MINISTRE DE L'ENSEIGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

Université Mouloud Mammeri de Tizi-Ouzou

Faculté de Génie Electrique et d'Informatique

Département d'Informatique

MEMOIRE DE MASTER II

Spécialité : informatique.

Option : Réseaux, Mobilité et Systèmes Embarqués (RMSE).

Thème :

Sécurité d'agrégation de données dans les réseaux de capteurs sans fil.

Présenté par

Ouziane mohamed.

Devant le jury d'examen :

Président : DAOUI.M

Promotrice : BEKADI.M.

Examinateur : ... …. … ...

Examinateur : … …. … .

Promotion : 2012-2013

A mon père A ma mère.
A ma sœur et mes frères.

A toute ma famille et mes amis

REMERCIEMENTS

Je tiens à remercie on première lieu nous bon dieu le tout puissant, pour m'avoir donnée le courage pour terminer ce mémoire.

Je tiens à remercier ma promotrice **Mme .BELKADI MALIKA**, pour avoir m'assuré la direction de mon mémoire, pour la qualité de son encadrement, ses remarques pertinentes, son soutien, et qui de parts son expérience, m'a permis de finaliser ce travail grâce à son suivi efficace et ses conseils avisées.

Je remercie aussi les membres du jury qui ont accepté d'étudier ce mémoire.

Une pensée particulière est adressée à mes chers parents, ma sœur et mes frères, pour leur soutien inconditionnel tout au long de la période d'études.

Merci à tous.

Abstract
The Wireless Sensor Networks (WSNs) are composed of a huge number of sensor nodes. The large number of nodes may lead to a huge amount of data in the network, causing the network to degrade performance and shorten its lifetime.
the data aggregation techniques of the messages may be a solution to reduce the number of messages exchanged in the network let us send of it only one message of data instead of envoy several. But this solution can have fatal consequences on the security of the network: the integrity of the messages, the authentification ,… In this document, propose to us a new solution for protected the aggregation of data against the attacks flood the aggregation of data. And one describes also the details of the installation of our solution and thereafter one gives to some results of simulation one presence of malicious sensors.

Résumé
Les réseaux de capteurs sans fil sont composés d'un grand nombre de capteurs avec de faibles ressources en énergie, en mémoire et en calcul. Dans le but d'étendre la durée de vie de ces réseaux, l'agrégation des messages s'avère comme une bonne solution pour diminuer le nombre de messages échangés dans le réseau en envoyant un seul message de données au lieu d'envoyer plusieurs. Mais cette solution peut avoir des conséquences néfastes sur la sécurité du réseau: l'intégrité des messages, l'authentification, . . . Dans ce document, nous proposent une nouvelle solution pour sécuriser l'agrégation de données contre les attaques d'agrégation de données. Et on décrit aussi les détails de la mise en place de notre solution et par la suite on donne quelques résultats de simulation en présence de capteurs malicieux.

Chapitre I : Généralité sur les réseaux de capteurs

Chapitre II : L'agrégation de données dans les WSN

Chapitre III : Sécurité et l'agrégation de données

Chapitre IV : Proposition d'une solution pour la sécurité d'agrégation de données

Chapitre V : Implémentation et Simulation

Introduction générale

Un capteur sans fil, fruit des récents progrès de la micro électromécanique et des nouvelles technologies, est un petit dispositif intelligent ayant la capacité de traitement local et de communication sans fil. Cependant, sa taille réduite limite ses capacités de calcul, de stockage, de communication et d'énergie ; ce qui rend son déploiement tout seul inutile. Par contre, sa coordination avec d'autres capteurs lui permet de surpasser ces limites. Un réseau de capteurs sans fil est la fédération d'un grand nombre de capteurs nommés nœuds. Il a pour objectif la surveillance et la collecte des paramètres physiques de l'environnement où il a été déployé.

La taille réduite des capteurs, la large gamme de types de capteurs disponibles ainsi que le fonctionnement autonome des réseaux de capteurs, leur confère un brillant avenir dans plusieurs domaines d'application tels que le domaine environnemental, le domaine militaire ou le domaine médical. Dans un futur proche, ils pourront être présents dans nos vies quotidiennes tout comme les micro-ordinateurs ou les téléphones portables le sont aujourd'hui.

Mais, le développement de tels réseaux est confronté au problème de la consommation d'énergie qu'est une importance primordiale. Chaque capteur est alimenté par une source d'énergie limitée et généralement irremplaçable. De ce fait, sa durée de vie (ou son autonomie) et, en conséquence, celle de tout le réseau sont limitées. Dès lors, une attention particulière à la consommation d'énergie est requise lors de la conception des réseaux de capteurs ainsi que des protocoles de communication qu'ils utilisent. Par exemple, les protocoles de routages doivent garantir l'acheminement des données captées vers les points de collecte, tout en réduisant la consommation d'énergie induite par les transmissions.

L'agrégation de données est une solution efficace pour économiser l'énergie dans le réseau de capteur sans fil. L'idée fondamentale est de fusionner les données de diverses sources, le routage avec l'élimination de la redondance, ce qui implique par conséquent la réduction de fait le nombre de transmissions et d'économiser l'énergie.

Cependant, plusieurs attaques sur les réseaux de capteurs sans fil ont pour objectif de nuire l'agrégation de données pour rendre le fonctionnement total du réseau inutiles. Par exemple si uniquement un seul capteur agrège les données saines avec une donnée falsifiée issue d'un nœud malicieux, alors tout le travail d'un ensemble de capteurs sera incorrect.

Donc, la problématique de notre présente étude est d'empêcher les attaques qui menacent l'agrégation de données. Pour cela, on propose une solution qui sécurise l'agrégation de données, en réduisant la consommation de l'énergie, car c'est le but de l'opération d'agrégation de données.

Pour mener à bien notre travail, nous l'avons organisé en cinq chapitres ;
Dans le premier chapitre, nous verrons les concepts généraux relatifs au domaine des réseaux de capteurs sans fil. Le second chapitre, porte sur l'agrégation de données. Le troisième chapitre traite le problème de la sécurité de l'agrégation de données. Dans le quatrième chapitre nous présentons notre approche pour sécuriser les données agrégées. Nous détaillent dans le cinquième chapitre, les étapes de l'implémentation, et par la suite nous donnons les résultats de simulations. Enfin, nous terminons par une conclusion générale.

Chapitre I:

Généralité sur les réseaux de capteurs.

I.1 Introduction

Un capteur sans fil, fruit des récents progrès de la micro électromécanique, des communications sans fil et des nouvelles technologies, est un petit dispositif intelligent ayant la capacité de traitement local et de communication sans fil. Cependant, sa taille réduite limite ses capacités de calcul, de stockage, de communication et énergétiques ; ce qui rend son déploiement, tout seul, inutile. Par contre, sa coordination avec d'autres capteurs lui permet de surpasser ses limites. Un réseau de capteurs sans fil est la fédération d'un grand nombre de capteurs, nommés nœuds, pour réaliser une tâche commune. Il a pour objectif la surveillance et la collecte des paramètres physiques de l'environnement où il a été déployé.
L'objectif de ce chapitre est de donner une vision d'ensemble sur les réseaux de capteurs sans fil (RCSF).

I.2 Capteur et réseau de capteur sans fil

I.2.1 Un Capteur sans fil [1]

I.2.1.1 Définition

Un capteur sans fil est un petit dispositif à un coût raisonnable, de quelques millimètres cubes de volume. Il a pour but de relever une grandeur physique comme l'humidité, la température et les vibrations, suivant l'environnement dans lequel il est déployé et l'objectif pour lequel il est conçu, puis la transformer en une mesure généralement électrique qui sera à son tour traduite en une donnée binaire exploitable et compréhensible pour un système d'information.

I.2.1.2 Architecture matériel d'un capteur [1]

Un capteur sans fil est doté, principalement, de quatre unités de base comme le montre la figure I.1 suivante:

- **L'unité d'acquisition des données (captage)**

Elle est constituée de deux composants, un dispositif qui intercepte les données du monde physique et les transforme en signaux analogiques (détection), et un convertisseur analogique/numérique CAN (Analog Digital Converter) qui transforme ces signaux analogiques en un signal numérique compréhensible par l'unité de traitement.

- **L'unité de traitement des données**

Elle est composée d'un microprocesseur ou d'un microcontrôleur associe généralement à une unité de stockage (mémoire). Elle est chargée d'exécuter les protocoles de communication, comme elle peut aussi effectuer des semi-traitements sur les données captées.

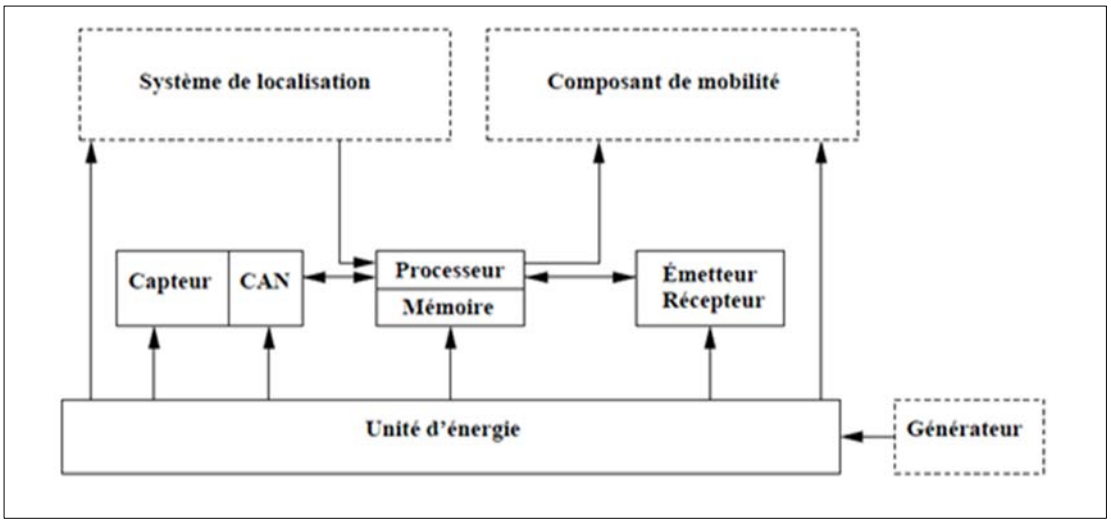


Figure. I.1- les composants fonctionnels d'un capteur sans fil.

- **Unité de communication (émetteur/récepteur)**

Elle est responsable des émissions et réceptions des données sur un médium sans fil (connecte le nœud au réseau). Elle se base sur les technologies sans fil à faible portée de communication : Zigbee (IEEE 802.15.4), Bluetooth (IEEE 802.15.1) ou Wi-Fi (IEEE 802.11).

- **Unité d'alimentation énergétique**

Elle est responsable de la gestion de l'énergie et de l'alimentation de tous les composants du capteur. Elle consiste, généralement, en une batterie qui est limitée et irremplaçable, ce qui a rendu l'énergie comme principale contrainte pour un capteur.

On peut également avoir d'autres unités suivant l'application. Parmi ces unités, nous citons:

- *L'unité de localisation (GPS)* : La plupart des techniques de routage et les tâches de détection des réseaux de senseurs nécessitent la connaissance de la localisation avec une grande précision ;
- *L'unité assurant la mobilité* : pour déplacer le nœud senseur assigné à une tâche par exemple la poursuite d'une cible ;
- *Un générateur d'énergie* ;

Ces capteurs que nous venons de décrire sont la pierre angulaire à l'édification de réseaux dits de capteurs sans fil.

I.2.2 Réseaux de capteurs sans fil

I. 2.2.1 Définition

Un réseau de capteurs sans fil (RCSF) est un type particulier des réseaux ad hoc [2] qui partage beaucoup de caractéristiques avec les réseaux embarqués sans fil [3]. Il est constitué d'un grand nombre de capteurs ou nœuds senseurs déployés aléatoirement d'une manière dense dans le champ de captage, ces capteurs coopèrent pour réaliser une tâche commune. Le RCSF est conçu dans le but de surveiller, détecter et traiter des phénomènes physiques captés

au niveau local, ou de les envoyer, à l'aide de la communication sans fil à un ou plusieurs points de collecte, appelés *puits*.

I.2.2.2 Architecture d'un réseau de capteurs

Un réseau de capteurs est généralement constitué d'un ensemble de nœuds capteurs, allant de quelques dizaines à plusieurs milliers, répartis de manière plus ou moins aléatoire dans une zone d'intérêt. Ces nœuds sont reliés à une ou plusieurs passerelles en utilisant une communication sans fil pour acheminer les données captées vers la station de base ou puits. Le puits transmet ensuite ces données via d'autres réseaux (Internet, satellite . . .) Vers l'utilisateur final pour l'analyse et la prise de décisions (figure.I.2). Des super-nœuds, possédant une énergie importante et une capacité de traitement et de stockage élevées, peuvent être introduits dans le réseau pour exécuter des tâches plus complexes comme la fusion des données issues des capteurs d'une même zone. Les capteurs qui ne sont pas au voisinage du puits doivent envoyer leurs données à d'autres capteurs qui prendront en charge leurs acheminements jusqu'au puits selon un mode de communication multi sauts.

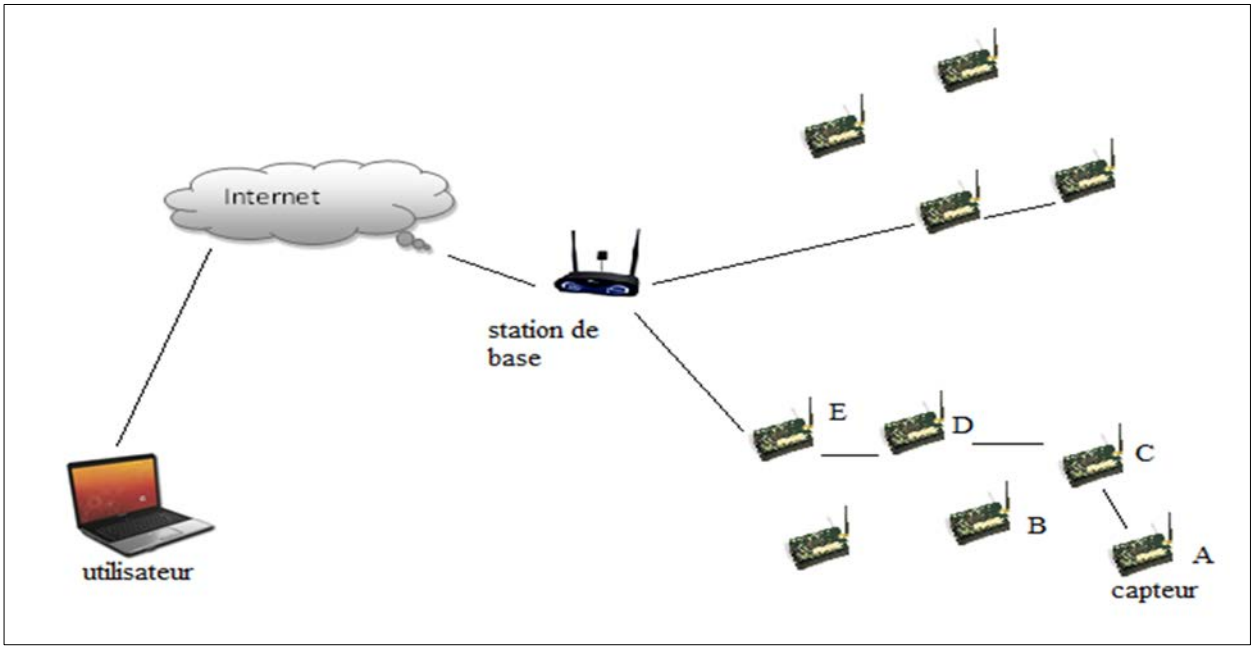


Figure. I.2- Architecture d'un réseau de capteur.

I.3 Classification des RCSFs

Pour classer les RCSFs, on peut utiliser plusieurs critères [4,5]. En effet, suivant le type d'application, ces réseaux ont des caractéristiques différentes. Ils se distinguent par le mode d'acquisition et de livraison des données au puits, la distance entre les nœuds capteurs et le puits, le modèle de mobilité dans le réseau, les capacités des nœuds du réseau, etc.

I.3.1 Selon le mode d'acquisition et de livraison des données au puits

Le modèle d'acquisition et de livraison des données au puits dépend de l'application et de ses exigences. On peut distinguer ceux cités là-dessous :

- **Le modèle orienté temps**

Ce modèle est appelé aussi continu (time-drivent en anglais) où l'acquisition et la transmission des données capturées sont liées au temps puisque les nœuds doivent périodiquement réveiller leurs émetteurs pour transmettre leurs données au puits. Ce type d'applications est dédié au domaine de collecte de données environnementales, par exemple le fonctionnement des capteurs de température déployés dans un champ agriculture qui transmettent à chaque instant précis et périodique un rapport au puits.

- **Le modèle orienté événements**

Dans ce cas, les capteurs envoient leurs données seulement si un événement spécifique se produit. On peut citer l'exemple de surveillance des feux dans les forêts où un capteur envoie des alarmes à la station de base dès que la température dépasse un certain seuil. Au départ, les applications de ce type de modèle étaient conçues à des fins militaires, comme la surveillance du déplacement d'objets dans le champ de bataille. Par la suite, elles ont rapidement trouvé de nouvelles perspectives comme le contrôle industriel, le contrôle médical des patients, la surveillance de buildings (barrages, ponts, voies de chemin de fer, etc.).

- **Le modèle orienté requêtes**

Un capteur envoie de l'information uniquement suite à une demande explicite de la station de base. Ce modèle est destiné généralement aux applications adaptées à l'utilisateur. Ce dernier peut requérir des informations à partir de certaines régions dans le réseau ou interroger les capteurs pour acquérir des mesures d'intérêts. Dans ce cas, des connaissances sur la topologie du réseau et l'emplacement des capteurs sont nécessaires.

- **Le modèle hybride**

Combine les trois modes de fonctionnement cités précédemment. Par exemple, dans un réseau conçu pour le suivi d'objets, le réseau peut combiner entre un réseau de surveillance et un réseau de collecte de données par événements c.-à-d. pendant les longues périodes d'inactivité des capteurs et lorsqu'aucun objet n'est présent, le réseau peut assurer une fonction de surveillance.

I.3.2 Selon le modèle de mobilité dans le réseau

On peut distinguer deux grandes catégories de réseaux selon que les nœuds capteurs et le nœud puits sont mobiles ou fixes :

- **Réseau statique**

Sers à la surveillance d'occurrences d'événements sur une zone géographique.

- **Réseau dynamique ou mobile**

Utilisé pour l'exploration de zones inaccessibles ou dangereuses.

I.3.3 Selon la distance entre les nœuds capteurs et le puits

Il y a des réseaux multi sauts (multi-hop WSN) et des réseaux à un seul saut (single-hop WSN).

- **Réseau à un seul saut**

Les nœuds capteurs sont dans le voisinage immédiat du puits ils envoient alors leurs données directement au puits sans passer par aucun nœud intermédiaire

- **Réseau multi sauts**

La distance entre certains capteurs du réseau et le puits dépasse leur portée maximale, donc ils ne peuvent pas atteindre directement le puits, alors pour envoyer leurs données, ils doivent le faire par l'intermédiaire d'autres capteurs. Ce genre de réseau a une large gamme d'application, mais il est difficile à le mettre en œuvre [5].

I.3.4 Selon les capacités des nœuds du réseau

On trouve dans cette classe les réseaux homogènes ou hétérogènes [6].

- **Le réseau homogène**

Tous les nœuds du réseau (capteurs, passerelles et puits) ont les mêmes capacités du point de vue énergie, calcul et stockage.

- **Le réseau hétérogène**

À la différence d'un réseau homogène, ce type de réseau a quelques nœuds sophistiqués qui ont plus de capacité de traitement et de communication que les nœuds normaux. Cela prolonge la vie du réseau. Ces nœuds sophistiqués peuvent être utilisés pour l'exécution des tâches complexes comme les coordinateurs et les chefs du cluster. Mais il est difficile de mettre en place un tel réseau puisque chaque type de nœuds aura un code propre à lui ce qui augmente le coût de développement.

I.4 Caractéristiques et facteurs critiques

Les principales caractéristiques et contraintes qui différencient les RCSFs par rapport aux autres réseaux sont :

1. *Ressources limitées* : les nœuds qui composent le RCFS sont de très petite taille jusqu'à l'ordre d'une poussière (smart dust), suivant leur utilité. Par conséquent, leurs capacités de traitement, de stockage, de communication et d'énergie sont très limitées.
2. *Durée de vie* : les capteurs disposent d'une ressource énergétique très limitée et généralement irremplaçable. Cette contrainte énergétique influe directement sur la durée de vie des capteurs, ainsi sur celle du réseau entier. La durée de vie d'un réseau de capteurs est définie de l'instant de son déploiement jusqu'au moment de l'épuisement de l'énergie du premier nœud [7] ou d'un certain pourcentage de nœuds [8].
3. *Topologie dynamique* : la topologie d'un RCSF peut être dynamique, ceci est dû à la mobilité des nœuds qui peuvent s'attacher à des objets mobiles, ou dû à l'ajout ou à la

suppression d'un nœud après le déploiement pour élargir le réseau ou pour remplacer les nœuds décédés ou défaillants.

4. *Passage à l'échelle* : à la différence des réseaux ad hoc, les RCSFs hébergent des milliers, voire des millions, de nœuds capteurs. Le défi à relever par les RCSFs est d'être capables de maintenir leurs performances avec ce grand nombre de capteurs.
5. *Autonomie* : les RCSFs sont des réseaux sans infrastructure fonctionnant sans intervention humaine. De ce fait, les RCSFs doivent être auto stabilisants : s'auto structurer, s'auto-organiser, s'auto reconfigurer, s'auto maintenir et s'auto sécuriser.
6. *Canal radio* : les capteurs d'un RCSF sont reliés entre eux par un médium sans fil qui est très hostile. La qualité du signal de transmission peut être dégradée à cause des collisions, des interférences, des diffractions, des réfractions, etc.
7. *Sécurité physique limitée*: cela se justifie par les contraintes et limitations physiques qui minimisent le contrôle des données transmises.
8. *Agrégation des données* : la transmission des données sur les réseaux de capteurs est l'une des tâches gourmandes en consommation d'énergie. Cependant, une approche répandue consiste à agréger les données au niveau des nœuds intermédiaires.

I.5 La communication dans les RCSFs

I.5.1 La pile protocolaire

Pour la prise en compte de l'hétérogénéité des équipements et la possibilité d'une adaptation facile, des règles communes de communication et de coopération formants un protocole de communication entre les équipements ont été définis. En général, la pile protocolaire est organisée selon un modèle en couche où chaque couche assure une fonctionnalité nécessaire à la communication d'une façon indépendante des autres couches, en utilisant les services de celles au niveau inférieur et en fournissant à son tour des services à celles de niveau supérieur. Vu les nouvelles exigences des réseaux de capteurs sans fil, la conception des protocoles de communication doit prendre en compte les contraintes propres aux RCSFs[9].

La figure I.3 montre la pile protocolaire utilisée par le puits (Sink) ainsi que par tous les autres capteurs. Elle comprend, en plus des 5 couches qui ont les mêmes fonctions que celles du modèle OSI, trois plans pour la gestion de : l'énergie, de la mobilité et de tâches.

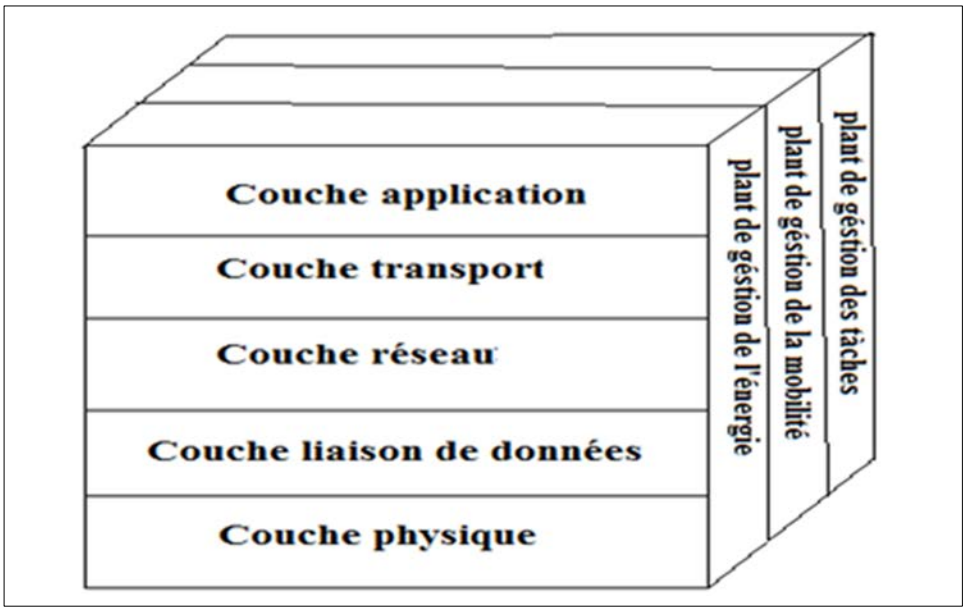


Figure. I.3-La pile protocolaire pour les RCSFs.

