الجـــــــمهورية الجـــزائـــريـــة الديمـــقـــراطـيـــة الـــشـعـبـيـــة
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التـــــــعـــــليـــــم العـــــالي و البـــــــحث العـــلـــمـــــــي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جـــــــامعة أبي بـكـــــــر بلـــقـــــايـد– تـلمســـــــان –
Université Aboubakr Belkaïd– Tlemcen –
Faculté de TECHNOLOGIE

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunications

Spécialité : Réseaux Mobiles et Services de Télécommunications

Par : BENAYAD Abdelhak

Sujet

Implémentation Et Sécurisation du Protocole De Routage AODV optimisé pour les RCSF (OAODV)

Soutenu publiquement, le 26 / 09 /2017, devant le jury composé de :

M. Merzougui Rachid	MCA	Univ. Tlemcen	Président
M. Moussaoui Djilali	MAA	Univ. Tlemcen	Examinateur
M. Kadri Benamar	MCA	Univ. Tlemcen	Encadreur

Année universitaire : 2016-2017

Dédicace

Je dédie ce modeste travail

A mes très chers parents

En signe de ma profonde reconnaissance pour leur amour, tous les sacrifices, les soutiens, les tolérances et les encouragements qu'ils sont consentis à mon égard

A mon frère et mes sœurs pour tous leur encouragements

A mon encadreur Monsieur KADRI pour
l'excellence de son accompagnement et la confiance qu'il nous a accordé.

A touts mes collègues du M2 RMST 2016/2017

A mes amies : Abdo, Brahim, Djamel, Lakhdar, Abdeljabar ……

A touts mes collègues et amis du travail : El Agrari, Mohamed, Salim, Hamza, Reda...

Et toute personne que je connais et qui me sont chers et tous ceux
qui m'aiment.

Remerciements

En préambule à ce mémoire nous remercions ALLAH
le tout puissant, qui nous a donné la force, la volonté et surtout

le courage pour accomplir ce modeste mémoire.

Nous souhaitons adresser nos remerciements les plus
sincères tout d'abord au corps administratif de
la faculté des sciences, à tous les enseignants de

la spécialité RMST qui ont contribué à la réussite de

cette formidable année universitaire.

ainsi qu'aux personnes du lab STIC qui nous ont apporté

leur aide pour l'élaboration de ce mémoire

Nous tenons à remercier sincèrement " notre encadreur Monsieur B. KADRI"
qui, s'est toujours montré à l'écoute et très disponible tout au long de la
réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et
le temps qu'il a bien consacré pour nous.

Nous expirons notre gratitude à Mr Merzougui Rachid, Mr Moussaoui Djilali d'avoir honoré
notre jury de soutenance.

Table des matières

LISTE DES FIGURES

LISTE DES TABLEAUX

Liste des abréviations

AODV : Ad-hoc On-demand Distance Vector

OAODV : Optimised ad-hoc On-demand Distance Vector

RCSF : Réseau de capteur sans fil

WSN : Wireless Sensor Network

UDP : User Datagram Protocol

PDR : Paquet Delivery Ratio

TEEN : Threshold sensitive Energy Efficient sensor Network protocol

PEGASIS :Power-Efficient GAthering in Sensor Information Systems

SPIN : Sensor Protocols for Information via Negociation

DD : Directed Diffusion

GEAR : Geographic and Energy Aware Routing

MECN : Minimum Energy Communication Network

PKI : Micro Public Key Infrastructure

WEP : Wired Equivalent Privacy

WPA : Wi-Fi Protected Access

ARC4 : Alleged RC4

PRGA : Pseudo-Random Generation Algorithm

ECC : Elliptic Curve Cryptography

ECDH : Elliptic Curve Diffie-Hellman

ECIES : Elliptic Curve Integrated Encryption Scheme

ECDSA :Elliptic Curve Digital Signature Algorithm

CPU : Central processing unit

RREQ : Route Request

RREQ ID :Identifier Route Request

RREP : Route Reply

RERR : Route Error

JDK : Java Development Kit

INTRODUCTION GENERALE

Aujourd'hui, les réseaux sans fil sont de plus en plus populaires du fait de leur facilité de déploiement. Ces réseaux jouent un rôle primordial au sein des réseaux informatiques. Ils offrent des solutions ouvertes pour fournir la mobilité ainsi que des services essentiels là où l'installation d'infrastructures n'est pas possible.

Les réseaux mobiles sans fil, peuvent être catégorisés en deux classes : les réseaux avec infrastructure qui utilisent généralement le modèle de la communication cellulaire, et les réseaux sans infrastructure ou les réseaux ad hoc.

Un réseau de capteurs sans fil (RCSF) est un réseau ad hoc, composé d'un ensemble de nœuds généralement dédiés à la collecte d'information, capables de communiquer entre eux afin de réaliser des tâches diverses. La facilité de déploiement de ce type de réseau constitue un atout qui les rend facilement intégrables dans une grande variété d'applications comme les applications militaires, environnementales, domotiques, industrielles, etc. Ses nœuds étant d'une petite taille, souvent de l'ordre d'une pièce de monnaie, cela permet leur déploiement dans des endroits à accès difficile ou dangereux pour l'être humain [1].

Du fait de la miniaturisation et de la production à grande échelle et à faible coût, les nœuds d'un RCSF sont limités en capacité de calcul et de mémoire ainsi qu'en ressources énergétiques. Ces contraintes représentent un véritable challenge pour le déploiement d'un réseau de capteur. Les protocoles nécessaires au fonctionnement des RCSF comme les protocoles de routage et de sécurité doivent prendre en compte les contraintes des nœuds du réseau tout en économisant le plus possible leur consommation d'énergie. Le but des nouvelles propositions et des nouveaux protocoles dédiés aux RCSF est de faire un compromis entre la qualité du service rendu par ces solutions et le respect des contraintes imposées par les nœuds.

La solution que nous avons proposée consiste à l'implémentation d'un protocole de routage adapté aux caractéristiques des RCSF, basé sur l'adaptation du protocole AODV par optimisations des nombre de requêtes utilisés pour la recherche et le maintient de la route, en proposant aussi une solution de sécurité des communications radio entre les nœuds de réseau, basé sur l'implémentation de l'algorithme de chiffrement symétrique de type RC4 et asymétriques de type ECC (Elliptic Curve Cryptography).

Le premier chapitre comporte des généralités sur les réseaux de capteurs sans fil : leurs architectures de communication, les contraintes de conception et leurs domaines d'applications, Nous allons présenter également les contraintes du routage, les critères de

performances des protocoles de routage pour les réseaux RCSF, en mentionnant les principaux protocoles de routage proposé pour les RCSF. Nous allons cités aussi, les exigences en sécurité et les vulnérabilités dans les RCSF ainsi que les solutions proposés.

Le second chapitre est consacré en premier partie à l'étude du protocole AODV et AODV optimisé, leur principe fonctionnement et leur caractéristiques. Dans la deuxième partie, nous allons présenter la solution de sécurité des communications radio dans le RCSF, adoptée pour notre projet, basé sur l'implémentation de l'algorithme de chiffrement symétrique RC4 et asymétrique ECC, toute en détaillant le fonctionnement de ces algorithmes.

Dans le troisième chapitre, nous allons présenter l'environnement de développement et d'implémentation exploité dans notre projet, nous décrivons les composants du capteur CM5000 utilisé, nous montrons les principaux composants, interfaces et fonctions utilisés pour l'implémentation des protocoles de routage AODV et AODV optimisé et les algorithmes de chiffrement symétrique RC4 et asymétrique ECC, dans les capteurs Telosb.

Le quatrième chapitre porte sur la présentation des différents tests effectués relatifs à la mesure des performances des protocoles de routage AODV et AODV optimisé et les algorithmes de chiffrement symétrique RC4 et asymétrique ECC implémentés, en analysant les résultats obtenus. Nous allons exposer dans ce chapitre, notre application web développé pour le contrôle de la température dans un environnement industrielle.

CHAPITRE I : RCSF ET LES PROTOCOLES DE ROUTAGE ET DE SECURITE DEDIES

I.1. Introduction :

Les récents progrès et les nombreuses avancées technologiques dans les domaines de la micro-électronique et les progrès atteints dans les domaines d'intégration et de la miniaturisation ont permis la fabrication des capteurs, faibles en cout et de plus en plus performants, capables de se disposer dans l'environnement de manière aisée sans altération du paysage ambiant. Sur un champ de captage, les capteurs coopèrent entre eux sans aucune intervention externe (humaine, infrastructure de base…etc.) pour former une infrastructure de communication dite réseau de capteurs sans fil.

En raison des caractéristiques particulières des RCSF qui les distinguent des autres réseaux traditionnels sans fil, notamment en capacités de calculs, de stockage et d'énergie des nœuds capteurs, le routage et la sécurité sont des tâches ardues et compliquées nécessitant un travail de collaboration de tous les nœuds appartenant au réseau

Dans ce chapitre, nous allons présenter les réseaux de capteurs sans fil, leurs architectures de communication, les contraintes de conception et leurs domaines d'applications. Nous allons discuter également sur les contraintes du routage, les critères de performances des protocoles de routage pour les réseaux RCSF, et les principaux protocoles de routage proposés pour les RCSF. Nous allons cités les exigences en sécurité et les vulnérabilités dans les RCSF ainsi que les solutions proposés.

I.2. Réseaux de capteurs sans fil:

I.2.1. Définition de RCSF

Un réseau de capteurs peut être vu comme un réseau de microsystèmes disséminés dans un espace donné et communicant entre eux via une liaison sans fil. L'espace où agissent les capteurs s'appelle un champ de captage. Ce qui est intéressant dans les réseaux de capteurs, c'est que les nœuds sont souvent composés d'un grand nombre de micro-capteurs capables de récolter et de transmettre des données environnementales d'une manière autonome.

Par conséquent, on peut définir un Réseau de Capteurs Sans Fil (RCSF) ou "Wireless Sensor Network" (WSN) comme un ensemble de dispositifs très petits, nommés nœuds capteurs, variant de quelques dizaines d'éléments à plusieurs milliers. Dans ces réseaux, chaque nœud est capable de surveiller son environnement et de réagir en cas de besoin en envoyant l'information collectée à un ou plusieurs points de collecte, à l'aide d'une connexion radio.

I.2.2. Architecture physique d'un capteur

Un capteur sans fil est un petit dispositif électronique capable de mesurer une valeur physique environnementale (température, lumière, pression, etc.) et de la communiquer à un centre de contrôle via une station de base. Les progrès conjoints de la microélectronique, des technologies de transmission sans fil et des applications logicielles ont permis de produire à coût raisonnable des micro-capteurs de quelques millimètres cubes de volume, susceptibles de fonctionner en réseaux. Un capteur est composé de quatre unités de base (voir Figure I-1):

a) **L'unité d'acquisition** : est composée d'un capteur qui va obtenir des mesures numériques sur les paramètres environnementaux et d'un convertisseur Analogique/Numérique qui va convertir l'information relevée et la transmettre à l'unité de traitement.

b) **L'unité de traitement** : est composée de deux interfaces, une interface pour l'unité d'acquisition et une interface pour l'unité de transmission. Cette unité est également composée d'un processeur et d'un système d'exploitation spécifique. Elle acquit les informations en provenance de l'unité d'acquisition et les envoie à l'unité de transmission.

c) **L'unité de transmission** : est responsable de toutes les émissions et réceptions des données via un support de communication radio.

Ces trois unités sont alimentées par une batterie comme le montre la figure ci dessous:

I.2.3. Architecture d'un RCSF

Tous les capteurs respectent globalement la même architecture basée sur un noyau central autour duquel s'articulent les différentes interfaces d'entrée-sortie, de communication et d'alimentation. La figure suivante montre un exemple d'un réseau de capteurs.

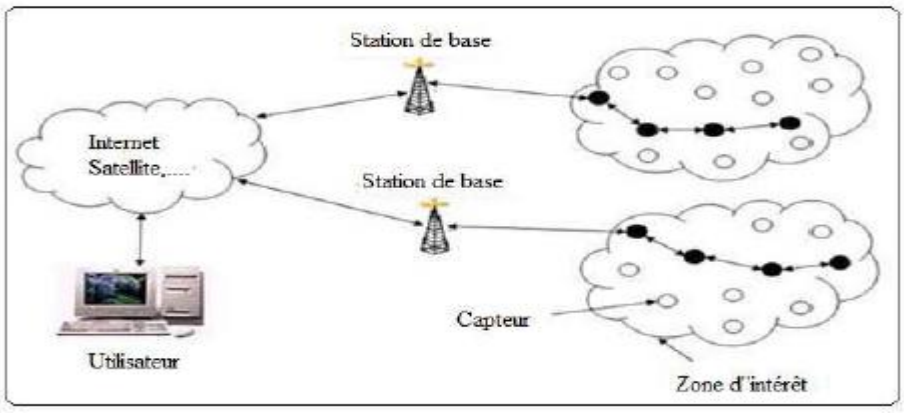


Figure I-1 : Exemple de réseaux de capteurs

Un RCSF est composé d'un ensemble de nœuds capteurs qui sont organisés en champs «Sensor Fields». Chacun de ces nœuds a la capacité de collecter des données et de les transférer au nœud passerelle par l'intermédiaire d'une architecture multi-sauts. Le nœud passerelle transmet ensuite ces données par Internet ou par satellite à l'ordinateur central pour analyser ces données et prendre des décisions [2].

Selon le type des capteurs qui sont organisés en champs «Sensor Fields», on peut distinguer deux architectures :

a) Les réseaux de capteurs sans fil plats : composés des nœuds qui ont les mêmes ressources en terme d'énergie, de calcul et de mémoire. Cette architecture est utilisée pour des réseaux ayant un déploiement massif des nœuds et un mode de communication sans fil en multi-sauts.

b) Les réseaux de capteurs sans fil hiérarchiques : composés des nœuds qui ne possèdent pas les mêmes rôles et par conséquent les mêmes ressources.

I.2.4. Fonctionnement des RCSF :

Il y a deux méthodes pour collecter les informations dans un réseau de capteurs [3]:

I.2.4.1. A la demande

Lorsque nous souhaitons avoir l'état de la zone de couverture à un moment donnée, le nœud puits émet des brodcasts vers toute la zone pour que les capteurs montent leur dernier relevé vers le puits. Les informations sont alors acheminées par le biais d'une communication multi-sauts comme s'est illustré par la figure I-2.

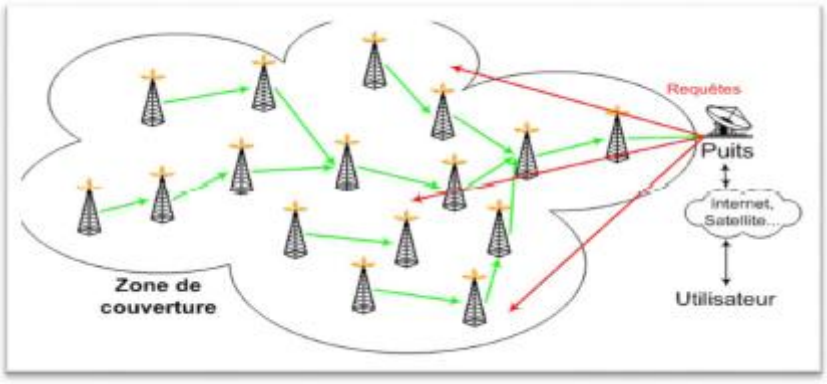


Figure I-2 : Collecte d'informations à la demande

I.2.4.2. Suite à un événement

Ce mode de communication de l'information est utilisé lorsqu'un événement se produit en un point de la zone de couverture (changement brusque de température, mouvement...), les

capteurs situés à proximité remontent alors les informations relevées et les acheminent jusqu'au puits (figure I-3).

