémentation. Certaines, par exemple, peuvent nécessiter une transmission des données fiables [07] alors que d'autres peuvent se permettent de perdre des données en cours de routage mais nécessitent une grande précision dans les données collectées.

Or les techniques actuelles nécessitent souvent de redévelopper, soit dans le meilleur des cas une partie, soit dans le pire des cas toute l'application. Un middleware peut alors

apporter une solution à ce problème en s'adaptant facilement à l'hétérogénéité des applications développables sur de tels réseaux. En effet, dans les systèmes traditionnels, un *middleware* est utilisé pour masquer l'hétérogénéité des systèmes mis en réseau et faciliter le développement d'applications distribuées. De plus, il est défini comme une couche logicielle servant d'intermédiaire entre le système d'exploitation et une application distribuée.

Cependant la conception d'un middleware pour les réseaux de capteurs n'est pas une démarche triviale. En effet, le nombre et la sévérité des contraintes existantes constituent un premier obstacle à son élaboration. De plus, les réseaux de capteurs étant une technologie particulièrement récente, il n'existe pas encore de consensus concernant la définition des interfaces du système d'exploitation. Celles-ci font toujours l'objet de recherche. En outre, de nombreuses applications exécutent des opérations matérielles directement sans passer par un composant système [22].

Malgré tout, certaines fonctionnalités récurrentes dans les applications actuelles (agrégation de données, contrôle et gestion du réseau) permettent d'établir des premiers éléments dans la définition d'un tel middleware. Dans ce cadre, des principes de conception ont néanmoins été proposés dans [23] et [24]. Un middleware doit donc être entre autre :

- Evolutif pour s'adapter aux contraintes ;
- Générique pour s'appliquer à différentes applications possédant des interfaces communes ;
- Adaptatif pour reconfigurer sa structure ;
- Réflectif pour changer de comportement en fonction de l'environnement et des circonstances ;
- Léger.

Nous avons présenté ici le contexte général, les spécificités et les contraintes du domaine dans lequel s'inscrit ce travail, nous allons, dans la section suivante, nous intéresser aux caractéristiques de TinyOS, le système d'exploitation de référence installé sur les capteurs.

De plus, des mécanismes configurables en fonction des besoins de l'application devront être proposés. Il devra également exister des mécanismes autorisant l'accès aux données détectées à partir d'application extérieures au réseau ainsi que des communications basées évènement et centrées sur les données.

10. Le système d'exploitation pour RCSF : TinyOS

TinyOS est un système d'exploitation open-source conçu pour des réseaux de capteurs sans fil. Il respecte une architecture basée sur une association de composants, réduisant la taille du code nécessaire à sa mise en place. Cela s'inscrit dans le respect des contraintes de mémoires qu'observent les réseaux de capteurs.

Pour autant, la bibliothèque de composant de TinyOS est particulièrement complète puisqu'on y retrouve des protocoles réseaux, des pilotes de capteurs et des outils d'acquisition de données. L'ensemble de ces composants peut être utilisé tel quel, il peut aussi être adapté à une application précise[16].

En s'appuyant sur un fonctionnement événementiel, TinyOS propose à l'utilisateur une gestion très précise de la consommation du capteur et permet de mieux s'adapter à la nature aléatoire de la communication sans fil entre interfaces physiques.

Disponibilité et sources : TinyOS est un système principalement développé et soutenu par l'université américaine de Berkeley, qui le propose en téléchargement sous la licence BSD et en assure le suivi. Ainsi, l'ensemble des sources sont disponibles pour de nombreuses cibles matérielles.

Event-driven : Le fonctionnement d'un système basé sur TinyOS s'appuie sur la gestion des évènements se produisant. Ainsi, l'activation de tâches, leur interruption ou encore la mise en veille du capteur s'effectue à l'apparition d'évènements, ceux-ci ayant la plus forte priorité. Ce fonctionnement évènementiel (event-driven) s'oppose au fonctionnement dit temporel (time-driven) où les actions du système sont gérées par une horloge donnée [19][25].

Langage : Comme nous l'avons évoqué plus haut, TinyOS a été programmé en **langage NesC** que nous allons détailler plus loin.

Préemptif : Le caractère préemptif d'un système d'exploitation précise si celui-ci permet l'interruption d'une tâche en cours. TinyOS ne gère pas ce mécanisme de préemption entre les tâches mais donne la priorité aux interruptions matérielles. Ainsi, les tâches entre-elles ne s'interrompent pas mais une interruption peut stopper l'exécution d'une tâche.

Temps réel : Lorsqu'un système est dit « temps réel » celui-ci gère des niveaux de priorité dans ses tâches permettant de respecter des échéances données par son environnement. Dans le cas d'un système strict, aucune échéance ne tolère de dépassement contrairement à un système temps réel mou. TinyOS se situe au-delà de ce second type car il n'est pas prévu pour avoir un fonctionnement temps réel[25].

Consommation : TinyOS a été conçu pour réduire au maximum la consommation en énergie du capteur. Ainsi, lorsqu'aucune tâche n'est active, il se met automatiquement en veille. Cibles de TinyOS :

Il existe de nombreuses cibles possibles pour ce système d'exploitation embarqué. Malgré leurs différences, elles respectent toutes globalement la même architecture basée sur un noyau central autour duquel s'articulent les différentes interfaces d'entrée-sortie, de communication et d'alimentation. Voici un schéma représentant cette architecture :

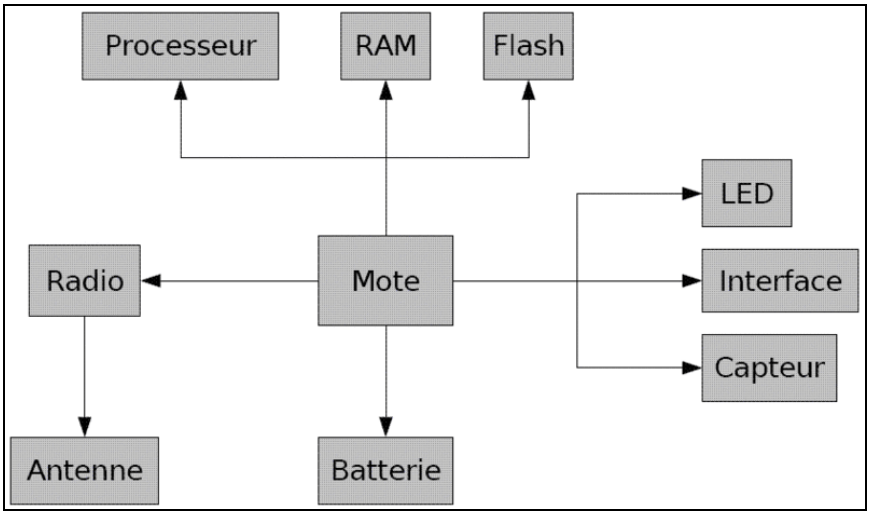

Figure 8 : schéma représentant l'architecture du TinyOS[25]

- Mote, processeur, RAM et Flash : On appelle généralement Mote la carte physique utilisant TinyOS pour fonctionner. Celle-ci a pour cœur le bloc constitué du processeur et des mémoires RAM et Flash. Cet ensemble est à la base du calcul binaire et du stockage, à la fois temporaire pour les données et définitif pour le système TinyOS.

- Radio et antenne : TinyOS est prévu pour mettre en place des réseaux sans fils, les équipements étudiés sont donc généralement équipés d'une radio ainsi que d'une antenne afin de se connecter à la couche physique que constitue les émissions hertziennes.

- LED, interface, capteur : TinyOS est prévu pour mettre en place des réseaux de capteurs, on retrouve donc des équipement bardés de différents types de détecteurs et autres entrées.

- Batterie **:** Comme tout dispositif embarqué, ceux utilisant TinyOS sont pourvus d'une alimentation autonome telle qu'une batterie [16].

11. Conclusion

Les RCSF possèdent des caractéristiques particulières qui les différencient des autres types de réseaux sans fil. Ces spécificités telles que la consommation d'énergie réduite, la scalabilité ou le routage incitent le besoin de concevoir de nouveaux protocoles d'accès au support, de routage, de sécurité, de transport ou d'application, qui s'adapteront aux caractéristiques des RCSF.

Chapitre 2 : la sécurité dans les réseaux de capteurs sans-fil

1 Introduction

Le degré d'utilisation des systèmes et réseaux d'information et l'environnement des technologies de l'information dans son ensemble ont évoluée de façon spectaculaire depuis 1992. Ces évolutions offrent des avantages significatifs mais requirent également que le gouvernement, les entreprises, les autres organisations et les utilisateurs individuels qui développent, possèdent, fournissent, gèrent, maintiennent et utilisent les systèmes et réseaux d'informations, portent une bien plus grande attention à la sécurité.

La sécurité est un enjeu majeur des technologies numériques modernes. Infrastructure de télécommunication (GSM, GPRS, UMTS), réseau sans fils (bluetooth, WiFi, WiMax), Internet, systèmes d'informations, routeurs, systèmes d'exploitation, applications informatiques, toutes ces entités présentent des vulnérabilités : faille de sécurité, défaut de conception ou de configuration. Ces systèmes tombent en panne, subissent des erreurs d'utilisation et sont attaqués de l'extérieur ou de l'intérieur par des pirates, des cybercriminels.

2 Conditions De Sécurité

Un réseau de capteur est un type spécial de réseaux. Il partage quelques vulgarisations avec un réseau informatique typique, mais 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 [26], à savoir :

2.1 Confidentialité Des Données

La confidentialité des données est la question la plus importante dans la sécurité de réseau. L'approche standard pour sécuriser le transfert des données est de crypter les données avec une clef secrète connue par l'émetteur et le récepteur

2.2 Intégrité des données

Un nœud intrus (adversaire) peut modifier les données transférées. Par exemple, un nœud malveillant peut ajouter quelques fragments ou manœuvrer les données dans un paquet.

Ce nouveau paquet peut alors être envoyé au récepteur original. La perte ou les dommages de données peut même se produire sans présence d'un nœud malveillant dû à l'environnement dur de communication. Ainsi, l'intégrité des données s'assure qu'aucune donnée reçue n'a été changée en transit.

2.3 Fraîcheur De Données

Même si la confidentialité et l'intégrité des données sont assurées, nous devons également assurer la fraîcheur de chaque message. Officieusement, la fraîcheur de données suggère que les données soient récentes, et elles s'assurent qu'aucun vieux message n'a été rejoué. Cette condition est particulièrement importante quand il y a des stratégies de partager-clef utilisées dans la conception. Des clefs typiquement partagées doivent être changées avec le temps.

Cependant, cela prend du temps pour de nouvelles clefs partagées d'être propagées au réseau entier. Dans ce cas-ci, il est facile pour l'adversaire d'employer une attaque de rejouer. Pour résoudre ce problème un compteur relatif au temps différent, peut être ajouté dans le paquet pour assurer la fraîcheur de données.

2.4 Auto-Organisation

Un réseau de capteur sans fil est typiquement un réseau adhoc, qui exige chaque nœud capteur soit indépendant et assez flexible à l'auto organisation. Il n'y a aucune infrastructure fixe disponible pour la gestion de réseau dans un réseau de capteurs. L'auto organisation apporte un grand défi à la sécurité du réseau de capteurs sans fil[26].

2.5 La Localisation

Souvent, l'utilité d'un réseau de capteur se fondera sur ses capacités de localiser automatiquement chaque capteur dans le réseau. Un réseau de capteurs conçu pour détecter des anomalies aura besoin de l'information précise d'endroit afin d'indiquer exactement l'endroit d'un défaut.

2.6 Authentification

Un adversaire n'est pas limité simplement à modifier le paquet de données. Il peut changer le jet entier de paquets en injectant les paquets additionnels. Ainsi le récepteur doit s'assurer que les données utilisées dans n'importe quel processus décisionnel proviennent de la source correcte.

D'autre part, en construisant le réseau de capteurs, l'authentification est nécessaire pour beaucoup de tâches administratives (coefficient de reprogrammation ou de contrôle). D'après ce qui précède, nous pouvons voir que l'authentification de message est importante pour beaucoup d'applications dans les réseaux de capteurs. Officieusement, l'authentification de données permet à un récepteur de vérifier que les données sont vraiment envoyées par l'expéditeur réclamé.

Dans le cas de communication bipartite, l'authentification de données peut être réalisée par un mécanisme purement symétrique : l'expéditeur et le récepteur partagent une clef secrète pour calculer le code d'authentification de message (IMPER) de toutes les données communiquées [26][27].

3 Vulnérabilités de la sécurité dans les RCSF

Les principaux problèmes de sécurité dans les RCSF émergent à partir des propriétés qui les rendent efficaces et attrayants, qui sont:

Limitation de ressources: l'énergie est peut-être la contrainte la plus forte aux capacités d'un nœud capteur. La réserve d'énergie de chaque nSud doit être conservée pour prolonger sa durée de vie et ainsi que celle de l'ensemble du réseau. Dans la plupart du temps, l'information transmise est redondante vus que les capteurs sont généralement géographiquement co-localisés.

La plupart de cette énergie peut donc être économisée par agrégation de données. Cela exige une attention particulière à détecter l'injection de fausses données ou la modification défectueuse de données, lors des opérations d'agrégation au niveau des nœuds intermédiaires.

La communication sans fils multi-sauts: en plus de fournir un déploiement simple, la communication sans fil a l'avantage d'offrir l'accès à des endroits difficilement

accessibles tels que des terrains désastreux et hostiles. Malheureusement, la portée de la communication radio des "motes" est limitée en raison de considérations énergétiques. La communication multi-sauts est donc indispensable pour la diffusion des données dans un RCSF. Cela introduit de nombreuses failles de sécurité à deux niveaux différents: attaque de la construction et maintenance des routes, et attaque des données utiles par injection, la modification ou la suppression de paquets. En outre, la communication sans fil introduit d'autres vulnérabilités à la couche liaison en ouvrant la porte à des attaques de brouillage et de style déni de service par épuisement des batteries.

Couplage étroit avec l'environnement: la plupart des applications de RCSF exigent un déploiement étroit des nœuds à l'intérieur ou à proximité des phénomènes à surveiller. Cette proximité physique avec l'environnement conduit à de fréquentes compromissions intentionnelles ou accidentelles des nœuds. Comme le succès des applications RCSF dépend également de leur faible coût, les nSuds ne peuvent pas se permettre une protection physique inviolable.

Par conséquent, un adversaire "bien équipé" peut extraire des informations cryptographiques des nœuds capteurs. Comme la mission d'un RCSF est généralement sans surveillance, le potentiel d'attaquer les nœuds et de récupérer leur contenu est important. Ainsi, les clefs cryptographiques et informations sensibles devraient être gérés d'une manière qui augmente la résistance à la capture des nœuds.

La figure suivante résume les problèmes de sécurité émergeant des caractéristiques d'un RCSF et les solutions à entreprendre :

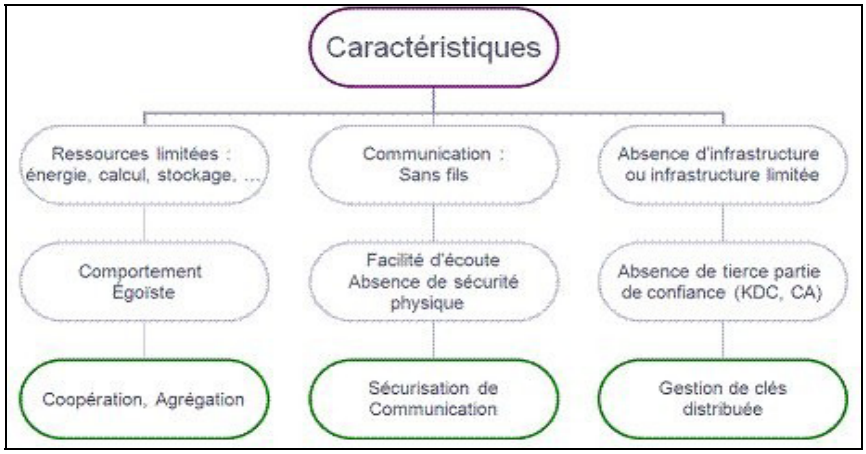

Figure 9 : Sécurité dans les RCSF : propriétés, challenges et solutions [20]

4 Bloques fonctionnels de la sécurité dans les RCSF

Comme illustré à la figure suivante, on distingue quatre blocks fonctionnels des solutions de sécurité dans les RCSF : la gestion de clés, la sécurité du routage, la sécurité de l'agrégation de données, et la sécurité de l'accès au canal.

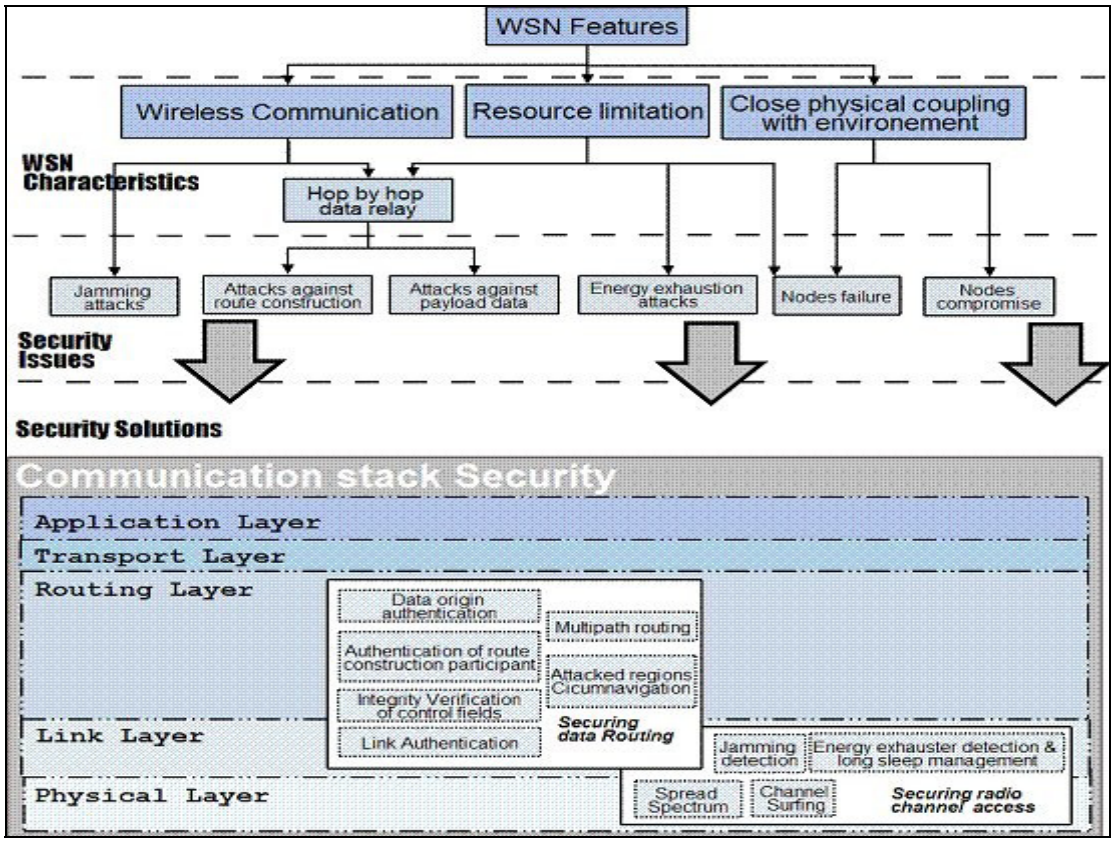

Figure 10 : Taxonomie des challenges et solutions de sécurité dans les RCSF [20]

5. Mécanismes de sécurité

Plusieurs mécanismes, basés généralement sur la notion de cryptographie, sont mis en place afin de répondre à la question de la sécurité dans les RCSF.