- **La couche physique**

Responsable d'envoi et de réception de données, en les modulant et les acheminant sur le média physique tou en choisissant les bonnes fréquences porteuses.

- **La couche liaison de données**

Elle est responsable du multiplexage du flux de données, du contrôle d'accès aux médias (MAC), la détection et la correction d'erreurs intervenues sur la couche physique. Elle assure une connexion fiable.
Parmi les protocoles de liaison de données, nous citons : SMACS (Self-ogranising Medium Acces Control for Sensor networks) et EAR (Eavestdrop and Register).

- **La couche réseau**

Elle s'occupe de 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 selon la métrique choisie (énergie, délai de transmission, débit ...etc.).

- **La couche transport**

Elle s'occupe du transport des données, de leur découpage en paquets, du contrôle de flux, de la conservation de l'ordre des paquets et de la gestion des éventuelles erreurs de transmission.

- **La couche application**

Elle est constituée de l'ensemble des applications implémentées sur le réseau de capteurs. Elle assure l'interaction du réseau avec l'utilisateur. Il s'agit donc de la couche la plus proche des utilisateurs, gérée directement par les logiciels.
la pile protocolaire d'un capteur est dotée de trois plan de gestion :

- **Le plan de gestion d'énergie**

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

- **Le plan de gestion de mobilité**

Détecte et enregistre le mouvement de capteur. Ainsi, un retour arrière vers l'utilisateur est toujours maintenu et le capteur peut garder trace de ses capteurs voisins. En déterminant leurs voisins, les capteurs peuvent équilibrer l'utilisation de leur énergie et la réalisation de tâche.

- **Le plan de gestion de tâches**

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

I.6. Domaines d'application des RCSFs

Le déploiement rapide des RCSFs autonomes à faible prix, même sur une très grande échelle géographique, a été la source d'émergence de plusieurs applications différentes en termes des besoins et des caractéristiques. On cite ci-dessous les principaux domaines d'application des RCSFs [1] [9] :

I.6.1 Domaine militaire

Les RCSFs peuvent être rapidement déployés et utilisés pour la surveillance des champs de bataille afin de fournir des renseignements concernant l'emplacement, le nombre, le mouvement, et l'identité des soldats et des véhicules, ou bien encore pour la détection des substances dangereuses(chimiques, biologiques et nucléaires) avant l'envoi des troupeaux militaires.

I.6.2 Domaine commercial

Dans la fabrication industrielle, des capteurs et des actionneurs sont utilisés pour les processus de suivi et de contrôle. Par exemple, dans une usine de traitement chimique à plusieurs étapes, il peut y avoir des capteurs placés en différents points dans le processus afin de surveiller la température, la concentration chimique, la pression, …etc. Les capteurs peuvent être aussi utilisés pour le contrôle environnemental des bâtiments pour permettre une meilleure gestion des ressources à faibles coûts. Un autre exemple est celui de l'utilisation des capteurs dans des musées scientifiques pour un apprentissage plus rapide des visiteurs.

I.6.3 Domaine médical

Parmi ses applications, on peut citer la surveillance d'état des patients et des taux de médicaments qui leurs ont été administrés, et l'aide à la localisation des médecins et des patients au sein d'un hôpital.

I.6.4 Domaine environnemental

Dans ce domaine, les capteurs peuvent être exploités pour détecter les catastrophes naturelles (feux de forêts, tremblements de terre…), traquer les mouvements animaux et surveiller les conditions d'environnement qui affectent les récoltes, les stocks et tout autre système d'agriculture.

I.6.5 Domestique

Les capteurs peuvent être embarqués dans des appareils, tels que les aspirateurs, les fours à micro-ondes, les réfrigérateurs, les magnétoscopes....etc., ces capteurs embarqués peuvent interagir entre eux et avec un réseau externe via Internet pour permettre à un utilisateur de contrôler les appareils domestiques localement ou à distance. [10]

Le déploiement des capteurs de mouvement et de température dans les futures maisons dites intelligentes permet d'automatiser plusieurs opérations domestiques telles que : la lumière s'éteint et la musique se met en état d'arrêt quand la chambre est vide, la climatisation et le chauffage s'ajustent selon les points multiples de mesure, le déclenchement d'une alarme par le capteur anti-intrusion quand un intrus veut accéder à la maison.

I.3.8 Autres applications [11]

Parmi les autres applications des réseaux de capteurs, on a : l'agriculture, des capteurs sont incorporés dans la terre pour donner des informations sur l'état du champ. 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. Grâce aux réseaux de capteurs, les entreprises pourraient offrir une meilleure qualité de service tout en réduisant leurs coûts. Les réseaux de capteurs peuvent également être utilisés pour : surveiller l'infrastructure, lutter contre le terrorisme, contrôle du trafic, détecter des intrusions (en plaçant à différents points stratégiques des capteurs), contrôler les stocks (savoir le lieu, la quantité, la forme de tous les produits, contrôler leur flux… etc.), l'urbanisme, l'ingénierie civile,…etc.

Figure I.4 : Exemple de domaines d'utilisation des RSCFs.

I.7. Principaux domaines de recherche

Dans la littérature, plusieurs travaux de recherche visent à proposer des solutions optimales et efficaces à un ou plusieurs problèmes des RCSFs illustrés précédemment. Les principaux domaines de recherche abordés dans les RCSFs sont les suivants :

✓ Efficacité énergétique

En raison de la ressource énergétique limitée, plusieurs solutions, à la fois matérielles et des logiciels, ont été proposées afin d'optimiser l'utilisation de l'énergie [12].

✓ Localisation

Vu le très grand nombre de nœuds capteurs sur un RCSF et leur déploiement d'une manière ad hoc, de nombreux systèmes de coordonnées spatiales et virtuelles ont été proposés, auxquels les nœuds capteurs peuvent s'identifier pour se localiser dans le RCSF.

✓ Routage

Plusieurs protocoles de routage ont été proposés pour les RCSFs pour minimiser les couts de communication, afin de réduire la consommation énergétique.

✓ Sécurité

Les applications utilisant les RCSFs ont souvent besoin d'un niveau de sécurité élevé. Or, de par leurs caractéristiques la sécurisation des RCSFs est à la source de beaucoup de travaux scientifiques et techniques proposant des solutions de sécurité efficaces [13].

I.8. Comparaison entre les réseaux de capteurs et MANET

Malgré les différences existantes, ces deux types de réseaux partagent beaucoup de points communs [11] :

- Tous les deux sont des réseaux sans fil, ce qui fait que la portée de communication est limitée.
- Tous les deux sont des réseaux ad hoc, c'est-à-dire, ils fonctionnent sans avoir besoin d'une infrastructure pour la gestion des échanges. De ce fait, ils ont besoin d'être auto configurables.
- Les réseaux sont souvent alimentés par des batteries.

Malgré les points sur lesquels ces deux types de réseaux convergent, ils diffèrent sur plusieurs aspects. Parmi les points sur lesquels ils divergent, on cite :

- Les entités d'un réseau MANET sont souvent utilisées directement par des êtres humains, comme les portables, les PDA. Dans les réseaux RCSFs les entités interagissent essentiellement avec l'environnement.
- Les capteurs peuvent être densément déployés par rapport au réseau MANET.
- Le fait que les nœuds du réseau dans un RCSF sont souvent déployés dans des environnements hostiles les rend vulnérables et risquent de tomber en panne beaucoup plus souvent que les nœuds dans un réseau MANET.
- La topologie des capteurs change fréquemment du fait des pannes des nœuds ou de leur mobilité.
- Les échanges de données dans les applications d'un réseau RCSF sont souvent du type collecte de données. Les nœuds doivent envoyer vers un puits des informations sur des phénomènes observés (modèles de communication many-to-one) alors que les applications des réseaux MANETs sont plus orientées calcul distribué et donc le trafic circule entre tous les nœuds du réseau et dans tous les sens (modèle de communication many-to-many).
- Les RCSFs sont qualifiés de réseaux à basse consommation et à bas débit, ceci n'est pas le cas des MANETs même si les nœuds d'un MANET sont alimentés par des batteries, ils sont facilement rechargeables.

I.9 Conclusion

Dans ce chapitre, nous avons présenté les concepts généraux nécessaires à la bonne compréhension des RCSFs. Dans le chapitre suivant, nous allons présenter l'agrégation de données dans ces réseaux.

Chapitre II:

L’agrégation de données dans les WSNs.

II.1. Introduction

Dans le chapitre précédent, nous avons présenté une vue générale sur les réseaux de capteurs (WSNs) et les domaines d'utilisations. Dans ce chapitre nous nous intéresserons à l'agrégation de données.

L'agrégation de données est considérée en tant qu'une des procédures fondamentales de résumer de l'information répartie pour économiser l'énergie et réduire au minimum la communication dans les réseaux WSNs. L'idée fondamentale est de fusionner les données de diverses sources, le routage avec l'élimination de la redondance, ce qui implique par conséquent la réduction du nombre de transmissions pour économiser l'énergie [2].

II.2. Types d'agrégation de données

Il existe deux approches pour agréger les données [18] :

II.2.1 Agrégation de données avec réduction de taille

Les données reçues par un nœud d'agrégateur de ses nœuds enfants sont traitées d'une telle manière de réduire sa taille ; par exemple, si un nœud agrégateur reçoit deux mesures de température de ses deux enfants, il pourrait calculer leur moyenne et expédier le résultat à l'agrégateur de niveau supérieur, donc il envoie un paquet simple au lieu de deux.

II.2.2 Agrégation de données sans réduction de taille

Le nœud agrégateur fusionne les paquets venant de différentes sources dans un paquet simple. Ceci permet une réduction des frais généraux, alors que la taille de données n'est pas réduite.

Selon [7], des techniques d'agrégation de données dans WSNs sont basées sur trois composants fondamentaux : protocoles de gestion de réseau appropriés, fonctions d'agrégation efficaces et manières efficaces de représenter des données. Une courte description de chaque composant est fournie dans le reste de cette section.

II.3. Le protocole de routage

Afin d'exécuter l'agrégation de données, un protocole spécifique de cheminement doit être conçu. En fait, alors que les protocoles d'acheminement traditionnels visent à trouver le chemin le plus court vers la destination (selon une métrique spécifique), on s'attend à ce qu'un protocole de cheminement approprié à l'agrégation de données choisisse les chemins qui assurent optimal agrégation de données avec le chemin le plus court vers la destination [18].

Une autre issue spécifique qui doit être tenue compte lors de la conception d'un protocole d'agrégation de données est la stratégie de synchronisation, c.-à-d. une certaine forme de la synchronisation parmi les nœuds. Selon l'application spécifique, une des politiques suivantes a pù être choisie:

II.3.1 Agrégation périodique

Des opérations d'agrégation sont déclenchées par un temporisateur. Donc, quand le temporisateur est déclenché, chaque nœud agrégateur agrège les données reçues dans la tranche de temps courante et fait suivre le résultat aux nœuds indiqués.

II.3.2 Agrégation périodique per-hop

Des opérations d'agrégation sont effectuées dès qu'un nœud aura des nouvelles (de données reçus) de tous ses nœuds enfants (c.-à-d. reçois les paquets de ces nœuds enfants). Un temps mort pour attendre les paquets est employé pour empêcher le protocole d'attendre infiniment au cas où les paquets de quelques enfants seraient perdus.

II.3.3 Agrégation ajustée par per-hop périodique

L'approche est identique à celle de l'agrégation périodique sauf que les temps morts de per-hop sont ajustés à chaque nœud, selon sa position dans la structure du réseau.

II.4. Fonctions d'agrégation

Afin d'exécuter l'agrégation, une fonction d'agrégation efficace est nécessaire, de sorte que les données venant de différentes sources puissent être combinées au nœud agrégateur. Par ''efficace'' nous voulons dire que ces fonctions doivent tenir compte de la disponibilité de ressources rares des nœuds capteurs et doivent faire face à leurs limitations graves.

Beaucoup de types de fonctions d'agrégation existent, selon l'application spécifique du capteur. Les exemples typiques sont des fonctions telles que : moyenne, le comptage, maximum et minimal. Spécifiquement, nous pouvons distinguer [19]:

II.4.1 Fonctions d'agrégation avec perte Vs sans perte

Des données peuvent être agrégées en perdant une partie d'informations originales contenter (agrégation avec perte par exemple : la somme, le moyenne, min, Max… etc.) ou en préservant tous (agrégation sans perte par exemple : la fusion d'informations). L'agrégation sans perte assure un rétablissement complet de toutes les lectures dans un capteur (BS), alors que l'approche avec perte ne permet pas une reconstruction parfaite.

II.4.2 Fonctions d'agrégation sensible à la redondance de données Vs des fonctions d'agrégation peu sensibles

Si un nœud agrégateur reçoit des données multiples de capteurs avec le même contenu, il pourrait soit tenir compte de l'information inutile ou simplement la jeter. Dans le premier cas, l'agrégation est sensible à la redondance de données, alors que dans le second elle ne l'est pas.

II.5. Conclusion

Il est important de souligner que le traitement de données est moins coûteux en consommation d'énergie que la communication. Il est donc plus avantageux de privilégier le traitement local des données en les agrégeant de sorte à réduire le nombre de transmissions et avoir des données plus significatives. Grâce à l'agrégation de données, une économie substantielle d'énergie ainsi qu'une amélioration de la qualité des données reçues par l'utilisateur peuvent être obtenues. Mais, l’information agrégée est très vulnérable au nœud malicieux, car une donnée agrégée représente un résumé des informations collectées. Ce qui nous motive à donner le chapitre suivant sur la sécurité d’agrégation de données.

Chapitre III:

Sécurité et l’agrégation de données.

III.1 Introduction

Après avoir présenté un aperçu sur l'agrégation de données, nous intéressons dans ce chapitre à la sécurité de l'agrégation de données dans les réseaux WSNs. Nous allons définir les types d'attaques et leurs méthodes de défense proposées dans la littérature. Nous terminons ce chapitre en citant quelques protocoles qui mettent l'accent sur la sécurité d'agrégation de données.

Avant de commencer nous allons d'abord définir une attaque, qui est une tentative d'accès non autorisé à un service, une ressource ou une information, ou bien une tentative de compromettre l'intégrité, la disponibilité, ou la confidentialité du réseau.

III.2 Objectife de la sécurité

Lorsque nous abordons le problème de la sécurité informatique, nous visons à atteindre certains objectifs, dont les principaux sont les suivants [22] :

III.2.1 L'authentification

Permet d'assurer l'identification des origines des messages, pour cela on exploite le Code d'Authentification de Message (MAC).

III.2.2 L'intégrité

Elle assure que les données reçues n'ont pas été altérées durant leur transit dans le réseau de manière volontaire ou accidentelle. Elle peut être assurée par l'utilisation des fonctions de hachage cryptographiques qui permettent d'obtenir pour chaque message une empreinte numérique.

III.2.3 La confidentialité

Elle consiste à préserver le secret des messages échangés et ne pas les révéler aux adversaires. La confidentialité peut être assurée par l'usage de la cryptographie.

III.2.4 La disponibilité

Elle signifie que le réseau est disponible pour assurer ses services et autoriser les parties communicantes lorsque ceci est nécessaire.

III.2.5 La fraicheur

Ce dernier service permet de garantir que les données échangées sur le réseau sont actuelles et ne sont pas une réinjection de précédents échanges interceptés par un attaquant.

Dans ce qui suit nous donnons une classification des attaques informatiques.

III.3 Classification des attaques

Il existe deux grandes classes d'attaques [22]:

III.3.1 Attaque passive

Tente de collecter ou d'utiliser des informations relatives au système, mais elle n'affecte pas les ressources du système, cette attaque ne modifie pas le contenue des paquets router dans un réseau. Cette attaque est difficile à détecter, mais assez facile à sécuriser.

III.3.2 Attaque active

Entraine la modification de l'information ou la création de fausses informations pour endommager le réseau.

Par exemple le déni de service est une famille des attaques classées comme une attaque active menaçant le fonctionnement de l'agrégation de données.

III.4 Les types de déni de service

Les dénis de service (DoS) sont définis comme un mauvais fonctionnement des capteurs d'une manière intentionnée ou par action malveillante. Le déni de service peut ne pas résulter d'une attaque, mais d'un simple événement empêchant le fonctionnement normal d'un de ses services.
Le déni de service le plus simple est d'empêcher le fonctionnement normal du capteur victime en lui envoyant énormément de messages sans importance. Les attaques de déni de service ciblent la réduction des capacités d'un réseau de capteurs sans fil. Les contraintes physiques de ces réseaux de capteurs, et la nature de leur environnement de déploiement, les rend vulnérables aux attaques de dénis de service plus que tout autre type de réseau.

Des solutions pour contrer les Dos existent, comme l'augmentation des ressources, l'authentification et l'identification du trafic.

Nous décrivons dans cette section certains dénis de service qui affectant l'opération d'agrégation de données et les moyens de sécurisation contre eux.

III.4.1 L'attaque Homing

Le déploiement des réseaux de capteurs WSN de grande taille utilise parfois le clustering pour le routage du trafic en optimisant l'énergie, par agrégation des données au niveau du cluster Head, l'attaquant peut profiter du clustering pour introduire un faux cluster Head disposant d'une forte transmission et qui invite plusieurs capteurs à rejoindre une grappe inexistante (figure III. 1).

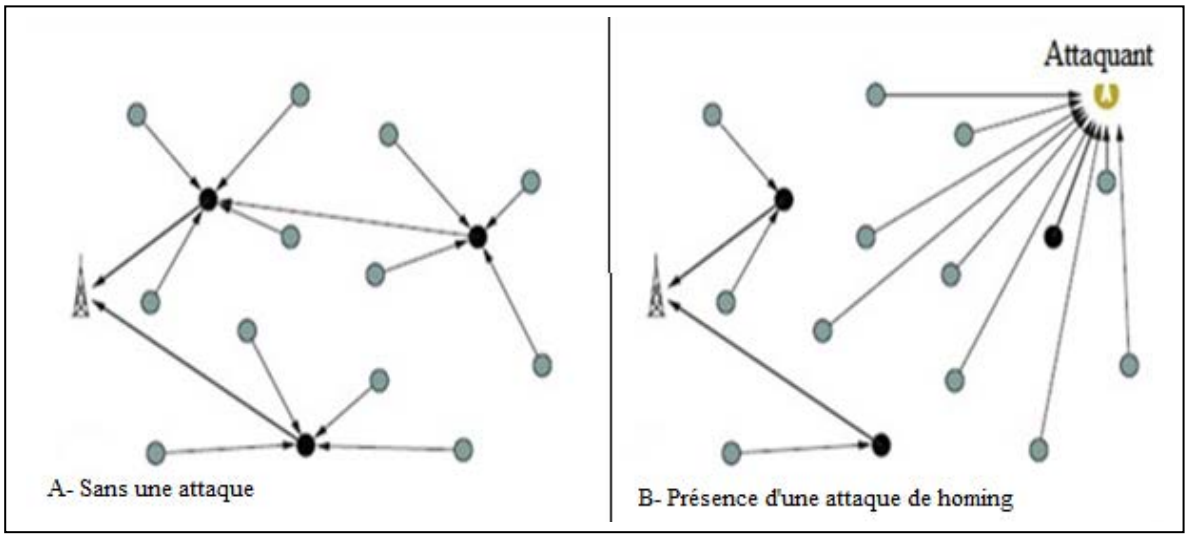


Figure. III.1 Réseau qui souffre d'un Cluster-Head malveillant [20].

III.4.2 L'attaque black holes

Dans l'attaque de black holes (ou trous noirs), les capteurs agressifs indiquent des routes à couts nuls aux autres capteurs ce qui constitue des trous noirs de routage dans le réseau. Ainsi, au fur et à mesure que la publication des messages se propage, le réseau achemine plus de trafic dans leurs directions.
Une des approches de défense contre Black holes est de n'autoriser que les capteurs authentifiés à envoyer des informations [21].

La figure III.2 montre un capteur malveillant qui s'est introduit entre des capteurs et une station de base afin de capter les paquets en transit entre eux.

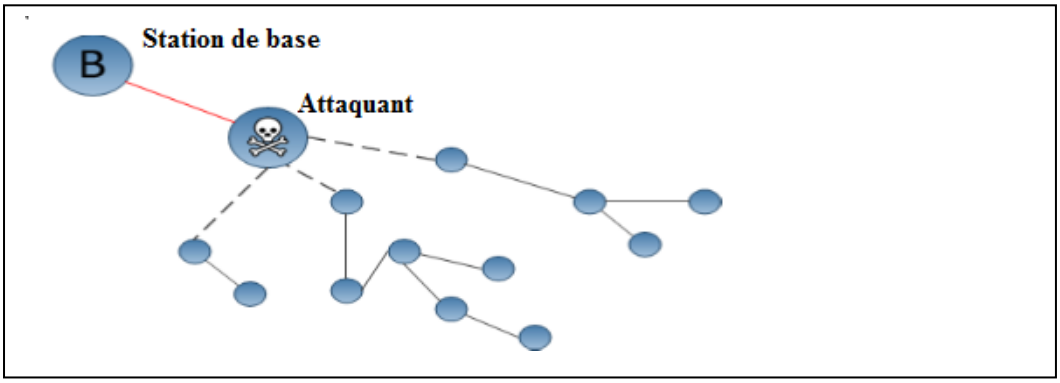


Figure. III.2- une vue de l'attaque black holes [22].

III.4.3 L'attaque Misdirection

Misdirection est une attaque active, qui renvoie un message vers de mauvaises directions en fabriquant une publication pour une route malveillante. Le protocole DSR est sensible à cette attaque. Un attaquant peut se forger une adresse source lors de l'envoi de requêtes, ainsi la réponse est retournée à un capteur victime. Cette méthode est utilisée afin d'affoler la victime ou bien de l'inonder.

Une des techniques de défense contre l'attaque de Misdirection, est l'authentification des capteurs, ainsi les tables de routage sont mises à jour, en tenant compte des informations authentifiées des capteurs de la route [2].

III.4.4 L'attaque Sink holes

Avec cette attaque, un capteur malveillant agit comme un "Black holes" afin de capter tout le trafic du réseau de capteurs. L'attaquant écoute les requêtes des routes demandées par les capteurs victimes, puis il envoie un message à ces capteurs leur signifiant qu'il a à sa disposition le meilleur chemin vers la station de base. Une fois que le capteur malveillant s'est introduit dans le chemin entre le capteur victime et la station de base, il peut affecter l'acheminement du message comme il le souhaite. Wood et al [21] décrivent l'attaque de "Sinkholes" comme une attaque de fausses courtes routes.
Une des approches de défense contre cette attaque est l'utilisation des algorithmes de routage résistants aux configurations arbitraires [23].

III.4.5. L'attaque Sybil

Dans attaques Sybil, un nœud prétend avoir des identités multiples. Dans ce cas, la plupart des messages seront adressés à l'attaquant Sybil étant donné qu'il figure à plusieurs endroits. Pour remédier à ce problème, une solution consiste à limiter le nombre de voisins de chaque nœud avec lesquels un nœud partagera des secrets.

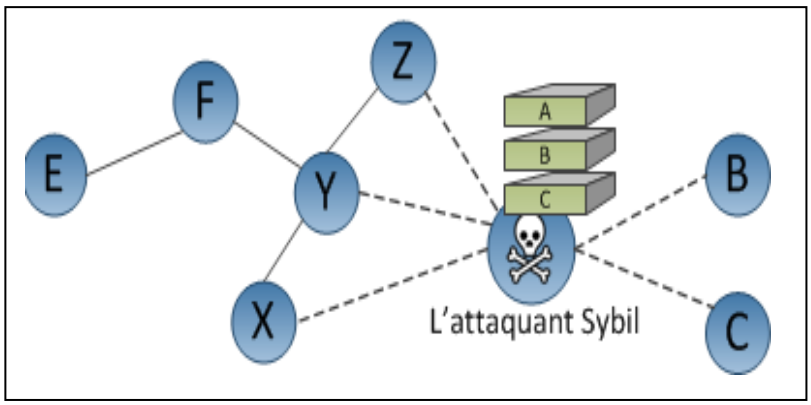


Figure. III.3- L'attaque Sybil [22].