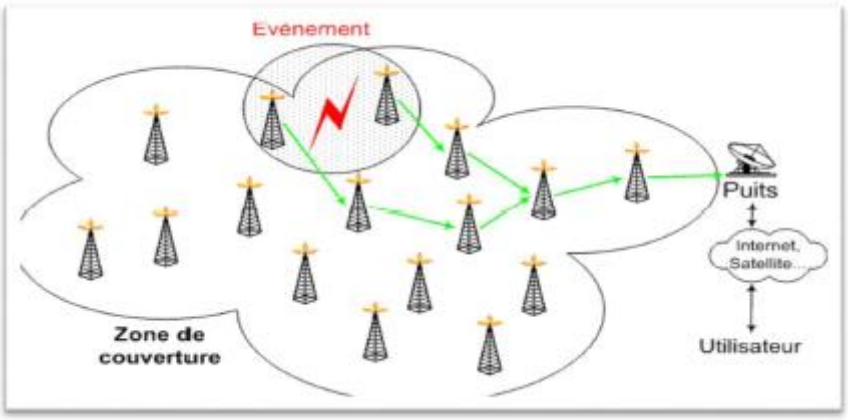


Figure I-3 : Collecte d'informations suite à un événement

I.2.5. Contraintes de conception des RCSF

Les principaux facteurs et contraintes influençant l'architecture des réseaux de capteurs peuvent être résumés comme suit:

a. **Tolérance aux pannes** : Les algorithmes et protocoles doivent tenir compte du fait qu'un nœud peut cesser de fonctionner par manque d'énergie ou parce qu'il a été détruit accidentellement par un animal ou intentionnellement par des personnes. Ils devront adapter leur niveau de tolérance aux pannes en fonction de l'hostilité du milieu dans lequel est déployé le réseau [7].

b. **Le facteur d'échelle (Scalability):** Le nombre de nœuds de capteurs augmente sur un réseau sans fil et ce nombre peut atteindre le million. Un nombre aussi important de nœuds engendre beaucoup de transmissions entre les nœuds et peut imposer des difficultés pour le transfert de données [2].

c. **Environnement de déploiement :** Dans la majorité des applications, les nœuds capteurs sont déployés dans des zones distantes, hostiles et sans aucune surveillance ni intervention humaine. Les capteurs doivent être conçus pour résister aux différentes conditions climatiques telles que la chaleur, l'humidité, le froid, la pression …etc [8].

d. **La topologie de réseau** : Le déploiement d'un grand nombre de noeuds nécessite une maintenance de la topologie. Cette maintenance consiste en trois phases : déploiement, post-déploiement (les capteurs peuvent bouger, ne plus fonctionner,...) et redéploiement de nœuds additionnels quand le taux de couverture de la zone d'intérêt diminue au-dessous d'un certain seuil [2].

e. **Le faible débit** : Le débit garanti par les capteurs est trop limité. D'où, il faudrait optimiser l'information envoyée au centre de contrôle, par exemple utiliser des nœuds agrégateurs pour agréger l'information localement avant de l'envoyer.

f. **Contrainte énergétique** : L'énergie est considérée comme la contrainte principale dans un réseau de capteurs. Déjà, comme pour tout réseau sans fil, il est important de tenir compte de cette contrainte car la plupart des machines fonctionnent sur batterie. Dans les réseaux de capteurs, il est pratiquement impossible de recharger les batteries de par le nombre élevé de capteurs déployés et de par la difficulté de l'environnement dans lesquels ils peuvent se trouver. Une fois vide, le capteur est considéré comme mort ou hors service. L'objectif à atteindre devient l'augmentation de la durée de vie du réseau de capteurs.

g. **Bande passante limitée** : afin de minimiser l'énergie consommée lors de transfert de données entre les nœuds, les capteurs opèrent à bas débit. Typiquement, le débit utilisé est de quelque dizaines de Kb/s. Un débit de transmission réduit n'est pas handicapant pour un réseau de capteur ou les fréquences de transmissions ne sont pas importantes [17].

h. **Sécurité physique limitée** : Les RCSF sont plus touchés par le paramètre de sécurité que les réseaux filaires classiques, cela se justifier par les contraintes et limitation physiques qui font que le contrôle des données transférées doit être minimisé. Les RCSF connaissent actuellement une grande extension et une large utilisation dans différents types d'applications, dont celles exigeant une grande sécurité. Leurs contraintes font que l'application des mesures classiques de sécurités, sur les RCSF, est restreinte [17].

i. **Agrégation de données** : les données fournîtes par les capteurs voisins sont très corrélées spatialement et temporellement. Ceci peut engendrer la réception par la station de base d'information redondante. Réduire la quantité d'information redondante transmise par les capteurs permet de réduire la consommation d'énergie dans le réseau et ainsi d'améliorer sa durée de vie.

I.2.6. Applications des RCSF :

a. **Domaine militaire** : Les RCSF sont le résultat de la recherche militaire. Ils sont utilisés dans la surveillance des champs de bataille pour connaitre exactement la position, le nombre, l'armement (chimique, biologique, nucléaire…etc.), l'identité et le mouvement des soldats et ainsi empêcher leur déploiement sur des zones à risques [8].

b. **Surveillance médicale** *:* En implantant sous la peau de mini capteurs vidéo, on peut recevoir des images d'une partie du corps en temps réel sans aucune chirurgie. On peut ainsi surveiller la progression d'une maladie ou la reconstruction d'un muscle [2].

c. **Application domestique :** Avec le développement technologique, les capteurs peuvent être embarqués dans des appareils, tel que les aspirateurs, les fours à micro-ondes, les réfrigérateurs, les magnétoscope, 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. En plaçant, sur le plafond ou dans le mur, des capteurs, on peut économiser l'énergie en gérant l'éclairage ou le chauffage en fonction de la localisation des personnes et seulement si nécessaires [18].

d. **Détection d'intrusions** : En plaçant, à différents points stratégiques, des capteurs, on peut ainsi prévenir des cambriolages ou des passages de gibier sur une voie de chemin de fer (par exemple) sans avoir à recourir à de coûteux dispositifs de surveillance vidéo [7].

e. **Domaine agricole et environnemental :** Les réseaux de capteurs sans fil sont très utiles dans la protection de l'environnement. Ils peuvent être utilisés pour la détection des feux de forêts, des inondations, surveillance des volcans, contrôle de la qualité de l'air par le suivi de l'évolution de la densité moyenne de Co_2, le déplacement des animaux…etc. Dans le domaine agricole, on cite le déploiement des capteurs sur un champ agricole afin d'identifier les zones sèches et permettre leur irrigation à temps [8].

f. **Applications du transport** : Il est possible d'intégrer des nœuds capteurs au processus de stockage et de livraison. Le réseau ainsi formé, pourra être utilisé pour connaître la position, l'état et la direction d'un paquet ou d'une cargaison [6].

g. **Contrôle de la pollution** : Les capteurs utilisés dans des applications industrielles offrent la possibilité de détecter et de contrôler des fuites de gaz ou de produits chimiques. Ces applications permettent de donner l'alerte en un temps record et de pouvoir suivre l'évolution de la catastrophe [2].

I.3. Protocole de routage des RCSF:

I.3.1. Définition de routage :

Le routage est un processus qui permet de sélectionner des chemins dans un réseau pour transmettre des données depuis un expéditeur jusqu'à un ou plusieurs destinataires. On parle de routage dans différentes domaines: réseaux téléphoniques et réseaux de transports.

Le problème de routage consiste à déterminer un acheminement optimal des paquets (de messages, de produits …etc.) à travers le réseau au sens d'un certain critère de performance. Il consiste à trouver l'investissement de moindre coût en capacités nominales et de réserves qui assure le routage du trafic nominal et garantit sa surveillance en cas de panne.

I.3.2. Contraintes de routage dans les réseaux de capteurs sans fil

Le routage dans les réseaux de capteurs diffère de celui des réseaux Ad Hoc dans les points suivants [9] :

- ✓ Il n'est pas possible d'établir un système d'adressage global pour le grand nombre de nœuds.
- ✓ Les applications des réseaux de capteurs exigent l'écoulement de données mesurées depuis des sources multiples vers la destination finale.
- ✓ Les différents capteurs peuvent générer les mêmes données à proximité d'un phénomène (problème de la redondance des données).
- ✓ Les nœuds capteurs exigent ainsi une gestion soigneuse des ressources.

I.3.3. Les critères de performance des protocoles de routage en RCSF

La performance des réseaux de capteurs sans fil est fondée sur les facteurs suivants :

● **Evolutivité** : l'évolutivité est un facteur important dans les réseaux de capteurs sans fil. Une zone de réseau n'est pas toujours statique, elle change selon les besoins des utilisateurs. Tous les nœuds dans le domaine du réseau doivent être évolutifs ou être en mesure de s'adapter aux changements dans la structure du réseau en fonction de l'utilisateur.

● **L'énergie** : chaque nœud utilise peu d'énergie pour des activités telles que la détection, le traitement, le stockage et la transmission. Un nœud dans le réseau doit savoir combien d'énergie sera utilisée pour effectuer une nouvelle tâche à laquelle il est soumis. L'énergie consommée peut varier selon le type de fonctionnalité ou l'activité qu'il a à accomplir.

● **Le temps de traitement** : il se réfère au temps pris par le nœud dans le réseau pour assurer l'ensemble de l'opération commençant par la détection, le traitement des données ou le stockage de données, la transmission ou la réception sur le réseau.

● **Le schéma de transmission** : la transmission de données par les nœuds de capteurs vers la destination ou la station de base se fait par un schéma de routage à un seul saut ou à multi-saut.

● **Synchronisation** : dans les communications radio entre les nœuds d'un RCSF, les capteurs écoutent en permanence les transmissions et consomment de l'énergie s'ils ne sont pas synchronisés les uns les autres. Pour cela, un nœud doit avoir la même notion de temps pour se mettre en veille et se réveiller que ses voisins.

● **Contrôle de paquets :** un paquet envoyé avant la transmission entre deux nœuds est appelé le paquet de contrôle. Le paquet de contrôle contient le nombre de bits de données envoyés, l'adresse du nœud de destination et certaines informations qui contribuent à éviter les collisions pendant la transmission.

I.3.4. Types de routage

Le routage dans les réseaux de capteurs sans fil se classe généralement selon plusieurs critères [19]:

I.3.4.1. Selon la structure du réseau

a. Routage à plat :

Appelé également routage centré où tous les nœuds ont les mêmes tâches à accomplir. Elle se base sur la collaboration de tous les nœuds du réseau. Parmi leurs avantages, la simplicité d'où la possibilité d'établir des communications sans surcoût où chaque nœud n'aura besoin que des informations de ses voisins directs. L'inconvénient est l'épuisement des ressources en énergie des nœuds proches de la station de base car tout le trafic vers cette dernière passe obligatoirement par eux.

b. Routage hiérarchique :

Cette approche est basée sur la formation de clusters (zones communes). Le principe est de router les données récoltées par chaque nœud du cluster à son chef de zone (Cluster Head), qui et après des traitements sur leurs parties communes, les transmettra à la prochaine destination. L'avantage est la réduction des coûts en communication et en énergie en minimisant le nombre de messages circulant sur le réseau, étant donné que les CHs appliquent des fonctions d'agrégat sur les données du cluster ce qui permet de les combiner. L'inconvénient concerne la taille du réseau. En outre, quand la taille du réseau augmente, le processus d'élection du Cluster Head devient critique et gourmand en ressources.

c. Routage basé sur la localisation :

L'identification des emplacements géographiques des nœuds capteurs sur la zone de captage est d'une importance capitale pour les mécanismes de routage de données dans les

RCSF. Ces informations de localisation permettent le calcul des positions des capteurs et les distances qui les séparent afin de construire les chemins les plus courts entre un nœud source et sa destination. Cette approche de routage est plus économe en énergie car elle dispense les nœuds capteurs d'employer les méthodes aléatoires ou probabilistes pour rechercher les routes. De plus, la localisation des nœuds permet de diffuser des requêtes uniquement à ces régions et éviter leur diffusion en mode broadcast et ainsi réduire le nombre de transissions d'une manière significative. L'inconvénient est la nécessité d'équiper les nœuds capteurs avec un système de localisation par satellite comme le GPS qui consomment énormément d'énergie.

I.3.4.2. Selon le Mode de transmission

a. Routage proactif :

Le calcul de routes se fait à priori ce qui facilite l'acheminement des données. Les informations des chemins à suivre par chaque donnée source vers une destination sur le réseau sont stockées dans une table de routage. Les tables de routage doivent être mises à jour régulièrement afin de corriger certains chemins coupés en raison du changement de topologie dus aux défaillances ou à la mobilité de certains nœuds capteurs. Cette mise à jour est assurée par la diffusion périodique des paquets de contrôle sur le réseau, ce qui n'est pas évident pour des réseaux de grande taille comme les réseaux de capteurs sans fil. L'établissement de routes se fait indépendamment des besoins réels de l'application et un bon nombre de ces routes est sauvegardé pour ne jamais être utilisées. Une autre limite concerne la taille des tables de routage, notamment pour des réseaux de grande taille, qui pourrait dépasser les capacités de stockage des nœuds capteurs.

b. Routage réactif :

Egalement appelé routage à la demande, le routage réactif permet de créer les routes selon les besoins de l'application. Lorsqu'une requête est diffusée sur le réseau, la procédure de découverte des routes est lancée par les nœuds concernés par cette requête, et les réponses sont acheminées sur les routes créées. Cette procédure est lancée également pour des applications orientées événements, pour chaque événement intéressant détecté. L'avantage d'établir des routes à la demande est la conservation d'énergie par rapport au routage proactif. La recherche de routes peut causer des lenteurs pour l'acheminement des données ce qui n'est pas approprié aux applications interactives et temps-réel [20].

c. Routage hybride (à la fois proactif et réactif) :

C'est une combinaison des deux concepts de routage proactif et réactif. Des tables de routage sont stockées sur les nœuds capteurs de façon à établir des routes sur leur voisinage proche (généralement en deux sauts maximums). Au-delà de leur voisinage, le routage devient réactif et des procédures de recherche de routes sont lancées. Cette approche combine les avantages des deux autres approches proactive et réactive et réduit considérablement la taille des tables de routage ainsi que les délais d'établissement de routes.

I.3.4.3. Selon Les fonctions du protocole [6] :

a. Protocole de routage multi-chemin :

Il se base sur l'adoption de plus qu'un chemin menant vers la destination, et ce, pour avoir des chemins de secours si jamais le chemin principal serait rompu.

b. Protocole de routage basé sur la négociation des données :

En détectant le même phénomène, les nœuds capteurs inondent le réseau par les mêmes paquets de données. Ce problème de redondance peut être résolu en employant des protocoles de routage basés sur la négociation. En effet, avant de transmettre, les nœuds capteurs négocient entre eux leurs données en échangeant des paquets de signalisation spéciales, appelés META-DATA. Ces paquets permettent de vérifier si les nœuds voisins disposent des mêmes données à transmettre. Cette procédure garantit que seules les informations utiles seront transmises et élimine la redondance des données.

c. Protocole de routage basé sur la QoS :

Ce type de protocoles tend à satisfaire certaines métriques, pendant la transmission des données vers la destination finale. Parmi ces métriques, nous citons : le délai de bout en bout, la gigue, PDR (Paquet Delivery Ratio), et l'énergie consommée.

I.3.5. Les protocoles de routage proposé pour les RCSF

Nous citons dans cette section quelques protocoles de routage proposés pour les réseaux de capteurs sans fil :

I.3.5.1. LEACH (*Low*-Energy Adaptive Clustering. Hierarchy) :

Son principe est de former des zones communes de calcul et de traitements en se basant sur la puissance du signal et le niveau d'énergie des nœuds capteurs. Chaque zone est dirigée par un chef de zone, jouant le rôle d'agrégateur et de routeur, en effectuant des traitements sur

les données reçues de son cluster et leur expédition vers la prochaine destination. Ce rôle de chef de zone est échangé entre les nœuds d'un cluster afin de répartir équitablement la consommation d'énergie entre eux.

I.3.5.2. TEEN et APTEEN (Threshold sensitive Energy Efficient sensor Network protocol)

TEEN est un protocole hiérarchique conçu pour être sensible aux changements improviste des attributs détectés tels que la température. L'architecture du réseau est basée sur un groupement hiérarchique où les nœuds les plus proches forment des clusters. Après la construction des clusters, le cluster head diffuse deux seuils aux nœuds. Qui sont la valeur minimale d'un attribut pour pouvoir être transmis et le degré minimale du changement de cet attribut. Le TEEN adaptatif (APTEEN) est une extension du TEEN basée sur la capture périodique des données et la réaction aux événements temps-réel. Quand la station de base forme les clusters, les cluster-heads diffusent les attributs, les seuils et le plan de transmission à tous les nœuds et effectuent également l'agrégation des données afin d'économiser l'énergie [9].