5.1. Définition de la cryptographie

Le mot « cryptographie » est composé des mots grecques: « crypto » signifie caché, « graphy » signifie écrire. C'est donc l'art de l'écriture secrète **[28].**

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 **[29]**.

La cryptographie est réalisée selon certains outils. Avant de les aborder, il est commode de définir la notion de clé qui sera utilisée tout au long de cette partie.

Une clé : Dans la cryptographie moderne, l'habilité de maintenir un message crypté secret, repose non pas sur les algorithmes, mais sur une information secrète dite clé qui est un paramètre utilisé en entrée d'une opération cryptographique et qui doit être utilisée avec les algorithmes pour produire le message crypté. **[28][30].**

5.2. Les outils cryptographiques

5.2.1. Le chiffrement

Le chiffrement est le système cryptographique assurant la confidentialité. Pour cela, il utilise des clés. Selon cette utilisation, on distingue deux classes de primitives : symétrique ou asymétrique.

- **Le chiffrement symétrique** Une même clé est utilisée entre deux nœuds communicants pour chiffrer et déchiffrer les données en utilisant un algorithme de chiffrement symétrique. Les algorithmes de chiffrement symétriques sont décomposés en deux catégories :
  - . Le chiffrement en chaîne est fait bit à bit sans attendre la réception entière des données. L'algorithme le plus connu est : RC4 (*Rivest Cipher 4*) **[31]**.
  - . Le chiffrement par bloc consiste à fractionner les données en blocs de taille fixe (64 bits, 128 bits). Chaque bloc sera ensuite chiffré une fois qu'il atteint a taille envisagée. Les algorithmes les plus utilisés sont **[32]** : DES (*Data Encryption Standard*), AES (*Advanced Encryption Standard*).

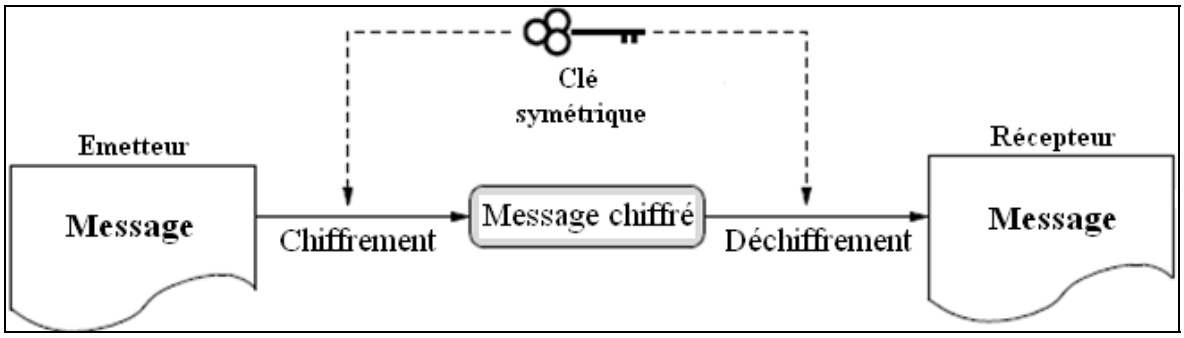

Figure 10 : Le chiffrement symétrique. [33]

Bien que l'on connaisse des algorithmes de chiffrement symétriques rapides et efficaces (comme RC4), ils peuvent exposer un grand contraste dans l'énergie dissipée pendant la construction et la distribution de clés. En effet, la distribution de clés est difficile car dans un système symétrique, chaque nœud a besoin d'une clé partagée avec chaque autre nœud du réseau. Donc on aura à gérer n*(n-1)/2 clés si on considère que le nombre de nœuds dans le réseau est égal à n **[28]**.

Cependant, de tels algorithmes cryptographiques pourraient être les plus appropriés pour les applications des RCSF. En effet, ils ne requièrent pas d'opérations mathématiques complexes pour crypter ou décrypter les données. Par conséquent, ils n'exigent pas de grandes dissipations énergétiques durant les phases de chiffrement et de déchiffrement.

- **Le chiffrement asymétrique** Deux clés différentes sont générées par le récepteur: une clé publique diffusée à tous les nœuds servant au chiffrement de données qu'ils vont émettre au récepteur, et, une clé privée maintenue secrète chez le récepteur servant pour le déchiffrement de ces données lorsque ce dernier les reçoit. Le point fondamental sur lequel repose la sécurité du chiffrement asymétrique est l'impossibilité de déduire la clé privée à partir de la clé publique. L'algorithme de chiffrement asymétrique le plus connu est : RSA (*Rivest Shamir Adleman*) [32].

Figure 11 : Le chiffrement asymétrique. [33]

Dans ces algorithmes, différents problèmes mathématiques se présentent à cause des calculs utilisés pour déchiffrer les données reçues. La complexité de telles opérations est très importante parce que les nœuds capteurs exigent une capacité de traitement plus levée et une dissipation d'énergie plus haute. En utilisant le chiffrement asymétrique, chaque nœud capteur souffre d'un autre problème dû au stockage de clés publiques de tous les nœuds restant du réseau [30].

Cela provoque une forte occupation des mémoires de chaque nœud. Cependant, la distribution de clés est moins pénible car leur échange est fortement simplifié. En effet, avec un système asymétrique, chaque nœud a besoin d'une paire de clés. Si on considère que le nombre de nœuds dans le réseau est égal à n, il faudra donc gérer 2.n clés [28].

Bien que le chiffrement asymétrique comporte des avantages, il est inutilisable dans les RCSF. Cela est dû à sa lenteur d'exécution et son coût en termes de capacité des ressources.

L'utilisation du chiffrement symétrique dans les RCSF possède quant à elle des inconvénients. Son problème majeur est de pouvoir trouver une méthode qui facilite l'établissement des clés entre les nœuds.

5.2.2. La signature digitale

La signature digitale est un système cryptographique assurant la non-répudiation de la source. Elle repose sur les clés asymétriques. L'émetteur (A) signe les données à transmettre avec sa clé privée (A) en produisant une signature digitale (1). Ce dernier est par la suite envoyé avec les données (2). Si elle peut être déchiffrée avec la clé publique (A) par le récepteur (B) et si son résultat est identique aux données reçues alors la signature est valide

(4), c'est-à-dire, les données proviennent bien de leur émetteur légitime qui ne pourra pas nier l'émission de ces données dans le futur.

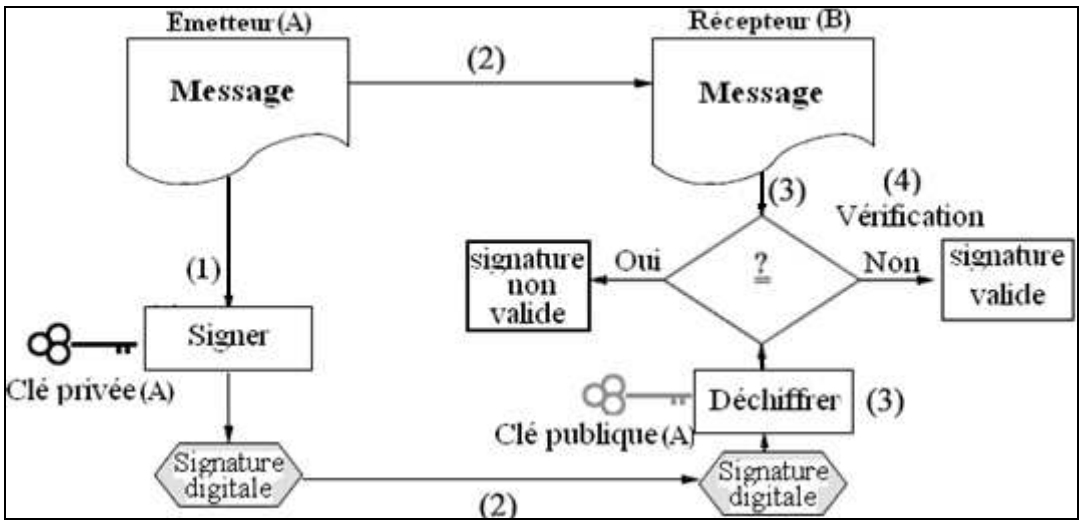

Figure 12: La signature digitale. [33]

5.2.3. La fonction de hachage

C'est le mécanisme qui assure l'intégrité de données. Cette fonction calcule une courte empreinte de taille fixe à partir d'une donnée de taille arbitraire (1). Etant donnée une fonction de hachage f, et un message à transmettre m. La fonction f doit remplir ces conditions: **[28]**

- Il est facile de calculer f(m), c'est-à-dire, de calculer l'empreinte à partir du contenu du message.
- Il est difficile de calculer m tel que f(m) = f, c'est-à-dire, de trouver le contenu du message à partir de l'empreinte. C'est pourquoi la fonction f est dite « à sens unique ».
- Il est difficile de trouver un autre message m2 tel que f(m) = f(m2), c'est-à-dire, il est difficile de trouver deux messages aléatoires qui donnent la même empreinte et cela mène à la résistance aux collisions. Cette empreinte est recalculée par le récepteur (2) afin qu'il la compare à celle calculée par l'émetteur. Si elles sont différentes (3), alors les données ont été altérées pendant leur transmission. Les fonctions de hachage les plus courantes sont: MD5 (*Message Digest 5*), SHA-1 (*Secure Hash Algorithm*) **[34]**.

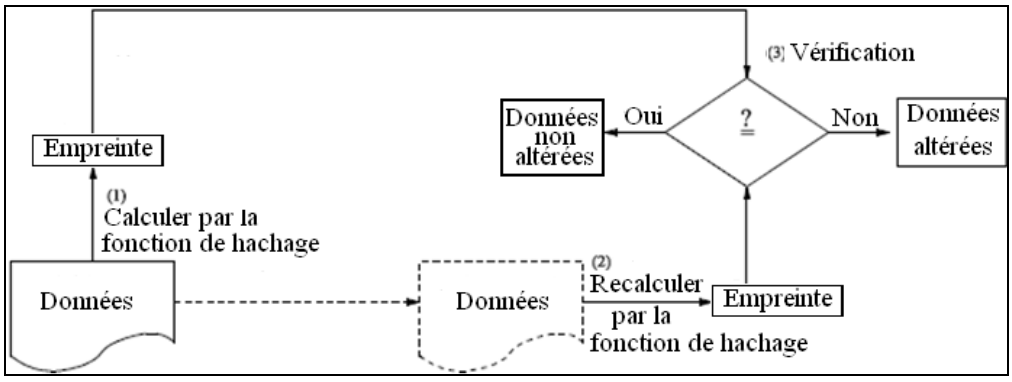

Figure 13 : La fonction de hachage. [33]

5.2.4. Le code d'authentification de message MAC

Le code d'authentification de message MAC (*Message Authentication Code)* fait partie des fonctions de hachage à clé symétrique assurant l'intégrité de données comme toute autre fonction de hachage, en plus, l'authenticité de la source de données. Cette clé est utilisée pour calculer le code MAC par l'émetteur (1). Ce code est par la suite envoyé avec les données (2).

Le récepteur calcule à son tour le code MAC avec cette même clé et le compare au code qu'il a reçu (3). S'ils sont bien identiques (4), alors la source est authentique et les données n'ont pas été altérées. Dans la pratique, HMAC *(keyed-Hash Message Authentication Code)* est utilisé [21].

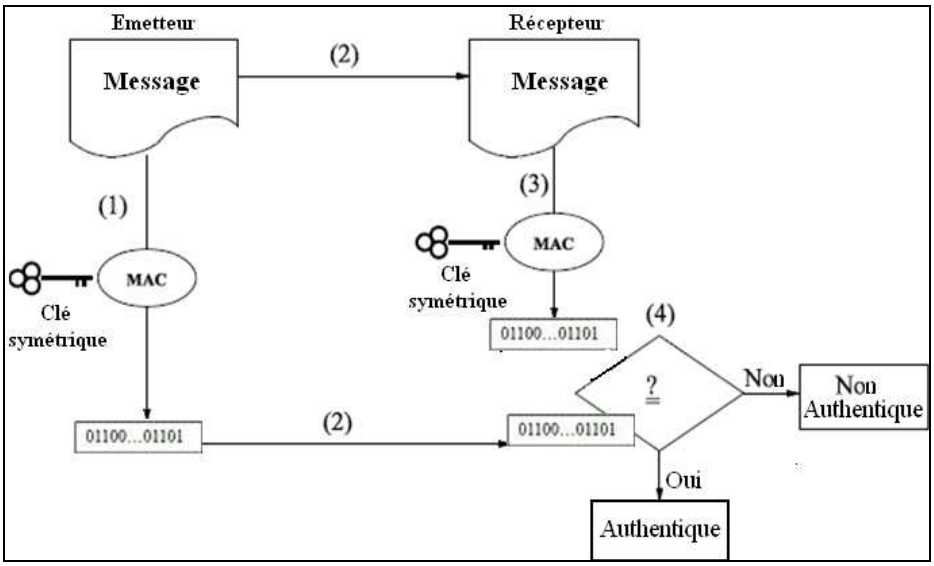

Figure 14 : Le code d'authentification de message MAC. [33]

6. La gestion de clés dans les RCSF

La gestion de clés fournit des mécanismes efficaces, sécurisés et stables de gestion de clés utilisées dans les opérations cryptographiques. Par conséquence, la gestion de clés est un service primordial pour la sécurité de n'importe quel système basé sur la communication. Sous les contraintes des RCSF, la conception d'un système de gestion de clés est un grand défi. Sélectionner une solution cryptographique appropriée pour les RCSF est un autre défi.

6.1. La fonction de gestion de clés dans les RCSF

6.1.1 Définition

La gestion des clés est un des aspects les plus difficiles de la configuration d'un système cryptographique de sécurité. Pour qu'un tel système fonctionne est soit sécurisé, chacun des utilisateurs doit disposer d'un ensemble de clés secrètes (dans un système à clés secrètes) ou de paire de clés publiques/privés (dans un système à clés publiques). Cela implique de générer les clés et de les distribuer de manière sécurisée aux utilisateurs ou d'offrir à l'utilisateur le moyen de les générer. Il doit aussi pouvoir enregistrer et gérer ses clés publiques et privées de manière sure. Dans les systèmes à clés publiques, la gestion des clés comprend la capacité à vérifier et à gérer les clés publiques des autres utilisateurs qui sont signées sous formes de certificats numériques.

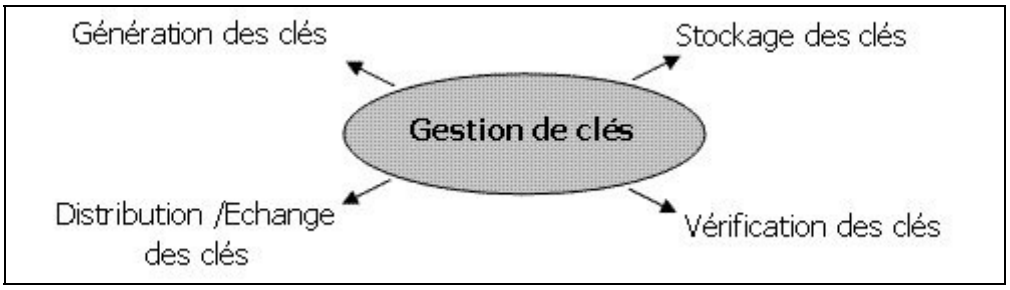

Figure 15 : Fonctions de la gestion de clés

6.1.2 Pourquoi la gestion de clés dans les RCSF ?

Après leur déploiement, les capteurs ont besoin d'établir des clés cryptographiques avec leurs voisins pour assurer des services de sécurité:

- ✓ Sécuriser le routage
- ✓ Sécuriser l'agrégation

✓ Coopération (authentification), etc.