III.5. Les types de ressources à sécuriser dans WSNs

Aujourd'hui, on dénombre plusieurs travaux de recherche ayant proposé des solutions aux différentes attaques portant atteinte à la sécurité des WSNs. Dans cette section, on s'intéresse essentiellement à la sécurisation du réseau (infrastructure).

III.5.1 Sécurisation des liens

La cryptographie asymétrique est la solution proposée pour l'établissement des liens sécurisés dans les réseaux. Malheureusement, celle-ci n'est pas adaptée aux réseaux de capteurs puisqu'elle est trop gourmande en temps d'exécution et en espace mémoire utilisé, et par conséquent en consommation d'énergie. De ce fait, la cryptographie symétrique est plus souhaitable pour les réseaux de capteurs.

III.5.2 Sécurisation des données agrégées

Pour assurer la confidentialité et l'intégrité, les données agrégées doivent être chiffrées. Il existe plusieurs publications qui ont proposé des solutions aux différents problèmes de sécurité des données agrégées. À titre d'exemple :

[23] propose un ensemble de techniques permettant d'assurer l'intégrité des données agrégées, mais elles n'assurent ni la confidentialité ni la protection contre le déni de service. De plus, ces techniques sont très difficiles à implémenter. [22] propose une technique qui fournit une authentification et une vérification de l'intégrité même en présence de quelques nœuds compromis. Malheureusement, elle ne fournit ni la confidentialité ni la disponibilité et elle est inefficace lorsque deux agrégateurs successifs sont compromis.

La sécurisation de l'agrégat peut se faire de point à point ou de bout en bout [26].

III. 5.2.1 Sécurité d'agrégation de point à point

La première solution pour le mode point à point est que chaque paire de nœuds voisins établit une clé secrète. Ensuite, chaque nœud déchiffre les données reçues de ces fils, ajoute sa valeur, les agrège, puis chiffre le résultat avec la clé qu'il possède avec son père. Cette méthode assure la sécurité, car un nœud compromis ne révèle que les clés de ce nœud avec ces voisins. Mais, elle est très couteuse. Elle nécessite l'établissement d'une clé secrète entre deux voisins et le déchiffrement de plusieurs messages, les agréger et les chiffrer en un seul message. De plus, la compromission d'un nœud près du puits permet à l'attaquant d'avoir un agrégat significatif.

- **Secure DAV, Secure Data Aggregation and Verification**

Dans beaucoup d'aspects, Secure DAV est un protocole semblable à SIA: les deux visent à assurer la protection contre les attaques furtives (c.-à-d. les attaques dans lesquelles l'agrégateur essaie d'une façon ou d'une autre de changer le résultat d'agrégation sans être découverte par les autres nœuds).Il emploie un mécanisme de vérification pour découvrir la falsification possible sur le résultat d'agrégation.

Néanmoins, Secure DAV présente quelques nouveaux dispositifs importants. Tout d'abord, Secure DAV emploie la cryptographie de courbe elliptique [23], maintenant le coût informatique à un niveau raisonnable. En particulier, l'algorithme de hachage (nommé CCE) permet à des nœuds capteurs de produire efficacement une signature, alors que la vérification de cette signature est beaucoup plus complexe, il est basé sur l'arbre de hachage de Merkle (figure III.4).

Pour réduire la taille de données à vérifier, l'agrégateur hache d'abord les mesures avec une fonction de hache cryptographique, par exemple : v3,0 = H (m0),

Supposons que la taille du hachage est plus petite que la taille des données. Pour construire l'arbre de hachage de Merkle, chaque valeur interne de l'arbre est dérivée de ses deux nœuds enfants : vi, j = H (vi+1,2j ||vi+1,2j+1) (où || dénote l'opération de concaténation). Par exemple, pour authentifier la mesure m5, l'agrégateur envoie m5 avec v3,4 , v2,3, v1,0 et m5 est authentique si l'égalité suivante est verfiée :

v0,0 = H (v1,0 || H (H (v3,4 || H (m5)) || v2,3)).

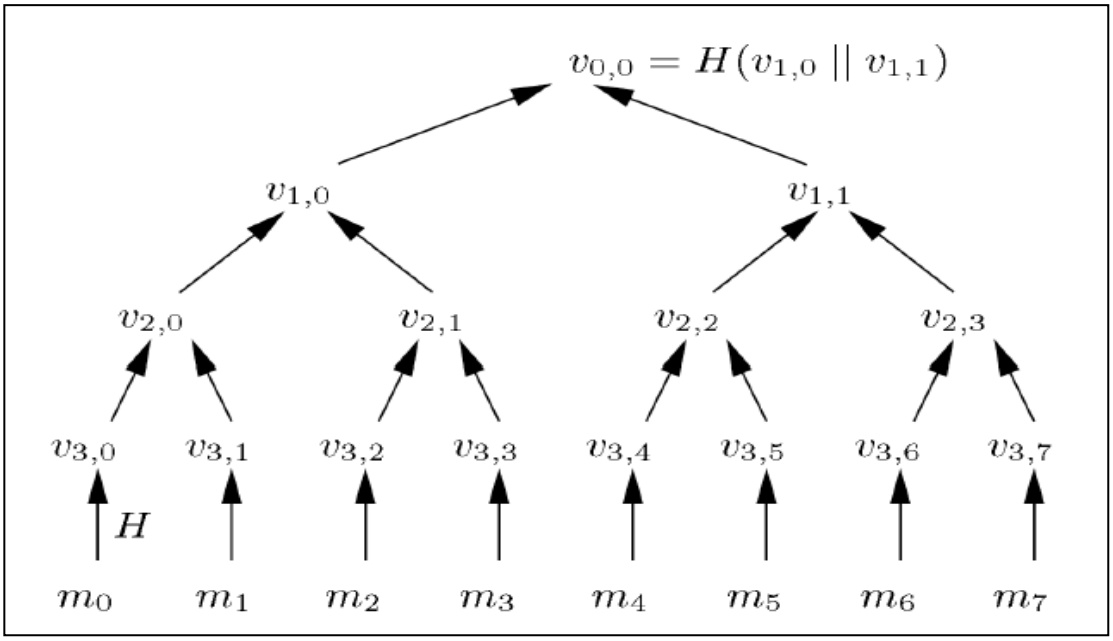


Figure. III.4- Arbre de hachage de Merkle employé pour commettre à un ensemble de valeurs.

❖ ESPDA, Energy-efficient Secure Pattern-based Data Aggregation

Une approche différente est poursuivie dans [23]. Les auteurs proposent un protocole de sécurité d'agrégation de données dans un WSN clustérisé, mais avec une différence principale en ce qui concerne les solutions présentées jusqu'ici : ce protocole n'exécute pas l'agrégation dans la signification stricte, mais choisit plutôt seulement des données non redondantes rapportées par des nœuds capteurs à la tête du cluster, qui l'expédie alors simplement au BS. Par conséquent, il n'y a aucune fonction d'agrégation dans ESPDA, mais juste une réduction de redondance optimisée de chaque donnée du cluster Head.

Spécifiquement, ESPDA emploie des codes de modèle afin d'exécuter l'agrégation. Comme expliqué par ses auteurs, les « codes de modèle sont des données élémentaires fondamentalement représentatives qui sont extraites à partir des données réelles de telle manière que chaque code de modèle ait certaines caractéristiques de la valeur réelle correspondante ». L'extraction de codes de modèle peut varier selon le type de données considéré.

Il est important de souligner que dans le code de modèle, le processus d'extraction à une certaine approximation (dont l'entité dépend de la précision de la génération de code de modèle) est supposé. Par exemple, si deux nœuds capteurs distincts mesurent deux températures légèrement différentes, disons 15°C et 17°C respectivement, le code de modèle correspondant peut être identique si la granularité est plus brute que 2°C.

❖ SELDA, Secure and rELiable Data Aggregation protocol

L'idée fondamentale de SELDA est que les nœuds capteurs observent des actions de leurs nœuds voisins pour développer des niveaux de confiance pour les nœuds voisins[26]. Des mécanismes de surveillance sont employés pour détecter la disponibilité de nœuds et les mauvaises conduites des voisins ; un exemple est montré par la figure III.5.

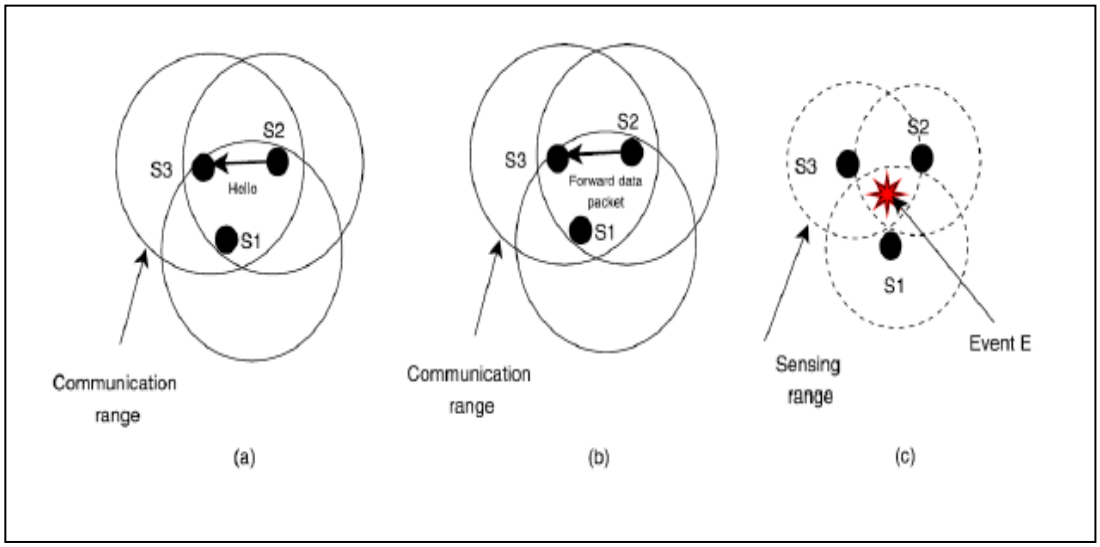


Figure. III.5- Mécanismes de surveillance de SELDA.

- Le S1 détecte la mauvaise conduite de disponibilité du nœud S3, si S3 ne répond pas à des messages de S2 (hello) pendant le temps.
- Le S1 détecte des mauvaises conduites de cheminement de S3, si S3 ne veut pas expédier des paquets de données de S2 correctement.
- L'événement E est détecté par S1, S2 et S3 si l'événement E est rapporté faussement par n'importe qui de ces nœuds, car la mauvaise conduite de détection d'un nœud compromis par la suite est détectée par les deux autres nœuds.

Les informations de confiance sont échangées par chaque nœud avec ses voisins afin de créer un réseau de confiance qui leur permet de trouver les chemins sécurisés et fiables vers des agrégateurs de données.

Grâce aux mécanismes de surveillance, SELDA peut détecter si un agrégateur de données est soumis aux attaques de DOS. Les auteurs réclament que SELDA implique des frais généraux tolérables à la communication, tout en augmentant considérablement la sécurité dans le processus d'agrégation.

III.5.2.2 Sécurité d'agrégation de bout en bout

La deuxième solution consiste à avoir une clé de groupe, au lieu d'une clé avec chaque nœud voisin, avec laquelle les membres du groupe déchiffrent et chiffrent les données jusqu'au puits. Bien que cette méthode réduit le coût de l'établissement des clés entre chaque paire de nœuds voisins, elle est, tout de même, coûteuse en termes de traitement (déchiffrement, agrégation et chiffrement) et la compromission d'un seul nœud révèle la clé du groupe.

Dans les solutions de bout en bout, l'idée est basée sur le fait que chaque nœud chiffre ses données avec une clé qu'il partage avec le puits et avec laquelle les agrégateurs chiffrent des données chiffrées reçues de leurs fils, sans accéder aux données en clair. [33] propose un chiffrement homomorphique par l'addition. L'idée générale consiste en :

1) chaque nœud *i* chiffre ses données *mi*, avec la clé *ki*, qu'il partage avec le puits : *chif (ki, mi)* ;
2) chaque agrégateur exécute cette fonction de chiffrement en utilisant comme paramètres d'entrée les valeurs chiffrées de ses fils : *chif (ki , mi) ⊗chif (kj , mj)*;

3) le puits déchiffre le message avec une clé qui dépend de l'ensemble des clés qu'il partage avec les nœuds et des identifiants des nœuds qui ont participé à l'agrégat.

❖ CDA, Agrégation cachée de Données [33]

CDA est l'une des premières propositions pour un protocole de sécurité d'agrégation de données de bout à bout, il utilise la notion de cryptographie homographique additive et multiplicative.

CDA a tous les avantages principaux des protocoles bout à bout tel que ; aucune opération cryptographique requise au nœud agrégateur, et assure la confidentialité bout à bout dans certaines conditions, mais il souffre, en même temps, des deux limitations graves de clé symétrique de cryptographie homomorphisme (PH) : d'un côté, l'attaquant peut violer la confidentialité du réseau entier par l'attaque de force brute ; de l'autre côté, la puissance d'énergie du PH adopté est considérablement plus haute que celle d'un algorithme traditionnel de RC5.

❖ CDAP [33]

Les auteurs de CDAP observent que la clé symétrique de PH est vulnérable à l'attaque de force brute et, en conséquence, ils proposent de se reproduire à une clef asymétrique PH. Un tel homomorphisme, cependant, est consomme trous de ressources informatique. Par conséquent, les auteurs proposent l'utilisation d'un groupe de nœuds plus puissants, appelés AGGNODEs. À la différence des nœuds capteurs traditionnels, ces nœuds ne sont pas exposés à des limites énergiques strictes et sont commis aux opérations de chiffrage et d'agrégation. Les auteurs déclarent que la proportion d'AGGNODEs requière au-dessus du nombre de nœuds capteurs traditionnels est dans l'ordre de 1 sur 50. CDAP peut être résumé dans les étapes suivantes :

1. AGGNODEs reçoit la clef publique de la BS et le réseau est déployé avec une topologie clustérisée statique.

2. AGGNODEs établit des clefs par paires secrètes avec des nœuds capteurs voisins.

3. Chaque nœud chiffre ses données et les envoie à l'AGGNODE indiqué.

4. AGGNODEs décrypte les données reçues, les agrège et enfin les chiffre avec le PH.

5. Des données cryptées sont expédiées par des nœuds et encore agrégées par des nœuds AGGNODEs sur un autre chemin vers la BS.

En observant cela dans les premières données captées, les nœuds emploient un algorithme de RC5, il est tout à fait évident que compromettant un AGGNODE infirme la confidentialité de toutes les lectures de capteurs envoyées par les nœuds fondamentaux et permet, quoi qu'il

soit, l'injection de fausses données. Les auteurs de CDAP, d'ailleurs, réclament qu'une telle attaque aurait seulement un effet local et, ainsi, c'est un inconvénient acceptable. En référence à la figure III.6 il peut noter que, si AGGNODE2 est compromis, il est autorisé à changer seulement ces données reçues par ses enfants.

D'une part, si AGGNODE1 était compromis et aucun mécanisme pour protéger des données agrégées n'était inclus dans le protocole, les dommages potentiels provoqués par l'attaquant seraient beaucoup plus larges. Cependant CDAP, par la clef asymétrique PH, assure qu'AGGNODEs ne peut pas manœuvrer ou violer la confidentialité des données agrégées reçues. Par conséquent, un AGGNODE compromis peut tricher seulement sur les lectures à un seul capteur reçues par ses enfants.

Au moyen de simulations, il a été montre que, en utilisant au moins 6 AGGNODEs sur 100 nœuds traditionnels, CDAP est bien plus efficace qu'un protocole de sécurité d'agrégation traditionnelle de hop-by-hop.

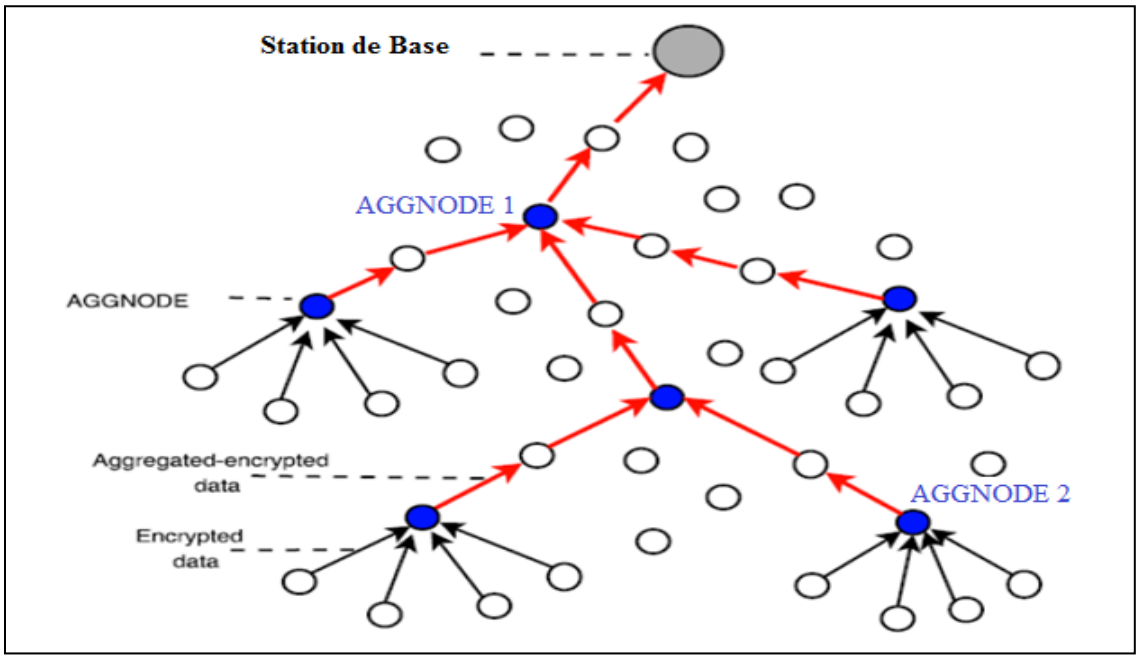


Figure. III.6- : Exemple de l'arbre d'agrégation de CDAP.

❖ EAED, Efficient Aggregation of Encrypted Data in WSNs

La dernière proposition examinée est celle de Castelluccia et autres [33]. Le protocole est très semblable à CDA, mais il est basé sur un PH différent, c.-à-d. un chiffrement de flux homomorphie additif.
L'idée fondamentale est celle de remplacer l'opération de XOR, traditionnellement utilisé, par une addition modulaire simple. Dans le suivant, ce nouveau PH est défini:

1) Initialement : choisir un nombre M plus grande que tous les valeurs qui peuvent être mesuré plu tard.
2) Cryptage d'une donnée :
- On a un message : m € [0, M-1].
- Soit k une clé aléatoirement générée tel que : k € [0, M-1].
- Calcule : C=Enc(m,K,M)= (m+k) modulo M.

3) Decryptage d'une donnée :
- Dec(C, K, M) = (C-K) modulo M.

4) Opération d'agrégation sur une donnée cryptée :
- Soit C1=Enc(m1,K1,M) et C2=Enc(m2,K2,M).
- Donc on obtient le résultat agréger comme suite : K=K1 + K2, C=C1 + C2 ;

Figure.III-7 : cryptographie de Castelluccia Mykletun Tsudik (CMT) [33].

Il faut noter que si n messages cryptés sont additionnés, alors M doit être plus grand que $\sum_{i=1}^{n} mi$, autrement l'exactitude n'est pas assurée, en fait si $\sum_{i=1}^{n} mi$ est plus grand que M le déchiffrage aura comme conséquence une valeur m\ qui est plus petite que M.

III.5.3 Sécurisation du routage

Pour plus d'efficacité énergétique, le routage des données vers le puits nécessite l'accès immédiat des nœuds aux données routées pour les compresser, les agréger ou supprimer les duplications. Cela permet aux attaquants, qui peuvent être plus puissants que les nœuds du réseau, de brouiller, de modifier, de supprimer ou d'accéder aux données, ou encore d'injecter des informations de routage malicieuses et de dévier le routage de ces données pour leurs propres intérêts.

III.6. Les outilles de sécurité de base dans les WSN

III.6.1 La cryptographie

Il existe deux types :

III.6.1.1 Cryptographie symétrique

La cryptographie symétrique également dite à clé secrète est la plus ancienne forme de cryptographie. Elle nécessite pour fonctionner que les deux parties en présence est au préalable échangées une clé commune connue seulement entre-deux : un émetteur et le récepteur.

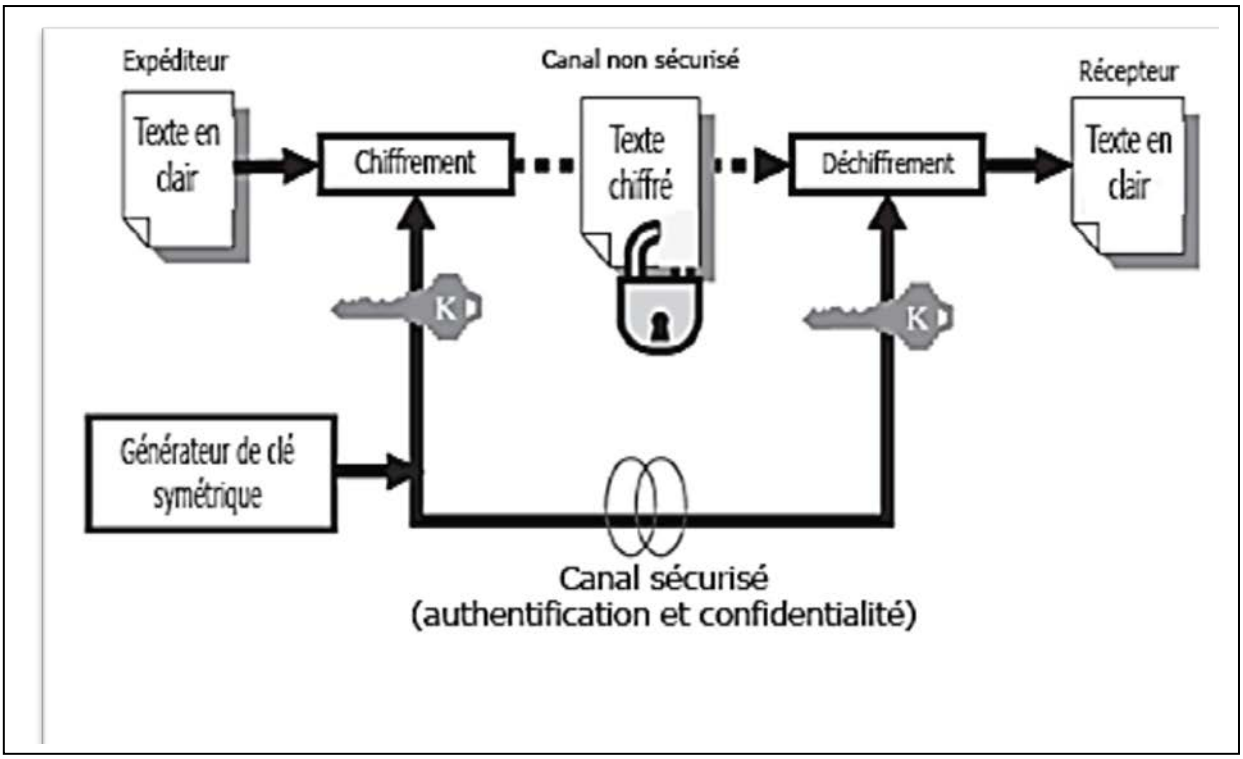


Figure- III.8. Chiffrement symétrique [9].