I.3.5.3. Les protocoles de routage «PEGASIS & Hierarchical-PEGASIS»

Power-Efficient GAthering in Sensor Information Systems (PEGASIS) est une version améliorée du protocole LEACH. PEGASIS forme des chaînes plutôt que des clusters de nœuds de capteurs afin que chaque nœud transmette et reçoive uniquement des données d'un voisin. Un seul nœud est sélectionné à partir de cette chaîne pour transmettre à la station de base. L'idée de PEGASIS est qu'il utilise tous les nœuds pour transmettre ou recevoir des données avec ses plus proches voisins. Il déplace les données reçues de nœud à nœud, puis les données seront agrégées jusqu'à ce qu'elles atteignent tous la station de base. Donc, chaque nœud du réseau est tour à tour un chef de file de la chaîne, ainsi que responsable pour transmettre l'ensemble des données recueillies et fusionnées par la chaîne de nœuds au niveau de la station de base [2].

I.3.5.4. Le protocole de routage SPIN :

SPIN (Sensor Protocols for Information via Negociation) est un protocole de routage qui se base sur le principe de métadonnées lors du processus de diffusion de requêtes sur le réseau. Le récepteur de la requête a la possibilité de choisir d'accepter la donnée ou non. C'est le principe de la négociation. SPIN est considéré comme le premier protocole data centric destiné aux RCSF. Tous les nœuds du réseau sont traités comme des nœuds de destination

dont la tâche principale est de recueillir un point de vue globale sur leur environnement. Egalement, SPIN s'attaque à la redondance de données transmises sur un réseau de capteurs sans fil [21].

I.3.5.5. Directed Diffusion (DD) :

Directed Diffusion ou diffusion dirigée est un protocole de propagation de données. C'est l'un des protocoles les plus intéressants dans le routage data Centric des RCSF. Son utilité réside dans sa capacité à réduire considérablement le nombre d'opérations de la couche réseau par la création de plusieurs chemins pour le routage d'informations et ainsi permettre une économie d'énergie conséquente aux capteurs. Cette réduction d'opération passe par une mise en place de quatre éléments : i) la nomination des données (décrire les intérêts) ; ii) propagation des intérêts ; iii) propagation des données ; iv) renforcement des chemins de transport de données. Le protocole DD à travers ces quatre éléments définit au préalable la nature de la donnée à capter, décrit cette donnée sous forme d'intérêt et diffuse cette requête sur le réseau. Les nœuds concernés commencent à capter et propager les données sur des chemins différents. A la réception des premières données par la station de base, elle procède au choix puis au renforcement des meilleurs chemins [8].

I.3.5.6. Rumour Routing :

Le protocole Rumour Routing utilise un procédé probabiliste, similaire au gossiping, afin de trouver un compromis entre l'inondation des intérêts et la propagation de données. Ce procédé probabiliste repose sur la méthode Monté-Carlo qui suppose que la probabilité que deux lignes se croisant au sein d'une région rectangulaire est de 0.69, et si on utilise cinq lignes passant par un point, la probabilité qu'une autre ligne se croise avec l'une des cinq lignes est de 0.997. Par conséquent, si on considère la source et le puits deux points distincts, on établit un certain nombre de mi-chemin depuis la source et d'autres mi-chemin depuis le puits et on aura une forte probabilité que deux mi-chemin se joignent créant ainsi un chemin complet entre la source et la destination. Ce procédé permet d'éviter l'inondation de données [22].

I.3.5.7. Le protocole de routage «GEAR »

Le protocole de routage GEAR (Geographic and Energy Aware Routing) a été suggéré par Y. Yu, D. Estrin. Il consiste à utiliser l'information géographique lors de la diffusion des requêtes aux régions cibles car les requêtes contiennent souvent des données géographiques. L'idée est de restreindre le nombre de données dans la diffusion dirigée en prenant en

considération uniquement une certaine région, plutôt que d'envoyer les données à l'ensemble du réseau. Avec le protocole GEAR, chaque noeud maintient le coût pour atteindre la destination en passant par ses voisins. Ce coût est divisé en deux parties : un coût estimé et un coût d'apprentissage. Le coût estimé est une combinaison de l'énergie résiduelle et de la distance jusqu'à destination. Le coût d'apprentissage est un raffinement du coût estimé qu'un nœud dépense pour le routage autour des trous dans le réseau [2].

I.3.5.8. Le protocole MECN (Minimum Energy Communication Network):

C'est un protocole de routage géographique conçu pour des nœuds capteurs équipés d'un GPS à basse puissance. L'idée est de trouver un sous réseau composé d'un nombre minimum de nœuds, proches entre eux, afin d'avoir une puissance de transmission faible et moins couteuse en énergie entre deux nœuds particuliers du réseau.

I.3.5.9. Le protocole AODV (Ad-hoc On Demand Distance Vector)

C'est un protocole de routage conçu par Charles E.Perkins et Elizabeth M. Royer basé sur le principe des vecteurs de distance et appartient à la famille des protocoles réactifs. Il représente essentiellement une amélioration de l'algorithme proactif DSDV mais réduit l'overhead (nombre de diffusions de messages) en ne calculant les routes que sur demande(AODV). Ce protocole utilise les deux mécanismes "découverte de route" et "maintenance de route", en plus du routage "nœud par nœud " et construit les routes par l'emploi d'un cycle de requête "route request/route reply ".

I.4. La sécurité dans les RCSF

I.4.1. Exigences en sécurité:

Un réseau de capteur est un type spécial de réseaux. Il partage quelques vulgarisations avec un réseau informatique typique, mais pose également des conditions uniques de ses propres caractéristiques. Par conséquent, un protocole de sécurité pour un RCSF, doit satisfaire une ou plusieurs conditions de sécurité, à savoir :

I.4.1.1. La confidentialité :

Elle consiste à s'assurer que personne ne pourra interférer dans la communication entre deux nœuds afin de récupérer les données lors de leur transfert. Les algorithmes cryptographiques de nos jours peuvent assurer une confidentialité suffisante en utilisant des clés de grande taille. Cependant, ces algorithmes sont lourds en termes de mémoire et de calcul, de plus pour les RCSF il faut considérer le temps et l'énergie qu'ils consomment. Il est

nécessaire d'adapter ces algorithmes pour qu'ils puissent être un bénéfice et non pas un handicap pour les RCSF [1].

I.4.1.2. Intégrité des données :

Un nœud intrus (adversaire) peut modifier les données transférées. Par exemple, un nœud malveillant peut ajouter quelques fragments ou manœuvrer les données dans un paquet. Ce nouveau paquet peut alors être envoyé au récepteur original. La perte ou les dommages de données peut même se produire sans présence d'un nœud malveillant dû à l'environnement dur de communication. Ainsi, l'intégrité des données s'assure qu'aucune donnée reçue n'a été changée en transit [23].

I.4.1.3. L'authentification d'un nœud :

C'est le fait de pouvoir vérifier et connaître avec une grande certitude l'identité d'un nœud. Elle pourra être assurée implicitement dans un système à chiffrement symétrique, car dans un tel système, deux entités partagent d'une façon sûre une seule clé secrète. Alors, seule l'entité qui possède la clé secrète pourra chiffrer/déchiffrer les données. Dans les systèmes asymétriques, l'authentification pourra être assurée lors de l'utilisation d'un certificat d'identification. C'est une relation sûre entre la clé publique et l'identité de l'entité. Le destinataire doit être capable d'identifier l'expéditeur en contactant un tiers de confiance qui a la responsabilité de signer, délivrer et vérifier la validité des certificats.

I.4.1.4. La non-répudiation :

C'est le fait qu'aucun nœud ne pourra nier (désavouer) le fait d'avoir échanger des informations. Autrement dit, la non-répudiation de l'origine prouve que les données ont été envoyées, et la non-répudiation de l'arrivée prouve qu'elles ont été reçues.

I.4.1.5. Fraîcheur De Données

Même si la confidentialité et l'intégrité des données sont assurées, il est nécessaire d'assurer la fraîcheur de chaque message. Officieusement, la fraîcheur de données suggère que les données soient récentes, et elles s'assurent qu'aucun vieux message n'a été rejoué. Cette condition est particulièrement importante quand il y a des stratégies de partage des clefs utilisées dans la conception. Des clefs typiquement partagées doivent être changées avec le temps. Cependant, cela prend du temps pour de nouvelles clefs partagées d'être propagées au réseau entier. Dans ce cas-ci, il est facile pour l'adversaire d'employer une attaque de rejouer. Pour résoudre ce problème un compteur relatif au temps différent, peut être ajouté dans le paquet pour assurer la fraîcheur de données [23].

I.4.2. Vulnérabilités de la sécurité dans les RCSF

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

I.4.2.1. Limitation de ressources :

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

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

I.4.2.2. La communication sans fils :

En plus de fournir un déploiement simple, la communication sans fil a l'avantage d'offrir l'accès à des endroits difficilement accessibles tels que des terrains désastreux et hostiles. Cela introduit de nombreuses failles de sécurité à deux niveaux différents, attaque sur la construction et maintenance des routes, et attaque des données utiles par injection, modification ou suppression de paquets. En outre, la communication sans fil introduit d'autres vulnérabilités à la couche liaison en ouvrant la porte à des attaques de brouillage et de style déni de service par épuisement des batteries.

I.4.2.3. Vulnérabilités matériel:

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

Par conséquent, un adversaire "bien équipé" peut extraire des informations cryptographiques des nœuds capteurs. Comme la mission d'un RCSF est généralement sans surveillance, le potentiel d'attaquer les nœuds et de récupérer leur contenu est important.

Ainsi, les clefs cryptographiques et informations sensibles devraient être gérées d'une manière qui augmente la résistance à la capture des nœuds [23].

I.4.2.4. Vulnérabilités réseau :

a) Attaques sur le protocole de routage : l'absence d'infrastructure fixe pénalise l'ensemble du réseau dans la mesure où il faut faire abstraction de toute entité centrale de gestion pour l'accès aux ressources ;

a) Attaques sur la confidentialité et l'authenticité des paquets transmises : les mécanismes de routage sont d'autant plus critiques dans les réseaux RCSF que les attaques peuvent exploiter le comportement coopératif durant le processus de routage, ou chaque nœud participe à l'acheminement des paquets à travers le réseau.

Les messages de routage transitent sur les ondes radio. Donc ces messages peuvent être interceptés, modifiés, interrompus ou relayés ce qui rend les protocoles de routage vulnérable [17].

I.4.2.5. Vulnérabilités Applicatives :

Vu le nombre important des nœuds capteurs dans les RCSF et la corrélation des données captées, l'agrégation de données est devenu cruciale, cela fait qu'elle est ciblée par beaucoup d'attaques afin de fausser les données reçues par le puits. Par conséquent, la sécurisation de cette opération devient de plus en plus indispensable. En route vers le puits, les données passent par un ou plusieurs nœuds agrégateurs et si l'un de ces nœuds est compromis, le message agrégé sera donc altérer, et son altération fausse les altérations qui suivent jusqu'à son arrivée au puits. [17]

I.4.3. Solutions adaptées aux communications des RCSF

Plusieurs solutions ont été développées pour contrer les attaques sur les RCSF. On distingue des solutions spéciales au routage où la fonction de sécurité concerne la protection de la donnée transmise, d'autres solutions sont conçues pour protéger les communications entre les nœuds capteurs en assurant la fiabilité des liaisons par des fonctions cryptographiques, et enfin des solutions qui se chargent de protéger les nœuds du réseau, de détecter et d'exclure les nœuds malicieux par des mécanismes de détection d'intrusions. Dans, on propose les mécanismes simples, les plus efficaces et les plus appropriés aux RCSF, regroupés selon l'objectif de la solution proposée :

I.4.3.1. Partitionnement de données

C'est une solution proposée dans le but d'empêcher les attaquants de récupérer l'intégralité des informations qui circulent sur le réseau. Le principe est de découper, au

niveau du nœud source, l'information à transmettre en plusieurs paquets de tailles fixes et chaque paquet de données sera envoyé sur un chemin différent. L'entité de destination finale, et après la réception de tous les paquets de données, procède à la reconstitution du message initial émis par le nœud source. Afin de saisir intégralement l'information échangée entre la source et sa destination, un attaquant doit scruter tous les chemins utilisés dans la transmission du message, ce qui est difficile voire impossible vu la taille des réseaux et le nombre important de chemins possibles entre chaque paire de nœuds. Cependant, cette solution est gourmande en énergie car elle implique un grand nombre de nœuds pour acheminer chaque paquet vers l'entité destinatrice [8].

I.4.3.2. Détection d'intrusions

a. Détection d'intrusions par localisation :

Les nœuds capteurs doivent être équipés de moyens qui permettent de connaitre leurs positions géographiques (exemple le GPS). Ce type de nœuds capteurs sont dits capteurs balises. Si un capteur souhaite faire partie du réseau, il adresse une demande d'insertion aux capteurs balises afin d'estimer sa localisation par rapport au domaine d'écoute. Les capteurs balises vont quadriller par la suite leurs zones d'écoutes respectives et demandent aux nœuds qui ont reçu la demande d'insertion de voter pour la zone de quadrillage qu'ils peuvent entendre. La zone qui obtient le plus de voix sera considérée comme celle où est censé se trouver le capteur.

b. Détection d'intrusions par indice de confiance :

Contrairement à la solution de détection d'intrusions présentée précédemment où des moyens supplémentaires et couteux sont exigés, la détection d'intrusions par indice de confiance consiste à générer une alerte pour tout scénario ou comportement suspect détecté.

I.4.3.3. La cryptographie

Basées sur les systèmes symétriques, asymétriques ou hybrides. La pré-distribution des clés de chiffrement dans un RCSF est le fait de stocker ces clés dans la mémoire des nœuds avant le déploiement :

a. La cryptographie symétriques

Les schémas de cette catégorie utilisent les mécanismes des systèmes symétriques dans le but d'établir une clé commune entre deux nœuds d'un RCSF. Cela est réalisé en trois étapes [1] :

- ✓ Pré-distribution de clés (Key predistribution) : les clés stockées dans la mémoire avant le déploiement constituent le porte-clés (key ring) du nœud. S'il existe une clé commune entre deux nœuds, ils peuvent alors créer un lien sécurisé entre eux.
- ✓ Découverte de la clé commune (Shared-key discovery) : le protocole de communication après le déploiement est chargé de découvrir la clé commune entre deux nœuds voisins.
- ✓ Etablissement ou constitution d'un chemin sécurisé par des clés (path-key establishment) : s'il n'existe pas de clé commune entre deux nœuds voulant communiquer, il faut alors trouver un chemin sécurisé entre eux. Ce chemin passe par un ensemble de nœuds qui contient déjà des liens sécurisés. Une fois ce chemin établi, les deux nœuds peuvent l'utiliser pour communiquer en toute sécurité.

b. La cryptographie asymétrique

Les schémas de cette catégorie utilisent les mécanismes des systèmes asymétriques dans le but d'établir une clé commune entre deux nœuds ou un groupe de nœuds d'un RCSF. La cryptographie asymétrique pour les RCSF peut être basée sur les PKI ou sur l'identité des nœuds :

a) **Les PKI** : utilisant le micro-PKI (Micro Public Key Infrastructure), qui est une version simplifiée des PKI conventionnelles. La station de base possède une clé publique et une autre privée. La clé publique est utilisée par les nœuds du réseau pour authentifier la station de base, et la clé privée est utilisée par la station de base pour déchiffrer les données envoyées par les nœuds. Avant le déploiement, la clé publique de la station de base est stockée dans tous les nœuds.

b) **L'identité des nœuds** : c'est un nouveau type de méthodes cryptographiques capable d'offrir les mêmes services cryptographiques qu'une PKI sans avoir besoin d'échanger les clés privées/publiques, de laisser des répertoires de génération de clés chez l'utilisateur, ni d'utiliser les services d'une entité tierce de confiance. La méthode se base uniquement sur les informations de l'identité des nœuds.