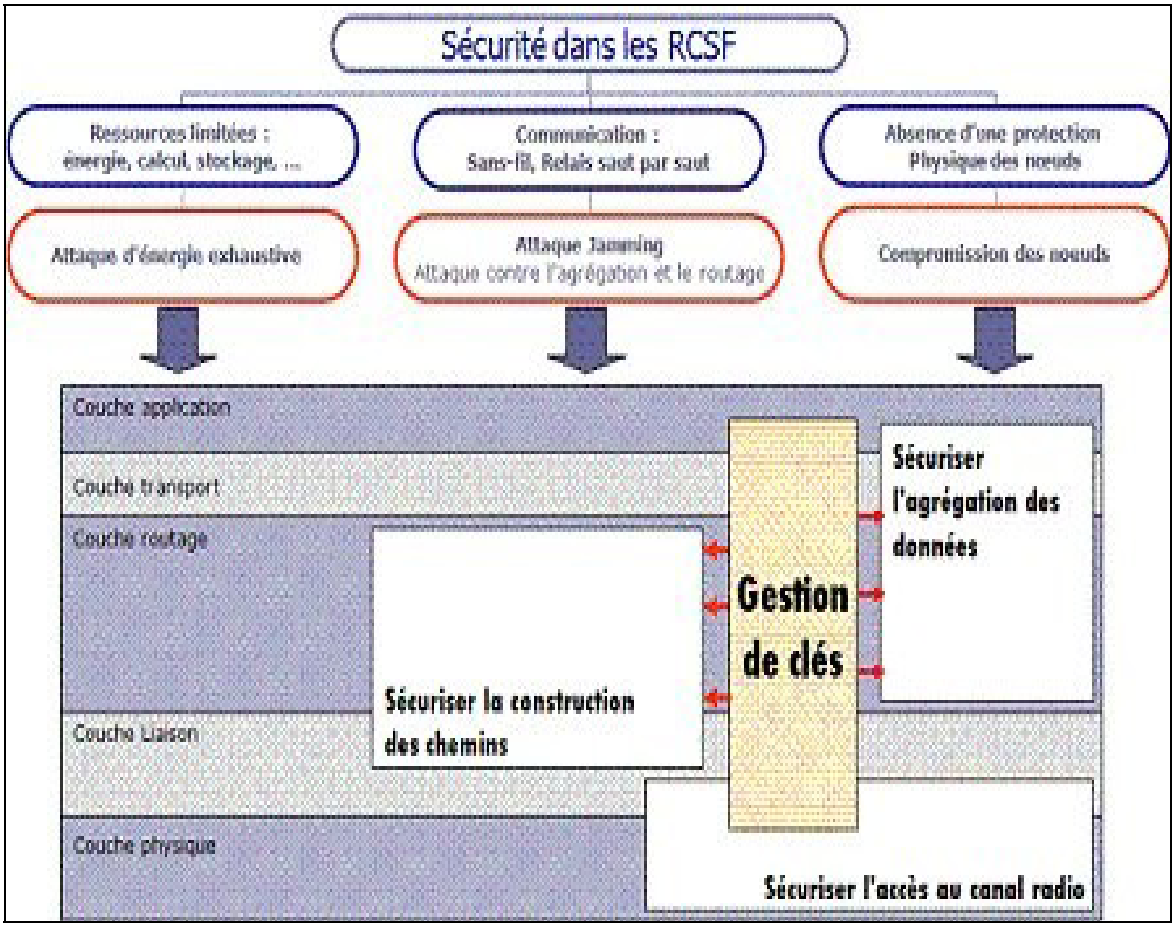

Figure 16 : Positionnement de la gestion de clé dans un RCSF sécurisé[20]

6.1.3 Contraintes de conception

La figure suivante résume les contraintes découlant des propriétés des RCSF, à prendre en compte dans la conception d'une solution de gestion de clés pour les RCSF.

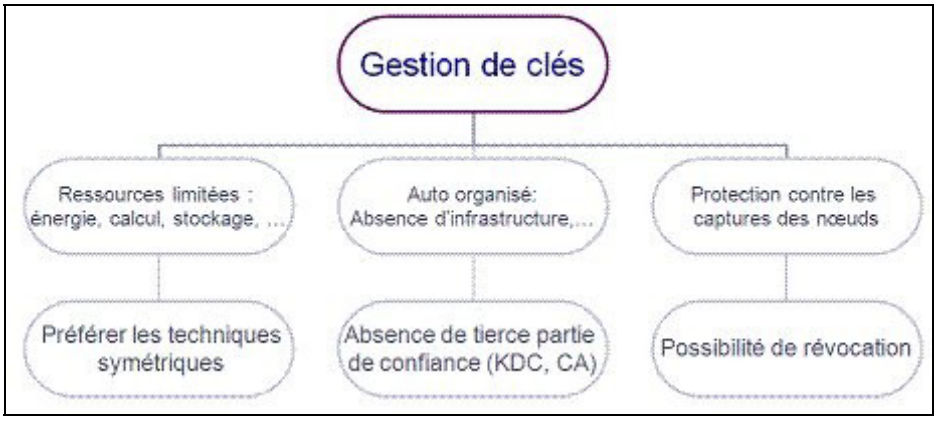

Figure 17 : Contraintes de conception de solutions de gestion de clés

6.1.4 Systèmes asymétriques ou symétriques ?

Dans les systèmes à clés publiques, l'échange de clé est fortement simplifié. Chaque partie communicante publie sa clé publique. Les clés publiques sont habituellement distribuées en utilisant des certificats numériques, utilisés par le destinataire pour authentifier la clé publique reçue ; toutes les communications avec cette partie seront alors cryptées avec cette clé. L'avantage principal d'utiliser des algorithmes à clés publiques est la facilité de gestion des clés et leur fiabilité.

Les inconvénients de cette approche incluent la consommation d'énergie due au calcule des algorithmes à clé publiques, la consommation d'énergie due à la transmission des certificats, et le stockage des clés connues pour être plus grandes que les clés symétriques. Employer des mécanismes de clés symétriques pour l'établissement de la confiance réduit considérablement la consommation d'énergie des nœuds capteurs et l'espace de stockage réservé pour accueillir ces clés.

Cependant, l'échange de clés dans les systèmes à clés symétriques est beaucoup plus compliqué. Habituellement, une seule clé symétrique est utilisée entre deux parties communicantes, sur une seule session ou sur une période limitée.

Bien que la cryptographie à clé publique comporte des avantages certains par rapport à la cryptographie à clé symétrique et malgré les recherches qui visent à les appliquer aux RCSF, la cryptographie à clé symétrique possède ses propres qualités qui la rend toujours la plus préférée pour les RCSF. Pour cette raison la plupart des schémas de gestion de clés proposés pour les RCSF sont basés sur la cryptographie symétrique.

Le problème majeur avec la cryptographie symétrique est de pouvoir trouver une méthode qui facilite l'établissement des clés entre les nœuds. La solution commune est d'utiliser une méthode de pré-distribution, dans laquelle les clés sont chargées dans les nœuds capteurs avant le déploiement.

La figure suivante illustre une taxonomie des solutions de gestion de clés basée sur la pré-distribution. Dans cette taxonomie, les protocoles sont classés selon la façon avec laquelle les nœuds voisins partagent des clés communes (probabiliste ou déterministe), et selon la topologie du réseau (hiérarchique ou plate).

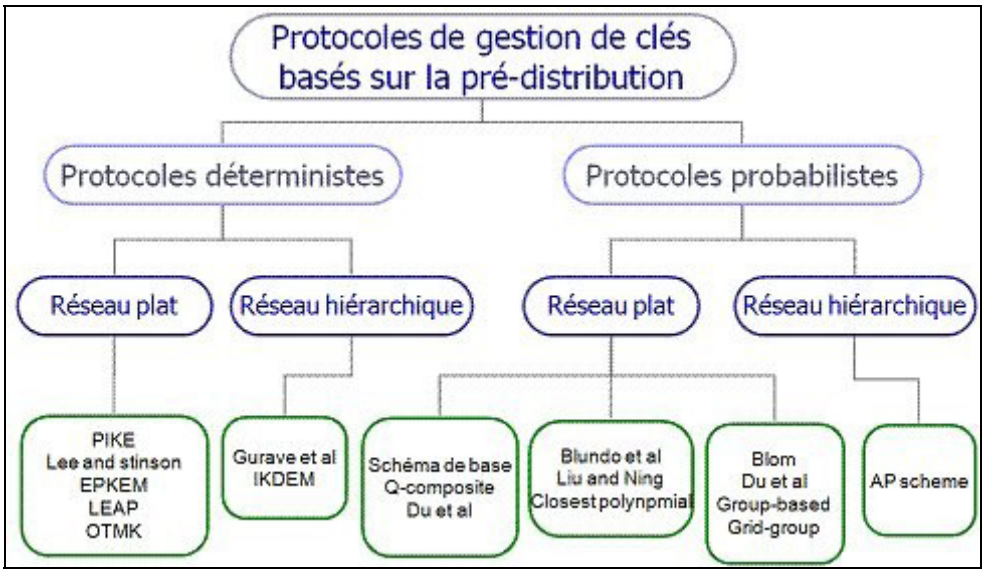

Figure 18 : Taxonomie de pré-distribution de clés pour les RCSF[20]

6.2 Schéma aléatoire de pré-distribution de clés deL.ESCHENAUER et D.GLIGOR

Eschenauer et Gligor ont proposé un schéma de gestion de clé basé sur la probabilité de partager une clé entre les nœuds d'un graphe aléatoire. Il fournit des techniques pour la pré-distribution de clé, la découverte de la clé partagée, l'établissement de chemin de clé, et la révocation de clé.

L'idée maîtresse de ce schéma, est de distribuer aléatoirement un certain nombre de clés, issues d'un ensemble fini à chaque nœud du réseau avant son déploiement. Deux nœuds quelconques seront en mesure de s'échanger des messages sécurisés s'ils possèdent une clé commune.

6.2.1 Phase de pré-distribution de clés

Un grand ensemble S de clés est générée (217-220 Clés). Pour chaque nœud, m clés sont choisies au hasard de l'ensemble S (S= {(kid1, key1), (kid2, key2),....}). Ces m clés sont stockées dans la mémoire du nœud et forment le trousseau de clés du nœud. Le nombre de clés |S| de l'ensemble est choisi de telle manière que deux sous-ensembles aléatoires de S de taille m auront une certaine probabilité p d'avoir au moins une clé en commun, par exemple pour une probabilité p=0.5 on a besoin d'un sous ensemble de taille m=75 clés de l'ensemble S de taille |S|=10,000 clés.

6.2.2 Phase de découverte de clés partagées

Les nœuds découvrent leurs voisins et plus particulièrement ceux avec qui ils sont en mesure de communiquer de façon sécurisée car ils possèdent une clé identique dans leur trousseau de clés respectif. Le protocole peut être de diffuser la liste des identités kidi des clés possédées. La clé partagée devient la clé de session du lien entre les deux nœuds. La figure suivante illustre cette phase :

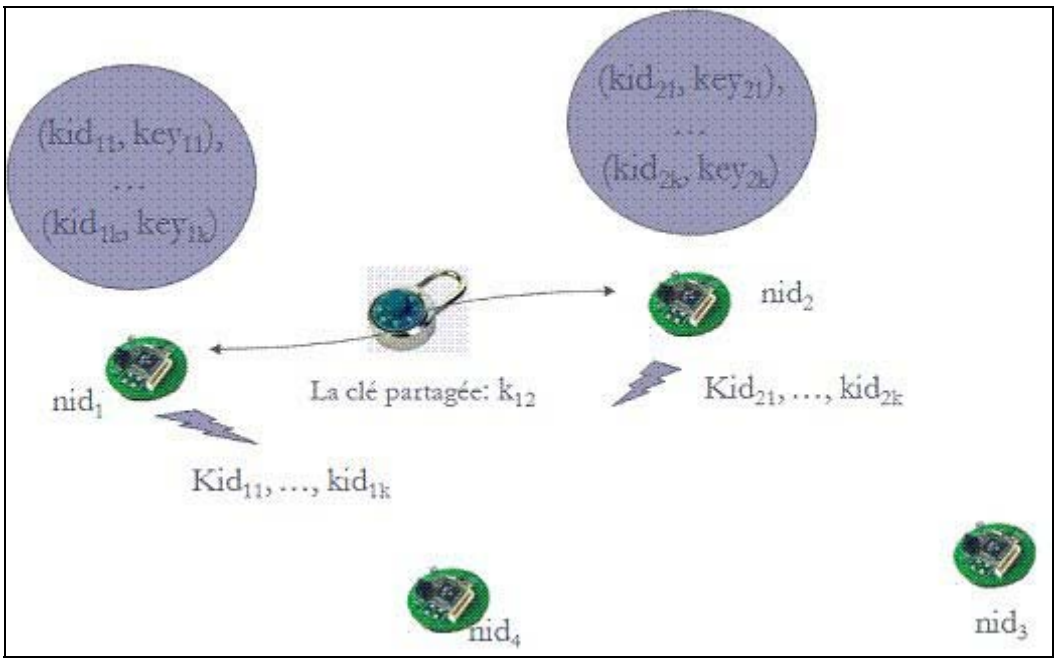

Figure 19 : Découverte des clés partagées[20]

6.2.3 Phase d'établissement de chemin de clé

Après la phase de découverte de clés partagées, le réseau devient un graphe connecté formé de quelques liens sécurisés. Les nœuds peuvent alors utiliser les liens existants pour mettre en place des clés partagées avec leurs voisins qui ne partageaient pas de clé en commun avec eux. La figure suivante illustre cette phase :

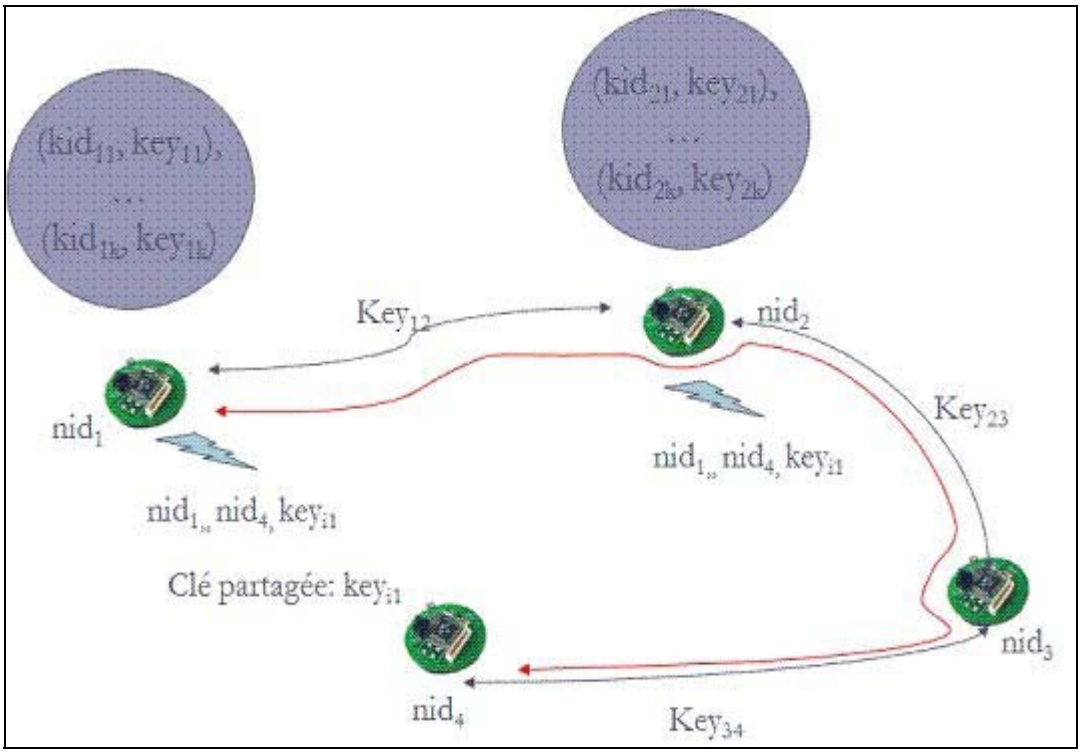

Figure 20 : Etablissement de chemins sécurisés

6.2.4 La révocation de clés

La révocation d'un nœud compromis se fait par l'élimination de leur trousseau de clés. Pour cela, un nœud contrôleur (qui a une grande connectivité et peut être mobile) annonce un message simple de révocation contenant une liste signée de k identificateurs des clés (kidi) pour que ces clés soient retirées des trousseaux de clés des autres nœuds.

La liste des identités est signée par une clé de signature Ke générée par le nœud contrôleur et envoyée en unicast à chaque nœud i en la chiffrant avec la clé Kci (la clé Kci est partagée entre le contrôleur et le ième nœud pendant la phase de pré- distribution de clés). Quelques liens seront disparus à cause de la suppression de clés du nœud compromis ce qui nécessite une reconfiguration de ces liens (par la découverte de clés partagées ou l'établissement de chemin de clé). La figure suivante illustre cette phase :

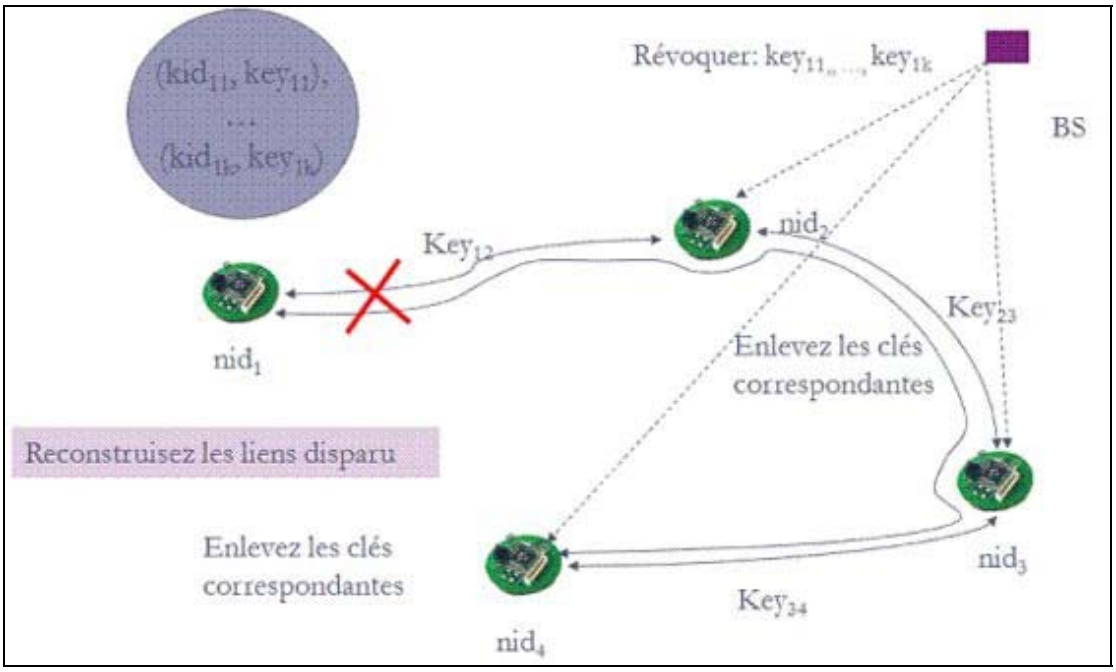

Figure 21 : Révocation de clés

6.2.4 Schéma q-composite de H.CHAN, A.PERRIG et D.SONG

Ce schéma est identique à celui de Eschenaur et Gligor sauf qu'au lieu d'exiger le partage d'une clé commune pour sécuriser un lien, une paire de nœud doit partager q clés avec q > 1 pour établir un lien sécurisé. La nouvelle clé utilisée pour la communication entre ces deux nœuds est le hash de toutes les clés partagées, par exemple pour deux nœuds quelconque qui partage q' clés ($q' \geq q$) la clé utilisée pour la communication est K = hash (k1 || k2 || ... || kq'). Plus le nombre de clé partagées augmente plus la résilience contre la capture du nœud augmente.