III.6.1.2 Cryptographie asymétrique

La cryptographie asymétrique, également appelée cryptographie à clé publique, repose sur l'utilisation de deux clés différentes pour chiffrer/déchiffrer. Ces deux valeurs de clés sont certes différentes, mais reliées entre elles : une clé publique (qui est diffusée) et une clé privée (gardée secrète), l'une permettant de chiffrer un message et l'autre de le déchiffrer. Ainsi, l'expéditeur utilise la clé publique du destinataire pour chiffrer un message que seul le destinataire (en possession de la clé privée) peut déchiffrer, garantissant la confidentialité du contenu.

Inversement, l'expéditeur peut utiliser sa propre clé privée pour signer un message et le destinataire peut vérifier la signature du message à l'aide de la clé publique correspondante.
Ce dernier mécanisme permet de faire de la signature numérique pour authentifier l'auteur d'un message. La figure III.9 illustre le mécanisme d'un chiffrement fondé sur la cryptographie asymétrique.
La cryptographie asymétrique permet donc d'une part de ne pas pré-échanger une clé entre deux personnes souhaitant communiquer entre elles, de garantir la confidentialité des données en chiffrant et déchiffrant des messages et de signer des messages vérifiables par tout le monde afin de garantir la non-répudiation d'une donnée.

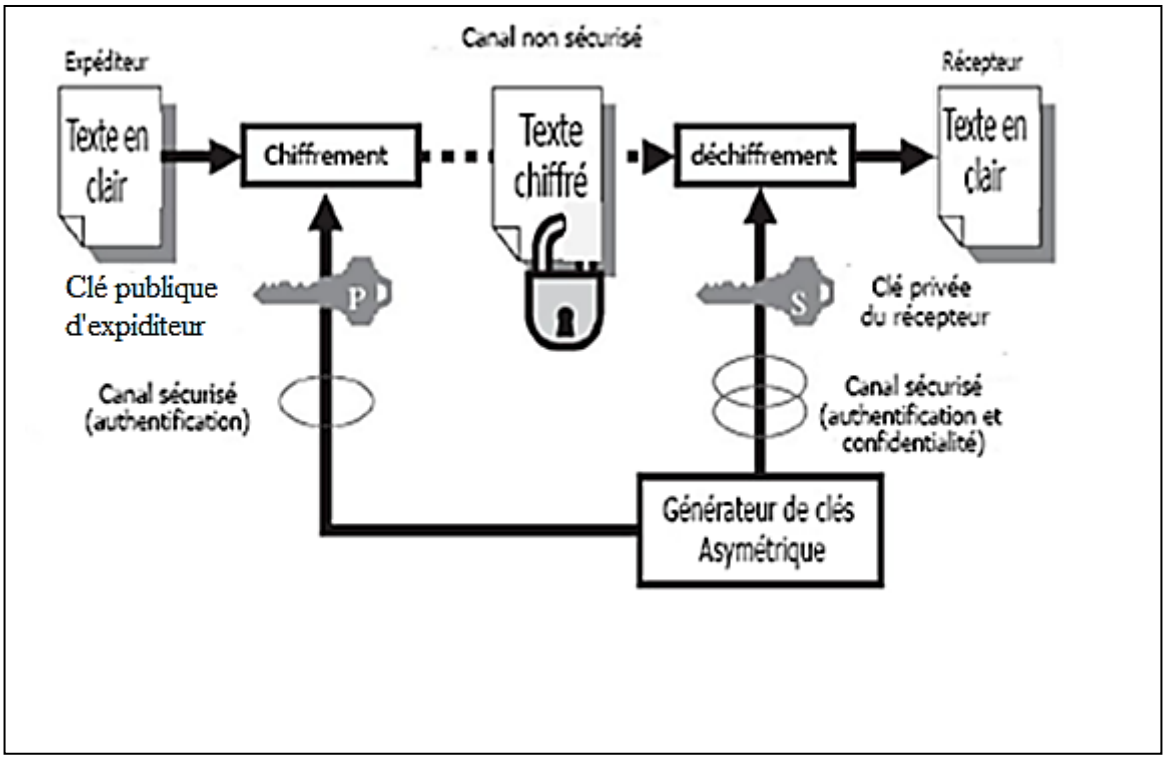


Figure- III.9. Chiffrement asymétrique [9].

Nous donnons ici quelques exemples d'algorithmes de chiffrement à clé publique :

- **RSA (Rivest Shamir Adleman)**

Le principe de RSA est basé sur la difficulté de décomposer un grand nombre en ses éléments premiers. Les tailles de clés de RSA recommandées aujourd'hui pour répondre aux exigences de sécurité sont de minimum 1024 bits (taille à la fois de la clé publique et de la clé secrète). Notons également qu'il est possible de construire un algorithme de signature fondé sur le même problème[25].

- **ECC (Elliptic Curve Cryptography)**

ECC se base sur la difficulté d'inverser le logarithme discret sur une courbe elliptique. Comme pour RSA, la cryptographie fondée sur les courbes elliptiques permet de construire des algorithmes de chiffrement et des schémas de signature.

Un des avantages majeurs de l'utilisation de cette forme de cryptographie est la faible taille des clés utilisées. En effet utiliser une clé de 160 bits dans un algorithme fondé sur ECC est équivalent à une clé RSA de taille 1024 bits. Donc ce genre d'algorithmes asymétriques attire l'attention des concepteurs de systèmes contraints et est également intéressant dans le cadre des réseaux de capteurs sans fil. Tel qu'ECC exige une puissance de calcul inférieur à celle de RSA [25].

III.6.2 Symétrique vs Asymétrique dans WSN

Chacune de ces deux formes de cryptographie possède ses propres avantages et inconvénients. Pour le cas des réseaux de capteurs, qui sont des réseaux contraints, la cryptographie symétrique est souvent privilégiée par rapport à la cryptographie à clé publique pour les deux raisons suivantes [29] :

- la vitesse de traitement des données est plus importante dans le cas d'un chiffrement symétrique, ce qui conduit à une consommation énergétique et un temps de calcul beaucoup moins important.
- La taille des clés et des paquets à chiffrer ou à signer est nettement inférieure pour un même niveau de sécurité dans le cas de la cryptographie symétrique.

Ces deux justifications sont cruciales pour un environnement comme un réseau de capteurs. C'est pour cette raison que la majorité de solutions de sécurité pour les WSNs se base sur la cryptographie symétrique. Néanmoins, la cryptographie asymétrique présente quelques avantages qui sont :

- Seule la clé secrète a besoin d'être conservée de manière secrète, ce qui n'est pas le cas dans le cadre de la cryptographie symétrique ou chaque couple de nœuds doit partager une clé. Dans ce dernier cas, les schémas de partage de clés sont donc plus compliqués à mettre en œuvre.
- La cryptographie à clé publique permet la gestion de certificats et donc permet de prouver les identités des personnes en présence contrairement à la cryptographie à clé secrète.

III.6.3 Fonction de hachage

Une fonction de hachage cryptographique consiste en général à l'application d'une fonction de compression à sens unique sur un bloc de données de taille quelconque pour générer une sortie de taille fixe n bits. La valeur n représente le degré de sécurité de la fonction de hachage. La fonction de hachage doit être en général facile à calculer et connue publiquement, mais très difficile à inverser.

Les fonctions de hachage sont généralement utilisées comme première étape pour vérifier l'intégrité d'un message ou bien pour générer des signatures numériques. En effet, imaginez que vous souhaitiez envoyer un fichier par mail, mais que ce fichier est de taille importante. Vous souhaitez de plus rassurer le destinataire sur la provenance de ce fichier et sur son contenu. Plutôt que de chiffrer votre fichier directement avec votre clé privée, vous allez hacher votre fichier et chiffrer le condensé obtenu avec votre clé privée. Vous enverrez ensuite votre fichier original ainsi que le condensé signé (la signature) à votre destinataire. Celui-ci va, lors de la réception, vérifier l'exactitude de la signature en vérifiant la signature du haché et la validité des données.

Une fonction de hachage cryptographique doit répondre aux trois critères suivants :

- Sens unique : quand la valeur hachée H = h(M) pour un message M est donnée, il est difficile de retrouver le message M ;

- Forte résistance aux collisions : il est difficile de trouver deux messages distincts M et M0 tel que h(M) = h(M0) et à partir d'un message M, de son haché H = h(M) et du code de la fonction de hachage, il est difficile de trouver un message M0 tel que

h(M0) = h(M).

Des exemples de fonctions de hachage sont la famille MD comme MD2, MD4 et MD5 et la famille SHA comme SHA-1 et SHA-2.

III.6.4 Codes d'authentification de message ou MAC

Un code d'authentification de message ou MAC peut être vu comme une signature bipartie en cryptographie symétrique : il s'agit de garantir d'une part l'intégrité d'un message M envoyé et d'autre part de vérifier l'identité de l'émetteur pour la personne possédant la clé symétrique partagée.
En effet, un MAC consiste à utiliser une fonction de hachage cryptographique combinée à une clé secrète connue seulement par les deux entités échangeant le message. Un MAC est donc un algorithme qui prend en entrée un message M, une clé K et qui produit un condensé ou ce qu'on appelle un tag : tag = MACK(M).
L'algorithme de vérification pour le receveur consiste à vérifier si pour le message M0 reçu on a bien tag =? =MACK(M0). Ainsi, si la relation est vérifiée, le receveur a bien la preuve que le message n'a pas été modifié au cours de la transmission et que l'émetteur du message est bien celui qui connaît également K [15].

III.7. Conclusion

Comme conclusion à ce chapitre, nous donnons d le tableau suivant qui résume les propriétés des protocoles cités précédemment, en termes d'exigences de sécurité vérifiées.

Protocoles	Confidentialité	Authentification	Intégrité	Freshnes	Disponibilité
SecureDAV	Oui	Oui	Oui	Non	Non
ESPDA	Oui	Oui	Oui	Oui	Non
SELDA	Non	Oui	Oui	n.d.	Oui
CDA	Oui	Non	Non	Non	Non
CDPA	Oui	Non	Non	Non	Non
EAED	Oui	Non	Non	Non	Non

Tableau III.1 : Exigences de sécurité assurées par les protocoles présentés [22].

En référence à ce tableau, il est important de souligner quelques points :

• Le niveau de sécurité réalisé par chaque protocole en ce qui concerne des exigences de sécurité n'est pas le même pour toutes les propositions ; en particulier, on peut assumer que la confidentialité bout à bout est une propriété plus forte en comparaison avec hop-by-hop. Généralement les exigences de sécurité assurées devraient être toujours évaluées dans le cadre du protocole considéré.

• Les protocoles bout à bout assurent seulement la confidentialité bout à bout de données ; l'accomplissement d'autres exigences de sécurité est laissé aux différents protocoles.

• Aucun des protocoles proposés, excepté SELDA [22], n’a traité la condition de disponibilité (et donc la protection contre des attaques de DOS). Même s’il pouvait apparaître comme faille de sécurité sérieuse, on devrait observer que, en raison des limites de transmission de puissance des nœuds de WSN, un attaquant peut bloquer toutes les communications dans le réseau sans grands efforts. Par conséquent, si un adversaire vise à perturber tous les services du réseau par une attaque de DOS, nous pouvons supposer qu'aucun mécanisme de sécurité ne peut protéger le réseau contre ses menaces. C'est la raison pour laquelle, il n'est pas très important de rechercher la disponibilité (prévue comme protection contre des attaques de DOS) dans ces protocoles.

Chapitre IV:

Proposition d’une solution pour la sécurité d’agrégation de données.

IV.1 Introduction

Dans ce chapitre nous présentons notre solution portant sur la sécurité d'agrégation de données qui repose sur l'authentification de la source du message reçu, autrement dit : si on a des messages des nœuds capteurs qui sont authentifiés donc l'opération d'agrégation sera faite de manière fiable, sinon il y a un risqué une d'attaques sur l'agrégation de données.

Pour cela nous commençons ce chapitre par une illustration des dommages causés par une attaque sur l'agrégation, ensuite nous donnons les étapes de cette attaque et nous terminons ce chapitre par la présentation de la solution proposée pour ce problème.

IV.2 Les dommages d'une attaque sur l'agrégation de données

Les dégâts qui peuvent arriver sur l'agrégation de données, c'est qu'un nœud malicieux peut attaquer le réseau en injectant de fausses données (figure IV. 1) ou en falsifiant le résultat d'une opération d'agrégation (figure IV.2). Dans ce cas, le nœud malicieux réussira à falsifier l'information captée dans toute une partie du réseau WSN, par conséquent tout le travail réalisé par les capteurs sera perdu

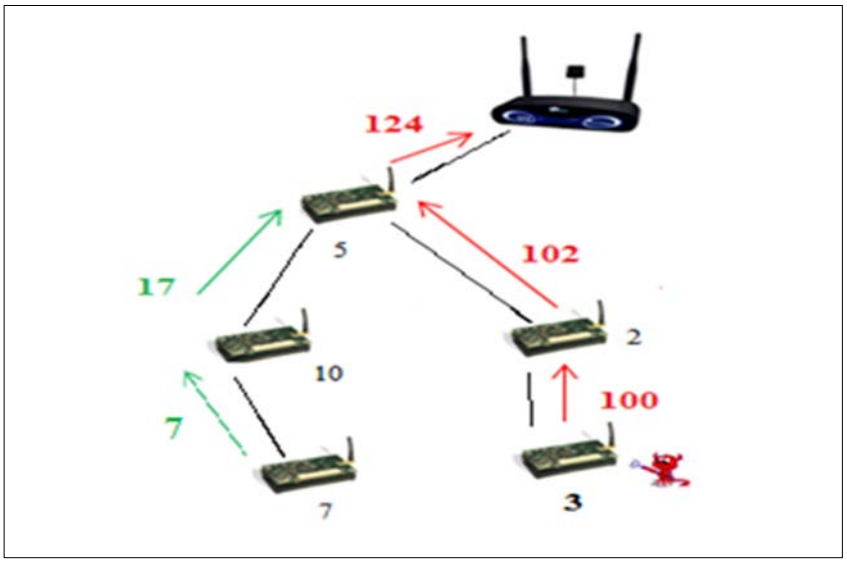


Figure. IV.1- un nœud malicieux injecte de fausses données.

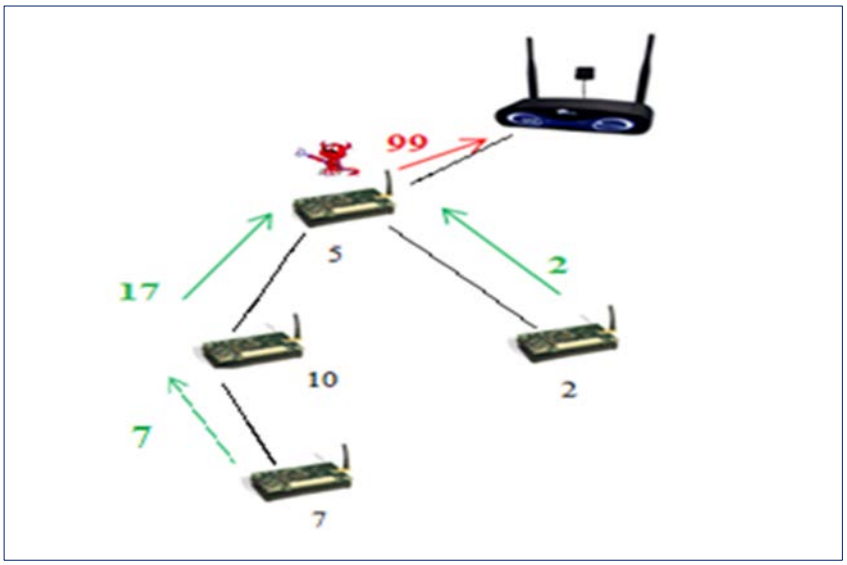


Figure. IV.2-Un nœud malicieux falsifie le résultat d'une agrégation.

IV.3 Les étapes d'une attaque d'agrégation de données

Une attaque sur l'opération d'agrégation de données commence avec une attaque passive telle que l'intrus ou le nœud malicieux essaie de comprendre l'architecture du réseau et de déceler les points faibles et les ressources importantes à attaquer (nœuds agrégateurs). Un autre but de cette attaque est que l'intrus intercepte les messages du réseau WSN et tente de tirer les informations pertinentes (mots de passes, informations confidentielles… etc.).

Cette attaque ne nuira pas le fonctionnement du réseau, mais c'est une étape importante pour créer une attaque active qui tente essentiellement dans notre cas d'étude d'injecter des données erronées (figure IV.1) ou de falsifier le résultat d'agrégation (figure IV.2).

On rappelle que plusieurs types d'attaques telles que : l'attaque sybile, l'attaque misdirection et l'attaque black holes… etc. (chapitre III), sont classées comme des attaques actives menaçant l'agrégation de données.

IV.4 Sécurité contre l'attaque de type passive

On rappelle que l'attaque passive est une interception des messages entre les nœuds, dans le but de comprendre la signification des données échangées. Pour résoudre ce problème, on va utiliser la cryptographie homomorphisme. de type asymétrique (c.-à-d. une clé de chiffrement et une autre clé pour le décryptage).

Dans ce cas, un nœud fils chiffre le message avec une clé A, et le transmet à son père, mais le père crypte aussi son message avec une autre clé B et agrège les deux messages sans comprendre la signification du message issu de son nœud fils.

IV.4.1 L'algorithme de cryptographie homomorphisme

Utilise cet algorithme, l'agrégateur à la possibilité d'effectuer des calculs sur les données malgré leur signification inconnue. Supposons que les nœuds *i* et *j* mesurent les valeurs *ri* et *rj*. Un nœud agrégateur peut additionner deux parts provenant de *i* et *j*, cette somme correspond à une nouvelle donnée qui code la valeur *ri* + *rj*. Cela est vrai pour un certain nombre de fonctions d'agrégation, par exemple la somme, la moyenne, le maximum, le minimum et le comptage.
Comme exemple pour les algorithmes de cryptographie homomorphisme nous avons :

- **Algorithme de Cryptographie à Courbe Elliptique(CCE)**

Algorithme de cryptographie à courbe elliptique (CCE)

Soit Karim veut envoyer un message à Nassim :

Choix des clés public :

Karim choisie un nombre entier **P** très grand et doit être premier, puis l'envoyé à Nassim.

Nassim reçoit **P** de Karim et choisie aussi un nombre **S** secret, puis il calcule : **B=S*P**, et transmet **B** à Karim

Donc **P** et **B** sont des clés publiques et **S** est une clé secrète de Nassim.

Cryptage :

Pour que Karim envoie un message vers Nassim :

1- Karim choisie un nombre entier qui doit être **secret** : **K** et calcule : $\mathbf{M_1 = K*P}$.
2- Calculer : $\mathbf{M_2 = Données + K*B}$.
3- Il envoie $\mathbf{M_1}$ et $\mathbf{M_2}$ à Nassim.

Décryptage :

Nassim déchiffre le message en calculant **:** $\mathbf{Données = M_2 - S*M_1}$.

Aggregation: $\mathbf{C= (M_{1,l} + M_{1,j}\ ,\ M_{2,l} + M_{2,j}) = (M_{1,1}\ ,\ M_{2,2})}$.

Figure. IV.3- Algorithme de CCE homomorphisme [25].

Remarque :
On rappelle que le CCE est un algorithme de cryptographie de type asymétrique, économique en pour la consommation de l'énergie, il nécessite de petite clés pour assurer un niveau de sécurité très élevée (figure. IV.4).

RSA	CCE
512	112
1024	160
2048	224
3072	256
7680	384
15360	512

Figure IV.4- comparaison la taille de clés assurent le même niveau de sécurité.

- **L'algorithme de cryptographie de Castelluccia Mykletun Tsudik (CMT)**

1) Initialement : choisir un nombre M plus grande que tous les valeurs qui peuvent être mesuré plu tard.
2) Cryptage d'une donnée :
- On a un message : m € [0, M-1].
- Soit k une clé aléatoirement générée tel que : k € [0, M-1].
- Calcule : C=Enc(m,K,M)= (m+k) modulo M.
3) Decryptage d'une donnée :
- Dec(C, K, M) = (C-K) modulo M.

4) Opération d'agrégation sur une donnée cryptée :
- Soit C1=Enc(m1,K1,M) et C2=Enc(m2,K2,M).
- Donc on obtient le résultat agréger comme suite : K=K1 + K2, C=C1 + C2 ;

Figure IV.5-algorithme de CMT homomorphisme [33].

Le CMT est classé comme un algorithme de cryptographie de type symétrique, il aussi économique en pour la consommation de l'énergie.

IV.5 Description de notre approche de sécurité sur l'attaque de type active

Notre solution de sécurité pour faire face à toutes les attaques qui menacent l'opération d'agrégation de données, est d'utilisé une nouvelle méthode, pour authentifier les messages reçus avant de les agrégées. Cette méthode se base sur la sauvegarde du contenu historique des données hachées (tags), tel que chaque capteur enregistre dans sa mémoire les tags des données reçues et envoyées. De cette façon, chaque capteur crée une partie historique des tags, qui font partie de son passé. Ensuite, les capteurs vont utiliser ces tags pour prouver l'identité des messages reçus, avant de les agréger.

Si un nœud fils veut transmettre un paquet de données à son nœud père, il choisit aléatoirement deux tags (de données hachées) de sa mémoire tampon, ensuite ils seront agrégés par l'opération de Xor pour avoir un seul tag :

Tag=Tag1 **xor** Tag2 ;

Ensuite, on va produire une signature numérique pour le message à envoyer dans le but de garantir que le message reçu (par l'autre nœud récepteur) n'est pas falsifié durent le routage par un nœud intrus, pour cela on va taguée : la donnée agréger et l'identifié du capteur émetteur du paquet, après on agrège tous les tags obtenus pour avoir un seul tag :

TAG de signature numérique = Tag **Xor** hachage(ID) **Xor** hachage (données agrégé) ;

tel que : Tag=Tag1 **Xor** Tag2, et ID est l'identifiant du nœud source du paquet.

Ce tag obtenu va être ajouté au paquet de données pour l'envoyer au nœud père. Et de l'autre côté, le nœud père qui reçoit ce paquet issu d'un nœud fils, tire le tag à partir du paquet reçu et essaie de reproduire un autre tag à partir de son propre tampon mémoire et le paquet reçu.

Puis il compare les deux tags (reçus et reproduit), s'ils sont similaires alors la source du paquet reçu est authentifiée (ou fiable) et le paquet n'est pas modifié, donc la donnée du paquet obtenue sera agrégée, hachée et enregistrée dans le tampon pour la future authentification sinon le paquet reçu est indésirable (intrus) par conséquent la donnée reçue ne peut pas être agrégé.

Les figures suivants, nous montrent le fonctionnement de notre système pour les deux capteurs émetteur du paquet 'i' et récepteur de ce paquet ;

IV.5.1 Capteur émetteur du paquet

IV.5.1 .1 Etape 1 : Production du TAG final (signature numérique) et envoie du paquet.