I.5. Conclusion

Dans ce chapitre, nous avons décrit les réseaux de capteurs sans fil, leurs architectures de communication, les contraintes de conception et leurs domaines d'applications. Nous avons décris les contraintes du routage, les critères de performances des protocoles de routage pour les réseaux RCSF et les principaux protocoles de routage proposé pour les RCSF. Nous avons cités les exigences en sécurité et les vulnérabilités dans les RCSF, ainsi que les solutions proposés.

Les protocoles de routage proposés pour les RCSF sont nombreux, mais ils ont tous un objectif commun qui réside dans l'acheminement des données collectées par les nœuds capteurs tout en essayant d'étendre la durée d'activité du réseau.

L'absence d'infrastructures comme les PKI dans les RCSF a obligé les nœuds à ne pas faire confiance au réseau et à créer des chemins sécurisés de la source des données jusqu'à la station de base.

Chapitre II : SECURISATION ET OPTIMISATIONS DU PROTOCOLE AODV

II.1. Introduction :

Le défi du déploiement d'un RCSF est la conception du réseau qui est généralement influencé par plusieurs contraintes, principalement la limitation des ressources de l'énergie, et de la sécurité. Il est donc nécessaire d'optimiser les protocoles de routages et d'adapter les algorithmes de sécurités dédiés à ce type de réseaux.

Dans ce chapitre, nous commençons par présenter le protocole AODV et son fonctionnement. Ensuite, nous détaillons le protocole AODV optimisé pour les RCSF et ses caractéristiques. Puis nous allons présenter l'algorithme de chiffrement symétrique RC4 et asymétrique ECC proposés pour l'implémentation dans les RCSF, leurs principes de fonctionnement, et leurs caractéristiques.

II.2. Le protocole AODV :

II.2.1. Présentation

Le protocole de routage AODV (Ad-Hoc On-Demand Distance Vector) est destiné aux réseaux ad hoc à nœuds mobiles. C'est un protocole multihop, adaptatif et dynamique par rapport aux conditions de la topologie, et permet d'éviter les situations de boucle fermée dans le routage.

Ce protocole utilise trois types de paquets, RREQ (Route Request), RREP (Route Response), et RERR (Route Error). Ces messages sont reçus à travers le protocole UDP en allouant une adresse IP à chaque nœud.

Quand une source cherche une route vers une destination, une requête RREQ est envoyée en broadcast à tout le voisinage. Chaque nœud recevant cette requête envoie de nouveau cette requête jusqu'à ce que celle-ci atteigne sa destination. La route est validée en renvoyant une réponse RREP via le chemin inverse de la requête jusqu'à sa source. Chaque nœud recevant la requête sauvegarde dans sa table l'adresse du nœud précédent. Le nœud suivant est reconnu dès la réception de la réponse.

Quand un lien est reconnu « rompu » avec un nœud, un message d'erreur est envoyé aux autres nœuds pour les prévenir que la route vers les destinations qui ne sont plus disponibles. Une réparation de la route est alors effectuée, soit par la source de la requête ou bien par le nœud se trouvant au niveau du lien brisé. **[14]**

II.2.2. Table de routage :

Le protocole AODV utilise une table de routage qui a les entrées suivantes:

- ✓ L'adresse IP de destination.
- ✓ Le numéro de séquence de la destination.
- ✓ Indicateur de validité du numéro de la séquence.
- ✓ Autres indicateurs de la route (valide, invalide, réparable, en réparation).
- ✓ Interface réseau.
- ✓ Compteur de hops.
- ✓ Prochain nœud.
- ✓ Liste des précurseurs.
- ✓ La durée de vie de la route (avant son expiration et son annulation).

II.2.3. Mécanismes de création des routes :

II.2.3.1. Le maintien du numéro de séquence:

Chaque table de routage doit inclure la dernière information sur le numéro de séquence pour l'adresse IP de la destination pour laquelle la table de routage est maintenue. Cette information est remise-à-jour si le nœud reçoit une nouvelle indication de changement du numéro de la séquence de la destination. L'indication se fait à travers les paquets reçus (RREQ, RREP et RRER). Une destination incrémente son propre numéro de séquence dans les deux cas suivants [5]:

- ✓ Juste avant qu'un nœud ait entrepris une découverte de route.
- ✓ Juste avant qu'une destination ait généré une RREP en réponse à une RREQ. Le nœud doit remettre-à-jour son propre numéro de séquence au maximum entre le numéro de séquence actuel et celui contenu dans le paquet RREQ.

II.2.3.2. Les entrées de la table de routage et la liste des précurseurs

Dans le cas où un nœud reçoit un paquet AODV de son voisin, ou crée ou remet à jour la route vers une destination particulière, celui-là recherche dans la table de routage l'adresse de la destination. Si aucune entrée ne correspond à cette destination, une entrée est alors créée dans la table de routage contenant le numéro de séquence du paquet AODV. La route est modifiée si le numéro de séquence est :

- ✓ Supérieur à celui de la destination, ou
- ✓ Egale à celui de la destination, mais le nouveau compteur de hop plus un est inférieur au compteur de hop actuel dans la table de routage, ou
- ✓ Inconnu.

Le champ de durée de vie de la table de routage est aussi une entrée de la table de routage. Ce paramètre est le temps d'expiration de la route valide et le temps de suppression de la route invalide.

Pour chaque route valide maintenue par le nœud sous la forme d'entrée à la table de routage, ce nœud maintient aussi une liste de précurseurs qui peuvent router les paquets sur cette route. Les précurseurs sont ceux qui ont déjà acheminé une réponse RREP vers ce nœud.

II.2.3.3. Génération et acheminement des requêtes RREQ

La requête est générée par un nœud si celui-là ne connait pas la destination ou qu'une route préalablement valide a été invalidée ou a expiré. Le numéro de séquence de destination dans la RREQ est le dernier numéro de séquence de destination connu, sinon un indicateur de numéro de séquence inconnu est activé.

Avant la transmission (en Broadcast) du paquet, la source sauvegarde le RREQ ID et l'adresse IP de la source (sa propre adresse) pour l'instant actuel. De cette manière, si la source reçoit le même paquet, celle-ci ne le retransmet pas. La source ne devrait pas générer plus qu'un maximum de paquets RREQ générés (RREQ_RATELIMIT) par seconde. Si une route n'est pas reçue en un temps inférieur à NET_TRA VERSAL_TIME millisecondes, la source peut générer un nouveau RREQ jusqu'à un maximum de retransmissions RREQ_ RETRIES.

Les paquets de DATA en attente d'une route (en attente d'une RREP après la génération d'une requête RREQ) doivent être bufférisés en mode FIFO. Si le RREQ_RATELIMIT pour une certaine destination est atteint, tous les paquets de DATA qui lui sont destinés doivent être rejetés.

Quand un nœud reçoit une requête, celui-ci crée ou remet à jour le nœud précédent si celui-là n'a pas de numéro de séquence valide dans sa table de routage. Par la suit, le nœud vérifie s'il a déjà reçu le paquet RREQ avec la même adresse IP source et le même RREQ ID dans le dernier intervalle PATH DESCOVERY TIME.

Si une requête est acceptée, le compteur de hops est incrémenté d'une unité. Ensuite, le nœud établit la route inverse vers la source de la requête. Cette route sera nécessaire si ce nœud doit acheminer une RREP vers la source de la requête. Quand une route est créée, les actions suivantes sont entreprises:

- ✓ Le numéro de séquence de la source de la RREQ est comparé au numéro de séquence de la destination contenu dans la table de routage et y est copié si le premier est supérieur au deuxième.
- ✓ L'indicateur numéro de séquence valide est activé (vrai).

- ✓ Le prochain nœud dans la table de routage devient le nœud à partir duquel le RREQ a été reçu (pas nécessairement la source originale de cette requête, mais probablement un nœud intermédiaire).
- ✓ Le compteur de hops est remplacé par celui contenu dans le message RREQ.
- ✓ Le champ de durée de vie de la route est remis à jour.

Si le nœud recevant la requête n'est pas supposé générer une réponse, celui-ci incrémente le compteur de hops par une unité et broadcast le requête. Si le nœud génère une réponse RREP, la RREQ est supprimée.

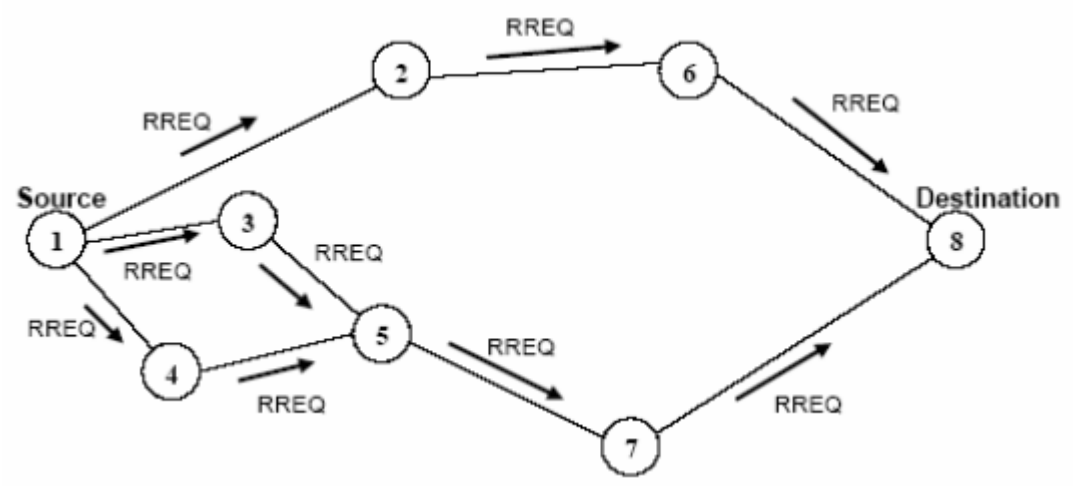


Figure II-1 La propagation du paquet RREQ.

- **RREQ**: contient essentiellement les champs suivants :

- ✓ Type (8 bits): ce champ indique le type de paquet, dans ce cas il prend la valeur 1
- ✓ Flags (drapeaux) (5 bits): ce champ contient cinq flags (J, R, G, D, U).
- ✓ (11 bits) Réservés: initialisé à la valeur 0 et ignoré à la réception du message.
- ✓ Hop Count (8 bits): il contient le nombre de sauts parcourus par RREQ.
- ✓ RREQ ID: il identifie la requête parmi les requêtes envoyées par la même source.
- ✓ Adresse IP destination pour laquelle une route est désirée.
- ✓ Numéro de Séquence de destination : Le dernier numéro de séquence reçu dans le passé pour n'importe quelle route vers la destination.
- ✓ Adresse IP de la source de la requête.
- ✓ Numéro de séquence de la source contenue dans la table de routage de ce nœud.

II.2.4. Génération et acheminement des réponses RREP

Un nœud génère une RREP si :

- ✓ Ce nœud est la destination de la requête, ou

- ✓ Ce nœud dispose d'une route active vers la destination de la requête; le numéro de séquence de la destination existant est valide et supérieur ou égale au numéro de séquence de la destination contenu dans le RREQ.

La réponse RREP est générée en copiant les adresses IP de la destination et de la source dans les champs équivalents. L'envoi de cette réponse se fait en mode unicast vers la destination qui était la source précédente (hop précédent) de la requête selon la table de routage. Le compteur de hops continu d'être incrémenté suivant le chemin inverse suivi par la réponse jusqu'à la source originale de la requête.

Chaque nœud intermédiaire recevant la réponse RREP place dans sa liste de précurseurs le nœud à partir duquel il vient de recevoir ce paquet. La route directe est donc établie à travers l'acheminement de la réponse ; sachant que la requête avait servi à établir le chemin inverse. Sachant aussi que si la réponse n'atteint pas la destination (qui est la source de la requête), celle-ci est déjà en train d'attendre NET_TRA VERSAL_TIME millisecondes avant d'entreprendre une nouvelle découverte de route et de retransmettre la requête ou d'en générer une nouvelle.

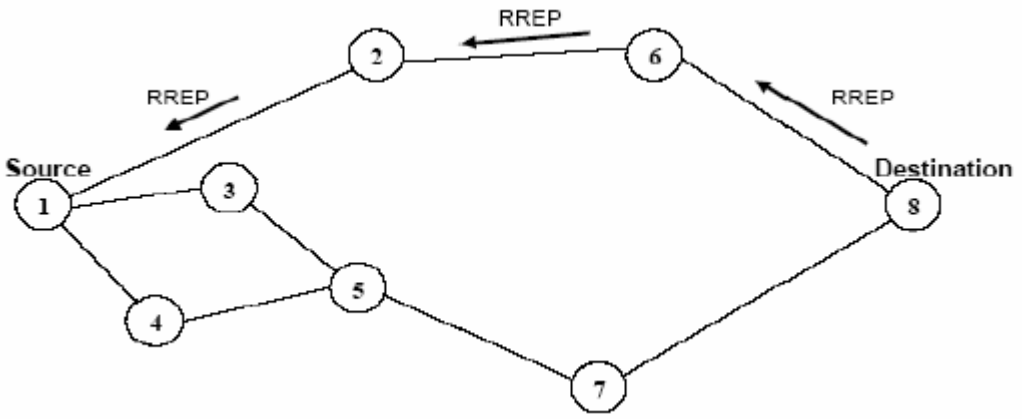


Figure II-2: Le chemin pris par RREP**.**

- **RREP**: contient essentiellement les champs suivants :

- Type (8 bits): ce champ indique le type de paquet, dans ce cas il prend la valeur 2.
- Flags (drapeaux) (2 bits).
- Préfix Sz (5 bits) : si la valeur de ce champs est différente de zéro, ce dernier indique que le nœud prochain peut être utilisé pour chaque nœud demandant cette destination et qui possède la même valeur de préfix Sz.
- Hop Count (8 bits): il contient le nombre de sauts entre la source jusqu'à la destination.
- Adresse IP de la destination du paquet RREQ.
- Numéro de séquence de la destination associé à cette route.
- Adresse IP du nœud qui crée la requête.

- Lifetime : le temps pour lequel chaque nœud qui reçoit RREP considère que la route est valide.

II.2.5. Les messages « Hello»

Les messages HELLO permettent d'offrir des informations sur la connectivité entre les nœuds. A chaque HELLO_INTERV AL millisecondes, chaque nœud vérifie s'il a envoyé au moins une requête RREQ ou un message MAC.

Les nœuds peuvent aussi écouter le canal pour vérifier si leurs voisins reçoivent des messages (Hello ou autres). Si dans un certain intervalle, un voisin ne présente aucune activité de ce genre, le lien avec celui-ci est considéré comme rompu.

II.2.6. Mécanismes de maintenance des routes

II.2.6.1. Maintien de la connectivité locale

Chaque nœud devrait s' assurer de sa connectivité avec son voisinage (ses prochains hops), que ce soit avec ceux qui participent à l'acheminement des paquets durant le dernier ACTIVE_ROUTE_TIMEOUT, ou bien avec ceux qui transmettent des messages Hello durant le dernier intervalle de réception de ce genre de messages donné par ALLOWED HELLO - LOSS * HELLO - INTERVAL.

II.2.7. Génération et acheminement des erreurs RERR

Les messages d'erreurs sont générés selon le processus suivant:

Si le nœud détecte qu'un lien est rompu avec le prochain nœud dans une route active ou dans sa table de routage lors du transfert de données ou qu'une opération de réparation a échoué, ou si ce nœud reçoit un paquet de données qui lui est parvenu d'une destination inconnue ou que ce nœud n'est pas en train d'effectuer une réparation de route, ou si ce nœud reçoit un message RERR d'un voisin pour une ou plusieurs routes.

Pour le premier cas, le nœud établit une liste de destinations injoignables, qui sont les voisins injoignables et toute autre destination contenue dans la table de routage utilisant ces voisins injoignables comme prochain nœud.

Pour le deuxième cas, il existe une seule destination injoignable. Pour le troisième cas, le nœud établit la liste des destinations in joignables à partir du paquet RERR.

Avant de transmettre un RERR, le nœud doit apporter des mises-à-jour nécessaires à la table de routage concernant les numéros de séquences des destinations injoignables.