Autrement, lorsque le nombre, exigé, de clés partagées augmente, il devient plus difficile à un attaquant avec un ensemble donné de clés de casser un lien.

Cependant, pour préserver une probabilité donnée p que deux nœuds partageant des clés suffisantes pour établir un lien sécurisé, il est nécessaire de réduire la taille de l'ensemble de clés S. Ceci permet à un attaquant de gagner un plus grand échantillon de S en cassant peu de nœuds. La figure suivante illustre un exemple de partage de clés avec q=2.

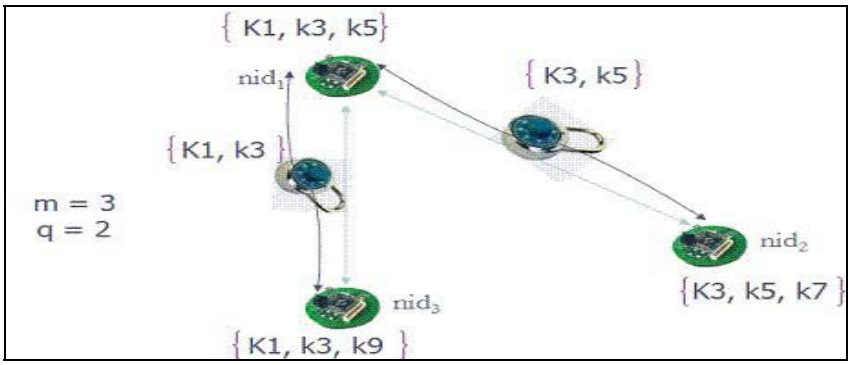

Figure 22 : Schéma q-composite

6.3 LEAP

LEAP est un protocole déterministe de gestion de clés pour les réseaux de capteurs sans-fils. Le mécanisme de gestion de clés fourni par LEAP support le traitement interne "in-network processing" tout en limitant l'impact de sécurité d'un nœud compromis sur son voisinage immédiat dans le réseau. LEAP support l'établissement de quatre type de clés pour chaque nœud : cllé individuelle, clé par paire, clé de groupe et clé globale.

6.3.1 Hypothèse de fonctionnement

LEAP est basé sur une clé initiale transitoire KIN chargée dans chacun des nœuds du réseau. Les auteurs de LEAP supposent que pour compromettre un nœud, l'adversaire nécessite un temps minimal Tmin : c'est le temps de brancher un câble série et le temps de copier le contenu de la mémoire du nœud compromis. LEAP exploite ce temps (de confiance) pour permettre à deux nœuds voisins d'établir, d'une manière sécurisée, une clé symétrique de session à partir de la clé initiale transitoire KIN. Après Tmin, la clé KIN est supprimée de la mémoire du nœud.

6.3.2 Chargement de la clé initiale

Le contrôleur (SB) génère une clé initiale KIN et charge chaque nœud avec cette clé. Chaque nœud u dérive une clé principale (Maser Key) Ku=fKIN(u), fk étant une fonction pseudo-aléatoire.

6.3.3 Découverte des voisins

Immédiatement après son déploiement, le nœud u essaye de découvrir ses voisins en diffusant un message HELLO qui contient son id. Aussi, il initie un *timer* qui sera déclenché après le temps Tmin. Le nœud u attend un ACK de chacun de ses voisins v qui contient

l'identificateur de v. l'ACK est authentifié en utilisant la clé principale Kv, qui est dérivée comme suit : Kv=fKIN(v). Comme le nœud u a la clé KIN, il pourra aussi dériver Kv, ainsi il pourra vérifier l'authenticité du ACK reçus :

u ==> *, u

v ==> u, v | MAC(Kv, u|v)

6.3.4 Etablissement de la clé par-paire

Le nœud u calcule sa clé par paire Kuv avec v, comme suit : Kuv=fKv(u).

Le nœud v peut de même calculer Kuv de la même manière.

Kuv sert comme clé entre u et v.

6.3.5 Effacement des clés

Lorsque le *timer* expire après Tmin, le nœud u efface KIN et toutes les clés principales Kv de ses voisins. Il est à noter que le nœud u n'efface pas sa clé principale Ku.

6.3.6 Sécurité de LEAP

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. de plus, aucun nœud dans le réseau ne possède la clé KIN. Un adversaire peut écouter clandestinement tout le trafic dans cette phase, mais sans la clé KIN il ne peut injecter des informations incorrectes ou déchiffrer les messages. Un adversaire compromettant un nœud après Tmin, obtient seulement les clés du nœud compromis. Quand un nœud compromis est détecté, ses voisins suppriment simplement les clés qui ont été partagée avec lui.

7. Sécurité du routage dans les RCSF

La couche de routage est le module responsable d'acheminer correctement une donnée d'un point du réseau vers un autre. Pour assurer ce rôle, la couche de routage est composée de deux blocs fonctionnels : la construction de routes et le relais des données. Le premier composant permet de construire un backbone connectant les nœuds aux destinations désirées via un ensemble de chemins.

Le deuxième composant quant à lui utilise ce backbone afin d'acheminer les données captées vers les utilisateurs finaux. Un adversaire désirant attaquer les réseaux peut alors s'en prendre à l'un des deux composants qu'il faudra protéger.

7.1. Attaques sur les protocoles de routage dans les RCSF

Vus les contraintes des RCSF, la plus part des protocoles de routage sont assez simple, et par conséquent assez vulnérables aux attaques. Un nœud malicieux peut opérer sur deux niveaux :

- Les données échangées entre les nœuds
- La topologie du réseau créée par le protocole

Ces attaques peuvent être classées en deux catégories : actives et passives.

7.1.1 Attaques actives

Attaque de "jamming"

Vu la sensibilité du média sans fil au bruit, un nœud peut provoquer un déni de service en émettant des signaux à une certaine fréquence. Cette attaque peut être très dangereuse car elle peut être menée par une personne non authentifiée et étrangère au réseau.

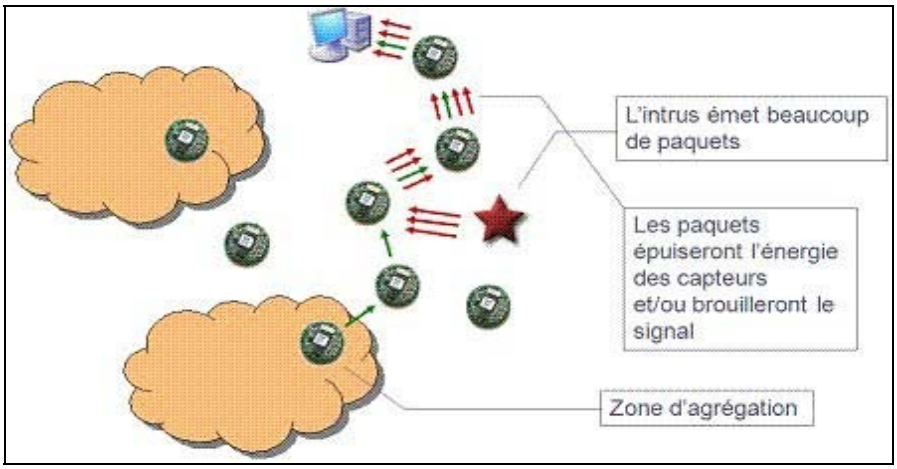

Figure 23 : Attaque de "jamming"

Attaque Sink hole

Dans une attaque sinkhole, le nœud essaye d'attirer vers lui le plus de chemins possibles permettant le contrôle sur la plus part des données circulant dans le réseau. pour ce faire, l'attaquant doit apparaître aux autres comme étant très attractif, en présentant des routes optimales.

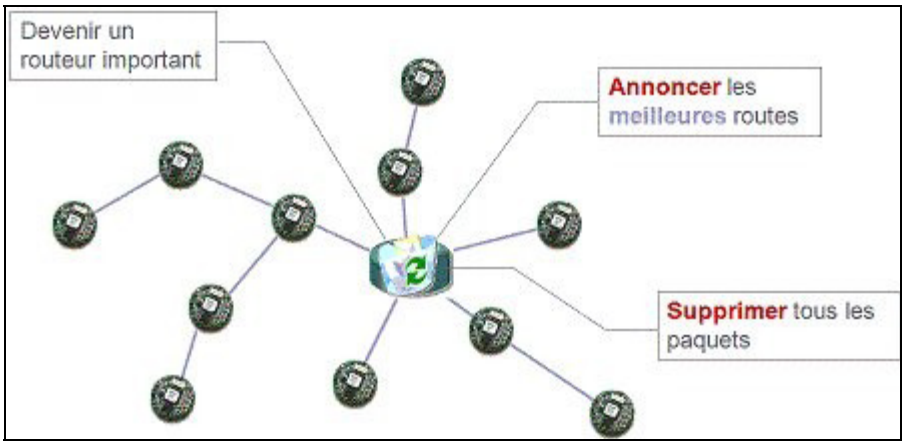

Figure 24 : Attaque sinkhole

Attaque Wormhole

Dans une attaque wormhole, un attaquant reçoit des paquets dans un point du réseau, puis les encapsule vers un autre attaquant pour les réintroduire dans le réseau. L'encapsulation peut se faire de deux manières:

- Multi-sauts: l'encapsulation multi-sauts permet de cacher les nœuds se trouvant entre les deux attaquants. Donc, les chemins passant par le nœud malicieux apparaissent plus courts. Cela facilite la création de sinkholes avec des protocoles qui utilisent le nombre de sauts comme métrique de choix de chemins.
- Communication directe: les routes passant par les attaquants sont plus rapides, car ils sont à un saut. Donc, cette technique peut être employée contre les protocoles qui se basent sur la latence des routes ou ceux qui utilisent la première route découverte.

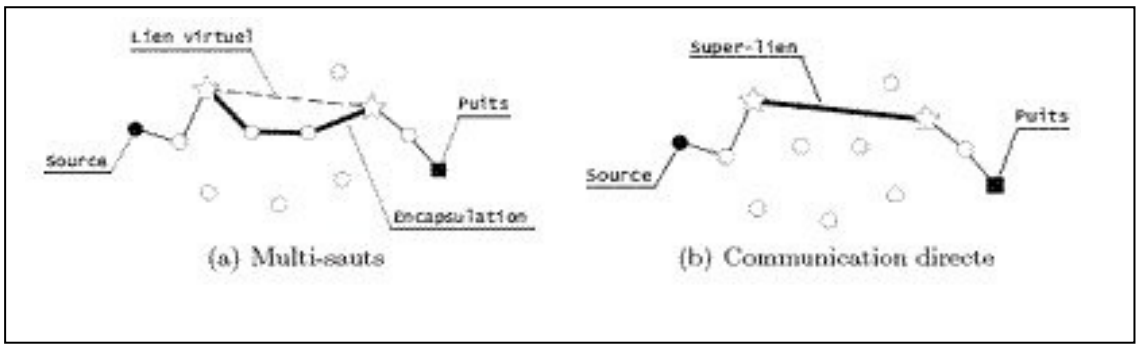

Figure 25 : Attaque Wormhole

Routing table poisoning

Certaines optimisations ont été développées afin d'augmenter la connaissance des chemins. Lorsqu'un nœud entend (en mode promiscuous) une information de routage, il met à jour sa table de routage locale en conséquence. Un nœud malicieux peut émettre un nombre important de fausses informations, remplissant ainsi les tables de routage des nœuds. Comme ces tables possèdent des tailles limitées, cela va engendrer un débordement, et les tables ne contiendront que de fausses routes.

Attaque Sybil

Dans certains algorithmes, la fiabilité du routage est implémentée par l'instauration d'une redondance de chemins. Un attaquant peut altérer ce genre de systèmes en "endossant" plusieurs identités, ce qui permet de créer plusieurs routes passant par le nœud malicieux, qui ne sont en réalité qu'un seul chemin.

Attaque Hello flooding

La faible portée de capteurs et la présence d'attaquant de classe laptop ont permis l'introduction d'une nouvelle attaque : hello flooding. Cette attaque se base sur le fait que la plus part des liens entre l'attaquant laptop et les capteurs sont unidirectionnels. Donc, un attaquant peut diffuser l'information d'une route optimale vers tous les nœuds du réseau en émettant avec un signal puissant, et tous les nœuds mettront à jour leurs tables locales. Lorsqu'un nœud veut communiquer, il ne pourra pas utiliser cette route car le prochain saut, qui est l'attaquant, est hors portée.

7.1.2 Attaques passives

Selective Forwarding

Tous les protocoles de routage supposent que les nœuds sont "honnêtes" et vont relayer normalement les paquets qui transitent par eux. Cependant, un attaquant peut violer cette règle en supprimant la totalité ou une partie de ces paquets. De plus, si l'attaquant au paravent utilisé une attaque sinkhole, il devient un routeur important dans le réseau. Donc, en abandonnant son rôle de routeur, les performances du système seront gravement dégradées.

Eavesdropping

Comme le média sans fil est un média ouvert, un nœud peut entendre toutes les communications de ses voisins. Cela peut divulguer d'importantes informations, comme la localisation d'un nœud important. La combinaison avec une attaque sinkhole aggrave d'avantage l'impact de cette attaque.

7.2 Types de solutions

Nous distinguons trois niveaux de solutions aux attaques sur le routage de données dans les RCSF :

La prévention contre les attaques actives: dans cette catégorie on utilise généralement des mécanismes cryptographiques pour protéger la signalisation qui sert à la construction de routes. C'est généralement des mécanismes d'authentification et de contrôle d'intégrité qui sont utilisés pour empêcher un nœud malicieux d'injecter, de modifier et/ou de supprimer une information qui servira pour la découverte, la construction ou la maintenance d'une route.

La détection de comportements suspects: dans cette catégorie on tente de déceler des comportements qui témoignent d'une attaque passive (manque de coopération, refus de relais de paquets, etc...).

La tolérance : dans cette catégorie, on introduit des mécanismes de tolérance de défaillance de nœuds à cause d'attaques ou de pannes. Le routage multi-chemin en est un exemple typique.

La figure suivante résume les catégories de solutions à préconiser pour faire face à différents types d'attaques :

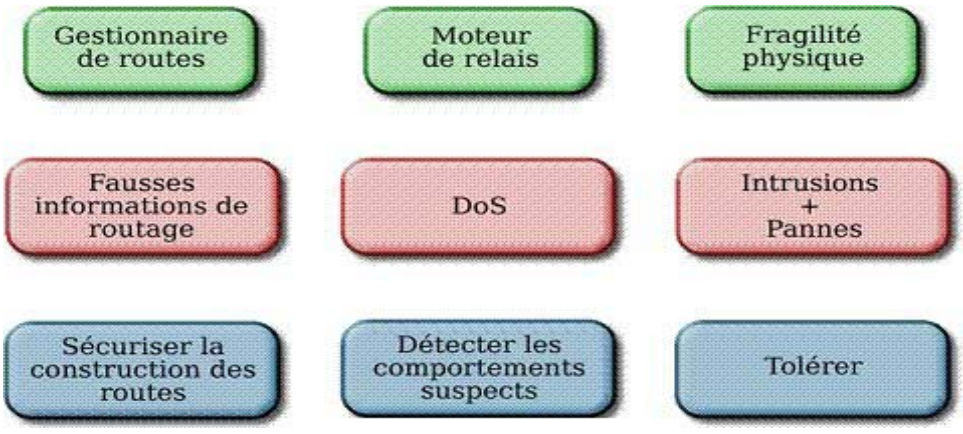

Figure26 : Catégories de solutions contre les attaques sur le routage

7.3 INSENS (Intrusion-tolerant routing for wireless sensor networks)

L'idée du protocole est de permettre à la SB de tracer une cartographie correcte du réseau qui permettra d'établir les tables de routage pour chaque capteur. Ces tables seront

transmises par la suite aux nœuds concernés de façon sécurisée. Le protocole vise deux objectifs :

- **Opérer correctement en présence d'intrus:** Pour cela, INSENS est un protocole tolérant aux intrusions, qui assure le routage même en présence d'intrus. Il construit plusieurs chemins indépendants pour chaque couple de communiquants. Les chemins indépendants ne partagent qu'un nombre réduit de nœuds/liens (idéalement, ils partagent seulement la source et la destination). Avec cette propriété, un intrus ne pourra altérer que les communications traversant un seul chemin.
- **Scalabilité et économie d'énergie**: Le calcul des chemins indépendants et l'établissement des tables de routage représentent une tâche assez lourde. INSENS effectue ces calculs dans les SB (station de base). Les résultats sont ensuite transmis vers chaque nœud d'une manière sécurisée. Le protocole se déroule selon les phases suivantes.

7.3.1 Initiation authentifiée de la construction de l'arbre

La SB doit d'abord dresser un arbre couvrant tout le réseau, dont elle est la racine. Cet arbre permettra d'acheminer les messages de contrôle entre les capteurs et la SB. Afin d'éviter le spoofing de SB, INSENS emploie un mécanisme de broadcast authentifié. Pour ce faire, la SB génère une chaîne de hachage à sens unique (ni)$0 \leq i \leq k$ comme suit : $ni+1=h(ni)$, $0<i<k$ où n0 est choisi aléatoirement, et nk est connu par tous les nœuds, et h est une fonction de hashage à sens unique.