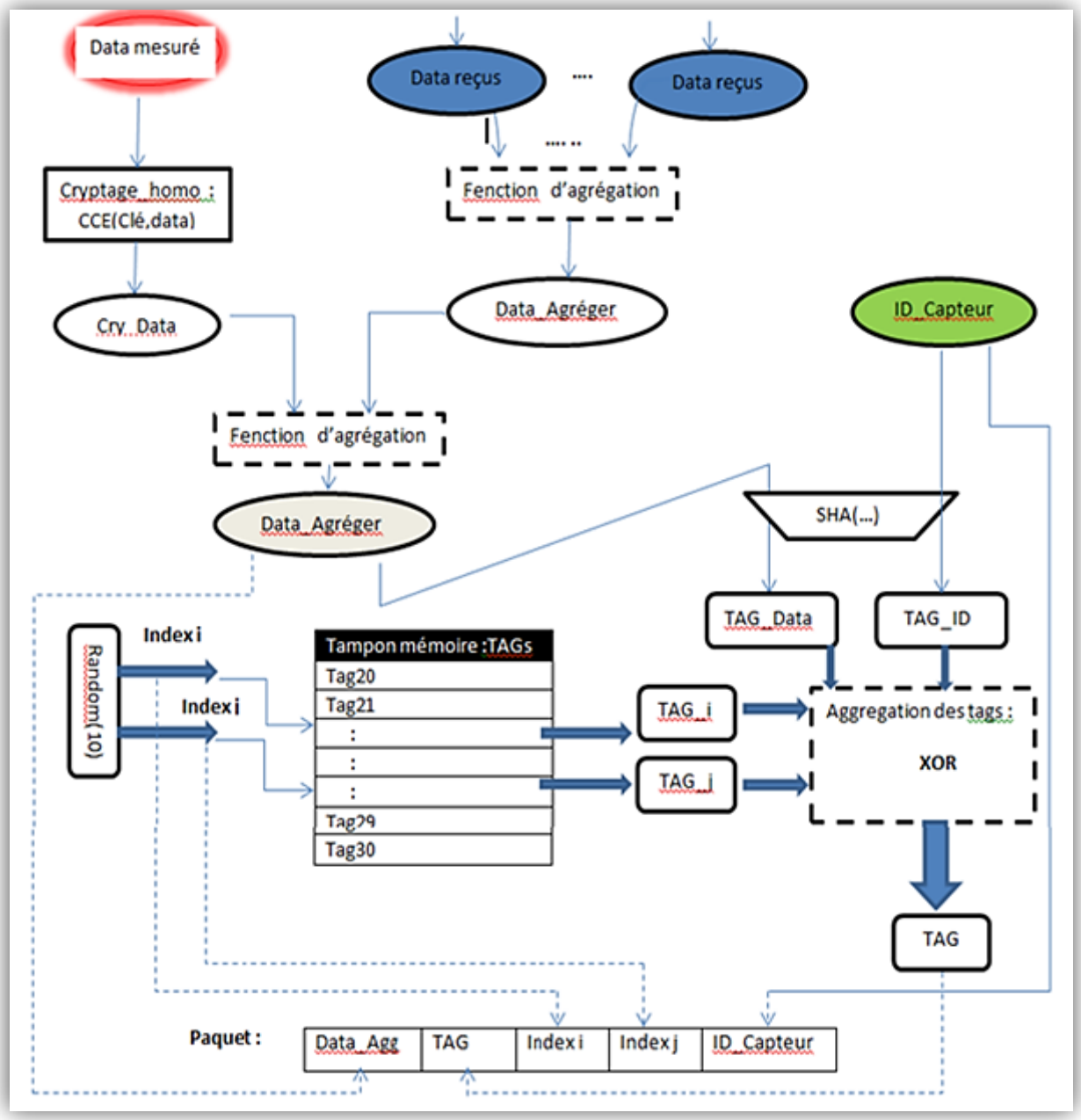


Figure IV.6-oganigramme de production de TAG, et envoie du paquet.

IV.5.1 .2 Etape 2 : la mis à jour du tampon mémoire des tags

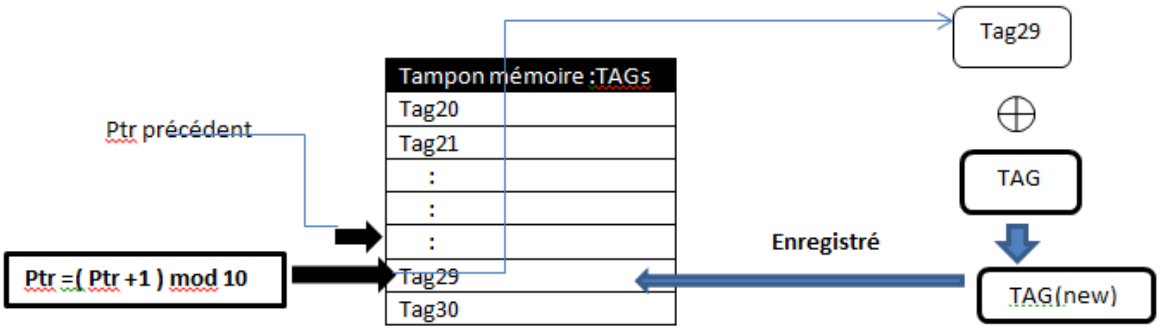


Figure IV.7-Mettre à jour le tampon mémoire des tags.

IV.5.2 Capteur récepteur du paquet

IV.5.2.1 Etape 1 : vérification de l'authentification du message reçus

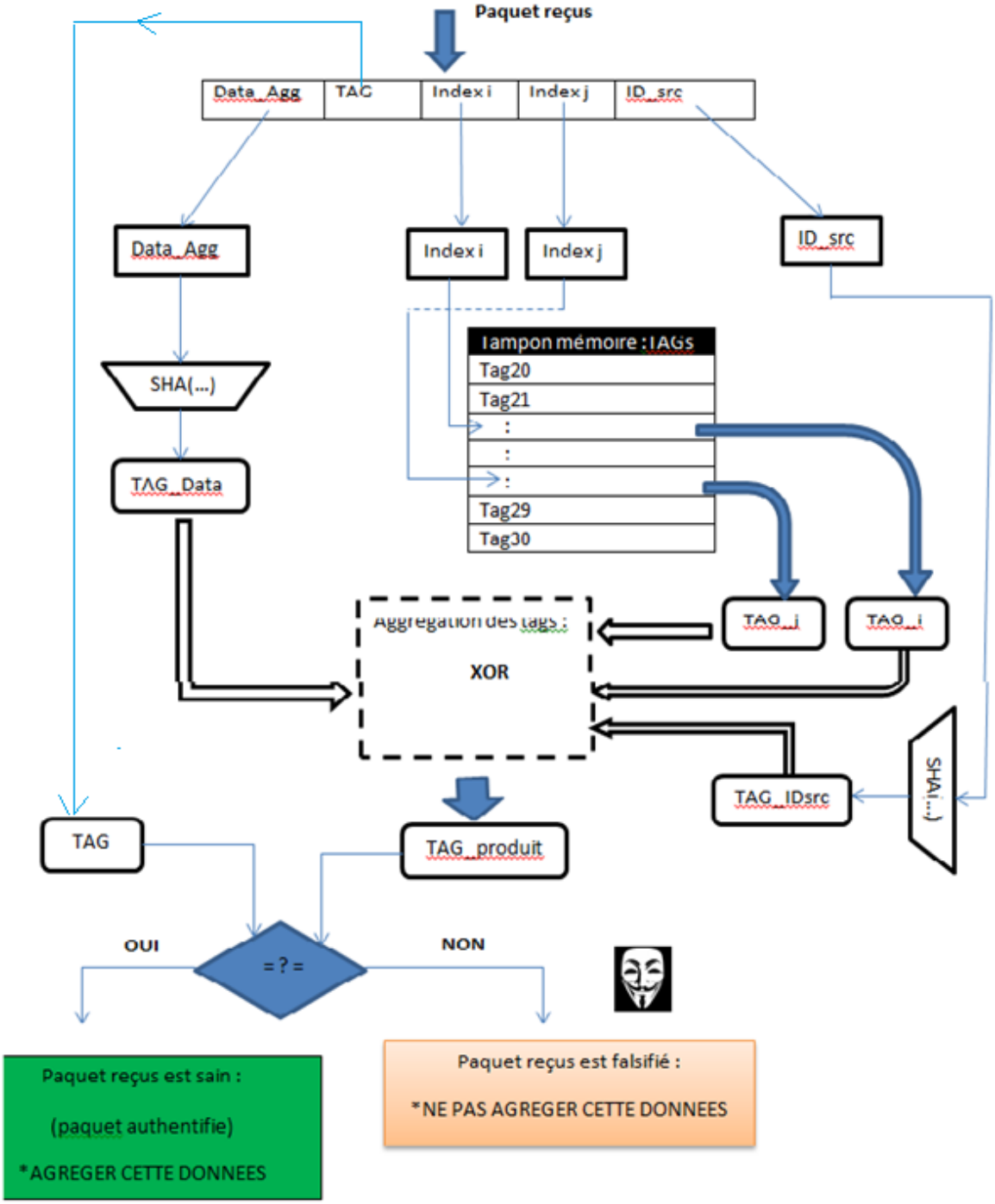


Figure IV.8- organigramme de vérification de l'authentification du message reçus.

IV.5.1 .2 Etape 2 : la mis à jour du tampon mémoire des tags, seulement si le paquet reçus est authentifie

Cette étape est identique à celle du capteur émetteur du paquet, mais la mise à jour du tampon mémoire se fait uniquement dans le cas où le paquet reçus est correct (non falsifier).

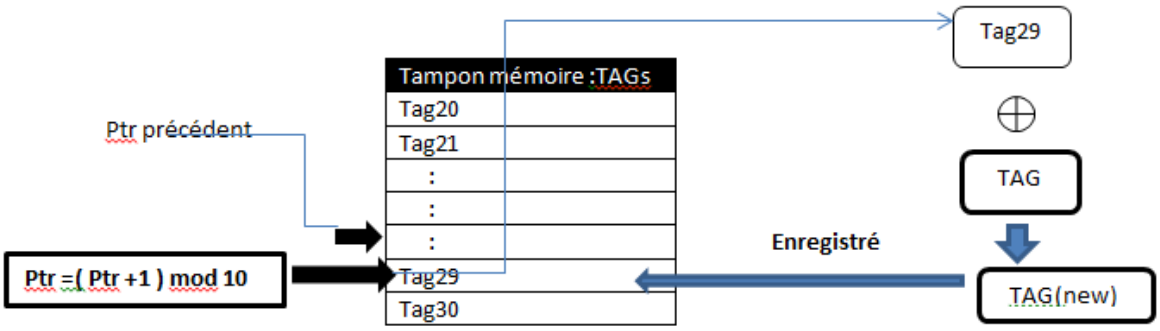


Figure IV.9-Mettre à jour le tampon mémoire des tags.

IV.5.2 Implication sur la sécurité

Un nœud malicieux qui arrive à l'instant *t+1* pour essayer d'injecter ou de falsifier les données, il sera obligé d'ajouter les données (c.-à-d. hachage) qui font partie de son passé (instant *t*) dans le message à transmettre, ce qui est impossible, car le nœud malicieux ne connaît pas tous les paquets déjà échangés.

De cette manière on vérifie l'authentification pour chaque nœud par son nœud père, c.-à-d. dans tout le réseau WSN chaque nœud père est responsable d'authentifie ses nœuds fils à chaque fois qu'ils émettent des messages dans le réseau.

Mais n'importe quelle technique utilisée possède des avantages et des inconvénients. La connaissance de ce(s) inconvénient(s) nous permettra d'avoir une meilleure réalisation de cette technique.

IV.5.3 Inconvénient de notre approche

On remarque bien que cette technique d'authentification repose essentiellement sur la ressource mémoire du capteur pour enregistrer les haches des données, ce qui nous permet de dire que si on a une architecture réseau clustérisée, cette méthode d'authentification devient inutile, car la mémoire est une ressource critique dans un capteur, et le cluster HEAD ne peut pas supporter de sauvegarder toutes les données issues des nœuds de son cluster.

Pour résoudre ce problème on va choisir une topologie arborisent du réseau WSN de type arbre binaire.

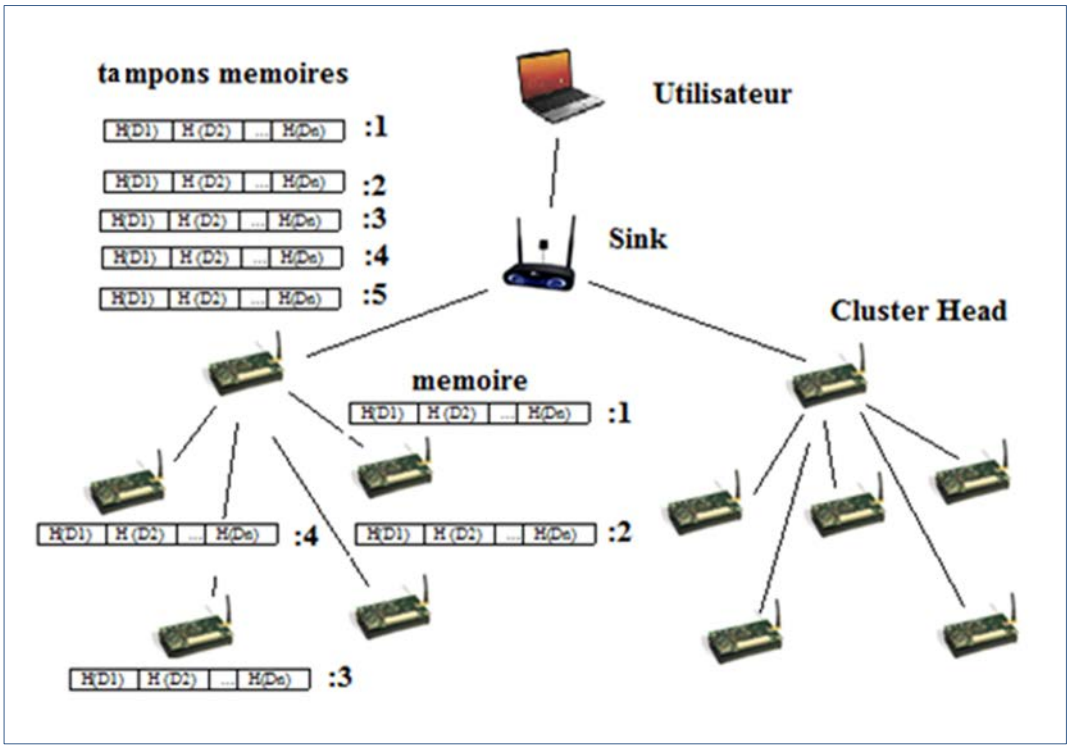


Figure. IV.10- saturation de mémoire dans notre solution d'authentification historique.

IV.5.4 La topologie d'arbre binaire du réseau WSN

Notre topologie sera un arbre binaire (celle utiliser pour la gestion de la structure de données), c'est un arbre qui possède un nœud principal nommé : racine ou root, est pour chaque nœud il aura au plus deux fils (fils gauche et droit).

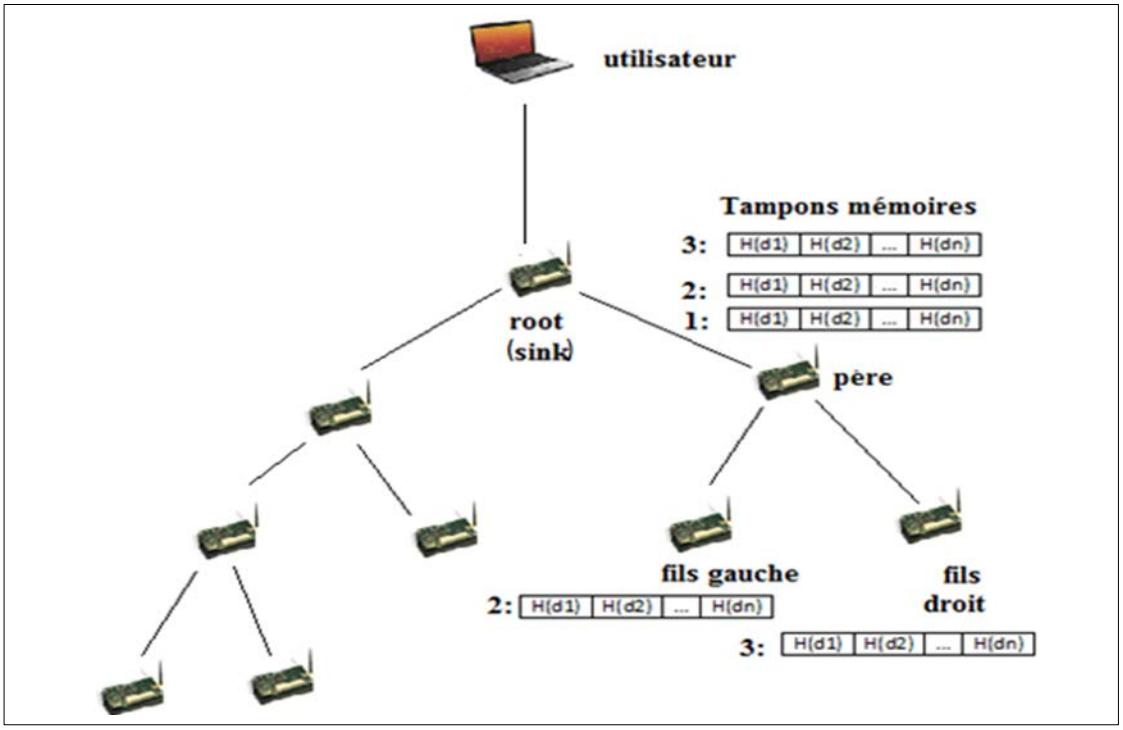


Figure. IV.11- topologie arbre binaire.

On remarque bien dans cette topologie, la présence de nœud : ROOT ou le SINK, on rappelle que le nœud ROOT son rôle principal est de gèrer les communications entre le réseau WSN et l'utilisateur, mais la tâche importante du nœud ROOT est la gestion de l'arbre : insertion de nouveaux nœuds et la maintenance de l'arbre en cas de problème.

IV.5.5 Algorithme de production de signature numérique et d'authentification du paquet reçus.

Pour présenter le fonctionnement de notre solution d'une façon algorithmique pour les figures : IV.6, IV.7, IV.8 et IV.9, on va supposer que le capteur X (fils) veut transmettre des données au capteur Y (père), et on met aussi la taille du tampon mémoire pour enregistrer les hachages des données agrégées de taille 10.

Pour cela les deux capteurs vont exécuter une étape commune, c'est l'initialisation de variables.

Algorithme d'initialisation (Capteur X (fils) & Capteur Y (père)) :

For (i=0 ; <10 ; i++) Tampon[i]=0xff; // tableau pour les données hacheés

Int Ptr=0 ; // pour gérer le tampon de la mémoire d'une façon cyclique.

String Buffer [5] ; // tableau de données à envoyer dans un paquet

DonnéesCryptéReçus=0 ;

Ensuite, si le nœud fils veut transmettre un paquet de données au nœud père, il doit exécuter les étapes suivantes :

Algorithme pour le capteur X (fils) :

If (capteurFils_2_veut_Envoyer_Data) {

Données_crypté =Cryptographie_homo(Data_temperature, Clé); // utilisent CCE ou CMT

Int Données_crypté _Agreger= Données_crypté + DonnéesCryptéReçus ; // agrégation de données ;

P=0 ; // variable

Tante que (P < NbrPaquet Reçus) alors { // agrégation de tous les données reçus existants

Données_crypté _Agreger = Données_crypté _Agreger +DonnéesCryptéReçus ;

P=P+1 ;

} fin tant que ;

TagDonnéesCrypter= hachage (Données_crypté _Agreger) ; // par le SHA-1

Tampon [Ptr]= TagDonnéesCrypte;

Ptr= (Ptr + 1) mod 10 ;

Int index_i= Random(10) ; // index du tag i

Int index_j= Random(10) ; // index du tag j

Int tag_i=tampon[index_i];

Int tag_j=tampon[index_j];

Int TagIdCapteur=hachage (ID_capteur) ;

Int SignatureNumerique= TagIdCapteur **Xor** TagDonnéesCrypter ;

Int TAG=tag_i **Xor** tag_j; // agrégation des deux tags

Int **TAGgenerale**=TAG **Xor** SignatureNumerique ;

Buffer[0]= Données_crypté;

Buffer[1]= TAGgenerale;

Buffer[2]=index_i;

Buffer[3]=index_j;

Buffer[4]=ID_capteur;

SendTo(Addr_NoeudPére, &Buffer);

}

Le tag général produit par le hachage, il permet d'authentifier la source du capteur émetteur du paquet, et de l'autre côté, il nous assure l'intégrité du message, c.-à-d. Que le message n'est pas altéré par un autre nœud intrus durant le processus du routage.

Enfin, le nœud récepteur vérifie l'authentification du paquet reçu avant d'agréger la donnée reçue, suivant cet algorithme:

Algorithme pour le capteur Y (père) :

if (ReceiveFrom(Addr_NoeudFils, &Buffer) == TRUE) {

Int Données_cryptéReçus=Buffer [0]; // la donnée mesurée du nœud fils

Int IDsourcePqt=PaquetReçus.IDsrc ; // récupérer Id source du paquet reçu

Int TAGReçus=Buffer [1]; // le hachage : tampon historique + signature numérique.

Int indexIreçus=Buffer [2]; // index du tag i

Int indexJreçus=Buffer [3]; // index du tag j

```
Int tag_i=Tampon [indexIreçus];

Int tag_j=Tampon [indexJreçus];

Int Tag= tag_i Xor tag_j;

Int TagIDsrc= hachage (IDsourcePqt) ; // utilisent le SHA-1

Int  TagDonnéesCrypterReçus= hachage (Données_cryptéReçus) ;

Int TAGReproduit=tag_i Xor tag_j Xor TagIDsrc  Xor TagDonnéesCrypterReçus;

If (TAGReçus == TAGReproduit) then { // authetification du paquet reçus à 100%

/* la source du message reçu est authentifiée : le message sera agrégé plus tard par le nœud père */

Ptr= (Ptr + 1) mod 10;

Tampon [Ptr]= hachage (Données_cryptéReçus) ; // mettre à jour la mémoire tampon

// ##### Agréger les données à nouveau …. #####

} sinon {   //##### une attaque est détectée sur agg de données #####

    /*  les mesures à apprendre :

    ▪ Ce paquet reçu ne sera pas agrégé avec les autres paquets sains.
    ▪ Envoyer un rapport  à la station de base (SINK)
    ▪ Ne pas accepter les paquets, qui viendront plus tard par ce nœud détecté (intrus).

  */

}

}
```

IV.6 Conclusion

Dans ce chapitre on a montré essentiellement notre approche pour sécuriser l'agrégation de données dans les réseaux WSN, c'est l'authentification des sources des nœuds basés sur le contenu historique pour un nœud, est ce contenu historique est obtenue par le hachage des données brutes au niveau des nœuds simples ou par des données agrégées au niveau des nœuds agrégateurs.

Dans le chapitre suivant on va implémenter et simuler notre approche sur une topologie arborescente binaire, pour cela nous somme obligés d'implémenter cette topologie d'abord ensuite nous implémentons notre approche de sécurité pour simuler notre propre protocole obtenu.

Chapitre V:

Implémentation et Simulation.

V.1 Introduction

Dans le précédent chapitre, nous avons présenté une solution pour sécuriser l'agrégation de données, contre les attaques qui menacent le processus d'agrégation.

Afin de vérifier l'efficacité de notre solution, nous avons en premier lieu implémenté un simple protocole (sans le sécurisé). Puis en deuxième lieu, nous lui avons ajouté le système de sécurité. Enfin, nous présentons les résultats obtenus à l'issue des simulations et nous donnons leurs interprétations.

V.2 Hypothèse de fonctionnement

Pour le bon déroulement de l'application, nous avons supposé les cas suivants :

- Les nœuds malicieux ne seront pas présents dans l'espace de déploiement des capteurs qu'après la configuration de tous les nœuds du WSN.
- La station de base (ou le nœud SINK), peut joindre n'importe quel nœud capteur du WSN via le lien radio.
- La station de base (SINK) ne doit pas avoir le problème de contrainte d'énergie et de sécurité.

V.3 Les outils logiciels utilisés

V.3.1 Le système d'exploitation : TinyOS

TinyOs est un système d'exploitation orienté événement (event-driven) conçu pour les réseaux de capteurs. Il est implémenté entièrement en Nesc et il respecte une architecture basée sur une association de composants, permettant de réduire la taille du code nécessaire à sa mise en place.
TinyOs occupe un espace mémoire très faible dans sa distribution minimale (512octets). Par conséquent, il s'adapte aux capteurs pourvus de ressources mémoires très limitées.
Pour autant, la bibliothèque de composants de TinyOs est particulièrement complète puisqu'on y retrouve des protocoles réseau, des pilotes de capteurs et des outils d'acquisition de données.
Une application s'exécutant sur TinyOs est constituée d'une sélection de composants systèmes et de composants développés spécifiquement pour l'application. Parmi les principaux composants systèmes, on trouve l'ordonnanceur TinyOs. Cet élément est responsable de la gestion des tâches et des évènements du système. Son choix déterminera le fonctionnement global du système et le dotera de propriétés précises telles que la capacité à fonctionner en évènementiel.

V.3.2 Présentation du NesC

Le système d'exploitation TinyOS s'appuie sur le langage NesC. Celui-ci propose une architecture basée sur des composants, permettant de réduire considérablement la taille mémoire du système et de ses applications. Chaque composant correspond à un élément matériel (LEDs, timer, ADC …) et peut être réutilisé dans différentes applications. Ces applications sont des ensembles de composants associés dans un but précis. Les composants peuvent être des concepts abstraits ou bien des interfaces logicielles aux entrées / sorties matérielles de la cible étudiée (carte ou dispositif électronique).

V.3.2.1 Développement

NesC permet de déclarer deux types de fichiers: les modules et les configurations.