- ✓ Le numéro de séquence de la destination injoignable est incrémenté dans les deux premiers cas de création du paquet RERR. Sinon, dans le troisième cas, le numéro est copié à partir de RERR reçu.
- ✓ L'entrée est invalidée en marquant la route comme invalide.
- ✓ Le champ de durée de vie de la route est remis-à-jour (dans ce cas, ce champ joue le rôle du temps de suppression de la route).

- **RERR** : Un message d'erreur de route contient essentiellement les champs suivants:

- Type (8 bits): la valeur de ce champ prend 3 dans le message RERR.
- Flag (1 bits): il contient un drapeau (N: No delete flag), celui-ci est indicatif lorsqu'un nœud est capable de réparer le lien, et informe les nœuds suivants qu'ils ne doivent pas supprimer le chemin.
- Reserved (15 bits): initialisé à la valeur 0 et ignoré à la réception du message.
- Dest Count (8 bits): il indique le nombre de destinations inaccessibles incluses dans ce message. Ce champ doit être supérieur ou égal à un.
- Adresse IP des destinations inaccessibles pour la raison de cassure de lien.
- Nombre de séquence de la liste des destinations inaccessibles.
- Adresse IP de destination.

II.3. Le protocole AODV optimisé pour RCSF

Pour le protocole de routage AODV, la recherche et le maintien des routes est effectué sur la demande du nœud source par la diffusions du RREQ afin de découvrir toutes les routes possibles vers une destination donnée, ce mécanisme est n'est pas adapté aux RCSF ayant moins de ressources comparées aux réseau sans fils MANET et dont le modèle de trafic dans ces réseaux est très souvent d'un nœud vers un autre nœud, contrairement dans les réseaux de capteurs dont le modèle de trafic est de type plusieurs nœud vers un nœud.

Dans son document [4], l'auteur propose une modification du protocole AODV adaptés au RCSF, en redéfinissant les spécifications de l'AODV conventionnel et optimisées pour les réseaux de capteurs, en tenant compte des caractéristiques suivantes:

- Le modèle de trafic des réseaux de capteurs qui est de type plusieurs vers un.
- Les contraintes des capteurs telles que la mémoire et la puissance de calcul.
- Les contraintes de bande passante.
- Le nombre de nœuds du réseau

II.3.1.1. Génération et acheminement des requêtes RREQ :

La diffusion fréquente, sur le réseau de capteur, des requêtes RREQ, vers la station de base, crie une surcharge dans le réseau qui engendre une dégradation de ses performances particulièrement pour un réseau à grande échelle.

La solution proposée dans le document [4] consiste à affecter la tâche de découverte de l'itinéraire par la station de base, qui lance périodiquement une requête RREQ sur le réseau destinée à informer chaque capteur sur le réseau de son emplacement. Chaque capteur sur le réseau rediffuse la requête RREQ reçue, et il met à jour sa table de routage en ajoutant la nouvelle route à sa table de routage. Les tables de routage de tous les nœuds sont composées seulement des routes conduisant à la station de base.

Suite à la réception d'une requête RREQ, le capteur utilise le numéro de séquence pour vérifier s'il n'a pas déjà traité cette requête RREQ, si oui, le capteur met à jour le prochain nœud pour la route vers la station de base, ils incrémentent le numéro de sauts dans le RREQ et le rediffuse. Si le RREQ est déjà traité, elle sera ignorée.

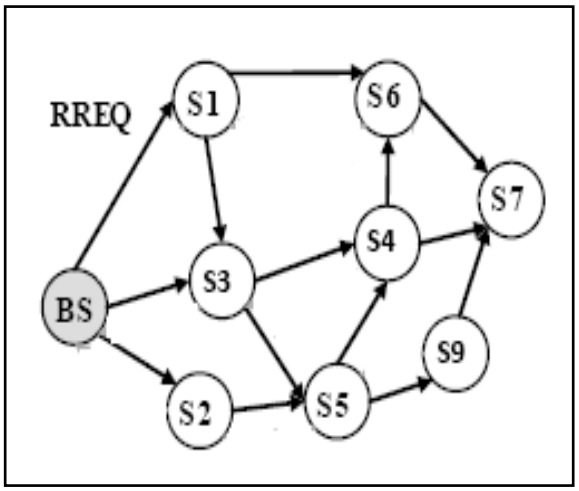


Figure II-3: découvert de route.

Afin d'éviter le sur-chargement du réseau et l'épuisement de la puissance des batteries des capteurs, les messages HELLO ne sont pas utilisés dans le protocole OAODV optimisé, la connectivité entre les capteurs est garantie en utilisant la requête RREQ envoyée périodiquement par la station de base et qui garantit la mise à jour des tables de routage. [4]

II.3.2. La Table de routage

Le protocole OADV optimisé conserve la même structure de la table de routage définie dans le protocole AODV conventionnel, cependant la table de routage ne contiendra qu'une

ou deux entrées ayant comme destination finale la station de base, ce qui minimise considérablement l'espace mémoire occupé par la table de routage, qui est insuffisant pour enregistrer des tables de routages ayant un nombre important des routes. [4]

II.3.3. Maintenance des routes :

Dans le protocole AODV optimisé, le nœud qui détecte la défaillance de la route n'envoie pas un message RERR, mais il attend la rediffusion du prochain RREQ par la station de base afin d'obtenir une nouvelle route. La période de diffusion de la requête RREQ est fixée par l'administrateur du réseau en fonction de la stabilité des nœuds et l'environnement de déploiement. [4]

II.4. Sécurisation des données dans le RCSF :

A cause de multiples vulnérabilités des RCSF, liés principalement aux propriétés de ce type de réseau, et afin de sécuriser la transmission des données contre les différentes attaques, nous avons proposé une solution de sécurisation basée sur la cryptographie symétrique et asymétrique.

La solution consiste à :

- L'utilisation d'un algorithme de chiffrement symétrique RC4, basé sur l'utilisation d'une seule clé privé entre chaque lien (station de base-capteur), afin de crypter les données transmises (sécurisation de l'information envoyée).
- L'utilisation de la signature ECDSA issue de l'algorithme ECC comme algorithme de chiffrement asymétrique, la proposition consiste à l'implémentation de la signature ECDSA dans tous les capteurs, après création des paires de clés privé et publique pour l'ensemble des capteurs. L'objectif est d'assurer la sécurité du protocole de routage et éviter la création de la route vers une destination autre que la Station de Base.
- La gestion des clés : chaque capteur contient sa propre clé privé pour l'algorithme RC4 et sa propre clé privé pour l'algorithme ECDSA, il contient aussi une table de tous les clés publiques de la signature ECDSA de tous les capteurs de son secteur dans le réseau.

Cette solution permet de garantir la confidentialité, l'authentification et l'intégrité

II.4.1. Algorithme de chiffrement symétrique implémenté RC4 :

II.4.1.1. Présentation :

Il s'agit d'un chiffrement par flux créé en 1987 par Ron Rivest. Diverses analyses ont démontré que la période du générateur est supérieure à 10. Il s'agit probablement du chiffrement par flots le plus utilisé actuellement. On le retrouve notamment dans le standard SSL/TLS, dans Oracle Secure SQL, ou encore dans le protocole WEP (Wired Equivalent Privacy, de la norme 802.11). Ce dernier fut remplacé par le WPA (Wi-Fi Protected Access), mais celui-ci utilise toujours le RC4.

Initialement gardé secret par la RSA, cet algorithme a été publié anonymement en 1994 sur Internet. Il existe une version allégée du chiffrement RC4, portant le nom de ARC4 (Alleged RC4), utilisable légalement. Le RC4 reste la propriété de RSA Labs.

Un chiffrement par RC4 est très rapide, comme le montre le tableau suivant :

Algorithme	Longueur de la clé	Vitesse (en Mbps)
DES	56	9
3 DES	168	3
RC4	Variable	45

Tableau II-1 : Comparaison entre les algorithmes de chiffrement.

II.4.1.2. Fonctionnement du RC4

Cet algorithme fonctionne sur les octets. Ainsi, la clé, de longueur variable, peut avoir une taille comprise entre 1 et 256 octets (de 8 à 2048 bits). Elle est utilisée pour initialiser un vecteur S de 256 octets. A tout moment, S contient une permutation de toutes les cellules le composant. La figure 6.8 illustre le principe du RC4.

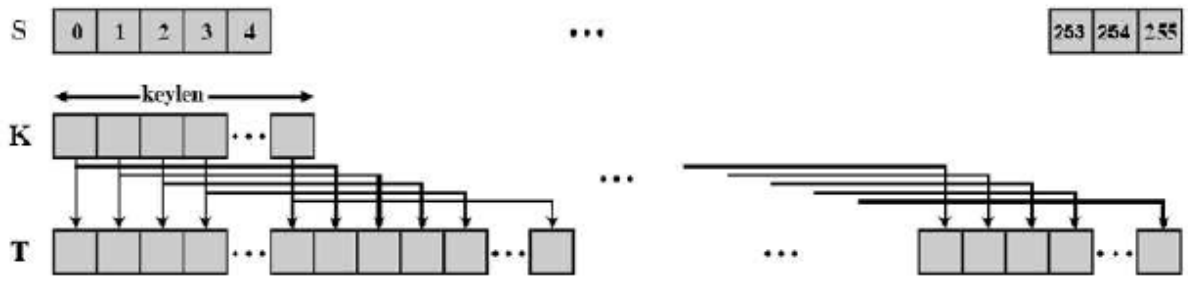


Figure II-4: Fonctionnement du chiffrement RC4.

II.4.1.3. Initialisation

Initialement, les cellules de S reçoivent une valeur égale à leur position (i.e., S[0]=0, S[2]=1, ...) Un vecteur temporaire de longueur T (de longueur égale à S) est également créé destiné à recevoir la clé. Si la longueur de la clé K est égale à 256 octets, K est simplement transféré dans T. Si K est inférieur à 256 octets, il est recopié dans T jusqu'à atteindre la taille de T. Ce procédé est illustré au point (a). On peut également le représenter algorithmiquement comme suit :

```
FOR  i = 0 TO 255
DO
S[i] = i;
```

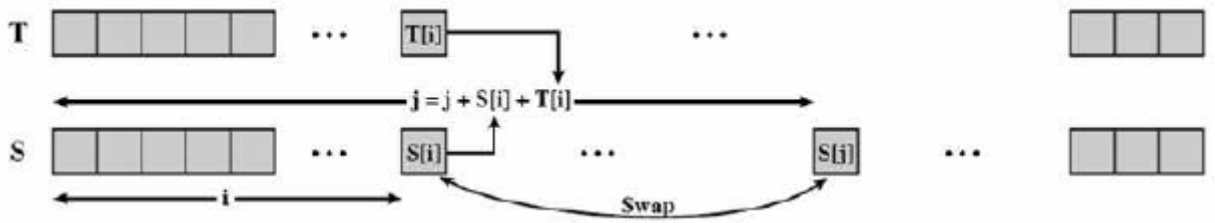


Figure II-5: Initialisation de RC4 (1).

Le vecteur temporaire T est ensuite utilisé pour produire la permutation initiale de S. Pour chaque cellule S[i] de S, celle-ci sera échangée avec une autre cellule de S selon un calcul basé sur la valeur comprise dans la cellule T[i] correspondante. Le point (b) illustre cette phase. Tout comme l'initialisation des vecteurs, on peut représenter la permutation initiale de S par un algorithme :

```
j = 0;
FOR  i = 0 TO 255 DO
   j = (j + S[i] + T[i]) mod 256;
   SWAP(S[i], S[j]);
T[i] = K[i mod keylen];
```

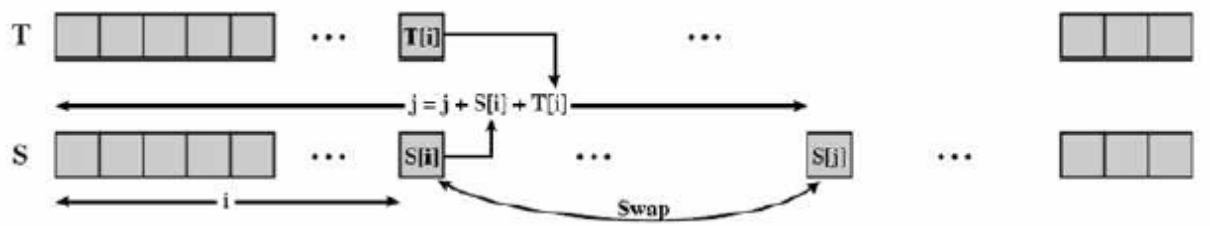


Figure II-6: Initialisation de RC4 (2).

II.4.1.4. Génération du flux :

A partir de cet instant, la clé d'entrée n'est plus utilisée. Pour chaque S[i], on procèdera à un échange avec un autre octet de S, selon un schéma basé sur la configuration courante de S. Une fois arrivé à S[255], le processus redémarre à la cellule S[0]. Le point (c) de la figure 6.8 présente la procédure. A nouveau, on peut illustrer algorithmiquement la méthode (on parle de PRGA pour Pseudo-Random Generation Algorithm) :

```
i, j = 0;
WHILE (true)
i = (i + 1) mod 256;
j = (j + S[i]) mod 256;
SWAP (S[i], S[j]);
t = (S[i] + S[j]) mod 256;
```

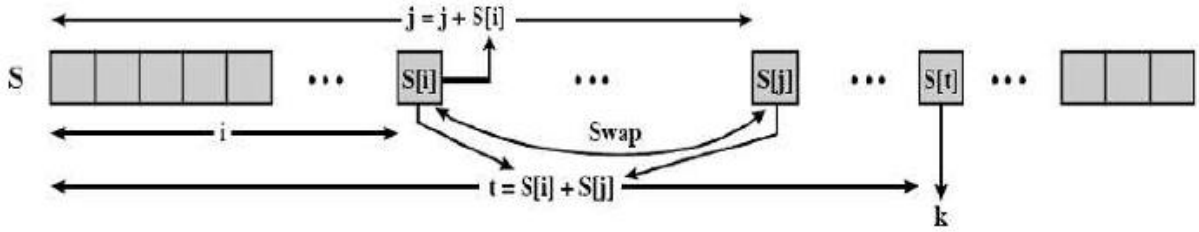


Figure II-7: Génération du flux RC4

La valeur de k est alors utilisée pour le chiffrement (⊕ avec le prochain octet de texte clair), ou pour le déchiffrement (⊕ avec le prochain octet de texte chiffré).

II.4.1.5. Robustesse du RC4

On peut tout d'abord remarquer que la sécurité repose uniquement sur la clé. En effet, c'est elle, et uniquement elle, qui détermine le flux de sortie du générateur. En 2001, une découverte affaiblit l'aspect sécuritaire du RC4. Fluhrer, Mantin et Shamir remarquèrent que les premiers octets en sortie du RC4 n'étaient pas tout à fait aléatoires. De fait, il semblerait que ces quelques premiers bytes divulgueraient des informations importantes sur la clé d'initialisation.

Le protocole WEP a été remplacé par le protocole WPA pour pallier les problèmes rencontrés avec a gestion des clés dans le RC4.

II.4.2. Algorithme de chiffrement ECC (Elliptic Curve Cryptography) :

A l'origine, les courbes elliptiques ont été créées afin de résoudre des problèmes mathématiques fondamentaux sans avoir la moindre arrière-pensée sur le fait qu'elles auraient un grand succès en cryptographie. Elles ont été proposées indépendamment par Victor Miller et Neal Koblitz en 1985 pour être utilisées dans la cryptographie. Leurs applications sont aujourd'hui au centre de la cryptographie moderne et remplacent de plus en plus les grands standards. Elles ont détrôné RSA de sa position dominante. En effet, les algorithmes basés sur les courbes elliptiques sont plus rapides et aussi sûrs avec des clés plus petites. [28]

En effet utiliser une clé de 160 bits dans un algorithme fondé sur ECC est équivalent à l'utilisation d'une clé RSA de taille 1024 bits, ce qui réduit la puissance de calcul exigée par ECC par rapport à celle exigée par RSA. [24]

II.4.2.1. Définition : [28]

Une courbe elliptique a pour équation générale (forme de Weierstrass) :

Équation II-1 : $\mathbf{y^2 + a_1xy + a_3y = (x^3 + a_2x^2 + a_4x + a_6)}$

Lorsque les coordonnées (x, y) sont définies sur un corps de caractéristique strictement supérieur à 3, alors on peut réécrire l'′equation de la courbe sous la forme :

Équation II-2 : $\mathbf{y^2 = (x^3 + ax + b)}$

Les variables et coefficients prennent des valeurs dans l'ensemble [0, p − 1] pour un certain nombre premier p, et où toutes les opérations sont calculées modulo p. L'équation devient :

Équation II-3 : $y^2 \bmod p = (x^3 + ax + b) \bmod p$

Les courbes elliptiques sont caractérisées par les points suivants : **[30]**

- ✓ Le point à l'infini est l'élément neutre du groupe.
- ✓ L'opposé d'un point est son symétrique par rapport à l'axe des abscisses.
- ✓ La multiplication est définie comme une répétition d'additions (ex : 3P = P + P + P)

La figure suivante présente un exemple de courbe elliptique sur lequel on désire additionner deux points P et Q.