Périodiquement, la SB génère une requête pour re-construire les tables de routage des capteurs. Le format du message est le suivant :

type|ows|size|path|MACRx

Le champ ows (One-Way Sequence number) désigne la valeur courante de la chaîne de hachage (i.e. pour la requête i, ows=ni). Chaque nœud maintient localement la dernière valeur reçue de ows, désignée par owsfresh. Lorsqu'un nœud x reçoit une requête, il vérifie sa fraîcheur et son authenticité par la relation suivante:

Trouver j, tel que j>0 et ows=hj(owsfresh)

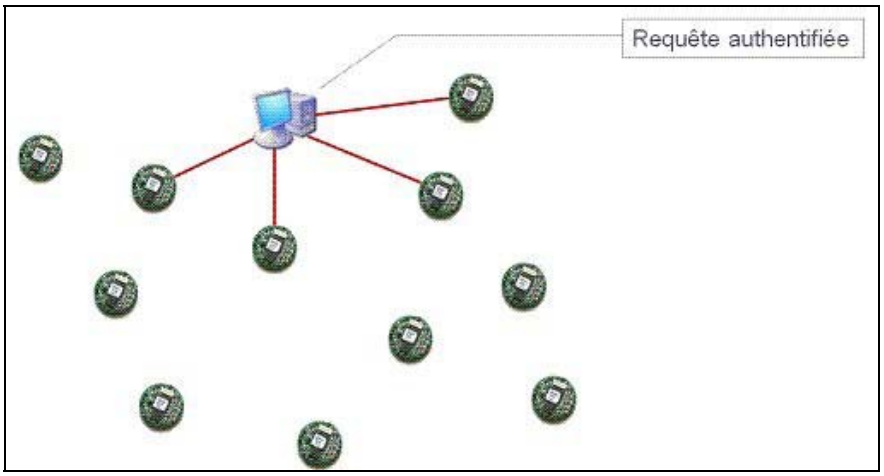

Figure 26 : Requête authentifiée de construction de l'arbre[20]

7.3.2 Construction de l'arbre par relayage de la requête

Un nœud x qui reçoit le message de requête précédent, ajoute son identificateur au champ path et calcule le champ MACRx :

$$MACRx = MAC(kx, size|path|ows|type)$$

où kx représente la clé sectrète partagée entre le nœud x et la SB. Le nœud doit aussi choisir son "upstream" vers la SB, qui représente le premier voisin qui a émis une requête valide, et sauvegarde son MACR, qui sera désigné par MACRupstream.

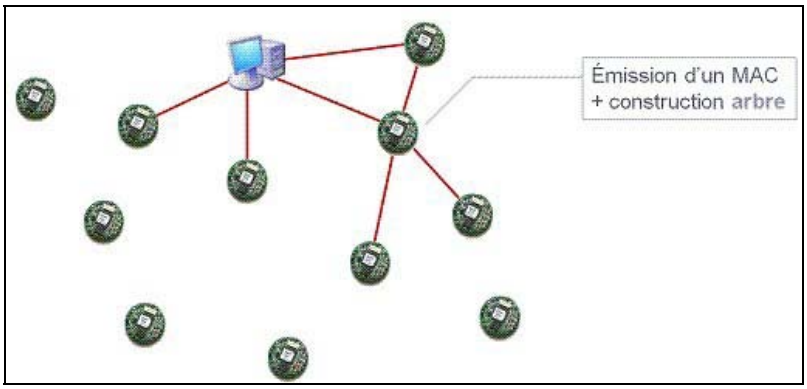

Figure 27 : Construction de l'arbre[20]

7.3.3 Route feedback

Après l'émission de la requête, le nœud attend un certain temps pour envoyer un feedback à la SB. Cette période lui permet de récolter les informations sur son voisinage, qui vont permettre à la SB d'établir une vue globale du réseau d'une manière sécurisée. Un message feedback contient les informations suivantes:

type|ows|path_info|nbr_info|MACRupstream|MACFx

Le champ "path_info" contient la liste des nœuds entre le nœud x et la SB, l'identification de x et son MACR:

IDx|size|path|MACRx

Le champ "nbr_info" contient la liste des identificateurs des voisins ainsi que leurs MACR :

size|IDa|MACRa|IDb|MACRb| . . .

Le champ MACFx est calculé comme suit :

MACFx=MAC(kx, path_info|nbr_info|ows|type)

Pour renforcer la sécurité, le routage du message s'effectue grâce au champ MACRupstream. Ainsi, lorsqu'un nœud reçoit un message feedback, il compare la valeur du MACRupstream reçue avec sa valeur liée au ows reçu, et relaie éventuellement le message.

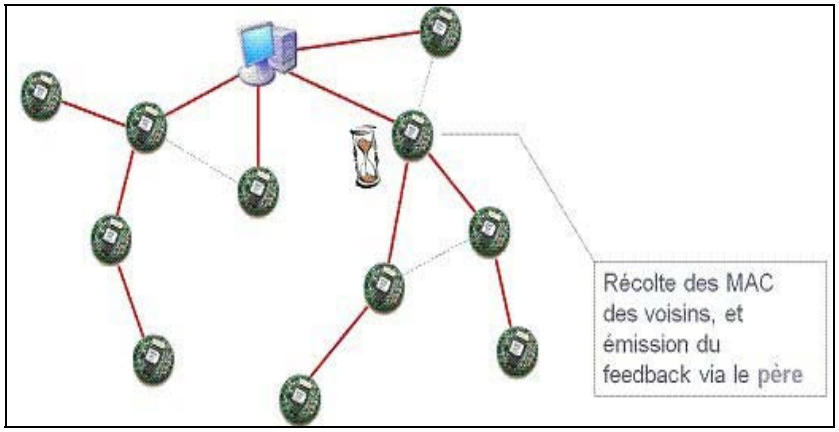

Figure 28 : Route feedback[20]

7.3.4 Construction des tables de routage

Après l'émission de la requête, la SB attend une certaine période avant d'entamer la construction des tables de routage. Pendant cette durée, elle traite les messages de feedback reçus afin de dresser le graphe du réseau. Pour vérifier l'information du feedback d'un nœud x, la SB recalcule le MACFx. Elle doit aussi comparer les valeurs des MACR de chaque voisin de i avec la valeur MACRi reçue dans le feedback du nœud i. Ainsi, les informations falsifiées sont rejetées, et un graphe correct peut être établi.

La SB peut ensuite rechercher les chemins indépendants et construire en conséquence les tables de relais pour chaque nœud. Les chemins indépendants sont choisis de telle sorte qu'ils regroupent le moins de nœuds possible en commun. L'algorithme utilisé emploie une méthode simple qui cherche d'abord le chemin le plus court entre chaque couple de communiquant.

Par la suite, l'algorithme essaye de trouver un autre chemin dans le sous-graphe ne contenant pas les nœuds du premier chemin, leurs voisins et les voisins des voisins. Si cela est impossible, le procédé est réitéré en rajoutant l'ensemble des voisins des voisins, puis en cas d'échec on rajoute l'ensemble des voisins. Pour chaque capteur, la SB calcule une table de relais qui contient une entrée pour chaque chemin passant par le capteur. Cette table est encapsulée dans le message suivant:

type|ows|size|routingTable|MAC

où le MAC est calculé comme suit :

MAC=MAC(kx,type|ows|size|table)

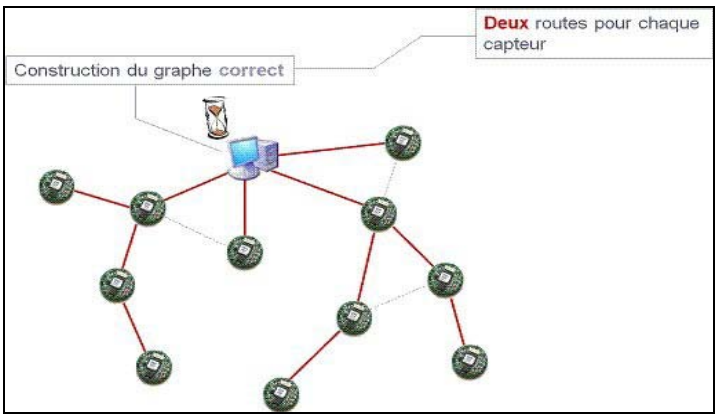

Figure 28 : Construction et distribution des tables de routage[20]

7.4 SecRoute

Le protocole SecRoute est un protocole de routage hiérarchique sécurisé. Le réseau est organisé en clusters ayant chacun un chef. Le nœud collecteur est supposé connaître cette organisation du réseau, et doit maintenir localement une table contenant une clé secrète de chaque capteur. Cette clé est supposée pré-chargée dans chaque nœud. De plus, chaque cluster doit posséder une clé permettant de sécuriser les échanges intra-cluster. Cette clé doit être connue par le cluster-head et tous les nœuds du groupe. Le protocole SecRoute ne spécifie pas

l'algorithme de construction de clusters, et suppose que les clusters ainsi que leurs clés sont établis par un autre protocole, comme LEAP.

7.4.1 Propriétés du SecRoute

Le protocole possède les propriétés suivantes:

- Les paquets de routage ne sont pas volumineux, car ils ne contiennent qu'une information partielle sur le chemin parcouru.
- Le protocole utilise une architecture à deux niveaux, dans laquelle les chefs agrègent les données des membres puis les transmettent au nœud collecteur.
- Le protocole emploie seulement des méthodes de chiffrement symétrique.
- Pour des raisons de sécurité, le protocole remplace les unicasts par des broadcasts locaux ciblés. En effet, en évitant les unicasts, un message émis est reçu par tous les voisins. Donc, cela permet de vérifier, lors du relais, l'intégrité du message émis par le prochain saut.
- Chaque capteur stocke une table de routage ayant le format suivant :

Source	*Pre*	*Next*
ID_{Soure}	ID_{pre_2}, ID_{pre_1}	D_{next_1}, D_{next_2}
⋮	⋮	⋮

Figure 29 : Format de la table de routage dans SecRoute

La table est organisée suivant l'adresse des sources. Le champ Pre (respectivement Next) indique les deux prochains sauts vers la SB (respectivement source) sur le chemin entre la source et la SB.

7.4.2 Découverte des chemins

Le nœud source initie une découverte des chemins en émettant vers ses voisins directs un paquet de requête RREQ contenant les informations suivantes:

IDsource|IDsink|IDRREQ|Nsource MAC(Ksource, IDsource|IDsink|IDRREQ|Nsource) avec :

- IDsource, IDsink, IDRREQ: les identificateurs de la source, du collecteur et de la requête.
- Nsource : un nonce généré par la source.

Lorsqu'un nœud reçoit une requête, elle n'est acceptée qu'avec l'unicité de son identificateur IDREQQ. Il met à jour par la suite sa table de routage en utilisant l'information des deux sauts précédents vers la source. Avant de relayer la requête, le nœud remplace les valeurs de IDpre et IDthis par, respectivement, IDthis et son identificateur :[30]

IDthis|IDpre|IDsource|IDsink|IDRREQ|NsourceMAC(Ksource,IDsource|IDsink|IDRREQ| Nsource)

7.4.3 Relais de la réponse

Lorsque le collecteur reçoit la première requête, il vérifie le MAC construit par la source en utilisant la clé relative à son identificateur IDsource, sauvegardée dans la table locale. Si le MAC est correct, le collecteur met à jour sa table de routage en utilisant les champs IDpre et IDthis. Il génère ensuite une réponse RREP ayant le format suivant :

IDpre|IDthis|IDnext|IDsink|IDRREQ|NsinkAC(Ksource,IDsource|IDsink|IDRREQ| Nsink)

La requête est émise en broadast local, ciblé à l'aide du champs IDnext. Lorsque le capteur voisin ayant l'identificateur IDnext reçoit cette réponse, il met à jour sa table de routage en conséquence, puis remplace les champs IDpre et IDthis par IDthis et son identificateur. [20]

Il doit aussi modifier le champ IDnext par l'identificateur connus lors de la phase de découverte de chemins. Si le nœud ne reçoit pas la réponse émise par IDnext après un certain temps, il ignore toutes les requêtes émises pendant la prochaine phase de découverte. Le nœud de relais doit aussi vérifier que la répon se émise par le prochain saut est valide, en s'assurant qu'elle est bien destinée au nœud à deux sauts contenu dans le chemin vers la source (i.e. le champ IDpre2 de la table de routage).

Lorsque la source reçoit la première réponse, elle vérifie le MAC généré par le collecteur et met à jour sa table de routage en ajoutant IDthis et IDpre comme prochains sauts vers le collecteur.[35][29]

7.4.4 Relais des données

Le relais des données est effectué en deux étapes. Les nœuds membres émettent leurs données vers le chef du groupe, qui va par la suite envoyer le résumé vers le collecteur.

L'établissement du chemin entre le chef et le collecteur s'effectue grâce au procédé décrit précédemment, i.e. : le chef représente la source de son groupe. Pour les communications intra-groupe, le protocole utilise la clé de cluster CK établie lors de sa création. Chaque donnée D d'un capteur est émise vers le clusterhead ID, encapsulée comme suit :[33]

ID\|{D}CK\|MAC(CK, ID\|{D}CK)

Le chef agrège les données des membres après vérification des MAC de chaque paquet, puis émet le résumé vers le collecteur. Pour cela, le chef opère comme étant la source de la donnée. Si aucune route vers le collecteur n'est connue, le procédé de découverte de chemins doit être effectué. En utilisant le chemin construit, le chef émet la donnée dans le paquet suivant :

IDthis\|IDnext\|IDsource\|IDsink\|QID\|{D}Ksource MA (Ksource,IDsource\|IDsink\|QID\|{D} Ksource)

Un nœud intermédiaire avec le même identificateur que IDnext relaie le paquet en remplaçant IDthis et IDnext. Si un nœud de relais ne reçoit pas le paquet émis par le prochain saut, une maintenance de la route doit être effectuée. Pour cela, il doit émettre vers la source un message d'erreur permettant d'enclencher à nouveau la procédure de découverte de routes. De plus, le prochain saut est ajouté à la liste noire (black list).

Cette liste contient les identificateurs dont le nœud doit ignorer leurs paquets de réponse. Lorsque le paquet atteint le collecteur, il vérifie le MAC en utilisant la clé de la source et peut donc décrypter la donnée et l'utiliser.

7.5 Sécurité de l'agrégation dans les RCSF

7.5.1 Attaques sur l'agrégation de données dans les RCSF

L'agrégation de données est nécessaire dans les RCSF pour minimiser les transmissions redondantes et donc économiser de l'énergie. Pour réaliser une opération d'agrégation, un nœud intermédiaire doit avoir accès aux données transmises par ses paires pour calculer l'information utile en utilisant une fonction d'agrégation comme : la moyenne, le maximul, le minimum etc.

Un nœud malicieux peut alors attaquer ce schéma en injectant de fausses données dans le réseau ou en falsifiant le résultat d'une opération d'agrégation. Dans ce cas, le nœud malicieux réussira à falsifier l'information captée dans tout une zone.[20]

La figure suivante montre le risque qui peut être encouru par une mauvaise :

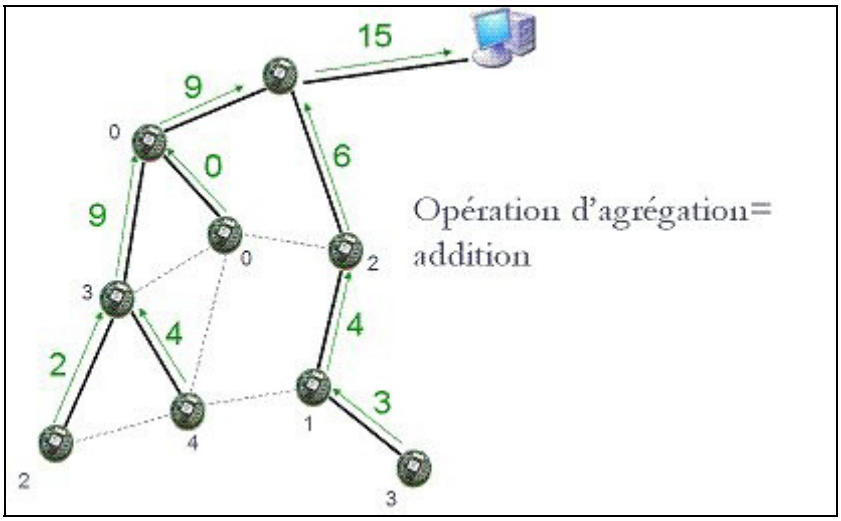

Figure 30 : Fonctionnement correcte de l'agrégation[20]

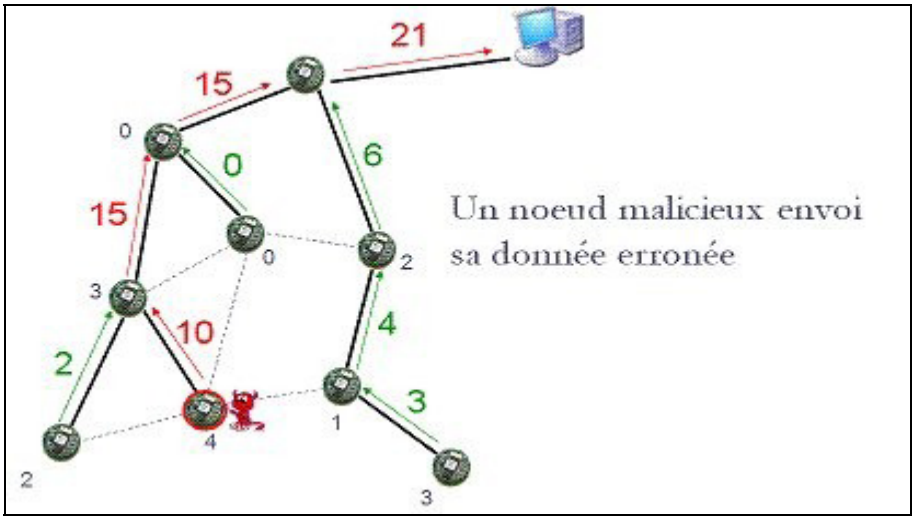

Figure 31 : Un malicieux injecte une fausse donnée[20]

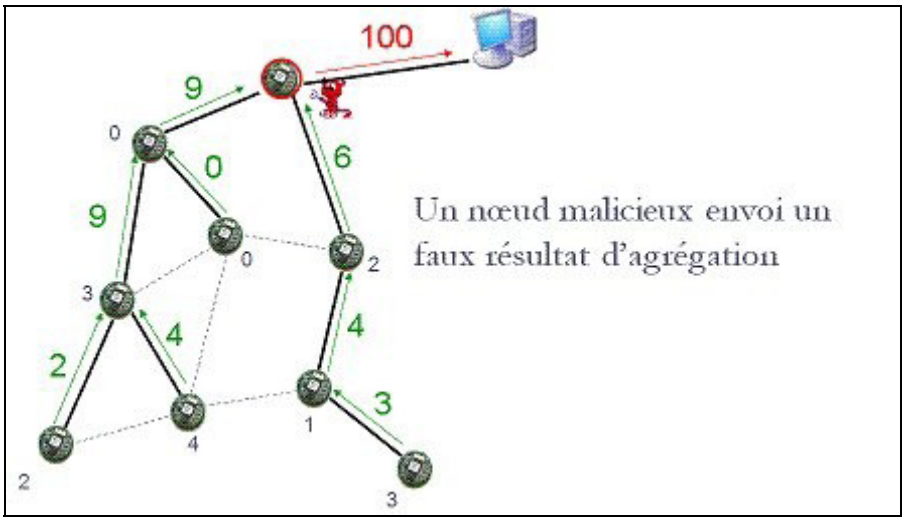

Figure 32 : Un malicieux falsifie le résultat d'une agrégation[20]

Toute la difficulté est comment permettre aux nœuds de relais d'avoir accès aux résultats intermédiaires pour calculer l'information utile par agrégation, et rendre cette opération exempt de risque de falsification, suppression ou modification ?