- Le fichier configuration est la définition des composants qui seront utilisés par l'application déployée sur le capteur.
- Les modules constituent les briques élémentaires de code et implémentent une ou plusieurs interfaces.
- Les interfaces sont des fichiers décrivant les commandes et évènements proposés par le composant qui les implémente.

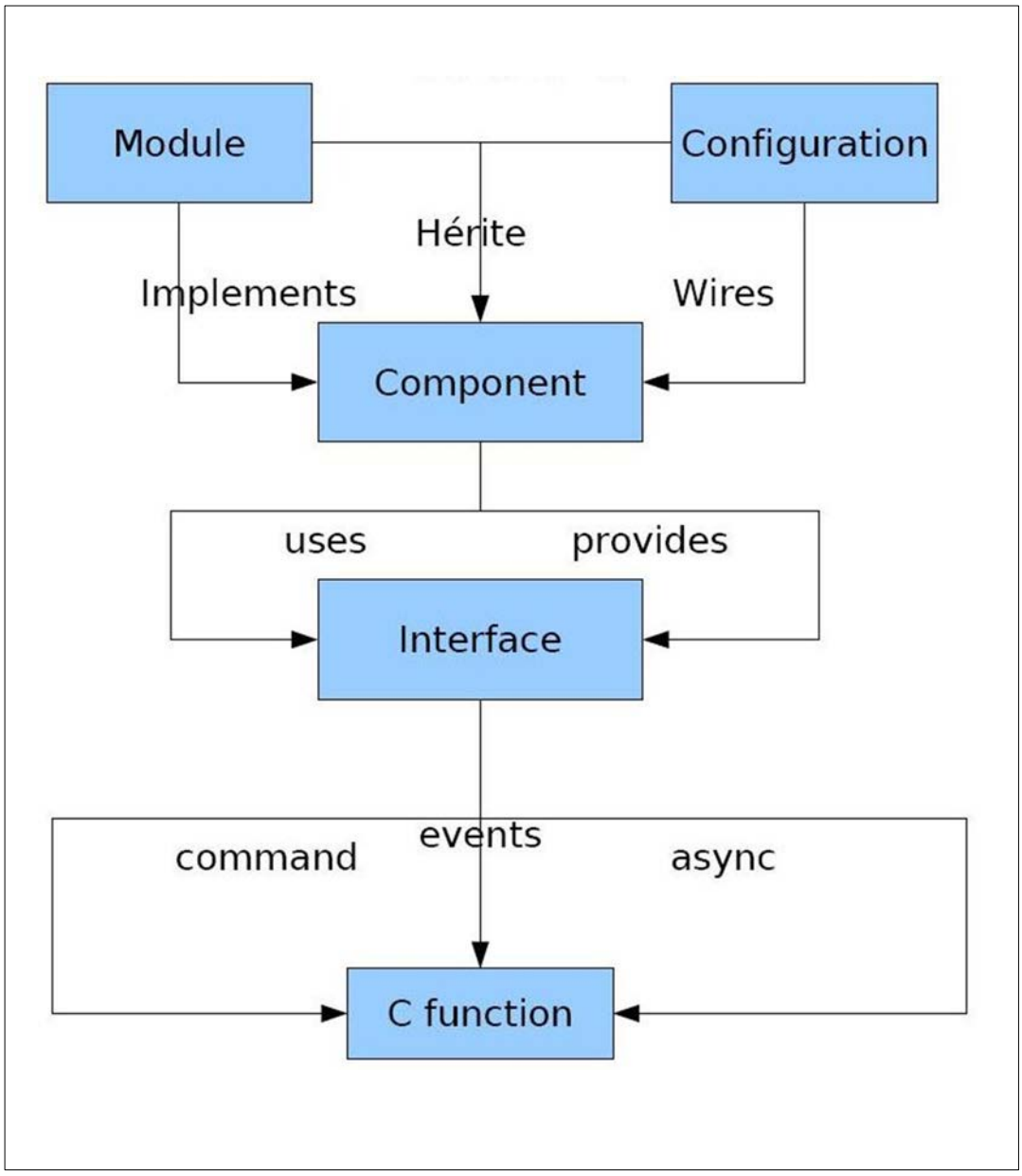


Figure V-1 : Organisation de l'architecture du langage NesC.

V.3.3. Compilation

Les capteurs possèdent en général la même architecture et sont programmables via le langage NesC. Cependant plusieurs firmes fabriquent de tels capteurs, ce qui implique quelques différences qui sont palliées par le compilateur de NesC appelé Ncc.

Pour effectuer une compilation, les fichiers doivent se situer dans le même répertoire contenant aussi un makefile.

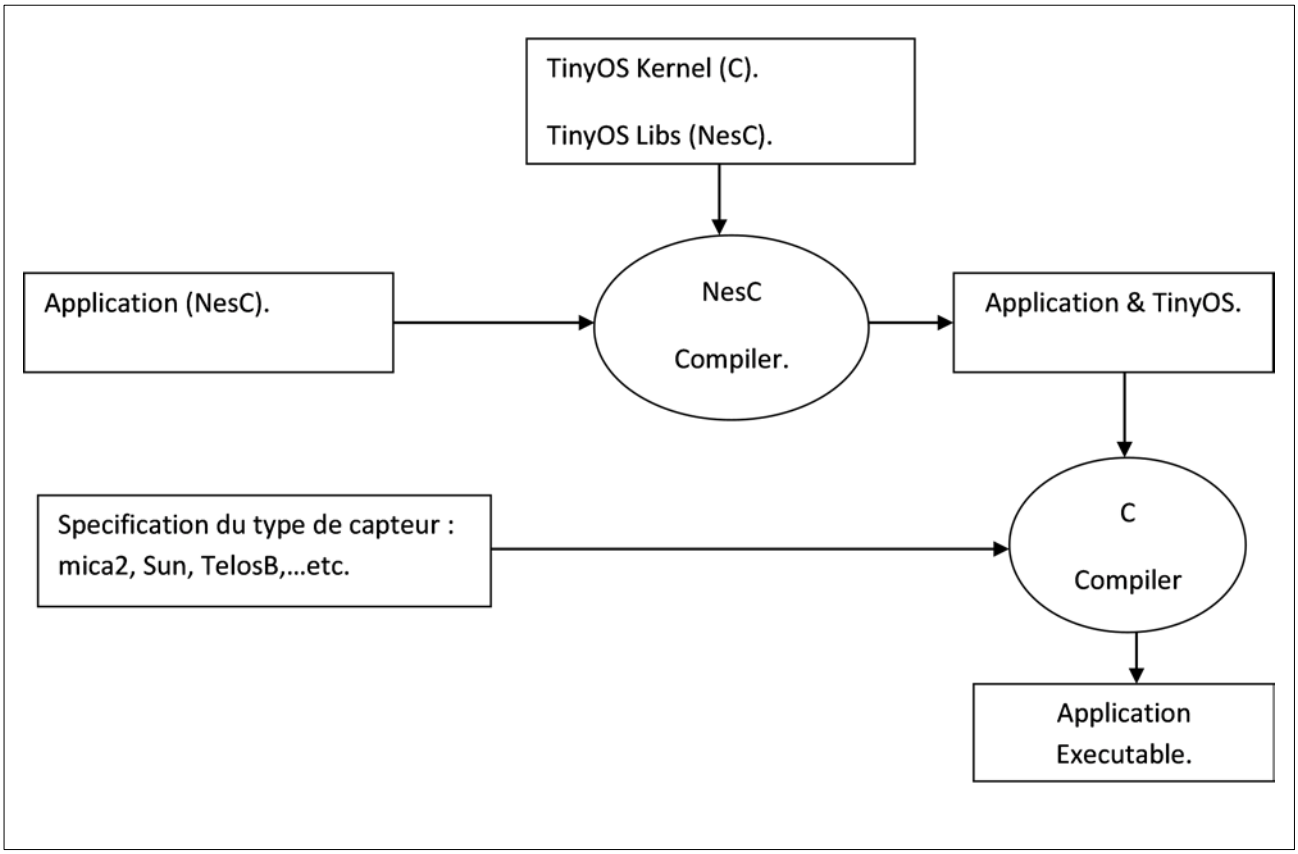


Figure V-2 : processus de compilation.

Le compilateur C compile le système d'exploitation du capteur 'TinyOS' avec l'application pour produire un fichier exécutable qui sera flashé dans la mémoire du capteur.

V.3.4 TOSSIM

Afin de simuler le comportement des capteurs, un outil très puissant a été développé et proposé sous le nom de TOSSIM.

TOSSIM simule le comportement des applications de TinyOS à un niveau très proche de ce qui se passe dans ces réseaux dans le monde réel.

Ce dernier est équipé d'une interface graphique (TinyViz) pour une meilleure compréhension et visualisation de l'état du réseau.

V.3.5 TinyViz

L'outil TinyViz est une application graphique qui nous permet d'avoir un aperçu de notre réseau sans avoir à déployer les capteurs dans la nature. Une économie d'effort et une préservation du matériel sont possibles grâce à cet outil. L'application permet une analyse étape par étape et en temps réel, en activant les différents modes disponibles.

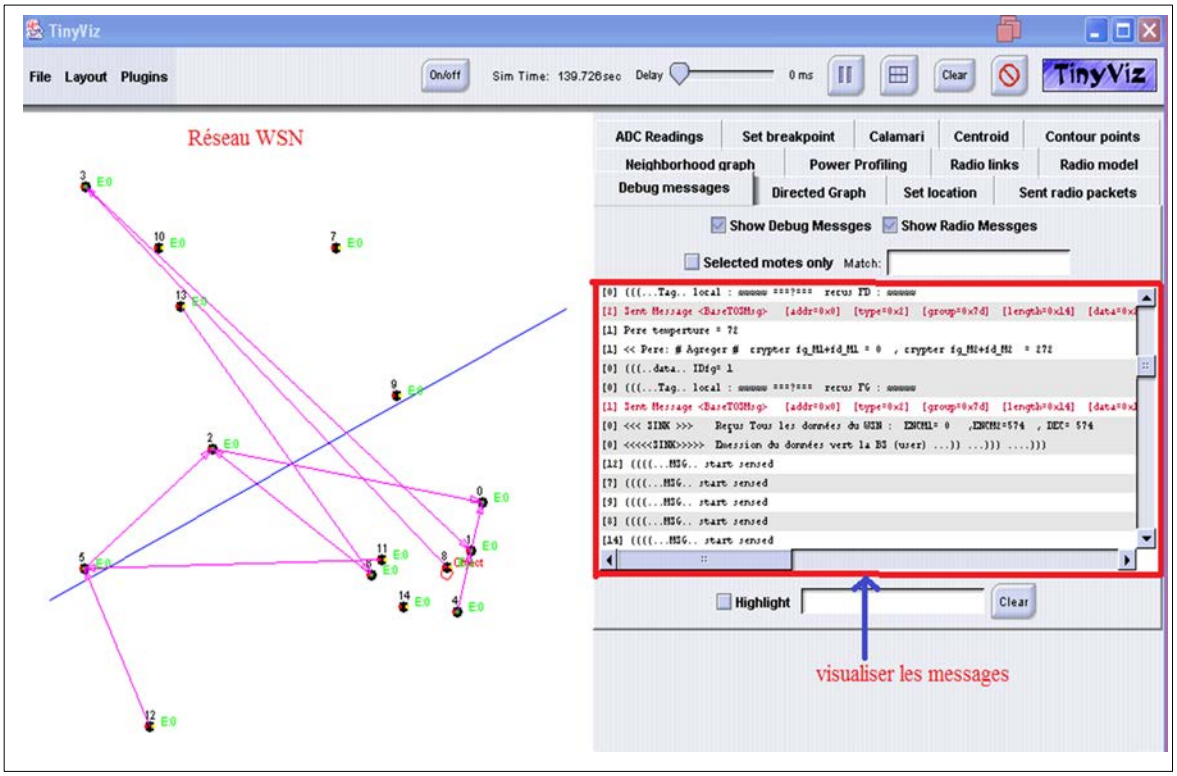


Figure V-3: Interface TinyViz.

V3.6 PowerTOSSIM

Est un outil qui permet de mesurer la consommation de l'énergie pour chaque capteur du réseau, et nous permet aussi de visualiser la consommation de l'énergie pour tous les composants d'un capteur : interface radio, CPU, LEDs… etc.

V.4. Implémentation

Dans le but d'implémenter notre solution de sécurité, nous avons besoin d'implémenter d'abord un protocole simple qui fonctionne sous une topologie arborescente binaire.

V.4.1 Implémentation d'une topologie arborescente binaire

Dans le but de mettre on marche notre système de sécurité contre les attaques menaçant l'agrégation de données on implémente un protocole de base, qui tien on compte :

- L'autoconfiguration des nœuds WSN, pour une topologie arborisente binaire.
- Le routage des paquets et l'agrégation de données.

V.4.1.1 Topologie arborescente binaire

Pour construire cette topologie, nous avons implémenté les nœuds, pour qu'ils puissent se configurer de manière à avoir pour chaque nœud père au plus deux fils : gauche et droit.

On effet, chaque nœud capteur du réseau WSN possède une adresse d'identification unique (TOS_LOCAL_ADRESS), utilisée pour envoyer un paquet de données. Alors, chaque nœud peut calculer l'adresse de ces deux fils (s'ils existent) de cette façon :

Fils gauche son adresse est : 2* TOS_LOCAL_ADRESS +1 ; Fils droit son adresse est : 2* TOS_LOCAL_ADRESS +2 ;

À titre d'exemple, si on a un nœud de l'adresse 17 qui possède deux fils inconnus, donc forcément on a l'adresse du fils gauche est : 35 et celle du fils droit est : 36.

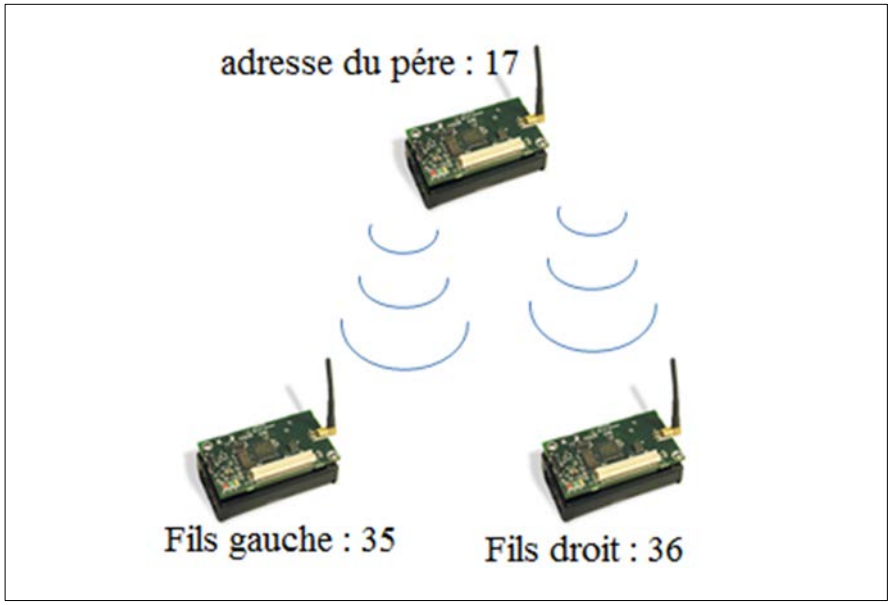


Figure V-4 : Exemple de configuration du réseau WSN.

V.4.1.2 Le routage des paquets et l'agrégation de données

Après avoir configuré la topologie de réseau WSN qui prend la forme d'un arbre binaire. On s'intéresse dans cette partie au routage et l'agrégation de données.

Chaque nœud contient dans sa mémoire l'adresse de son père établie lors de la configuration initiale du protocole, par la suite, le routage des paquets de données, débute par l'émission de données d'un nœud fils vers son père, et de ce dernier vers son grand-père, jusqu'à arriver à la destination qui est le nœud SINK (figure V-5).

L'agrégation de données s'effectue dans les nœuds pères, tels que chaque nœud père agrège au maximum trois données : celle du fils gauche et droit et celle du capteur agrégateur lui-même (figure V-5).

Dans notre cas, pour agréger les données, on a choisi l'opération d'addition, uniquement pour montrer le fonctionnement de notre système de sécurité contre les attaqués d'agrégation de données dans les WSN, cela signifie que notre système va fonctionner dans n'importe qu'elle opération qui agrège les données (min, Max, Addition, Moyen… etc.).

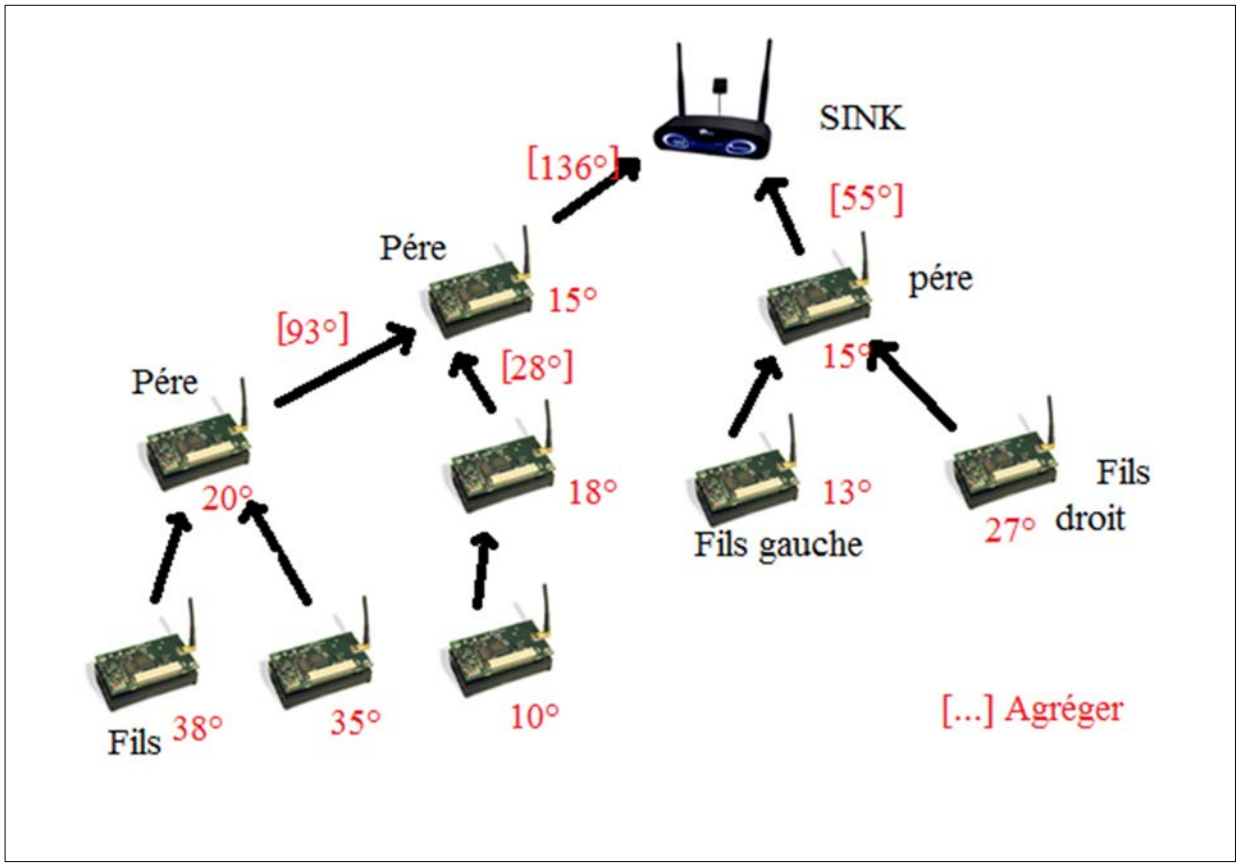


Figure V-5 : Routage et l'agrégation des données dans l'ARB.

V.4.2 Les types de paquets

On a utilisé les paquets suivants pour établir les communications dans le WSN.

V.4.2.1 Paquet de gestion de clés

Utiliser pour la distribution par diffusion de deux clés publiques : P et B nécessaires pour la cryptographie à courbe elliptique.

SrcId	Identification du nœud émetteur (SINK)
Pkey	la clé publique P
Bkey	La clé publique B

V.4.2.2 Paquet initialisé le protocole :

Ce paquet permet au nœud SINK d'informer les nœuds du WSN, pour commencer le processus de collecte de données, de leur environnement.

SrcId	Identification du nœud émetteur (SINK)

V.4.2.3 Paquet de données :

Un paquet pour transmettre les données agrégées.

SrcId	Identification du nœud émetteur.
DstId	Identification du nœud destinateur.
PntM1	La donnée cryptée M1 (AGREGER).
PntM2	La donnée cryptée M2 (AGREGER).
Tag[5]	le tag du message AGRÉGER : tag1 xor tag2 xor DstId xor PntM1 xor PntM2.
Indx1	Index 1 du tampon mémoire pour obtenir un tag1.
Indx2	Index 2 du tampon mémoire pour obtenir un tag2.

V.4.2.4 Paquet d'alertes d'intrusion :

Si une attaque sur l'agrégation de données est détectée dans le réseau WSN, le paquet suivant sera envoyé au nœud SINK, pour informer la station de base (SINK).

SrcId	Identification du nœud émetteur de l'alerte.
Intrus	Identification du nœud intrus.
POS	La position du nœud intrus : fils gauche ou droit.

V.5 Simulation

Dans cette seconde partie du chapitre, on va simuler notre solution de sécurité d'agrégation de données sous une plateforme d'un capteur MICAZ (voir annexe B).

V.5.1 Détection des attaques sur l'agrégation de données

V.5.1.1 Détection d'une attaque d'injection de faux paquets

Pour cela, nous avons programmé le capteur qui porte l'adresse 7 (figure V-6), pour qu'il fonctionne comme un nœud malicieux, qui essaie d'injecter uniquement des données incorrectes pour erronées l'agrégation de données.

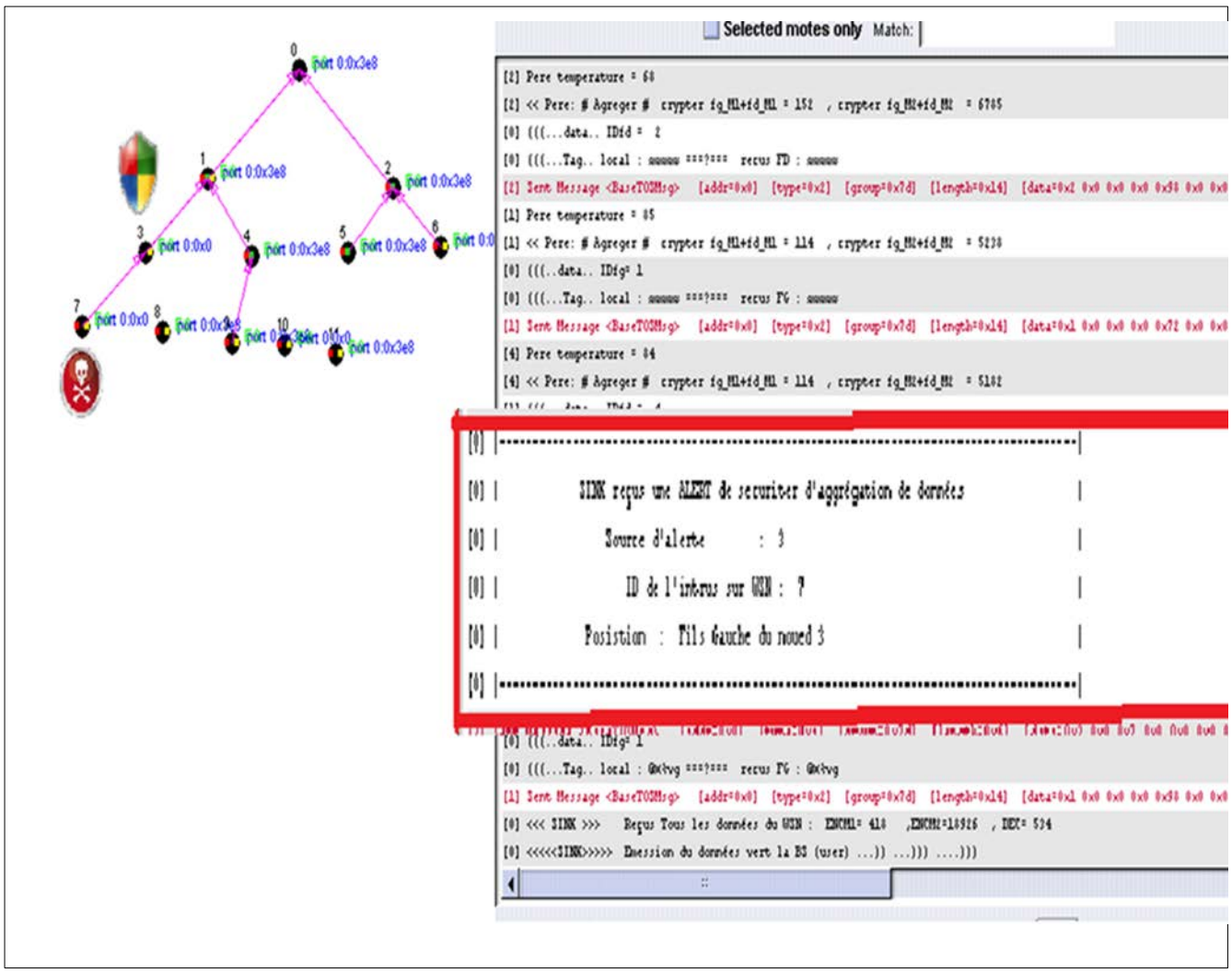


Figure V-6 : détecter une attaque d'injection de paquets.