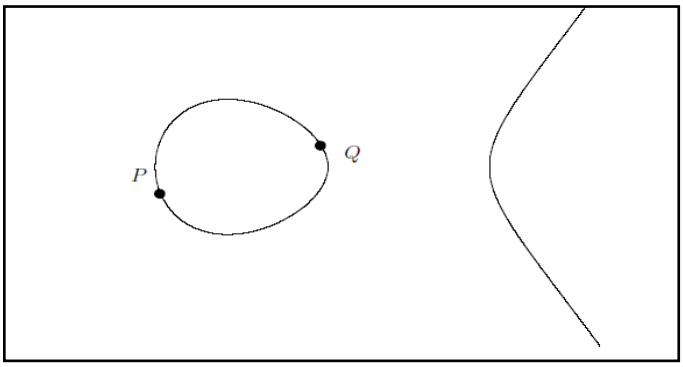


Figure II-8 : Exemple d'une courbe elliptique E.

Pour cela, il faut tracer la droite qui passe par ces deux points. Cette droite coupe la courbe en un troisième point, comme nous pouvons le voir sur **la figure II-9**.

Il faut alors prendre son point symétrique par rapport à l'axe des abscisses pour obtenir un point R tel que R = P + Q sur le groupe des points de la courbe (voir **la figure II-10**). Le point symétrique par rapport à l'axe des abscisses est en fait l'opposé d'un point dans le groupe. Si l'on veut doubler un point, il faut choisir pour droite la tangente de la courbe en ce point et poursuivre le même protocole (**la figure II-11**). Les coordonnées sont dans un corps de base.

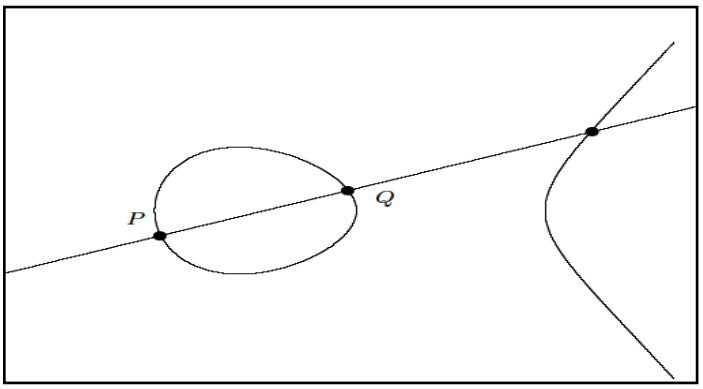


Figure II-9 : Droite passant par P et Q.

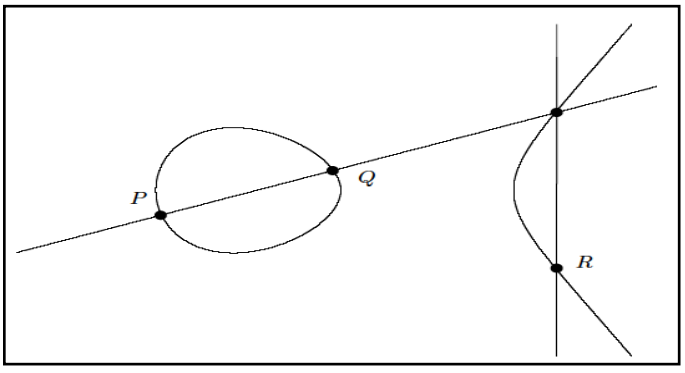


Figure II-10 : Calcul du point R de E tel que P + Q = R.

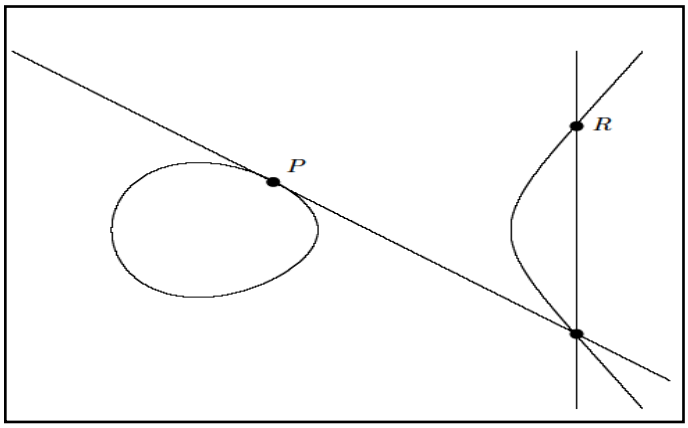


Figure II-11 : Calcul du doublement d'un point.

Pour calculer les coordonnées exactes du point R, nous utilisons l'équation de la courbe et celle de la droite. Ainsi, nous pouvons directement calculer les coordonnées de R. Nous évaluons le coefficient directeur γ de la droite.

Si P = (x_P , y_P) et Q = (x_Q, y_Q) avec P ≠ Q, alors on détermine R = P + Q = (x_R, y_R) comme suit :

Équation II-4 : $X_R = (\lambda^2 - x_P - x_Q) \bmod p$

Équation II-5 : $y_R = (\lambda (x_P - x_R) - y_P \bmod p$

Où

Équation II-6 :

$$\lambda = \begin{cases} \left(\frac{y_Q - y_p}{x_Q - x_P}\right) mod\ p\ si\ P \neq Q \\ \left(\frac{3x^2{}_p + a}{2y_p}\right) mod\ p\ si\ P = Q \end{cases}$$

II.4.2.2. La cryptographie sur courbes elliptiques (ECC) :

Pour utiliser les courbes elliptiques en cryptographie, il faut trouver un problème difficile (tel que la factorisation d'un produit en ses facteurs premiers dans le cas du RSA). Considérons l'équation suivante :

Équation II-7 : $Q = kP$ où $Q, P \in E_P(a, b)$ et $k < p$

Il est facile de calculer Q connaissant k et P, mais il est difficile de déterminer k si on connait Q et P. Il s'agit du problème du logarithme discret pour les courbes elliptiques : logP (Q). Dans une utilisation réelle, le k est très grand, rendant l'attaque par force brute inutilisable. [29]

a. ECC pour l'échange de clés (ECDH) :

ECDH (Elliptic Curve Diffie-Hellman) est l'adaptation de la méthode d'échanges de clés de Diffie-Hellman en se basant sur les courbes elliptiques. Soit un grand entier premier q, et les paramètres a et b satisfaisant l'équation N° II.3. Cela nous permet de définir $E_q(a, b)$. Prenons ensuite un point de départ $G(x_1, y_1)$ dans $E_q(a, b)$ dont l'ordre n est élevé. L'ordre n d'un point sur une EC est le plus petit entier positif tel que nG = O. [29]

Eq(a, b) et G sont rendu publiques.

L'échange d'une clé par ECC entre deux entités A et B se déroule comme suit :

- A choisit un $n_A < n$ qui sera sa clé privée. A génère alors sa clé publique $P_A = n_A \times G$.
- B choisit un $n_B < n$ qui sera sa clé privée. B génère alors sa clé publique $P_B = n_B \times G$.
- A génère la clé secrète $K = n_A \times P_B$ et B génère la clé secrète $K = n_B \times P_A$.

b. ECC pour chiffrer des données (ECIES):

Même si la cryptographie par courbes elliptiques est souvent employée pour l'échange d'une clé symétrique, elle est aussi utilisée pour chiffrer directement les données. ECIES (Elliptic Curve Integrated Encryption Scheme) est un algorithme de chiffrement à clé publique basé sur les courbes elliptiques qui est une variante de celui d'El Gamal standardisée. [29]

Il faudra ici encoder le texte clair m comme un point Pm de coordonnées x et y. C'est ce point qui sera chiffré. Il faut ici aussi rendre publique un point G et un groupe elliptique Eq(a, b). Les utilisateurs doivent également choisir une clé privée et générer la clé publique correspondante.

Pour chiffrer le message, A détermine aléatoirement un nombre entier positif k et produit C_m comme un couple de points tel que :

Équation II-8 : $C_m = \{kG, P_m + kP_B\}$

On remarquera l'utilisation de la clé publique de B. Pour déchiffrer, B devra multiplier le premier point par sa clé privée, et soustraire le résultat au second point reçu :

Équation II-9 : $\mathbf{P_m + kP_B - n_B(kG) = P_m + k(n_BG) - n_B(kG) = P_m}$

c. ECC pour la signature numérique (ECDSA) :

ECDSA (Elliptic Curve Digital Signature Algorithm) est un algorithme de signature numérique à clé publique. La méthode de signature de cet algorithme est une variante de l'algorithme de signature DSA (Digital Signature Algorithm), proposée par le NIST (National

Institute of Standards and Technology) en 1991 pour être utilisée dans son standard de signature DSS (Digital Signature Standard). Elle est basée sur les courbes elliptiques et varie très peu de la méthode de signature El Gamal. [29]

Soient E une courbe elliptique d'ordre premier P, Q un point appartenant à E et H une fonction de hachage dans $[1, p-1]$. Supposons qu'Alice doive signer et envoyer un message. Afin de signer ce message, Alice doit d'abord créer une clé privée k_A et une clé publique $P_A = k_A * Q$. Bob doit disposer de la clé publique d'Alice pour pouvoir vérifier l'authenticité de la signature. Alice et Bob procèdent ensuite aux étapes présentes dans le Tableau II.4.

Etapes	Signature côté Alice	Vérification côté Bob
01	Calculer $e = H(m)$	Calculer $e = H(m)$
02	Choisir $k \epsilon [1, p]$ au hasard	Vérifier que $r, s \epsilon [1, p-1]$
03	Calculer KQ	Calculer $w = s^{-1} \, mod \, p$
04	Convertir $x(KQ)$ en un entier $\bar{x}$	Calculer $t_1 = ew, t_2 = rw \, mod \, p$
05	Calculer $r = \bar{x} \, mod \, p$	Calculer $X = t_1 Q + t_2 P_A$
06	Calculer $s = k^{-1} \, (e + k_A r) mod \, p$	Convertir $x(X)$en un entier $\bar{x}$
07	Envoyer (m, r, s)	Accepter si $\bar{x} = r \, mod \, p$

Tableau II-2 : Les étapes de l'algorithme ECDSA.

II.5. Conclusion

Dans ce chapitre, nous avons présenté le protocole AODV et le protocole AODV optimisé, leurs fonctionnements et leurs caractéristiques. Nous avons aussi détaillé le fonctionnement de l'algorithme de chiffrement symétrique RC4 et asymétrique ECC proposés pour l'implémentation dans les RCSF.

Le protocole de routage AODV Optimisé proposé permet de réaliser le routage dans les RCSF en tenant comptes des différentes caractéristiques de ces réseaux. La solution de sécurité proposée basée sur l'utilisation de l'algorithme de chiffrement symétrique RC4 et de la signature ECDSA est adaptée aux RCSF.

CHAPITRE III : IMPLEMENTATION DU PROTOCOLE OAODV SUR CAPTEURS TELOSB

III.1. Introduction :

Dans ce chapitre, nous présentons l'environnement de développement et d'implémentation exploité dans notre projet, nous décrivons les composants du capteur CM5000 utilisé, nous montrons les principaux composants, interfaces et fonctions utilisés pour l'implémentation des protocoles de routage AODV et AODV optimisé et des algorithmes de chiffrement symétrique RC4 et asymétrique ECC, dans les capteurs Telosb.

III.2. Systèmes d'exploitation et langage de programmation utilisés:

III.2.1. Le système d'exploitation TinyOS :

Les contraintes des RCSF ont motivé l'université de Berkeley et d'autres contributeurs à développer un système d'exploitation destiné aux RCSF afin de faciliter l'implémentation et l'exécution de protocole dédiés à ce type de réseaux.

L'objectif consiste à minimiser la taille du code afin de respecter les contraintes de ressources énergétiques et physiques des nœuds capteurs. Ce système a l'avantage de permettre une programmation simple et puissante tout en gardant la portabilité du code pour les nombreuses plateformes supportées. Il est utilisé par plus de 500 universités et centres de recherche dans le monde vu la caractéristique open-source qu'il détient. Il respecte une architecture basée sur une association de composants et utilise une programmation entièrement réalisée en langage NesC. [7]

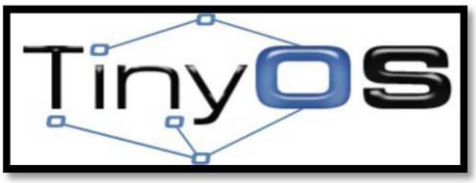


Figure III-1 : Sigle de TinyOS.

Les propriétés de Tinyos sont :

- ✓ Une taille de mémoire réduite.
- ✓ Une basse consommation d'énergie.
- ✓ Des opérations robustes.
- ✓ Applications orientées composants.

- ✓ Programmation orientée évènement : Généralement sur TinyOS, un programme s'exécute suivant le déclenchement des événements. Sinon, les capteurs restent en veille ce qui maximise la durée de vie du réseau.

Le fonctionnement de TinyOS s'appuie sur la gestion d'évènements. Ainsi, l'activation de tâches, leur interruption ou encore la mise en veille du capteur s'effectuent suite à l'apparition d'évènements. L'implémentation de composants s'effectue en déclarant des tâches, des commandes ou des évènements : [1]

- ✓ Une tâche est un travail de longue durée (à l'échelle du temps processeur).
- ✓ Une commande est l'exécution d'une fonctionnalité précise dans un autre composant.
- ✓ Un événement est l'équivalent logiciel d'une interruption matérielle.

Pour notre projet, nous avons installé le Tinyos de la version 2.1.2.

III.2.2. Le langage NESC :

Grâce à NesC il est possible de créer une application par un assemblage de composant nécessaire à l'application. Chaque composant correspond `a un élément matériel (LEDs, timer, ADC ...) et peut être réutilisé dans différentes applications.

Un composant est constitué alors de trois parties essentielles : interfaces, modules et configurations.

a) Interface d'un composant

Un composant offre et utilise des interfaces. Ces interfaces sont l'unique point d'accès au composant et sont bidirectionnelles. Une interface définie un ensemble de fonctions appelées commandes et un autre ensemble de fonctions appelées événements. Pour qu'un composant puisse appeler les commandes d'une interface il doit implémenter les événements définis dans l'interface. Un même composant pourrait offrir et utiliser plusieurs interfaces différentes ou plusieurs instances d'une même interface.

b) Modules et configurations

Il existe deux types de composants dans nesC: modules et configurations. Les modules définissent le code de l'application en implémentant une ou plusieurs interfaces. Les configurations sont utilisées pour assembler d'autres composants, en reliant les interfaces utilisées par des composants aux interfaces offertes par d'autres composants. Ceci est appelé

"wiring" (câblage). Toute application nesC est décrite par une configuration qui assemble les composants de l'application.

III.3. Outilles utilisés dans l'implémentation et la simulation :

III.3.1. VMware Workstation version 10.0.1:

VMware Workstation est un logiciel qui permet de faire tourner une machine virtuelle sur un ordinateur, permettent aux utilisateurs d'exécuter plusieurs systèmes d'exploitation, notamment Linux et Windows, en tant que machines virtuelles.