Il existe deux grandes catégories de solutions selon le mécanisme cryptographique utilisé :

- ✓ Solutions basées sur le cryptage de bout en bout : dans cette catégorie on utilise des mécanismes cryptographiques qui sécurisent l'information captée de bout en bout tout en permettant aux nœuds intermédiaires de réaliser les opérations d'agrégation. Dans cette catégorie, la vérification de l'information ne se fait généralement qu'au niveau du collecteur, ce qui engendre une forte contamination de la fausse information.
- ✓ Solutions basées sur le cryptage de proche en proche : dans ce cas, la véracité de la l'information est vérifiée de proche en proche et son rejet peut se faire à n'importe quel niveau de l'arbre couvrant le RCSF. [27]

La figure suivante résume cette classification :

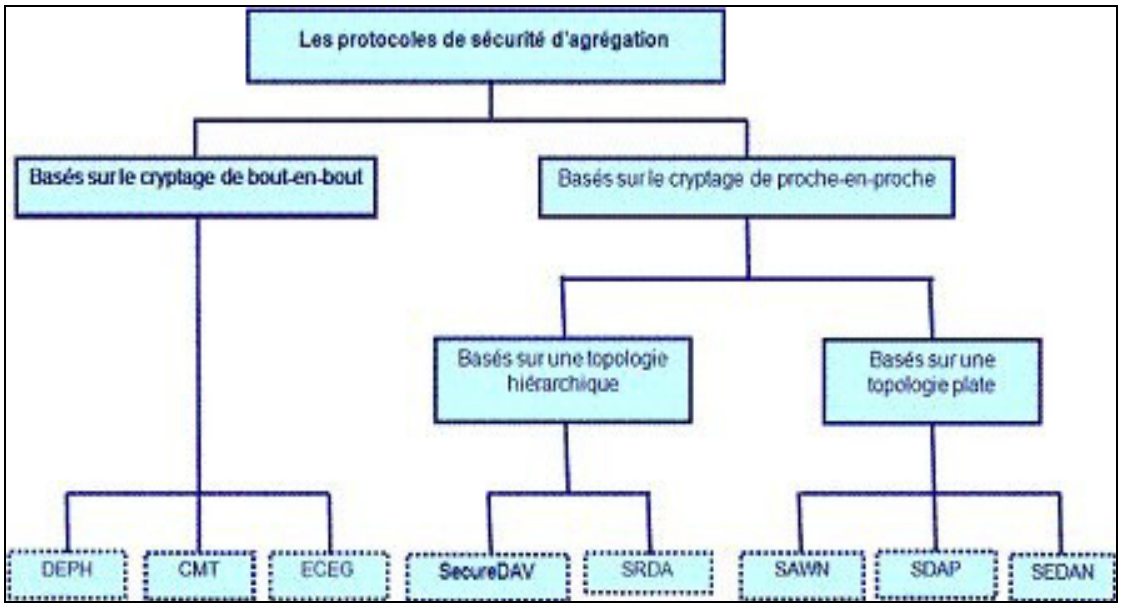

Figure 33 : Classification des solutions d'agrégation sécurisée[33]

7.5.2 SAWN (Secure Aggregation for Wireless Networks)

SAWN suppose que deux nœuds consécutifs ne peuvent pas être compromis simultanément. Il se base sur la vérification à deux sauts : un nœud vérifie si l'agrégation des données de ses petits fils, réalisée par son fils, est correcte. La vérification de l'agrégation se fait d'une manière différée dans le temps en utilisant le protocole µTESLA pour l'authentification des clés utilisées dans l'authentification des données et de leurs agrégations.

Dans ce qui suit nous supposons que les nœuds ont le moyen de vérifier l'authenticité des clés partagées entre les nœuds et la SB, lorsque cette dernière révèle les clés pour la vérification.[35][20]

Chaque nœud feuille transmet sa lecture à son père. Les messages incluent la lecture des données du nœud, son id, ainsi qu'un MAC calculé grâce à la clé KAi. Cette dernière est partagée entre le nœud A et la station de base, mais n'est pas encore connue par les autres capteurs. Le nœud père stocke le message ainsi que son MAC jusqu'à la révélation de clé KAi par la station de base. A cet instant, il vérifiera le MAC et envoi une alarme en cas de différence. L'agrégation des lectures est exécutée dans chaque étape intermédiaire. Les nœuds attendent pendant un temps indiqué pour recevoir des messages de leurs fils et retransmettent ensuite les messages et les MACs qu'ils reçoivent directement de leurs fils immédiats.[31]

Les nœuds agrègent les données qu'ils reçoivent de leurs petits-fils (via leurs fils) et transmettent le MAC de la valeur d'agrégation. Après l'arrivée de tous les messages à la

station de base, cette dernière révèle les clés temporaires des nœuds. Une fois que la clé (KAi) est révélée, les nœuds passent à la clé temporaire suivante (KAi+1).

Considérons l'arbre d'agrégation illustré par la figure suivante. L'exemple illustre le i-ème tour où l'en utilise les clés Kxi pour authentifier les messages transmis:

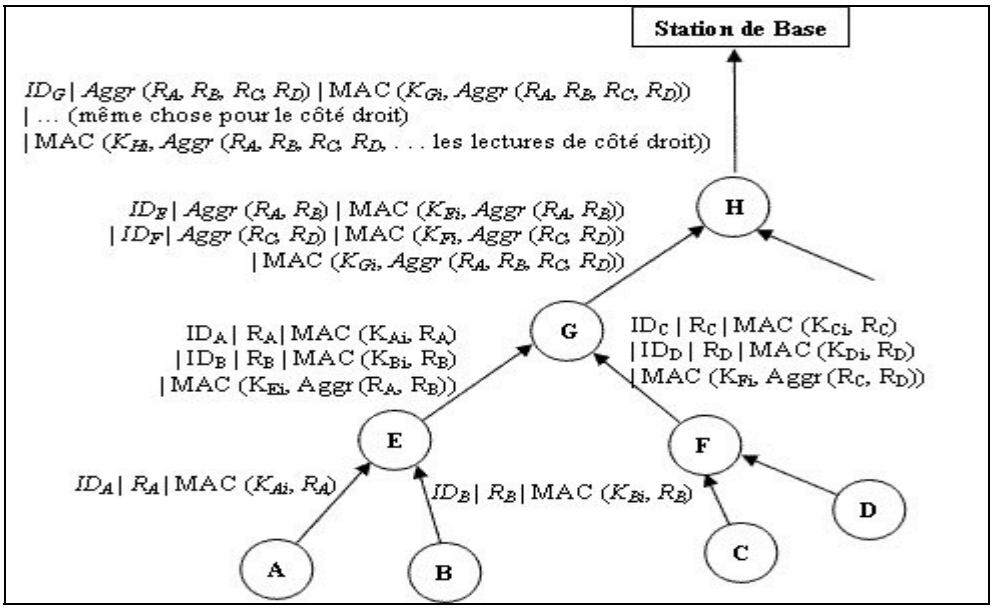

Figure 34 : Exemple d'arbre d'agrégation sécurisé avec SAWN[21]

- **Transmission de données**

Les nœuds A, B, C et D envoient des données à la station de base via l'arbre d'agrégation construit avec un protocole de routage.[28]

- Les nœuds feuilles envoient des données à leur père. Les messages incluent des MACs calculés avec la clé d'authentification courante:

A==>E : RA | IDA | MAC(KAi,RA)

Chaque clé est utilisée pour authentifier un seul message ce qui empêchera l'attaque de rejeu.

- Les nœuds intermédiaires reçoivent les messages de leurs fils. Le nœud père ne peut pas encore vérifier le MAC car la clé du fils ne sera révélée que pendant la phase de vérification. Pour le moment, le père stocke le message et le MAC. Le nœud intermédiaire attend les paquets des fils, et envoie ensuite un message à son père contenant les lectures des fils, leurs MACs, ainsi que le MAC calculé sur la valeur d'agrégation:

E==>G : RA | IDA | MAC(KAi,RA) | RB | IDB | MAC(KBi,RB)|MAC(KEi, Aggr(RA,RB))

Il n'y a aucun besoin de transmettre la valeur d'agrégation calculée, puisque G peut calculer Aggr(RA,RB) depuis les valeurs RA et RB. Ce n'est pas aussi nécessaire de transmettre l'IDE à G, parce que G connaît la topologie du réseau donc peut déterminer le nœud qui envoie le message.

- Le nœud G reçoit les messages des nœuds E et F. Pour chacun d'eux, G calcule les valeurs de l'agrégation des lectures de ses petits-fils c'est-à-dire (A, B, C et D). Il transmet alors les valeurs agrégées de ses petits-fils, l'ID de ses fils et leurs valeurs de MAC. G calcule aussi et transmet le MAC de la valeur d'agrégation suivante :

Aggr(RA,RB,RC,RD)=Aggr(Aggr(RA,RB),Aggr(RC,RD)).

Puisque la fonction de l'agrégation est connue à tous les nœuds, le MAC calculé par E authentifiera la valeur calculée par G. Les lectures des capteurs et les valeurs de MAC reçues à partir de E et F sont stockées pour vérification postérieure.[33]

G==>H : IDE | Aggr(RA,RB) | MAC(KEi, Aggr(RA,RB)) | IDF | Aggr(RC,RD) | MAC(KFi,Aggr(RC,RD)) | MAC(KGi,Aggr(RA,RB,RC,RD))

De la même façon, le nœud H reçoit des messages de G et d'une autre branche, et transmet à son tour le message agrégé à la station de base. Noter que la longueur du message n'augmente pas si le réseau était plus profond.

- La station de base reçoit le message de H. Elle peut calculer la valeur de l'agrégation finale, Aggr(RA, RB, RC, RD, ...) en utilisant Aggr(RA,RB,RC,RD) et les autres valeurs de ses nœuds fils.

- **Validation des données**

Le but du protocole SAWN est d'authentifier toutes les lectures qui ont participé à la valeur d'agrégation, sans pour autant recevoir toutes ces lectures. Pour valider les données (lecture des nœuds et valeurs d'agrégation), la station de base révèle la clé courante Kxi qu'elle partage avec chaque nœud x du réseau, en émettant un seul message contenant toutes ces clés. Ce message de révélation des clés est authentifié par un MAC en utilisant une clé authentifiée grâce à µTESLA.

Si un nœud détecte un message erroné dans l'étape de validation de données, il envoie un message d'alarme. Une alarme est émise par un parent quand il détecte que le MAC d'agrégation d'un fils est contradictoire avec les données des petits-fils, ou bien quand les MAC des données eux-mêmes sont erronés.[31][20]

7.5.3 Protocoles basés sur le cryptage de bout en bout

Les protococles de cette catégorie utilisent une clé partagée entre chaque nœud et le nœud collecteur pour garantir l'intégrité des données transmises dans le réseau.

Comme les contenus des données sont cryptés, les nœuds utilisent un type de cryptographie particulier appelé "Privacy Homomorphism (PH) pour pouvoir exécuter l'agrégation.

Un algorithme est PH si et seulement si en ayant E(x) et E(y) on peut calculer E(x ¤y) sans décrypter x et y. Ainsi il vérifie la propriété suivante :

EK1(x1) ¤ EK2(x2)=EK1+K2(x1 ¤ x2), où Ki sont les clés et xi sont les données.

Le point unique de vérification dans ce type de protocoles est le nœud collecteur.

Ce dernier ayant toutes les clés utilisées pour crypter les données dans le réseau.

Castelluccia, Myldetun et Tsudik ont proposé le protocole CMT basé sur l'hypothèse que chaque nœud utilise une clé symétrique partagée entre ce nœud et le nœud collecteur. L'idée de ce protocole est que chaque nœud fait l'addition modulaire entre sa clé stockée et sa sonnée. Pendant la phase d'acheminement des données, l'agrégation se fait sur ces données qui sont déjà cryptées. L'algorithme suivant montre les différentes étapes de ce protocole :[34][20]

L'algorithme de CMT

Paramètre :

Sélection d'un grand nombre entier M.

Cryptage :

Le message $m \in [0, M-1]$.

Aléatoirement générer une clé $k \in [0, M-1]$.

$C = (m + k) mod M$.

Décryptage :

$m = (c - k) mod M$.

Agrégation :

$c_{12} = (c_1 + c_2) mod M$.

Figure 35 : Algorithme CMT[21]

La taille du paquet dans ce protocole dépend de la taille de M, et une seule addition modulaire suffit pour l'agrégation et le cryptage. Ainsi, ce protocole ne consomme pas beaucoup d'énergie.

- **Protocole Elliptic Curve ElGamel**

Ce protocole utilise un algorithme cryptographique à courbe élliptique ElGamel (ECEG) qui est une approche asymétrique qui ne consomme pas autant d'énergie que les systèmes asymétriques classiques comme RSA. L'algorithme suivant illustre l'agrégation et la vérification dans ce protocole :[34]

L'algorithme de ECEG

Paramètre :

Une clé privé x.

Une clé publique(G, H), G et H des points dans **ECEG**, $H = xG$.

Cryptage :

$C = [c_1, c_2] = [kG, kH + mG]$ = un point dans **ECEG**.

Décryptage :

$mG = (kH + mG) - x(kG)$.

Agrégation :

$C_{12} = C_1 + C_2 = [(c_{11} + c_{21}), (c_{12} + c_{22})]$.

Figure 36 : Algorithme ECEG [21]

8 Conclusion

Nous avons abordé dans ce chapitre l'aspect de sécurité dans les RCSF et les contraintes de ces derniers qui rendent impossible l'application des méthodes classiques de sécurité dans de tels réseaux. Nous avons aussi introduit les différents protocoles et solutions de gestion de clés proposés pour les RCSF. Nous avons vu que les protocoles basés sur la méthode de pré-distribution sont les plus appropriés aux RCSF, pour leur faible coût.

L'inadaptation de la cryptographie asymétrique _a conduit les recherches dans la gestion de clés vers la cryptographie symétrique et ainsi aux méthodes de pré-distribution de clés. Nous croyons que cette situation peut changer dans l'avenir. Des méthodes de cryptographie asymétriques _a faible coût comme la ECC ont un avenir prometteur pour sécuriser les RCSF et méritent des études plus approfondies.

Chapitre 3: APPROCHE DE SÉCURITÉ PROPOSÉE

1. Introduction

Dans cette section nous avons proposé un nouveau mécanisme de sécurité dédié aux RCSF. Notre objectif principal est de sécuriser le processus de transfert des données vers la station de base. Le protocole proposé protège les données transférées contre les attaques des nœuds intrus en utilisant un mécanisme de sécurité basé sur l'utilisation de message de contrôle MAC (Message Authentification Code) pour l'authentification.

2. Approche De Sécurité Proposée

Le principe de base de notre protocole est l'élaboration d'un système de détection d'intrus adapté pour les réseaux de capteurs sans fil. Dans un système de détection d'intrus on peut citer trois approches :

- **L'approche distribuée :** dans cette approche les capteurs sont chargés d'effectuer toutes les vérifications de sécurité et de détecter le comportement malicieux sans avoir coopéré avec ses voisins ou bien la station de base.
- **L'approche centralisée :** la station de base est le seul responsable de toutes les vérifications de sécurité, elle détecte les intrus et fournit des chemins sécurisés.
- **L'approche hybride :** dans cette approche la station de base et les capteurs travaillent ensembles pour garantir la sécurité du réseau. Tous les nœuds capteurs envoient des informations concernant le réseau à la station de base, et cette dernière traite ces informations afin de prendre des décisions concernant la sécurité du réseau.

Notre approche proposée est classée dans cette dernière (Approche hybride).

2.1 Principe de base du protocole de sécurité proposée

La station de base maintien une table de confiance dans laquelle chaque nœud dans le réseau possède un certain degré de confiance. Le degré de confiance est en fonction du nombre d'apparitions du nœud dans un chemin de routage suspect. Ce dernier est un chemin qui contient un ou plusieurs nœuds intrus.

Lors de la phase de communication la station de base maintient à jour sa table de confiance, elle utilise pour cela un message de contrôle basé sur le mécanisme MAC (Message Authentification Code). Rappelons qu'un MAC [8] ou code d'authentification de message peut être vu comme une signature bipartie en cryptographie symétrique : il s'agit de garantir d'une part l'intégrité du message envoyé et d'autre part la vérification de l'identité de l'émetteur pour la personne possédant la clé symétrique partagée.

Un MAC est donc un algorithme qui prend en entrée un message *m*, une clé *K* et qui produit un condensé, un tag :

$$tag = MACK(m).$$

L'algorithme de vérification pour le receveur consiste à vérifier si pour le message reçu *m0* on a bien $tag = MACK(m0)$, i.e. si le message reçu n'a pas été modifié en cours de route. Dans le cas d'un réseau de capteurs, il s'agit de vérifier les propriétés garanties par un MAC de bout en bout : le MAC doit être vérifié entre la station de base et chacun des nœuds envoyant un message.

Chaque nœud envoi dans des périodes aléatoires un MAC, lorsque les deux massages sont arrivés à la station de base, le contenu des deux est comparé après le décryptage du premier, si le contenu des deux est identique donc le chemin traversé est sécurisé, dans le cas contraire le chemin est signalé comme non sécurisé.

Dans le deuxième cas la station de base déclenche un processus dont l'objectif est de détecter à quel niveau le message est modifié ou falsifié.

Pour ce faire la station de base vérifie tous les chemins non sécurisés et recherche un nœud en intersection dans deux ou plusieurs chemins non sécurisés, s'il existe donc il se déclare dans une liste noire et son degré de confiance se dégrade d'une valeur égale à leur apparition, les nœuds possédant les plus faibles valeurs de confiance seront considérés comme des nœuds intrus et les chemins de routage intégrant ces nœuds (nœuds intrus) sont reconfigurés en les écartant.