D'après cette figure, on remarque que le capteur numéro 3 détecte le faux paquet injecté par le capteur malicieux de numéro 7.

V.5.1.2 Détection d'une attaque de falsification de données

Dans ce cas on simule le comportement du capteur numéro 15, dans le but d'effectuer une attaque avancée avec les étapes suivantes :

- Tout d'abord le capteur malicieux va se mettre sur l'écoute du réseau WSN.
- Ensuite il capture un paquet de données agrégé.
- Puis le capteur malicieux modifie le champ de données crypté.
- Et enfin, le capteur malicieux envoie le paquet falsifié dans le réseau.

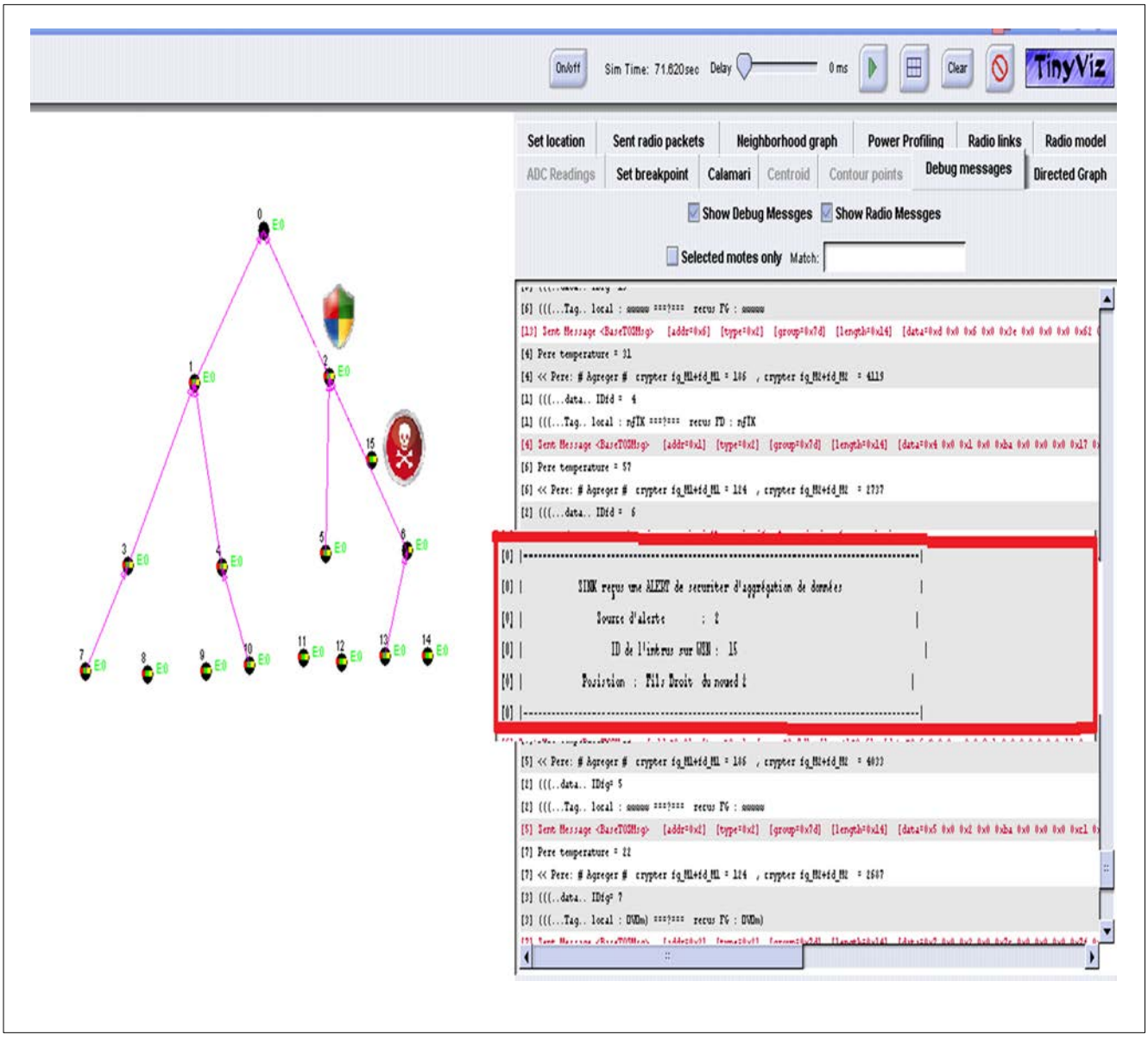


Figure V-7 : une attaque de falsification de données.

Dans cette figure le nœud malicieux (numéro 15), capte un paquet issu du nœud agrégateur numéro 6, ensuite il modifié le résultat de données agrégé avant d'envoyer le paquet au nœud 2.

Donc, le nœud 2 reçoit le paquet falsifié, mais comme il a procédé à l'opération de vérification en utilisent notre solution de sécurité (décrit dans le chapitre 4) pour chaque paquet reçu, il a détecte une attaque sur l'agrégation de données et comme résultat la donnée du paquet falsifié ne sera pas agrégée avec les autres données.

Remarque sur les paramètres de simulation de la consommation d'énergie:

Dans ce qui suit, nous effectuent les simulations avec paramètres suivants, en utilisent l'outil powerTOSSIM pour mesurer la consommation de l'énergie :

- Trois minutes comme temps de simulation.
- Vingt nœuds de capteur dans le réseau WSN.
- Dans notre cas on a surveillé la consommation de l'énergie du réseau uniquement pour le CPU et l'interface radio c.-à-d. on n'a pas tenu compte de la consommation de l'énergie a les Leds ou a les autres périphériques du capteur.

V.5.2 Consommation de l'énergie Vs nombre d'attaques

Dans cette partie nous montrent l'énergie consommée dans notre solution en utilisant la cryptographie de CCE et la cryptographie CMT en fonction de nombre de nœuds malicieux existants dans notre réseau.

Les nœuds malicieux vont réaliser une attaque sur l'agrégation de données, tel qu'ils envoient des paquets de données incorrects, et de l'autre côté on surveille l'évolution de la consommation d'énergie dans notre réseau WSN.

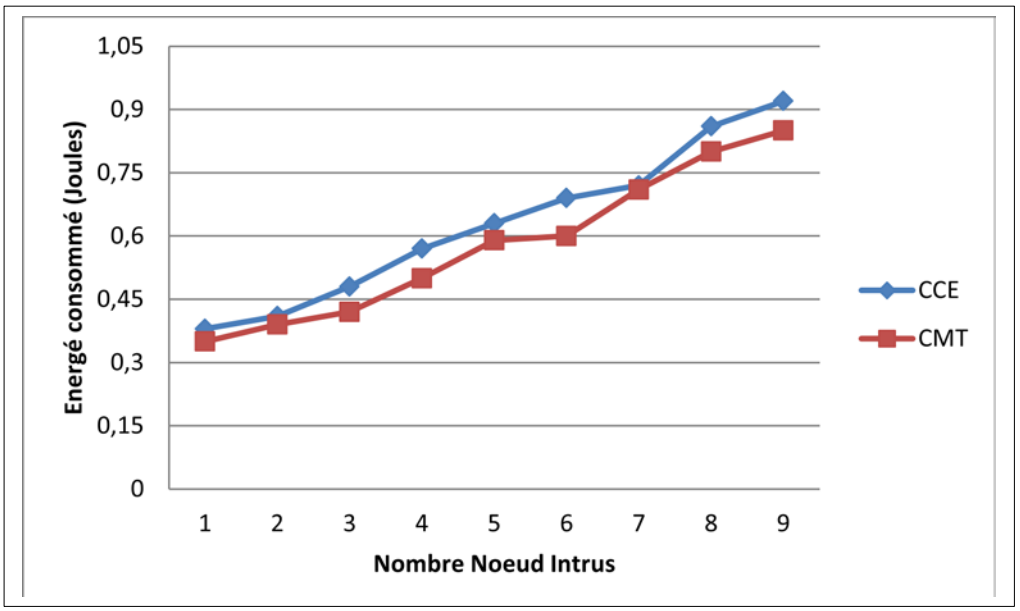


Figure V-8 : variation de la consommation de l'énergie en fonction du nombre des nœuds intrus (en utilisent le CCE et CMT).

On remarque d'après la figure V-8, que la consommation de l'énergie du WSN change on fonction du nombre de nœuds malicieux existant dans le réseau WSN, tel que tant que le nombre de nœuds malicieux est important, la consommation de l'énergie augmente aussi.

Et ça revient essentiellement à l'augmentation de transmettions des messages de détection des attaques au nœud SINK, autrement si un nœud détecte une attaque, il va informer le nœud SINK par l'envoie d'un message d'alerte, et si on a plusieurs attaques, donc forcément

il y aura plusieurs émissions de messages d'alertes ce qui augmente la consommation de l'énergie dans le WSN.

On remarque aussi, que la consommation de l'énergie en utilisant le CCE et le CMT sont presque égale.

V.5.3 Consommation de l'énergie dans notre solution de sécurité d'agrégation de données

Afin de visualiser la consommation de l'énergie dans notre solution de sécurité, nous avons procédé avec les deux cas suivants :

- 1ere cas: nous avons simulé le fonctionnement des capteurs (c.-à-d. l'agrégation de données), sans présence de notre système (solution) qui sécurise les données agrégées.
- 2éme cas : nous avons simulé l'agrégation de données avec notre solution de sécurité des données agrégées en utilisant le CCE et le CMT.

1) En utilisant la Cryptographie à Courbe Elliptique (CCE)

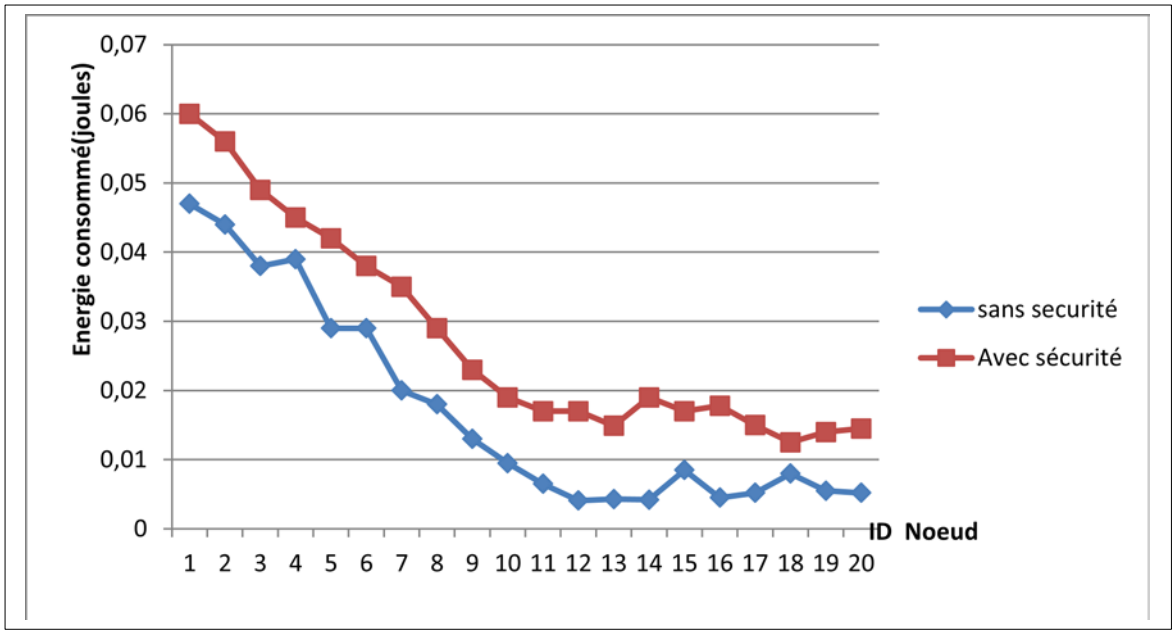


Figure V-10 : consommation de l'énergie par nœud en utilisent le CCE.

La figure V-10 , nous montre la consommation de l'énergie par nœud dans les deux cas : avec sécurité d'agrégation de données (en rouge), et sans sécurité d'agrégation de données (en bleu), tel qu'on remarque que notre solution de sécurité d'agrégation de données consomme un peu plus d'énergie, et cette consommation d'énergie est une conséquence des opérations de :

- La cryptographie et le hachage des données.
- La distribution des clés.

- Les engagements de sécurités : les variables, les contrôles des paquets reçus, l'envoi des tags… etc.

On remarque aussi que la consommation de l'énergie pour les capteurs qui se trouvent dans un niveau moins profond dans la topologie arbre comme les nœuds (0, 1, 2, 3, 4, 5, 6) consomment plus de l'énergie comparativement aux autres capteurs qui se trouvent à la profondeur de l'arbre, parce que tant que les capteurs sont moins profonds dans l'arbre, tant qu' ils effectuent plus de l'opération d'agrégation de données, plus de contrôle de sécurité , et reçoivent plus de paquets pour faire le routage.

2) En utilisant la Cryptographie de Castellucia Mykletun Tsudik (CMT)

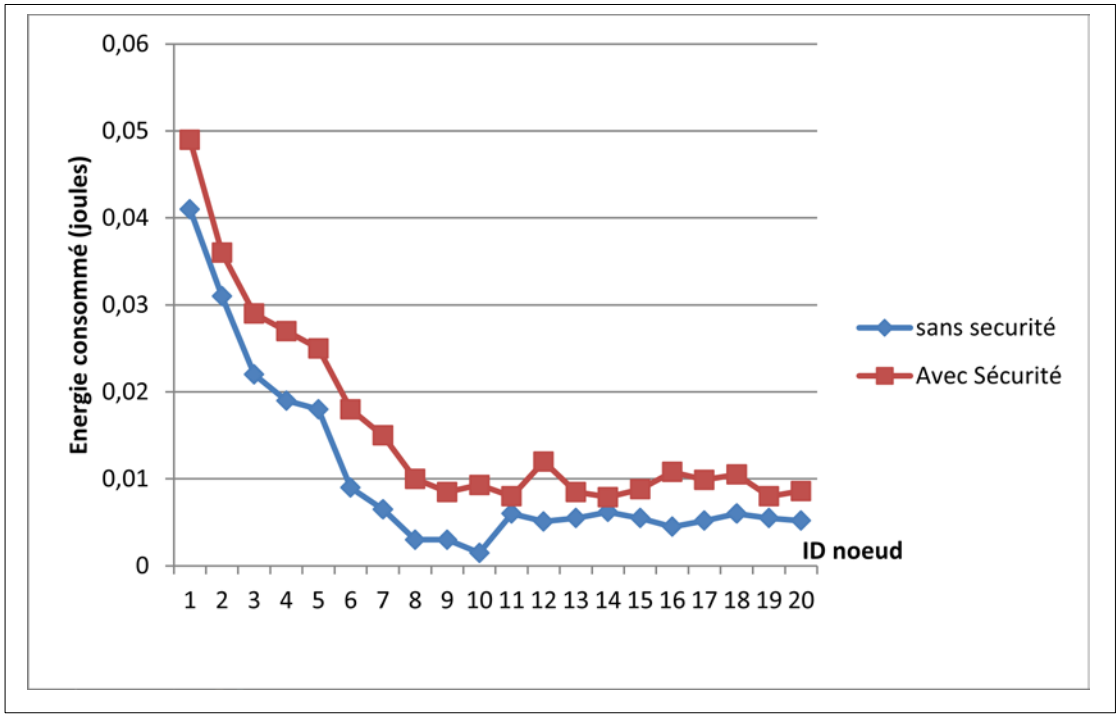


Figure V-11 : consommation de l'énergie par nœud en utilisent le CMT.

La figure V-11, nous montre aussi la consommation de l'énergie par nœud dans les deux cas : avec sécurité d'agrégation de données (en rouge), et sans sécurité d'agrégation de données (en bleu), tel qu'on remarque aussi que notre solution de sécurité d'agrégation de données en utilisant le CMT consomme aussi un peu d'énergie malgré que le CMT est un programme de cryptographie symétrique.

V.5.4 Variation de la consommation de l'énergie en fonction : de nombre de nœuds de réseau et du nombre de nœuds intrus

La figure suivante nous montre, la variation de la consommation de l'énergie du réseau en fonction du nombre de nœuds du réseau et en fonction du nombre de nœuds intrus existant dans le réseau.

Remarque : Nous avons utilisé dans ce cas la cryptographie à courbe elliptique.

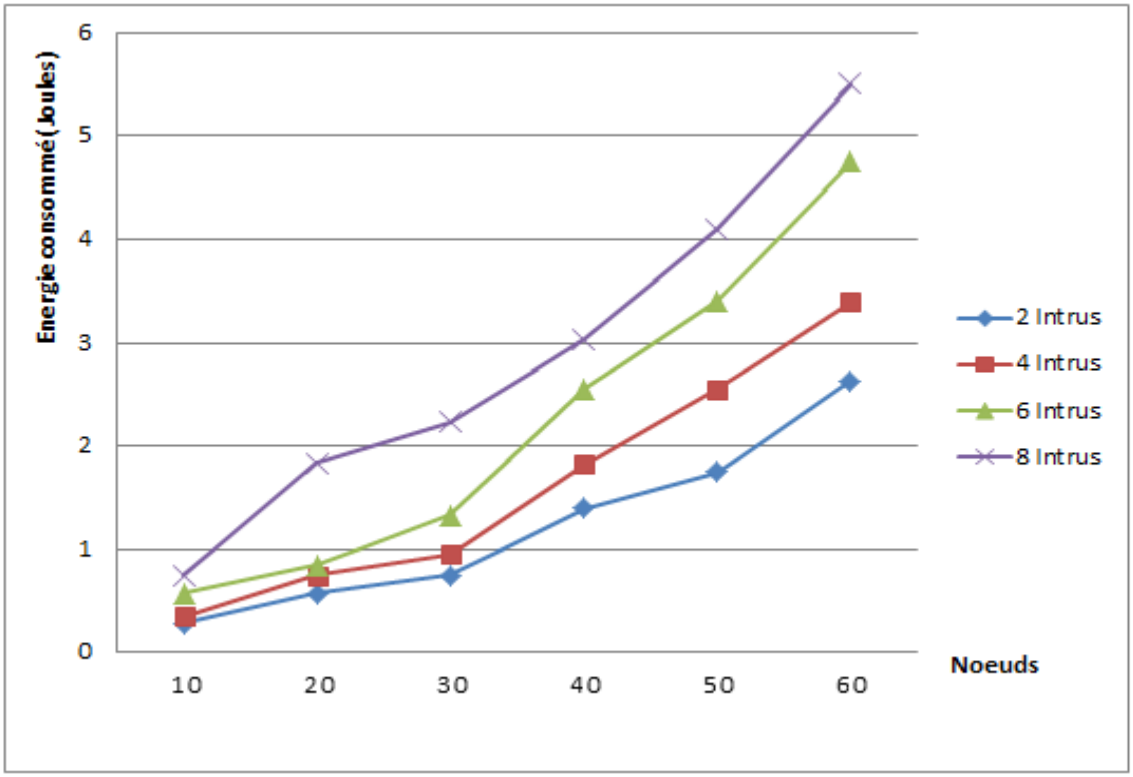


Figure V-12 : variation de la consommation de l'énergie en fonction de nombre de nœuds de réseau et en fonction de nombre de nœuds intrus.

D'après cette figure on remarque que la consommation de l'énergie dans le réseau se varié en fonction de la taille du réseau et en fonction du nombre de nœuds intrus existant dans le réseau, tel que si la taille du réseau augment et si le nombre de nœuds malicieux existant dans le réseau est important alors l'énergie consommée augment aussi.

En constate d'après toutes les simulations précédentes, que pour sécuriser l'agrégation de données nécessite de sacrifie un peu d'énergie pour assurer une sécurité, c'est le frais de la sécurité.

Assuré une sécurité sur les données agrégées impliqué une nécessite de sacrifie de l'énergie.

V.6 Conclusion

Dans ce chapitre, essentiellement nous avons implémenté et simulé notre solution de sécurité d'agrégation de données dans les deux cas : on utilisant la cryptographie asymétrique de CCE et la cryptographie symétrique CMT.

D'après les résultats de simulation effectuée sur la consommation de l'énergie dans notre solution on a constaté que pour assurer une sécurité sur les données agréger, implique pour le réseau WSN une consommation supplémentaire de l'énergie soit on utilisant la cryptographie symétrique ou asymétrique.

Conclusion générale et perspective

Les réseaux de capteurs sans fil constituent un axe de recherche très fertile et peuvent être appliqués dans plusieurs domaines différents. Cependant pour que ces réseaux puissent mener à bien leurs missions ils doivent assurer un certain niveau de sécurité, et plus précisément contre les attaques informatiques qui menacent le bon fonctionnement de ce type de réseau.

Dans ce contexte, notre travail a pour objectif d'apporter une nouvelle solution au problème lié à la sécurité d'agrégation de données dans les réseaux capteurs. Tel que les données agrégées peuvent être falsifiés ou nuire par une attaque informatique, ce qui crée des conséquences fatales, comme prendre une mauvaise décision au niveau de l'utilisateur final du réseau.
Notre solution de sécurité pour faire face à toutes les attaques qui menacent l'opération d'agrégation de données, est d'utilisé une nouvelle méthode, pour authentifier les messages reçus avant de les agrégées. Cette méthode se base sur la sauvegarde du contenu historique des données hachées (tags), tel que chaque capteur enregistre dans sa mémoire les tags des données reçues et envoyées. De cette façon, chaque capteur crée une partie historique des tags, qui font partie de son passé. Ensuite, les capteurs vont utiliser ces tags pour prouver l'identité des messages reçus, avant de les agréger.

La puissance de cette solution, est que si une attaque qui arrive à l'instant 't', pour faire une attaque, il doit connaitre les historiques des tags pour chaque capteur ce qui est impossible, car les contenus historiques font partie du passé du réseau.

Pour assurer la confidentialité, des données agrégées, nous avons utilisé aussi dans notre solution, l'un des puissants algorithmes de cryptographie, nommée la cryptographie à courbe elliptique, économique en consommation d'énergie et puissant du côté cryptage et on a utilisé aussi le CMT qui aussi très économique pour la consommation de l'énergie.

Et comme perspective, on envisage l'adaptation de notre solution dans une topologie clustérisée, on diminue par exemple le nombre de tags enregistré, pour économiser la mémoire où on utilise une technique de compression.

Références Bibliographiques

[1] I.F. AKYILDIZ, W. S. SANKARASUBRAMANIAM, E. CAYIRCI: "Wireless Sensor Networks: A Survey", Computer networks, 2002, 38.

[2]P. MOHAPATRA, S. V. KRISHNAMURTHY: "Ad Hoc Networks Technologies and Protocols", Springer Verlag Telos, 2004, ISBN: 0-387-22689-3.

[3] F. STAJANO: "Security for Ubiquitous Computing", John Wiley and Sons, New York, 2002, ISBN:0-470-84493-0.

[4]Ian F. Akyildiz, Weilan Su, Yogesh Sankarasubramaniam et Erdal Cayirci "A Survey on Sensor Networks", IEEE Communication Magazine, p.102-114, August 2002.