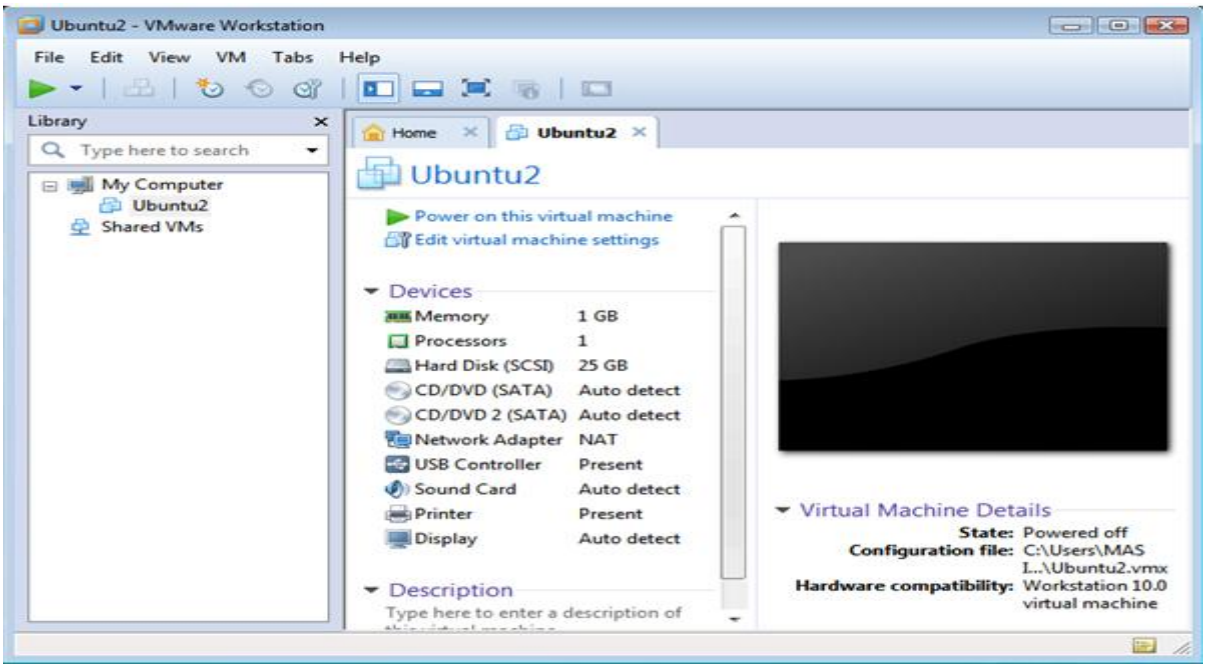


Figure III-2 : Machine VMware.

III.3.2. Ubuntu version 14.04 :

Ubuntu est une distribution GNU/Linux basée sur Debian. Initialement conçu pour tous les ordinateurs de bureau (fixe ou portable), Ubuntu propose également une version serveur et depuis peu, une version pour appareil mobile (Ubuntu Netbook Remix). On a utilisé la version 14.04 (du 17 avril 2014).

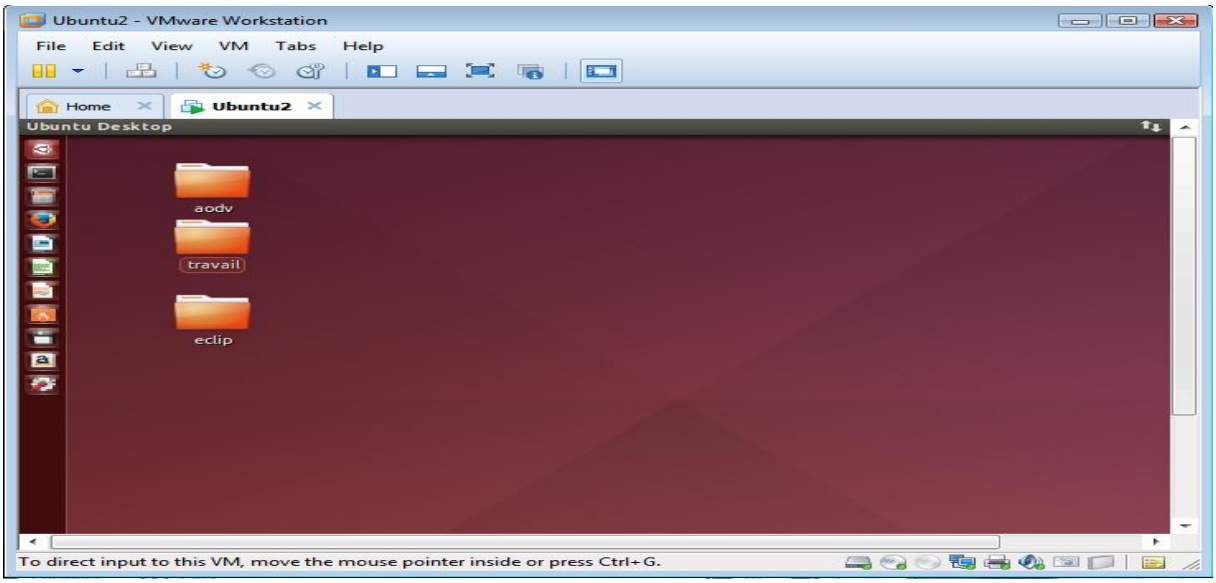


Figure III-3 : Ubuntu 14.04.

III.3.3. Java Jdk version 1.8.0.:

Le Java Development Kit (JDK) désigne un ensemble de bibliothèques logicielles de base du langage de programmation Java, ainsi que les outils avec lesquels le code Java peut être compilé. Avant d'installer le Tinyos version 2.1.2, nous avons installé le java jdk 1.8.0. Une fois installée, la plate-forme d'exécution Java est lancée au démarrage d'Ubuntu.

III.3.4. Eclipse neon.2 et yeti2 :

Eclipse est un environnement de développement Java intégré (IDE), c'est un logiciel qui simplifie la programmation en proposant un certain nombre de raccourcis et d'aide à la programmation. Il est développé par IBM. Nous avons installé la version neon.2 (4.6.2) d'éclipse :

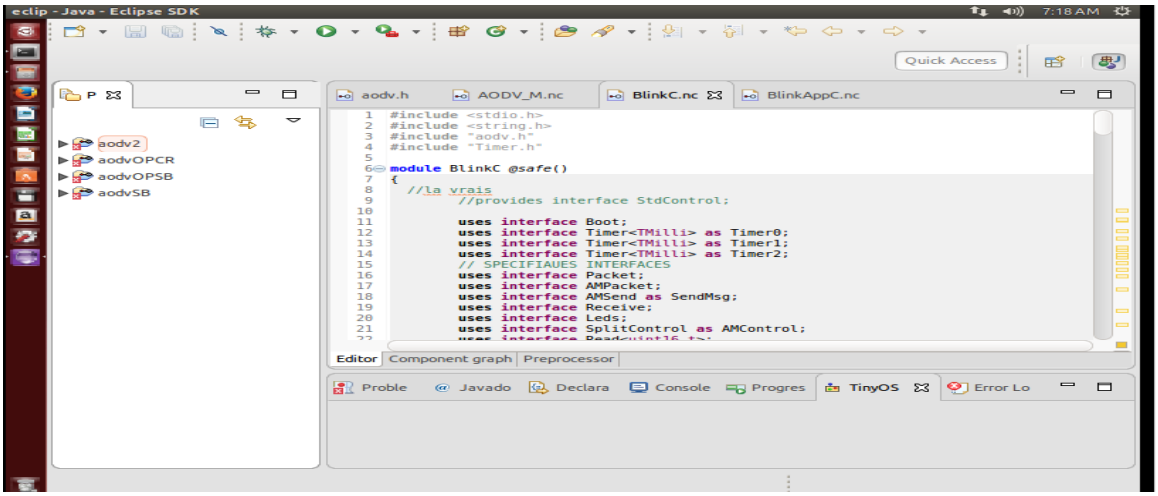


Figure III-4 : Eclipse neon.2.

Yeti est un éditeur NesC implanté comme plugin pour Eclipse. La liste des fonctionnalités fournies par Yeti comprend: la mise en surbrillance de syntaxe, la validation du code en temps réel et l'achèvement du code. Yeti est conçu pour fonctionner avec TinyOS 2.x et n'a qu'un support limité pour TinyOS 1.x.

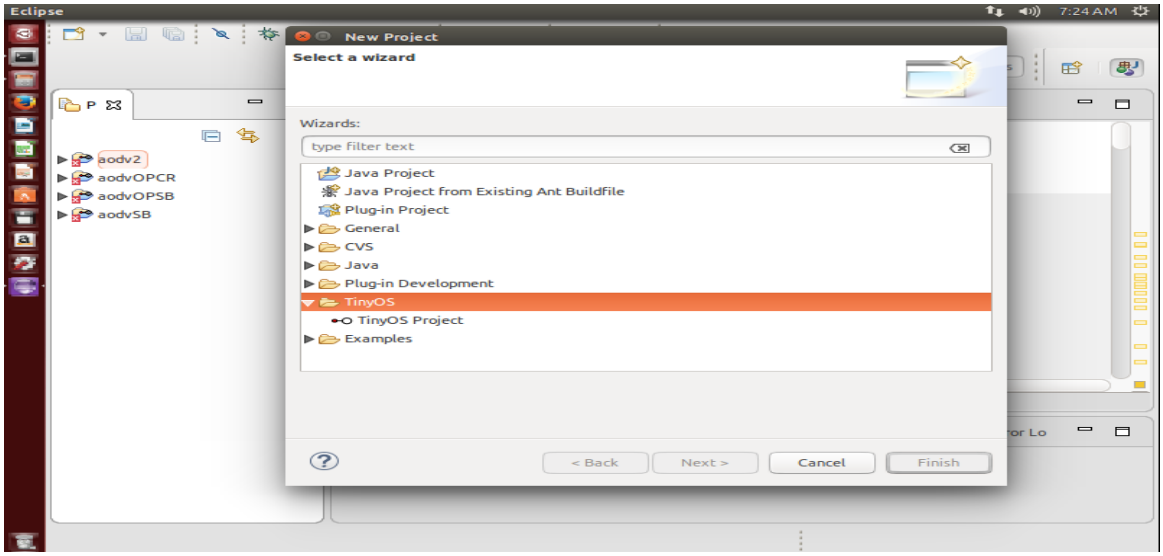


Figure III-5 : Eclipse avec Yeti2.

III.3.5. Port série :

Pour l'affichage des résultats et des messages, nous avons utilisé le port série fournit par Ubuntu, en exploitant la fonction printf :

```
printf("HELLO': %u\n");
```

III.4. Cartes TelosB :

Deux familles de cartes sont utilisées pour la recherche et l'évaluation des solutions proposées dans le domaine des RCSF, la famille « Mica », composée de trois types de cartes : MicaZ, Mica2 et Mica2dot, et la famille « Telos », composée de deux types de cartes TelosA et TelosB. Les deux familles ont été produites par « Crossbow ». L'implémentation est effectuée sur les cartes TelosB (CM5000).

Le CM5000 est un capteur sans fil compatible avec IEEE 802.15.4 basé sur la conception de plate-forme open source "TelosB" développée et publiée par l'Université de Californie, Berkeley ("UC Berkeley").

Figure III-6 : Capteur CM5000

III.4.1. Composants du capteur CM5000 :

Ces capteurs sont composés de :

- ✓ Microcontrôleur MSP430.
- ✓ Module radio CC2420 RF.
- ✓ Capteur de température, capteur d'humidité et capteurs de lumière.
- ✓ Boutons de réinitialisation.
- ✓ 3xLeds.
- ✓ Support de batterie 2xAA.
- ✓ Interface USB.
- ✓ Antenne embarquée.

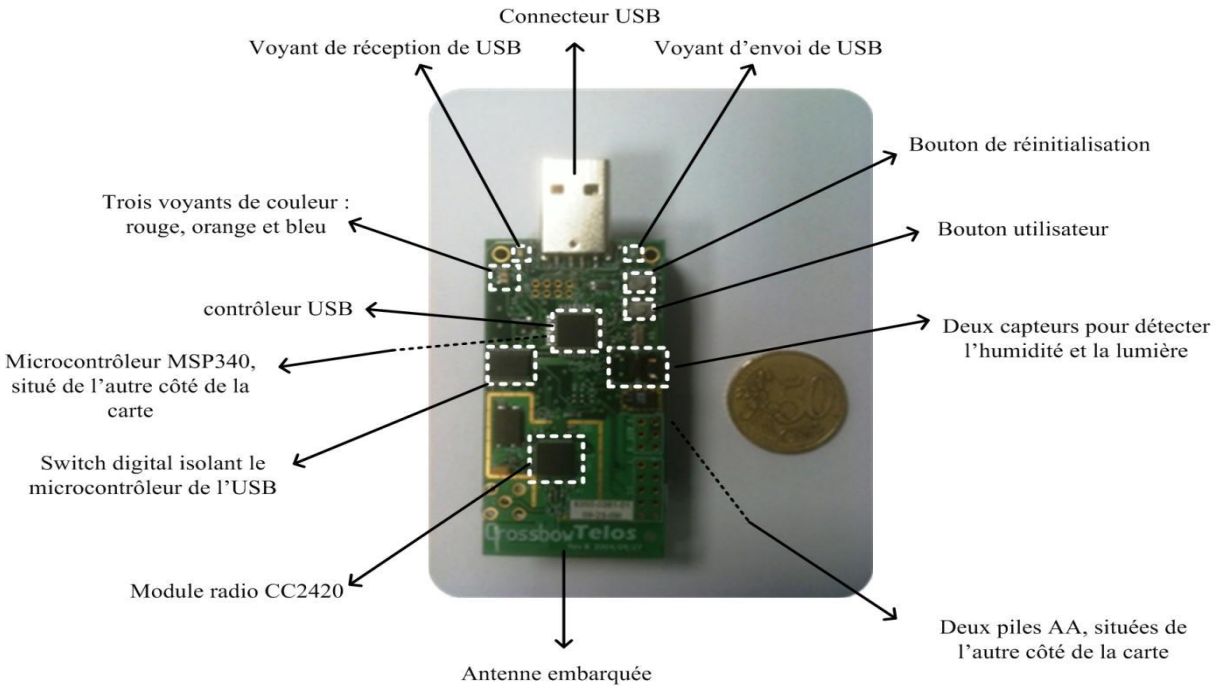


Figure III-7 : Composants de capteur Telosb

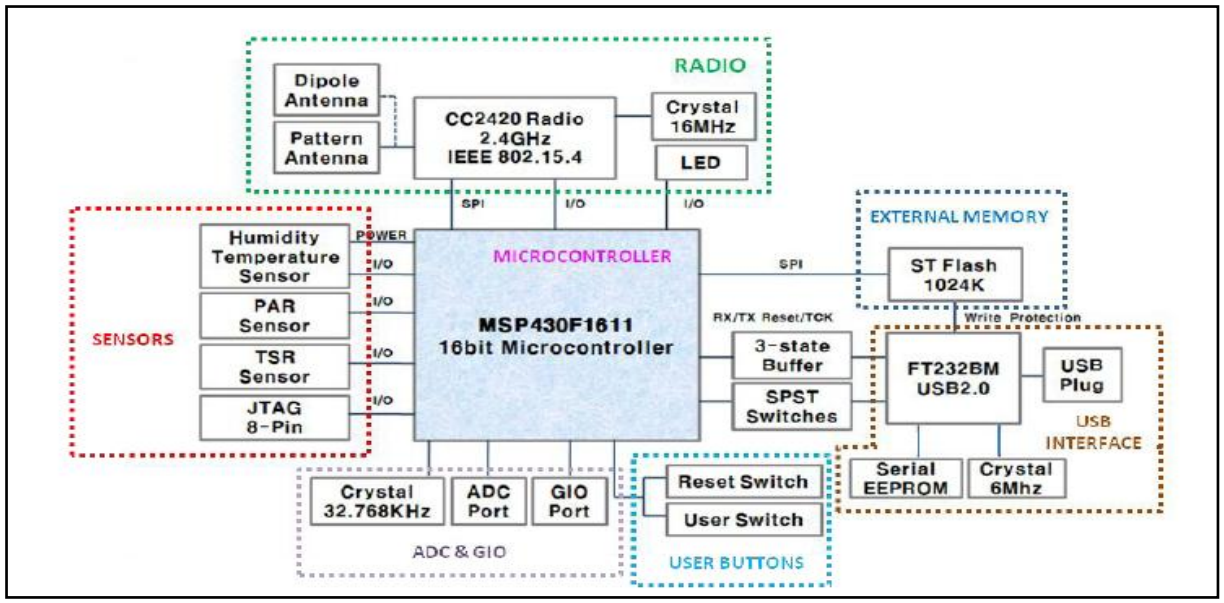


Figure III-8 : Diagramme block du capteur.