Le choix d'envoi de messages de contrôle dans des périodes aléatoires est justifié par le fait que si ces derniers sont envoyés en même temps et dans des périodes fixées d'avance cela va entraîner la congestion et la saturation du réseau.

De plus si l'ensemble des nœuds intrus coopèrent entre eux, ces derniers peuvent détecter le processus d'envoi de messages de contrôle et donc d'éviter d'être détectés.

3. Les grandes étapes de notre approche

Le mécanisme de sécurité proposé est composé de plusieurs algorithmes où chaque un est dédié à une application bien spécifique.

3.1 Création des tableaux TN et TC

Dans cet algorithme les tables des nœuds appartenant au chemin sécurisé ou non sécurisé sont crées

```
DEBUT
    SB : station de base.
    TN [0 .. x] : Tableau des nœuds de chemin non sécurisé.
    TC [0 .. x] : Tableau des nœuds de chemin sécurisé.
    i : identificateur des nœuds
    MessCR_Ni: Message chiffrée du nœud Ni
    MessC_Ni: message non chiffrée.
    CH[Ni]: le chemin traverse par le message envoyée par Ni
    Si SB reçoit tous les message Alors
        i = 1, j=0, k=0
        While i<= x do
            Si MessCR_Ni ≠ MessC_Ni Alors
                TN [j] = CH[Ni]
                j= j+| CH[Ni] |
            Sinon
```

```
                TC [j] = CH[Ni]
                k =k+|CH[Ni]| {| | = langueur }
            Fin si
            i=i+1
        Fin While
    Fin si
Fin
```

Figure 37 : Algorithme de création des tables TC et TN

La figure 38 représente l'algorithme responsable du Calcul du nombre d'apparition des nœuds dans la table TN. (Même algorithme pour TC)

```
DEBUT
    App [0..N]: tableau d'apparition des nœuds dans TN respectivement TC
    Y : langueur du tableau TN
    App [0..N] = 0 ,   h=0,  m=0 ;
      While h <= y  Do
        j=0 ;
        While j <= y  Do
            Si TN[h] = TN[j]   Alors
                App[m] = App[m] + 1 ;
                j=j+1;
            Sinon
                j=j+1;
            Fin Si
```

```
        Fin While
            h =h+1;
    Fin While
Fin
```

Figure 38 : du Calcul du nombre d'apparition des nœuds dans la table TN

La figure 39 représente l'algorithme dédié à la création de la table qui contient la liste noire des nœuds (nœuds suspect)

```
DEBUT
    TBL [0..n]: Tableau des nœuds suspect (Black liste).

    i = 0,  j= 0,  k = 0;
    While i<= x Do
        Si App_Ni TN[i] > App_Nj TC[j]  Alors
            TBL[k]= id Ni
            i =i+1, j      j+1,  k=k+1;
        Sinon
            i = i+1, j=j+1,  k =k+1;
        FinSi
    Fin While
Fin
```

Figure 39 : Algorithme de création du black liste

3.2 Création de la table de confiance

Enfin la table de confiance est mis à jour en utilisant l'algorithme représenté dans la figure 40.

```
DEBUT
        TCON[0..x]  :Tableaux de niveau de confiance.
        i =0 ;
        While i<= x Do
            Si Ni existe dans TBL[] Alors
                TCON[i] =  TCON[i] – AppNiTN[i] ;
                i     =   i+1;
            Sinon
                i=    i+1;
            FinSi
        Fin While
Fin
```

Figure 40 : Algorithme de création de la table de confiance

4. Concept de base du protocole de routage HEEP

L'organisation des nœuds appartenant à la même grappe (Cluster) sous forme d'une chaîne permet d'améliorer et de réguler la dissipation d'énergie, ce qui permet de réduire la charge sur le CH (cluster-head). En effet, les nœuds communiquent uniquement avec leurs proches voisins et non pas directement avec leur CH, ce que économise d'avantage d'énergie et offre une meilleure utilisation de la largeur de bande. L'agrégation des données au niveau de chaque nœud dans la chaîne réduit la quantité de données échangées entre les nœuds et leur CH, ce qui a pour effet de préserver les réserves d'énergie de ces derniers.

La figure 41 montre comment les nœuds seront organisés à l'intérieur des grappes (Clusters), le nœud N_0 transmet ses données à son proche voisin N_1, N_1 quant à lui agrège les données reçues avec les siennes et les transmet à son autre voisin jusqu'à atteindre le CH qui les transmet directement à la BS ou en utilisant l'approche multi sauts pour préserver d'avantage d'énergie. Donc, dans cette nouvelle organisation (grappe à chaînes), tous les nœuds du cluster vont transmettre leurs données colletées à leurs CHs respectifs en se reliant à travers la chaîne, tandis que chaque CH doit recevoir les données collectées des nœuds entêtent de la chaîne.[35]

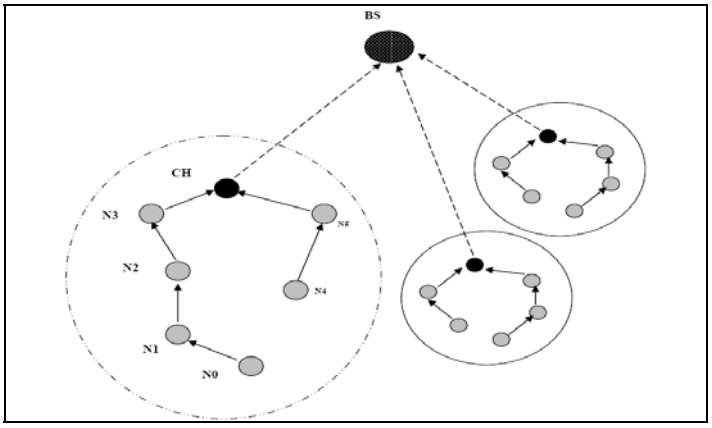

Figure 41 : Organisation des nœuds dans le réseau.

A l'inverse de LEACH, le nombre de nœuds qui communiquent avec le CH est considérablement réduit. Ceci implique une meilleure économie d'énergie et prolonge d'avantage le temps de vie des CHs, car si ces derniers meurent (épuisent leur réserve d'énergie), tous les nœuds du cluster vont perdre leur pouvoir de communication avec la BS et par conséquent le cluster tout entier est considéré comme invalide (ne communique pas avec la BS).

Le protocole HEEP adopté le concept de la rotation aléatoire du rôle de CH proposé par LEACH, qui régule la dissipation d'énergie et évite que les nœuds choisis comme CHs meurent plus rapidement. Cependant, par opposition à LEACH, HEEP réutilisé le concept de PEGASIS il organise les nœuds du cluster sous forme de chaîne, ce qui a pour effet de prévenir que les nœuds les plus éloignés des CHs épuisent leur réserve d'énergie.

En ce qui concerne l'accès au médians (*Media access*), la même méthode proposée dans LEACH a été utilisé, où les CHs établissent un plan de transmission (*TDMA schedule*) qui attribut à chaque nœud le temps exact pendent lequel il doit transmettre ses données collectées.[35]

5. Analyse De Sécurité

Dans cette section nous allons évaluer les performances de notre protocole de sécurité par rapport aux conditions de sécurité suivante :

5.1 Confidentialité de données et authentification des paquets

Afin de garantir la confidentialité et l'authentification des messages nous utilisons une clé secrète de session, une fois un intrus est détecté la station de base reconfigure le réseau avec une nouvelle clé secrète, ce qui ne donne pas aux intrus le temps d'explorer la clé secrète, la méthode sécurisée de la distribution des clés n'est pas prise en charge dans notre approche, donc on suppose qu'il existe un mécanisme sécurisé de distribution de clé secrète pour chaque session.

5.2 Intégrité des données

Dans notre protocole de sécurité on se focalise sur la détection des intrus qui touche l'intégrité des données, on a utilisé pour cela le mécanisme MAC (message authentification code). Pour notre protocole le MAC est utilisé comme un message de contrôle pour détecter les attaquants qui falsifient les données, ou injectent des paquets. Lorsqu'un attaquant transfère un message de contrôle il modifie le contenu de *m0 et MACK*(*m0*) d'une manière aléatoire, une fois les deux messages reçus par la station de base elle détecte qu'il existe un chemin non sécurisé qui automatiquement contient un ou plus d'intrus, dans ce cas la station de base exécute l'algorithme de recherche d'intrus.

5.3 La Localisation

L'objectif principal de notre approche est de détecter les nœuds intrus pour les écarter, une fois un intrus est détecté, leur endroit est localisé par la procédure d'intersection du chaines.

6. Implémentation

Afin d'implémenter notre protocole, nous avons réutilisé le code d'implémentation du protocole HEEP, et effectué certaines modifications à plusieurs niveaux. Ces modifications consistent en l'injection de nouveaux morceaux de code comme par exemple la procédure d'envoi de donnée, procédure de réception de paquets par la station de base, procédure de création de TDMA schedule pour les chaînes de nœuds, ..etc.

6.1 Choix du langage et de l'environnement d'implémentation

Nous avons choisi comme langage d'implémentation pour notre protocole le langage TCL (*Tool Command Language*) [33, 34]. Ce dernier est connu comme étant un langage de commandes *interprété et* extensif. En effet, les programmes écrits en TCL sont des fichiers texte constitués de commandes TCL qui sont traités via un interpréteur TCL au moment de l'exécution. L'avantage d'implémenter notre algorithme en TCL est de pouvoir facilement l'interpréter avec l'interpréteur TCL intégré dans le simulateur NS2 [12], ce qui nous permet de simuler son fonctionnement afin d'évaluer ses performances.

Nous avons choisi comme plate- forme d'implémentation, le simulateur NS2 [12] version NS2.30, sous le système d'exploitation *LUNIX MANDRIVA 2007*. Ce choix est basé sur le fait que NS2 est le leader en terme d'extensibilité puisque on peut facilement ajouter nos propres protocoles, que ce soit au niveau de la couche réseau, mac, application, etc.… De plus, NS2 est bâti selon les idées de conception par objets, de réutilisation du code et de modularité. Il est devenu aujourd'hui un standard de référence dans ce domaine, son utilisation est gratuite et il est exécutable tant sous UNIX que sous Windows. NS2 est construit autour du langage de programmation TCL dont il est une extension. Le noyau de ce dernier est implémenté en commandes C++.

Du point de vue utilisateur, la mise en œuvre de ce simulateur se fait via une étape de programmation qui décrit la topologie du réseau et le comportement de ses composants.

Ensuite, vient l'étape de simulation proprement dite et enfin l'interprétation des résultats. La description de la topologie du réseau peut être facilement effectuée en utilisant des primitives de base comme par exemple *Nodes*, *Links*, *Agents*, et *Applications*, où la primitive *Nodes* représente un nœud dans le réseau, la primitive *Links* représente le support de

communication, la primitive *Agents* est utilisée pour implémenter différents protocoles réseaux, et la primitive *Applications* est chargée de la génération des données ainsi que la réalisation de plusieurs tâches spécifiques.[35]

6.2 Etapes d'implémentation de notre protocole

L'implémentation de notre protocole passe par plusieurs étapes à savoir :

6.2.1 Préparation de l'environnement d'implémentation

La préparation de l'environnement d'implémentation consiste à installer le simulateur de réseau NS2 sous le système d'exploitation LUNIX MANDRIVA 2007. On a choisi la version NS2.30 puisqu'elle prend en considération la topologie des réseaux sans fil. L'installation de NS2.30 sous LUNIX MANDRIVA 2007 s'effectue suivant 3 étapes qui se résument en :

- Copier le package d'installation NS2.30 dans le répertoire USER du système ;
- Taper la commande d'installation './Install' dans le terminal de commandes, dans le répertoire NS2.30 précédemment copié ;
- Modifier les variables d'environnement et cela consiste à ajouter les lignes de code à suivre dans le fichier '.bashrc' ;

```
export
PATH=${PATH}:/home/user/ns-allinone-2.30/bin:/home/user/
ns-allinone-2.30/tcl8.4.13/unix:/home/user/ns-allinone-
2.30/tk8.4.13/unix:/home/user/ns-allinone-2.30/nam-1.12
export
LD_LIBRARY_PATH=/home/user/ns-allinone-2.30/otcl-1.12,
/user/ns-allinone-2.30/lib
export
     TCL_LIBRARY=/user/ns-allinone-2.30/tcl8.4.13/library
```

- Et enfin taper la commande './make' dans le terminal de commandes afin de compiler NS2 et de générer tous les fichiers NS2 nécessaires à son fonctionnement.

- Après avoir installé le simulateur de réseau NS2, il faut installer le protocole de routage HEEP. Pour ce faire, il faut suivre les étapes suivantes :
- Obtenir le package d'installation du protocole HEEP.
- Placer le package d'installation dans le chemin suivant : 'user/ns-allinone-2.30/ns-2.30' ;
- Décompresser le package d'installation en tapant la commande 'gunzip mit.tar.gz' et puis la commande 'tar -xvf mit.tar' dans le terminal de commandes ;
- Ajouter les lignes de code à suivre dans la fiche de compilation NS2 nommée 'Makefile' qui se trouve dans le répertoire ns-allinone-2.30.

- Ajouter la ligne de commande ***'DMIT_uAMPS'*** dans la partie de déclaration ***'DEFINE list'***
- Ajouter la ligne de commande ***'I./mit/rca I./mit/uAMPS'*** dans la partie ***'INCLUDE list'***
- Ajouter les lignes de commande à suivre juste avant la ligne de commande ***'gaf/gaf.o \'***

  ***'mit/rca/energy.o mit/rca/rcagent.o ***
  ***mit/rca/rca-ll.o mit/rca/resource.o ***
  mac/mac-sensor-timers.o mac/mac-sensor.o mit/uAMPS/bsagent.o \'

- Taper la commande 'make clean' dans le terminal de commandes afin d'effacer l'ancien fichier de compilation.
- Et enfin recompiler de nouveau NS2 afin de prendre en considération l'ajout du nouveau protocole de routage HEEP en tapant la commande ***'nohup make 2>error.log >make.log &'*** dans le terminal de commandes.

6.2.2 Implémentation de notre protocole

Après avoir préparé notre environnement d'implémentation, on entame l'étape d'implémentation de notre protocole qui consiste à modifier le code d'implémentation du protocole HEEP afin qu'il corresponde à l'architecture de fonctionnement de notre protocole.

Pour cela, on a ajouté diverses fonctions et procédures telles que : la procédure d'envoi des données, procédure de réception des données, procédure de création de TDMA schedule pour les chaînes de nœuds, ….etc.

En effet le, code d'implémentation proposé est composé de plusieurs fichiers TCL dont chacun joue un rôle bien précis. Ainsi, et pour réutiliser ce dernier, nous avons dû analyser et comprendre le rôle de chacun de ces fichiers. Etant donné qu'un fichier TCL est présenté comme une classe qui contient plusieurs sous classes, le protocole proposé est vu comme un ensemble de classes qui se communiquent pour réaliser une tâche commune.

Le protocole proposé est une application spécifique des protocoles de sécurité au niveau routage de données. Il est implémenté comme sous classe par rapport à la classe Applications du simulateur NS2.

7. Conclusion

Dans ce travail nous avons mis en place un nouveau mécanisme de sécurité dédié aux protocoles de routage pour les RCSF. Ce mécanisme consiste à protéger le processus de transfert de données en se basant sur un algorithme spécial de détection des nœuds intrus. Les simulations conduites avec notre protocole prouvent les performances de ce dernier. En perspective on projette d'essayer notre mécanisme de sécurité sur d'autres protocoles de routage afin de raffiner les performances de notre algorithme.

Chapitre 4 : simulation

1. Introduction

Afin d'évaluer les performances de notre mécanisme de sécurité, nous avons simulé son fonctionnement à l'aide du simulateur réseau NS2. Comme protocole de routage nous avons choisi le protocole HEEP, afin de tester notre mécanisme de sécurité. Le protocole HEEP est un protocole de routage basé sur le clustering à chaînes dans lequel le réseau est reconfiguré sous forme de cellules. Dans chaque cellule (cluster) les chemins de routage de données sont construits sous forme de chaînes de nœuds adjacents afin de réduire les distances de transmission.

2. Présentation du simulateur NS2

NS-2 est un logiciel de simulation open source gratuit (libre) à évènements discrets. Il est développé dans le cadre du projet VINT qui regroupe plusieurs laboratoires de recherche comme AT&T institut de recherche à Berckeley (ACIRI), Xerox PARC et Sun Microsystems. Ce simulateur est développé en C++ et OTCL, est utilise le langage TCL comme langage de création des scénarios de simulation. Supporte les réseaux sans fil et filaires, avec plusieurs protocoles des différentes couches (Physique, MAC, réseaux, transport … etc.).

Par sa possibilité d'extension, NS2 est le simulateur le plus utilisé dans le domaine de recherche pour les tests des nouveaux protocoles proposés. Pour plus d'information sur NS2 ou sur son installation, veuillez consulter [36].

3. Environnement de simulation

Notre modèle d'expérimentation est établi sur 100 nœuds, dispersés aléatoirement sur une surface carrée de 100 m². Nous assumons que tous les nœuds ont une position fixe durant toute la période de simulation. Les paramètres de notre simulation sont résumés dans le tableau suivant :

Paramètres	Valeur
Localisation de la BS	(20, 175)
Le nombre des nœuds	100
Le nombre de cluster	5
Energie initiale	2 J
La taille des paquets	512 Byte
Taille du paquet de contrôle	8 Byte

Table 2. Paramètres de simulation.

La duré de simulation est fixée à 60s, et le nombre de nœuds intrus est égal à 10. On suppose que les intrus modifient tous les paquets qu'ils retransmettent.

4. Résultats de simulation

La simulation conduite évalue les performances de notre mécanisme de sécurité en termes de résistance contre les attaques de type modification et falsification des données.

Le temps de simulation est divisé en plusieurs intervalles de 10 secondes. Durant chaque intervalle de temps nous avons mesuré le nombre de paquets de données et de contrôle envoyés à la station de base, le nombre de paquets de données et de contrôle modifiés, ainsi que le nombre de nœuds intrus détectés.

Les résultats de la simulation sont résumés dans le tableau 3, les figures 39 et 40.

	0s-10s	10s-20s	20s-30s	30s-40s	40s-50s	50s-60s
Nombre de paquets de données envoyés	89	214	274	259	146	74
Nombre de paquets de contrôle envoyés	18	31	10	22	13	6
Nombre de paquets de données modifiés	14	23	7	0	0	0
Nombre de paquets de contrôle modifiés	15	23	5	0	0	0
Nombre d'intrus détectés	3	6	1	0	0	0

Table 3: Résultats de simulation.