[5] K. Sairam, N. Gunasekaran, S. Redd, "Bluetooth in wireless communication",IEEE Communications Magazine, Vol. 40, pp. 90–96, 2002.

[6] M. J. HANDY, M. HAASE, D. TIMMERMANN: "Low Energy Adaptive Clustering Hierarchy with Deterministic Cluster-Head Selection", Fourth IEEE Conference on Mobile and Wireless Communications Networks, Stockholm, erschienen in Proceedings, S. 368-372, ISBN 0-7803-7606-4 World Scientific Publishing Co. Pte. Ltd., 2002.

[7] L. ESCHENAUER, V. D. GLIGOR: "A Key Management Scheme for Distributed Sensor Networks », ACM CCS, 2000, pp. 41–47.

[8] C. F. GARCIA-HERNANDEZ, P. H. IBARGUENGOYTIA-GONZALEZ, J. GARCIAHERNANDEZ, J. A. PEREZ-DIAZ: "Wireless Sensor Networks and Applications: a Survey", IJCSNS International Journal of Computer Science and Network Security, 2007, Vo.7, No.3, pp. 264-273.

[9] LEHSAINI Mohamed.''Diffusion et couverture basées sur le clustering dans les réseaux de capteurs : application à la domotique''.p10.2009.

[10] theseMakhoul08.

[11] T. WATTEYNE: "Energy-Efficiency Self-Organisation for Wireless Sensor Network", Doctorate Thesis, Institut National des Sciences Appliquees de Lyon, N° 2008-ISAL-0082, 2008.

[12]Shio Kumar Singh et al, "Application, Classification, and Selections of Energy-Efficient Routing Protocols for Wireless Sensor Networks", INTERNATIONAL JOURNAL OF ADVANCED ENGINEERING AND TECHNOLOGIES (IJAEST), Vol No. 1, Issue No. 2, p 85-95

[13] National Bureau of Standards, Data Encryption Standard, FIPS-Pub.46.National Bureau of Standards, U.S. Department of Commerce, Washington D.C., January 1977.

[14] V. Miller, Use of elliptic curves in cryptography, CRYPTO 85, 1985.

[15] TaherElGamal, "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory,v. IT-31, n. 4, 1985, pp.469-472 or CRYPTO 84, pp.10-18, Springer-Verlag.

[16] W. Diffie and M. E. Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp: 644-654.

[17] E. Fasolo, M. Rossi, J. Widmer, M. Zorzi, In-network aggregation techniques for wireless sensor networks: a survey, IEEE Wireless Commun. 14(2) (2007) 70-87.

[18] A. Harris III, R. Kravets, and I. Gupta, "Building Trees Based On Aggregation Efficiency in Sensor Networks", Med-Hoc-Net 2006, Lipari, Italy,June 2006.

[19] S. Madden et al.,"TAG: a Tiny AGgregation Service for Ad Hoc Sensor Networks", OSDI 2002, Boston, MA, Dec. 2002.

[20] B. Zhou et al., "A Hierarchical Scheme for Data Aggregation in Sensor Network", IEEE ICON '04, Singapore, Nov. 2004.

[21] A. Mahimkar and T. S. Rappaport, “SecureDAV: A Secure Data Aggregation and Verification Protocol for Sensor Networks”, IEEE GLOBECOM 2004, Dallas, TX, Nov. 2004.

[22] B. Przydatek, D. Song, A. Perrig, SIA : secure information aggregation in sensor networks, in: Proceedings of SenSys ’03, 2003, pp. 255-265.

[23] Ralph C. Merkle. A certified digital signature. In Proc. Crypto ’89, pages 218-238, 1989.

[24] A. Mahimkar, T.S. Rappaport, SecureDAV: a secure data aggregation and verification protocol for wireless sensor networks, in: Proceedings of the 47th IEEE Global Telecommunications Conference (Globecom), November 29 - December 3, Dallas, TX,2004.

[25] D. Johnson, A. Menezes and S. Vanstone, “The Elliptic Curve Digital Signature Algorithm (ECDSA)”, Springer-Verlag, 2001.

[26] Yang, Y., Wang, X., Zhu, S., and Cao, G. 2008. SDAP: A secure hop-by-hop data aggregation Protocol for sensor networks. ACM Trans. Inf. Syst.Secur. 11, 4, Article 18 (July 2008).

[27] H. C¸ am, S. Ozdemir, P. Nair, D. Muthuavinashiappan, H.O.Sanli, Energyefficient and secure pattern based data aggregation for wireless sensor networks, Comput. Commun., Elsevier 29 (4) (2006) 446-455

[28] S. Ozdemir, Secure and reliable data aggregation for wireless sensor networks, in: H. Ichakawa et al. (Eds.), LNCS 4836, 2007, pag. 102-109.

[29] Niels Ferguson, Bruce Schneier, “Practical Cryptography”, John Wiley & Sons (2003).ISBN 0471223573.

[30] SuatOzdemir, “Functional Reputation Based Reliable Data Aggregation and Transmission for Wireless Sensor Networks”, Computer Communications, Elsevier, vol. 31, no. 17, pp. 3941-3953, Nov. 2008.

[31] D. Wagner, “Cryptanalysis of an Algebraic Privacy Homomorphism”, in: Proceeding of the 6th Information Security Conference (ISC03), Bristol, UK, October 2003.

[32] S. Ozdemir, “Secure data aggregation in wireless sensor networks via homomorphic encryption”, Journal of The Faculty of Engineering and Architecture of Gazi University 23 (2) (2008) 365-373. ISSN:1304-4915.

[33] C. Castelluccia, E. Mykletun, G. Tsudik, “Efficient aggregation of encrypted data in wireless sensor networks, in: Proceedings of the Conference on Mobile and Ubiquitous Systems: Networking and Services, 2005, pp.109-117.

Annexe A:

Installation TinyOS sous windows xp.

Installation de Cygwin

Cygwin est un environnement d'émulation Linux qui permet d'avoir un Shell et de compiler et exécuter les programmes Linux dans un système d'exploitation défirent de Linux.

- Télécharger le fichier tinyos-1.1.0-lis.exe de la source :

http://www.tiniyos.net/dist-1.1.0/tiniyos/windows/

- Exécuter ce fichier pour installer la version 1.1.0 sous Windows XP. L'installation se fait automatiquement. Un raccourci de Cygwin est sauvegardé sur le bureau.

Installation de TinyViz

TinyViz est un outil facultatif, nous permettant de visualiser les résultats de simulation, telle que TinyViz nous propose plusieurs options pour visualiser la simulation : voire les paquets, les liens radio, modifie la fréquence radio… etc.

Les concepteurs développent au fur et à mesure l'outil TinyViz sans mettre à jour les fichiers sources déjà existants dans les anciennes versions. Cela ne permet pas de lancer TinyViz dans des conditions normales. Pour pouvoir le lancer, il est nécessaire de passer par les étapes suivantes:

1- Après avoir installé TînyOS-1.0

2- Accéder par le Shell on tape : **cd** /opt/tinyos-l.x/tools/java et : make

3- Installer les mises à jour de NesCl.1.1 et TinyOS1.1.15.
Pour se faire, rechercher sur le net http://www.tinyos.net/dist-1.1.0/tinyos/windows/ ces mises à jour en téléchargeant le rpm et le mettant dans C:\tinyos\cygwin\home\<users>
et taper dans le Shell **:**
rpm -ivh --ignoreos nesc-1.1.2b-l.cygwin.i386.rpm
rpm -ivh --ignoreos --force tinyos-1.1.15Dec2005cvs-l.cygNNin.noarch.rpm

4- Editer le fichier 'Makefile' qui est dans :

C:\tinyos\cygwin\opt\tinyos-l.x\tools\java\net\tinyus\sim et on écrirait cette instruction: **net/tinyos/message/avrmote/*.class** comme suit :

net/tinyos/sim/ui net/tinyos/sim/plugins/plugins.list

net/tinyos/sf/*.class net/tinyos/util/*.class net/tinyos/packet/*.class

net/tinyos/message/*.class **net/tinyos/message/avrmote/*.class**

5- Aller au Shell et taper: cd /opt/tinyos-l.x/tools/java/net/tinyos/sim
make clean # supprimer les fichiers déjà compiler
make # compiler

6- Accéder à l'application qui va être simulée. On prend par exemple, notre

application ‘’Protocole ‘’. Accéder au Shell et faire: **cd opt/tinyus-l.x/apps/Protocole**

make pc # la compilation sur notre machine pc

export PATH="$TOSROOT/tools/java/net/tinyos/sim:$PATH" #exporter le lien aux variables d’environnement PATH

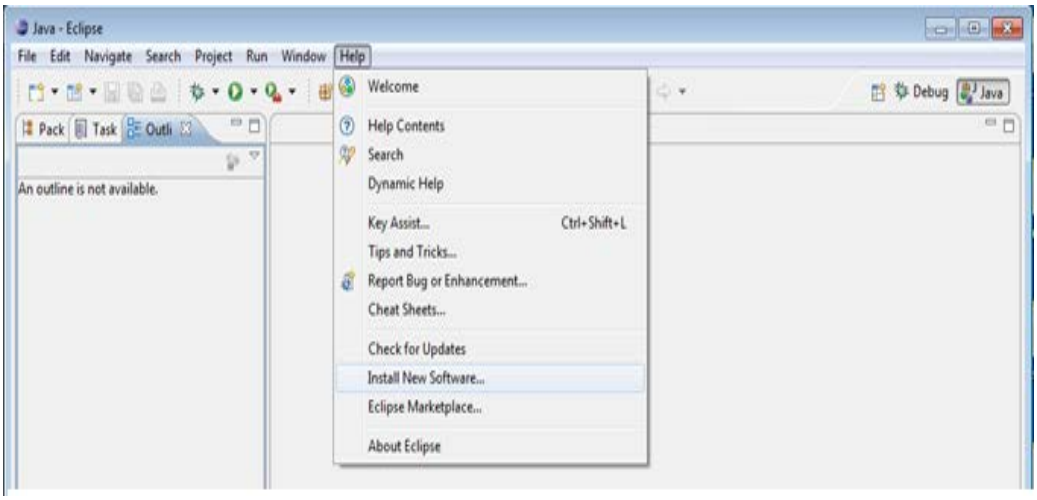
TinyViz -run build/pc/main.exe 12 # 12 est le nombre de nœuds capteurs.

Maintenant il nous reste que de configurer éclipse pour préparer un environnement de programmation utilise le langage nesC.

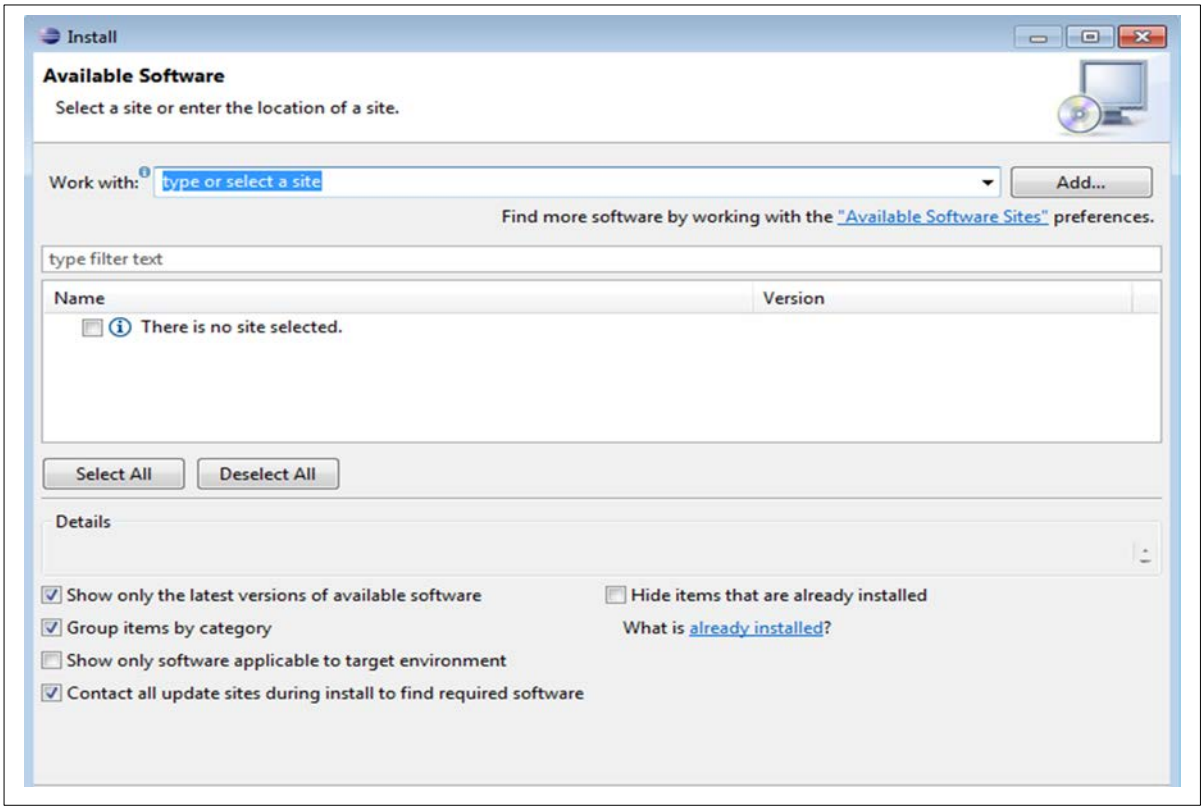
Installation du plug-in Yeti 2

Pour l'installation de l'IDE Eclipse, il suffit de télécharger une version d’Éclipse située à l'adresse suivante : http://www.eclipse.org/downloads/ après avoir décompressé le fichier téléchargé, on lance Éclipse à partir du raccourci créé à cet effet.

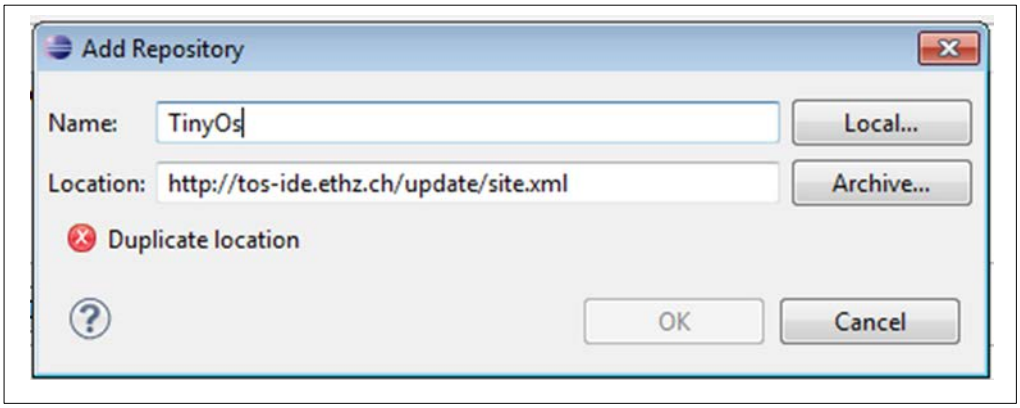
Afin de paramétrer Eclipse pour qu'il fonctionne avec une perspective TinyOS, on doit lui ajouter le plug-in Yeti 2, et ceci à partir du menu « help » de l'IDE Eclipse en suivant la démarche «help→Install New Software».



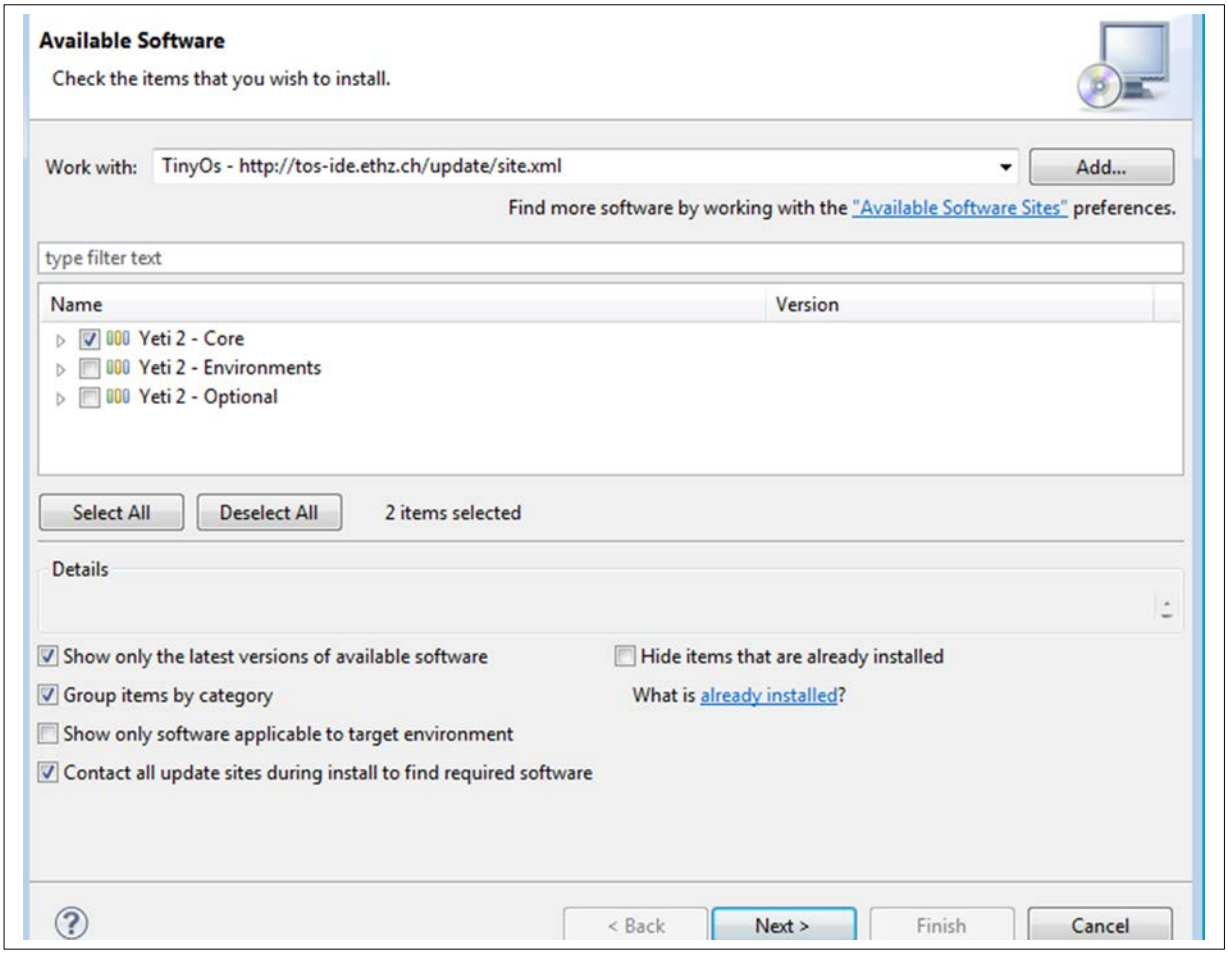
Cliquer sur le bouton : Add



Tapez le lien suivant : http://tos-ide.ethz.ch/update/site.xml.



Choisir les plugins nécessaires au fonctionnement de TinyOS sous Éclipse, puis cliquer sur « next » pour lancer l'installation.



Maintenant tout est prêt pour créer un projet TinyOs sous l'IDE Eclipse, pour cela on tape « File→New→Project », puis on sélectionne «TinyOs project».

Annexe B:

Capteur MicaZ.

Chaque capteur MicaZ se compose de deux modules : une carte MPR2600 et une carte MTS400 (Figure). La carte MPR 2600 contient un microcontrôleur et un transceiveur. Le

Microcontrôleur contient une mémoire flash de 128kB, une mémoire vive de 4kB et un processeur de 4MHz. C'est la partie de calcul et de traitement des informations des capteurs. Sur la carte, il y a également un module transceiveur à fréquence radio de 2.4 GHz allant jusqu'à 250kb/s. Ce dernier est le module de communication permettant des transmissions sans fil entre les capteurs et de capteurs vers la station de base.

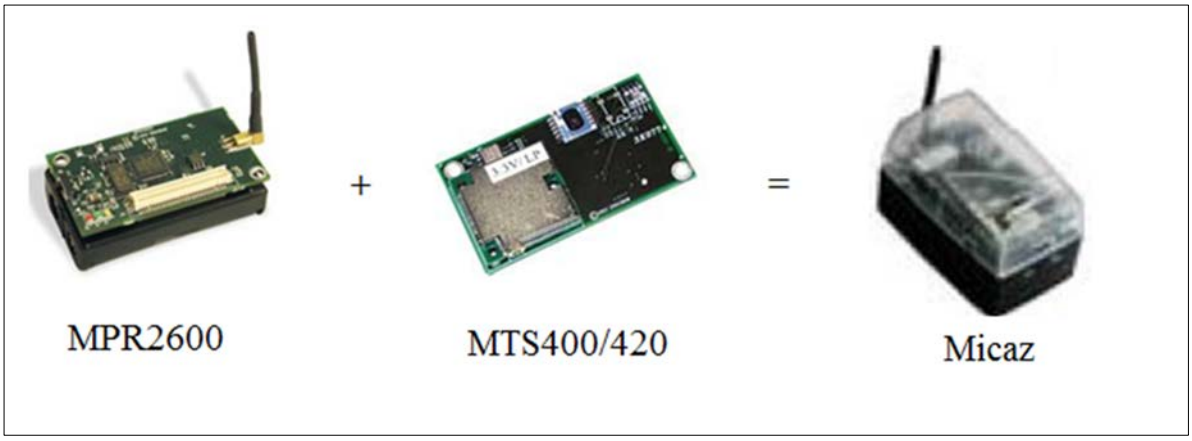


Figure : composition d'un capteur MICAZ.

Une partie indispensable pour un capteur est le module "senseur". En effet, c'est le module qui permet de convertir des valeurs ambiantes en valeurs numériques telles que la température, les vibrations, etc. Pour les capteurs MicaZ, la carte MTS400 donne des mesures assez complètes de différents facteurs environnementaux : la température, l'accélération, la pression, la luminosité et l'humidité. Cette carte est dotée un connecteur spécifique afin de connecter avec la carte MPR2600 pour transmettre les mesures ambiantes. Une nouvelle génération de carte "senseur" MTS420 est également fournie chez Crossbow avec une fonctionnalité supplémentaire. Elle donne toutes les mesures de MTS400 et elle offre également la position géographique GPS du capteur qui permet de les utiliser dans les applications où on a besoin de la localisation de capteurs.

Annexe C:

Techniques de compilation et simulation sous TOSSIM.

Dans cette partie on va citer, les commandes de base pour exploiter le simulateur TOSSIM, et précisément les commandes nécessaires pour la compilation et l'exécution.

Pour des raisons de test et pour afficher des messages sur notre console noire de cygwin, nous avant procéder à la compilation du notre protocole sous notre ordinateur, utilisant cette commande :

$ make pc

Et pour des raisons de simulation, nous avons généré un code exécutable par la compilation destiner pour qu'elle puisse être fonctionné sur le capteur MicaZ, pour cela on utiliser cette commande :

$ make micaz

Remarque : la simulation que nous avons faite sur le capteur MicaZ est une simulation virtuelle sous le simulateur TOSSIM.

Dans le but d'activer l'interface TinyViz du simulateur TOSSIM, nous avons besoin d'exporter le lien de TinyViz aux variables d'environnement du système Windows :

$ export PATH="$TOSROOT/tools/java/net/tinyos/sim:$PATH"

Si on veut visualiser les messages crée par des commandes d'appel système comme 'printf(…) ou DBG(…)', on doit tapez cette commande sur notre console comme suite :

$ export DBG=usr2

et pour visualiser la consommation d'énergie pour tous les capteurs sous TinyViz, on a besoin de tapez aussi cette commande sur la console :

$ export DBG= `power`

Et enfin, pour lancer l'exécution de notre protocole déjà compilé, on tape l'une des commandes suivantes :

- Pour la simulation sous l'interface TinyViz on a :
  $ TinyViz -run build/pc/main.exe 15 // 15 est le nombre de nœuds à simuler
- Si on veut suivre les évolutions de consommation de l'énergie sous TinyViz, on tape la commande suivante :
  $ TinyViz -run 'build/pc/main.exe –p' 15 // 15 est le nombre de nœuds à simuler
- Et pour la simulation sans l'interface TinyViz on a :
  $ ./build/pc/main.exe 15 // 15 est le nombre de nœuds à simuler