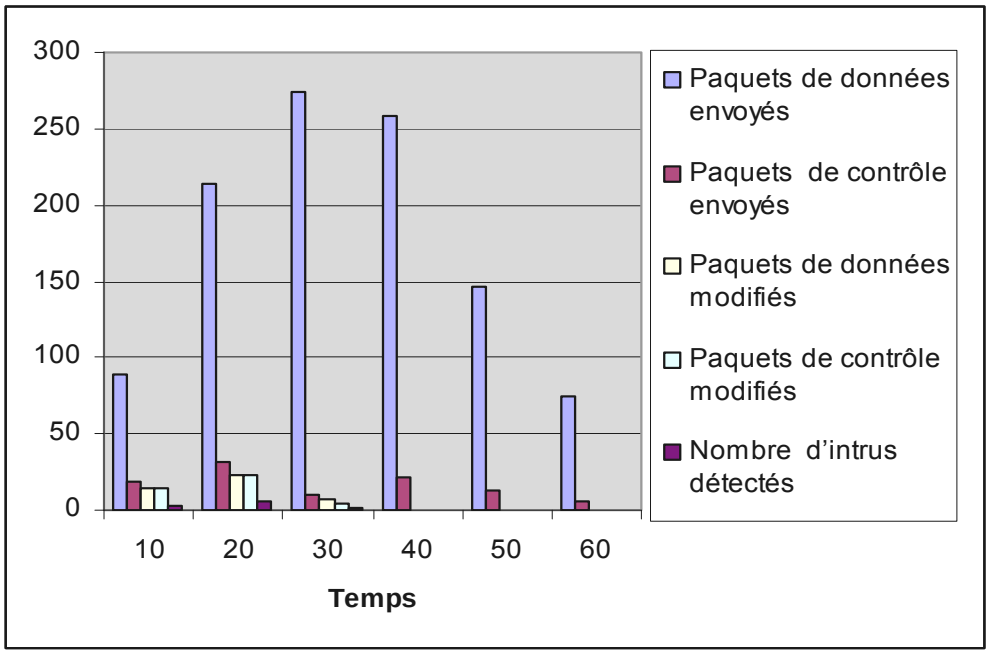

Figure 39 : Résultats de simulation

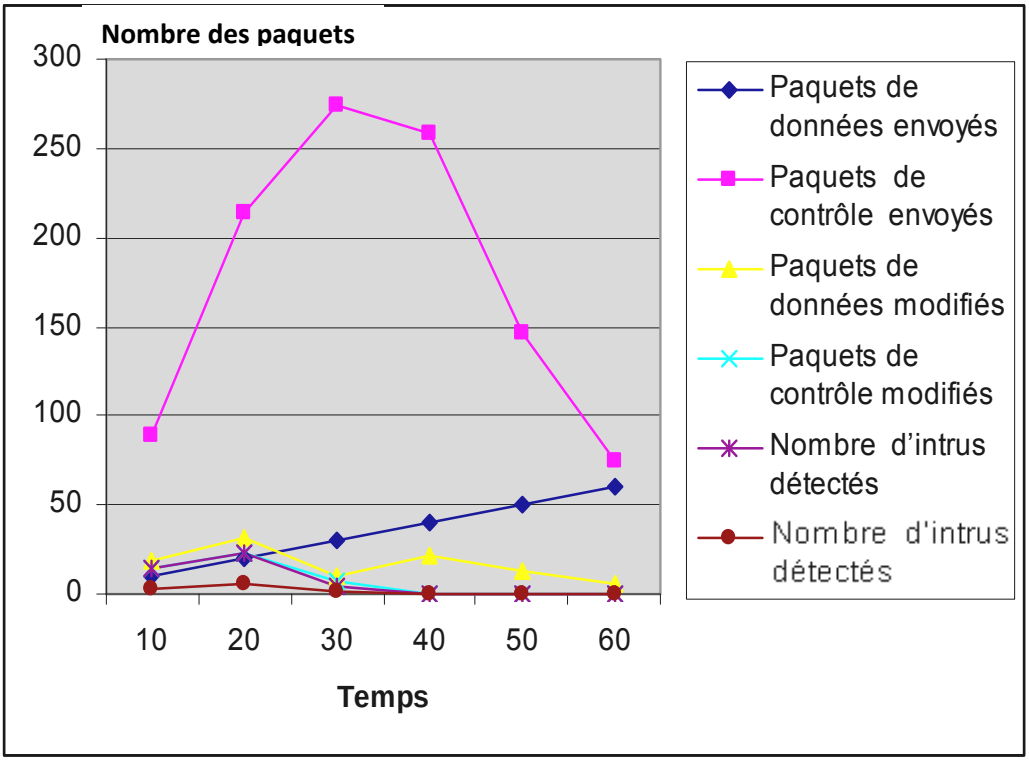

Figure 40 : Résultats de simulation

La comparaison de notre approche avec les protocoles HEPP et LEACH-C on pourcentage de nombre des paquets modifiés donnée les résultats suivants :

	0s-10s	10s-20s	20s-30s	30s-40s	40s-50s	50s-60s
	Pourcentage de nombre des paquets modifiés					
Le Protocole HEEP	10.5	31.66	60.38	45.95	30.88	79.12
Le protocole LEACH-C	5.32	23.45	15.06	69.26	71.07	41.77
Approche proposée	12,46	49.22	19.18	0	0	0

Table 4: pourcentage de nombre des paquets modifiés

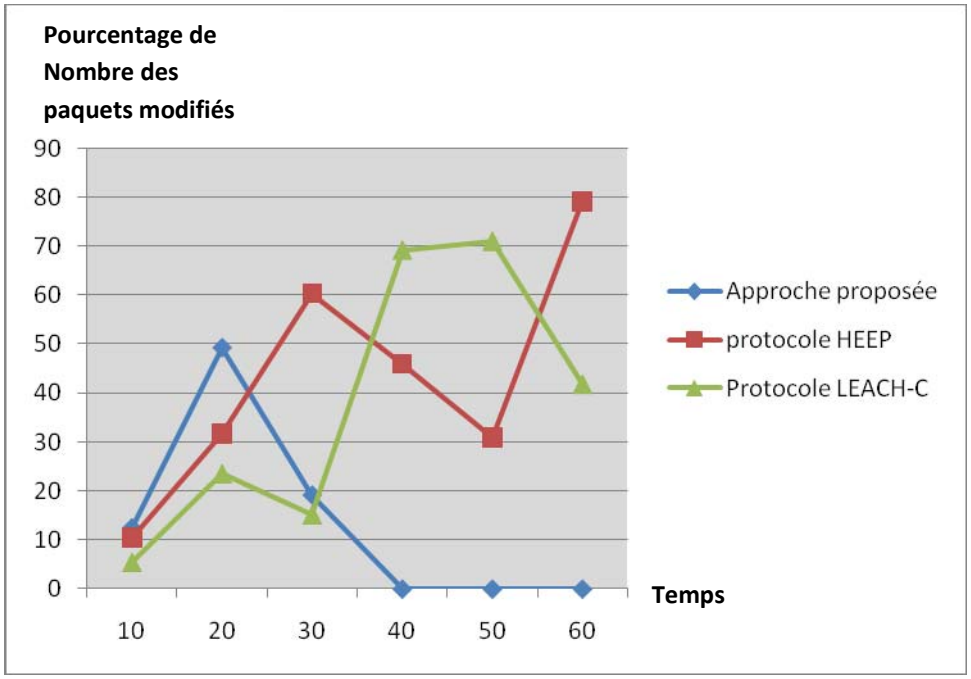

Figure 41 : comparaison entre les protocoles HEEP, LEACH-C et notre approche

Les résultats de simulation démontrée que notre approche de sécurité garantir une communication sécurisée après deux ou trois cycles de reconfiguration du cluster, par contre les protocoles HEEP et LEACH-C présentent un faille de sécurité durant toute la période de simulation.

L'énergie est un facteur très important pour n'importe quelle approche de sécurité proposé dans RCSF, pour cela on a mesurée et comparée la consommation d'énergie de notre approche avec les protocoles HEEP et LEACH-C.

	0s-10s	10s-20s	20s-30s	30s-40s	40s-50s	50s-60s
	nombre des nœuds vifs					
Le Protocole HEEP	100	100	96	92	79	68
Le protocole LEACH-C	100	100	92	84	70	65
Approche proposée	100	98	85	66	59	56

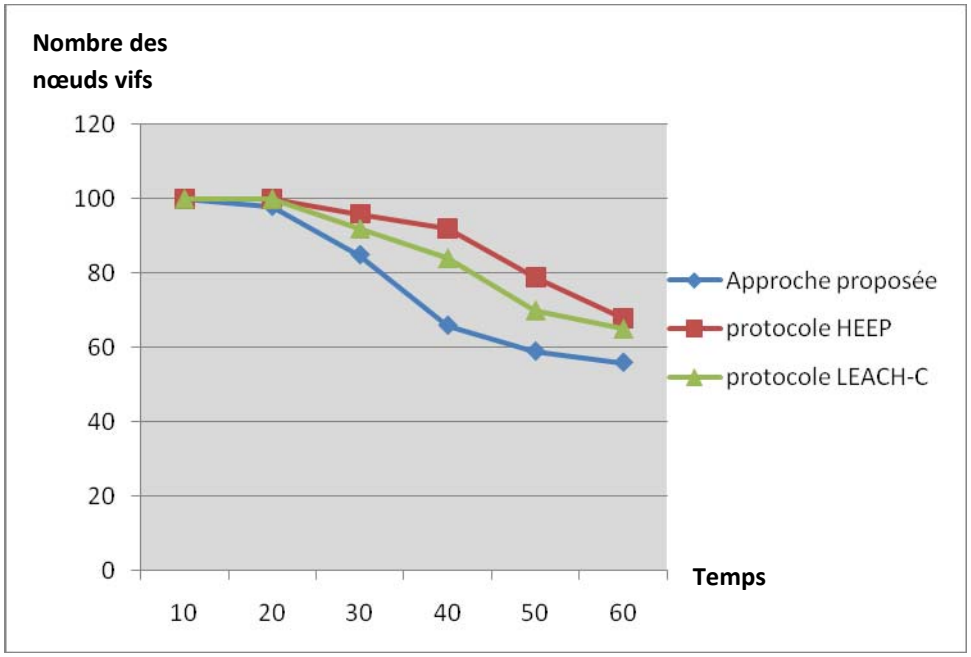

Figure 42 : comparaison de vivacité des nœuds entre les protocoles HEEP, LEACH-C et notre approche

L'énergie consommée à l'envoi des messages de contrôles et à la reconfiguration du cluster, cause la déférence de mortalité des nœuds de notre approche avec les deux autre protocoles HEEP et LEACH-C, mais cette déférence est acceptée pour garantir une communication sécurisée.

5. Conclusions

Les résultats de simulation démontrent la capacité de notre algorithme à sécuriser le processus de transfert de données. La détection des nœuds intrus augmente au fur et à mesure que nouveaux nœuds capteurs participent au transfert de données, le pourcentage de mortalité des neouds est diminué lorsque les intrus sont détectés.

On peut déduire que la fiabilité de notre approche est élevée lorsque le temps de simulation est augmenté.

Conclusion générale

Dans ce mémoire, nous avons mis en avant les caractéristiques essentielles des réseaux de capteurs sans fil, ainsi que les besoins et les défis de la sécurité dans ces derniers. Nous avons étudié aussi quelques schémas de gestion de clés qui permettent d'offrir le service de sécurité de base pour n'importe quel système basé sur la communication. L'ensemble des protocoles de gestion de clés proposés pour les RCSFs, se basent principalement sur la cryptographie à clé symétrique et la méthode de pré-distribution de clés afin d'achever l'établissement de clés entre les entités communicantes dans le réseau.

Nous avons étudié un ensemble de ces protocoles de gestion de clés qui sont classés dans plusieurs catégories selon la topologie du réseau (hiérarchique ou plate) et la façon dont les nœuds voisins partagent des clés communes (probabiliste ou déterministe). D'après les critiques de ces solutions, nous avons constaté que le défi dans la conception des schémas de gestion de clés est de trouver un compromis entre un système efficace et les contraintes caractérisant les RCSFs.

Références

[01] http://www.commentcamarche.net, site de documentation informatique, Septembre 2005.

[02] http://www.francetelecom.com/rd, site de la division R&D de Francetelecom, Septembre 2005.

[03] Pujolle G., « Les réseaux Editions 2005 », éditions Eyrolles, 2005.

[04] http://www.epfl.ch, site de l'école polytechnique fédérale de Lausanne, Septembre 2005.

[05] Zheng J., Myung L., « Will IEEE 802.15.4 make ubiquitous networking a reality ? », The city college of CUNY, IEEE Communications Magazine, juin 2004.

[06] Michel HUBIN. *Traité sur les capteurs et la conception instrumentale.*2000.

http://perso.wanadoo.fr/michel.hubin/capteurs/instru.htm

[07] Franck L. Lewis. Wireless Sensor Networks. In Smart Environments : Technology, Protocols and Applications, ed. Diane Cook and Sajal Das, John Willey, New York, 2004.

[08] David Culler, Deborah Estrin and Mani Srivastava. *Overview of Sensor Networks*. In IEEE Computer, vol. 37, no. 8, pp 41–49, august 2004.

[09] Laetitia Mailhes. Une nouvelle génération de mémoires magnétiques. Dans Les Echos, no. 19611, Innovation, pp. 30, 22 février 2006.

[10] I. Khemapech, I. Duncan and A. Miller. *A survey of wireless sensor networks technology*. In PGNET, Proceedings of the 6th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking & Broadcasting, June 2005.

[11] *A Standard Smart Transducer Interface*, Sensors Expo, Philadelphia, Oct. 2001.

[12] Akyildiz I., Su W., Sankarsubramaniam Y., Cayirci E., «A Survey on Sensor Networks», Georgia Institute of Technology, IEEE Communications Magazine, Aout 2002.

[13] Holger K., Willig A., « A short survey of wireless sensor networks », Technical university Berlin, Telecommunication Networks Group, Octobre 2003.

[14] Culler D., Estrin D., Srivastava M., « Overview of sensor networks », University of California, Berkeley, IEEE Computer Society, Aout 2004.

[15] Fleury E., Chelius G., Mignon T., « minimisation de l'énergie dans les réseaux de capteurs », Laboratoire CITI/INSA de Lyon, 2003.

[16] Séverine Sentilles « Architecture logicielle pour capteurs sans-fil en réseau » Rapport de recherche, Université de Pau et des Pays de l'Adour, juin 2006

[17] Demirkol I., Ersoy C., Alagöz F., « MAC Protocols for Wireless Sensor Networks: a Survey », 2003

[18] Bouabdellah Kechar, « Problématique de la consommation de l'énergie dans les réseaux de capteurs sans fil », Séminaire LIUPPA, Université de Pau et des Pays de l'Adour, 14 Octobre 2007.

[19] Lyes Khelladi, Nadjib Badache « Les réseaux de capteurs: état de l'art », Rapport de recherche, Algérie, Février 2004.

[20] Yacine Challal, « Réseaux de Capteurs Sans Fils », Cours, Systèmes Intelligents pour le Transport, Université de Technologie de Compiègne, France, 17 Novembre 2008.

[21] Noureddine LASLA« La gestion de clés dans les réseaux de capteurs sans-fil » mémoire de magister, Institut National de formation en Informatique (I.N.I) Oued-Smar, Alger.

[22] E. Yoneki and J. Bacon. *A Survey of Wireless Sensor Network Technologies: Research Trends and Middleware's Role.* Technical Report UCAM-CL-TR646, University of Cambridge, 2005.

[23] Jan Blumenthal, Matthias Handy, Frank Reichenbach, Dirk Timmermann. *SeNeTs –Test and Validation Environment for Applications in Large-Scale Wireless Sensor Networks*. In *2nd IEEE International Conference on Industrial Informatics, 2004*.

[24] Yang Yu, Bhaskar Krishnamachari, and Viktor K. Prasanna. *Issues in designing middleware for wireless sensor networks*. In IEEE Network, Vol. 18, No.1, pp. 15-21, 2004.

[25] Patrice KADIONIK «Réseau De Capteurs Sans Fils » projet avance, école national d'électronique informatique et radiocommunication de bourdeaux.

[26] J. P. Walters, Z. Liang, W. Shi, and V. Chaudhary, “Wireless Sensor Network Security: A Survey ”, *Department of Computer Science Wayne State University*.

[27] H. Krawczyk and M. Bellare and R. Canetti, "HMAC : Keyed-Hashing for Message Authentication", RFC 2104, 1997, February.

[28] Hatem Bettahar, Yacine Challal, « Introduction à la sécurité informatique », Supports de cours, Systèmes Intelligents pour le Transport, Université de Technologie de Compiègne, France, 15 Octobre 2008.

[29] www.securiteinfo.com, « Le Grand Livre de SecuriteInfo.com », 19 Février 2004.

[30] Mohamed-El-Amine Chetibi, Souad Djerroud, « Sécurisation des échanges sur les réseaux Wifi », Thèse d’ingéniorat, Institut National de formation en informatique INI, Algérie, Juin 2008.

[31] Emmanuel. Bresson, «Cryptographie: chiffrement par flot», Séminaire de la cryptographie, Page(s): 22-34, Laboratoire de cryptographie, Université de Paris XII, 2001/2002.

[32] D.Baker, H.X.Mel, «La cryptographie décryptée», Livre, Nombre de Pages: 413, Edition Campus Press, 2001.

[33] A.Bachir, A. Ouadjaout, L. Khelladi, M. Bagaa, N. Lasla, Y.Challal, « Information Security in Wireless Sensor Networks», Handbook/Encyclopedia on Ad Hoc and Ubiquitous Computing, edited by: Agrawal Dharma P., and Xie Bin., World Scientific, 2009.

[34] Ghislaine Labouret, « Introduction à la cryptographie », Supports de cours, Cabinet Hervé Schauer Consultants-HSC, 09 Février 2001.

[35] Djallal Boubiche « protocole de routage pour les réseaux de capteur sans fil » mémoire de magister, université de batna, juin 2008.

[36] Nabil Ouazene «Pour une QoS au niveau de la Couche MAC dans les Réseaux sans Fil » mémoire de magister, université de batna, Avril 2009.