III.4.2. Microcontrôleur TI MSP430F1611 :

Ce microcontrôleur appartient à la famille Texas Instruments MSP430 de microcontrôleurs ultra légers. Cette architecture comporte cinq modes de faible puissance qui sont optimisés pour améliorer la durée de vie de la batterie pour les applications de mesure. Ce composant dispose d'une CPU de 16-bit, des registres de 16-bit, d'un oscillateur à commande numérique permet le changement des modes de faible puissance au mode actif en moins de 6 µs.

Le MSP430 dispose de deux compteurs de16 bits, d'un convertisseur A/D rapide de 12 bits, de deux interfaces de communication synchrones et asynchrones séries universelles (USART), I2C, DMA et 48 pins I/O. En outre, la série MSP430F161x offre un adressage RAM étendu pour des applications à forte intensité de mémoire.

III.4.3. Module RF 2.4 GHz IEEE 802.15.4 :

Le CC2420 est un émetteur-récepteur RF à 2,4 GHz IEEE 802.15.4 conçu pour les applications sans fil faible puissance et basse tension. CC2420 comprend un modem de transmission numérique de bande de base à spectre étalé, offrant un gain d'étalement de 9 dB et un débit de données efficace de 250 kbps.

Figure III-9 : Le modem RF CC2420

En mode de transmission ou réception, le tampon de type FIFO est limité à 128 octets. Il permet de tamponner les données avant leur transmission ou réception. Une séquence de préambule est automatiquement insérée avant le champ. Ce champ doit toujours être le premier octet écrit dans le tampon d'émission.

III.4.4. Les capteurs d'humidité et de température Sensirion® SHT11 :

Le CM5000 utilise le capteur Sensirion® SHT11, spécialement conçu pour les mesures de température et d'humidité de l'air. Ce composant est géré par le capteur à l'aide d'une interface à deux fils. La précision de la mesure de la température est de ± 0.05°C, tandis que pour l'humidité la précision est de ± 4.5%RH.

Les mesures sont reçues en tant que valeurs de 2 octets à partir du convertisseur analogique numérique interne. Ils doivent être convertis en utilisant les formules suivantes :

Équation III-1 : $\text{Temperature(Tc)} = -39.60 + 0.01 * \text{la valeur mesurée.}$

Équation III-2 :$\text{Humidité}(\%) = -4 + 0.0405 * \text{val mesurée} + (-2.8 * 10^{-6}) * (\text{val mesurée}^2)$

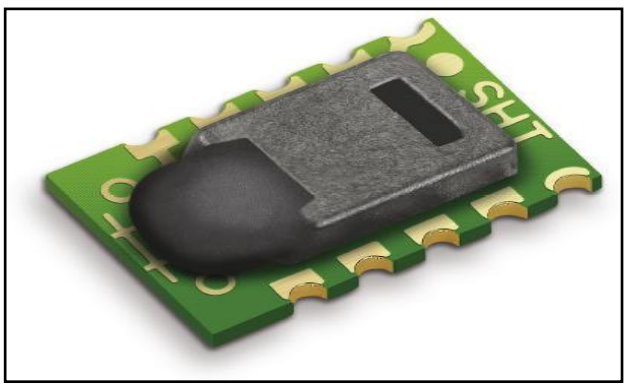


Figure III-10 : Détecteur Sensirion® SHT11

III.4.5. Détecteur de lumière Hamamatsu® S1087 / S1087-01:

Le CM5000 fournit deux détecteurs de lumière, le Hamamatsu® S1087 pour les mesures de la gamme visible et le S1087-01 pour la gamme infrarouge.

Ces capteurs analogiques sont connectés aux pins ADC4 et ADC5 du microcontrôleur. La lecture de données mesurées est effectuée selon la conversion suivante :

Équation III-3 : $\text{S1087 Light(lx)} = (2.5 * (\text{adc\_Light\_value}) / 4096) * 6250$

Équation III-4 : $\text{S1087} - 01 \text{ Light(lx)} = (1.5 * (\text{adc\_TSR\_value}) / 4096) * 1000$

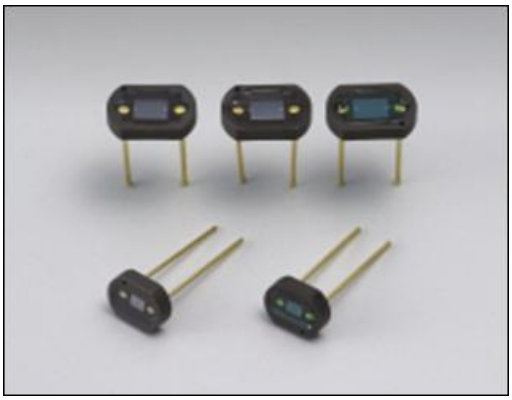


Figure III-11 : Hamamatsu® S1087

III.4.6. Interfaces USB :

Le CM5000 dispose d'une interface USB. Cela permet de transmettre les informations du capteur à un PC ou un périphérique similaire. Il est également utilisé pour reprogrammer le capteur avec un nouveau microprogramme.

Au cœur de l'interface USB, se trouve un FTDI® FT232BM qui fournit une compatibilité USB-UART.

Figure III-12 : FTDI® FT232BM

III.4.7. Mémoire flash ST® M25P80 :

Le capteur CM5000 dispose d'une mémoire flash externe de 1 Mo.

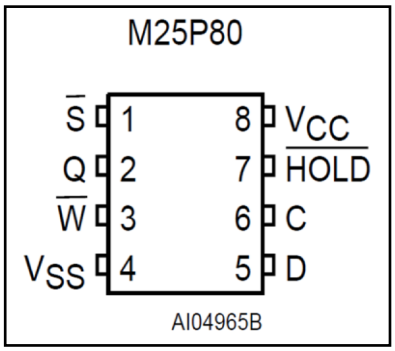


Figure III-13 : Mémoire flash ST® M25P80

III.4.8. Bottons d'utilisation :

Deux boutons sont disponibles sur le capteur, l'un fournit une réinitialisation matérielle qui redémarre l'appareil, et l'autre peut être programmé car il est directement connecté au microcontrôleur.

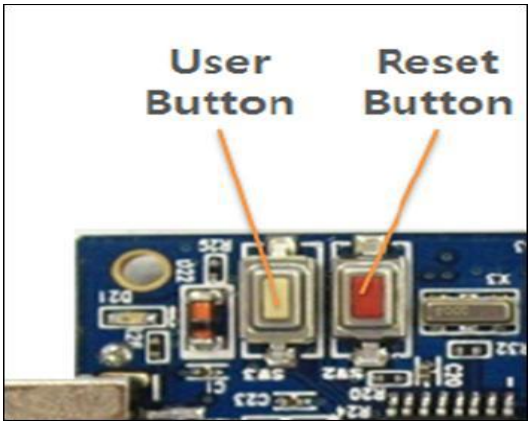


Figure III-14 : Bottons d'utilisation.

III.4.9. Alimentation du capteur :

Le capteur CM5000 peut être alimenté soit en branchant l'USB sur un ordinateur hôte, soit en utilisant des piles. Le support de batterie fourni permet d'utiliser des piles 2xAA. Le capteur dispose un régulateur de tension MICREL MIC5207, qui adapte l'entrée de puissance à la tension nécessaire.

III.5. Interfaces et composants du NESC utilisés :

Plusieurs interfaces et composants sont utilisés dans les deux algorithmes AODV et OAODV à savoir :

III.5.1. Initialisation :

Interface ‘Boot’ : elle est utilisée pour notifier les composants après le démarrage de TinyOS (initialisation de tous ses composants).

III.5.2. Compteurs :

Le composant ‘TimerMilliC’ qui est utilisé par l’interface temporelle de TinyOS ‘Timer’, afin d’exécuter les fonctions et les commandes après chaque période comme des événements. Dans notre projet, nous avons utilisé trois (03) compteurs.

III.5.3. Emission et réception :

1. Le composant ‘AMSenderC’ : géré par plusieurs interfaces, comme l'interface ‘AMSend’. Cette interface fournit des commandes pour envoyer un message et annuler l’envoi de message en attente. Elle fournit aussi un événement pour indiquer si un message a été envoyé avec succès ou non. Nous avons utilisé la fonction ‘AMSend.send’ pour envoyer le paquet à tous les nœuds que se trouve sous la couverture radio en spécifiant l’adresse soit broadcast (AM_BROADCAST_ADDR) pour le message RREQ ou adresse de destination pour le message RREP, un événement est signalé après une tentative de transmission de message (succès ou non).
2. Le composant ‘AMReceiverC’ : géré par l'interface ‘Receive’, cette interface est utilisée pour la réception des messages. Elle fournit également des commandes pour obtenir la longueur de la charge utile d'un message.
3. l'interface ‘Packet’ et ‘AMPacket’ : Elles Fournissent les accessoires de base pour le type de données message_t. Ces interfaces fournissent des commandes pour effacer le contenu d'un message et obtenir sa longueur de charge utile.

III.5.4. Utilisation des LEDs :

Le composant ‘LedsC’ : c’est un composant géré par l'interface Leds, utilisée pour manipuler les trois Leds.

III.5.5. Lecture des données captées :

Pour collecter la température mesurée au niveau des capteurs, nous avons utilisé l’interface ‘Read’, avec le composent ‘SensirionSht11C’. Nous avons accéder au résultat de

lecture par l'utilisation de l'événement 'readDone', en plaçant la valeur mesurée dans un paramètre val. La température réelle est calculée par la formule suivante :

Équation III-5 : $\text{temp} = -39.6 + 0.01 * \text{val}$

III.5.6. La communication radio entre les capteurs :

TinyOS fournit un certain nombre d'interfaces pour la gestion des services de communication. Toutes ces interfaces et ces composants utilisent une abstraction de mémoire commune appelée 'message_t', qui est implémentée comme une structure nesC.

Nous avons définit dans notre projet, un format de message appelé (AODV_pkt_t), utilisé pour envoyer les données par radio, qui contient les champs présenté dans le tableau suivant :

Type (RREQ, RREP, data)	N° de séquence	Nœud Destination	Nœud Source	Nombre de Hop	Nœud précédent	Data (06 champs)
1 octet	1 octet	1 octet	1 octet	1 octet	1 octet	12 octet

Tableau III-1 : Architecture de la trame utilisée

On utilise les interfaces 'Packet' et 'AMPacket' pour accéder au type de données abstraites 'message_t'. Le composant 'AMSenderC' doit être utilisé pour éviter l'interférence des communications radio entre les nœuds.

III.6. Implémentation des protocoles de routage (AODV et OAODV) :

III.6.1. Fonctions principales utilisées :

On a utilisé plusieurs fonctions pour la concrétisation des deux algorithmes AODV et OAODV, parmi les principaux fonctions utilisées on site :

III.6.1.1. Fonctions d'utilisation des requêtes RREQ : composés de :

a. La fonction d'envoi de RREQ :

Cette requête est envoyée par les capteurs dans le cas du protocole AODV ou par la station de base dans le cas du protocole OAODV :

```
void sendRREQ(uint8_t dest) {

     AODV_pkt_t* p ;
     p= (AODV_pkt_t*)call Packet.getPayload(&pkt,sizeof(AODV_pkt_t));

        atomic {
            free=1;
            }
     p->seq       = rreq_seq_++;
     p->dest      = dest;
     p->src       = TOS_NODE_ID;
     p->hop       = 1;
     p->prv       =TOS_NODE_ID;
     p->type      =RREQ;

        printf("envoi RREQ vers': %u\n", p->dest);

     add_rreq_cache( p->seq, p->dest, p->src, p->hop );
     if (call SendMsg.send (AM_BROADCAST_ADDR ,&pkt, sizeof ( AODV_pkt_t)  )==SUCCESS){
  atomic {
                free=0;
              }
     }
  }
```

Figure III-15 : Algorithme sendRREQ.

b. La fonction de réception de RREQ :

```
void recvRREQ(AODV_pkt_t* p) {
uint8_t me=TOS_NODE_ID ;
uint8_t src;
uint8_t dest;
uint8_t seq;
uint8_t hop;
uint8_t prv;

  seq=   p->seq;
  dest=  p->dest;
  src=   p->src;
  hop=   p->hop;
  prv=   p->prv;
     if ( rreq_cache_rep(p->src, p->seq ))return;
     if(is_rreq_cached( src, dest, seq, hop )){
  add_rreq_cache( p->seq, p->dest, p->src, hop );}
  add_route_table( seq, src, prv, hop ) ; // exploiter les info de routage
  if(dest==me){
    sendRREP(src);
  }
  else {
    forward_RREQ(p);
  }}
```

Figure III-16 : Algorithme recvRREQ.

c. **La fonction de retransmission de RREQ :** Cette requête est envoyée à partir des nœuds intermédiaires :

```
void forward_RREQ(AODV_pkt_t* p) {

  AODV_pkt_t* p1 = (AODV_pkt_t*)call Packet.getPayload(&pkt,sizeof(AODV_pkt_t));
     call Leds.led0Toggle();
      printf("forward_RREQ avec seq= %u\n",p->seq);
  if (seqR != p->seq) {
    seqR=p->seq;
  }
  else{
    return;
  }
      p1->seq       = p->seq;
      p1->dest      = p->dest;
      p1->src       = p->src;
      p1->type      =p->type;
      p1->hop++;
      p1->prv       =TOS_NODE_ID;
        if (call SendMsg.send (AM_BROADCAST_ADDR ,&pkt, sizeof ( AODV_pkt_t)  )==SUCCESS){

  atomic {
                free=0;
              }
      }
}
```

Figure III-17 : algorithme forward_RREQ.

III.6.1.2. Fonctions d'utilisation des requêtes RREP :

Ces requêtes sont utilisées seulement dans le cas du protocole AODV :

a. **La fonction d'envoi de la requête RREP :** elle est envoyée par la station de base, pour répondre aux requêtes RREQ envoyées par les capteurs cherchant la route.

```
void sendRREP(uint8_t dest) {
  uint8_t next;



      AODV_pkt_t* p = (AODV_pkt_t*)call Packet.getPayload(&pkt,sizeof(AODV_pkt_t));

      p->seq       = rreq_seq_++;
      p->dest      = dest;
      p->src       = TOS_NODE_ID;
      p->hop       = 1;
      p->prv       = TOS_NODE_ID;
      p->type       = RREP;
      next= get_next_hop(dest);

      if (call SendMsg.send (next ,&pkt, sizeof (AODV_pkt_t)  )==SUCCESS){
        printf("sendRREP vers: %u\n", p->dest);
  atomic {
                free=0;
              }
      }
}
```

Figure III-18 : Algorithme sendRREP.

b. La réception de RREP par les nœuds.

```
void recvRREP(AODV_pkt_t* p) {
     uint8_t dest;
     uint8_t me;
     me=TOS_NODE_ID ;
     dest  = p->dest;
     printf ("recvRREP de: %u\n", p->src);
     add_route_table(p->seq,p->src,p->prv,p->hop) ;

     if(dest!=me) forward_RREP(p);
     free=0;
}
```

Figure III-19 : Algorithme recvRREP.

c. La retransmission de RREP :

```
void forward_RREP(AODV_pkt_t* p){
    uint8_t dest;
      uint8_t next;
      AODV_pkt_t* p1 = (AODV_pkt_t*)call Packet.getPayload(&pkt,sizeof(AODV_pkt_t));
 atomic {
     free=1;
     }
      dest=p->dest;
      next= get_next_hop(dest);
      p1->seq       = p->seq;
      p1->dest      = p->dest;
      p1->src       = p->src;
      p1->type      =p->type;
      p1->prv =TOS_NODE_ID;
      p1->hop++;

        printf("forward_RREP dest': %u\n", p->dest);
      if (call SendMsg.send (next ,&pkt, sizeof (AODV_pkt_t)  )==SUCCESS){
  atomic {
                free=0;
            }
      }
 }
```

Figure III-20 : algorithme forward_RREP.

III.6.1.3. Fonctions de manipulation des requêtes DATA :

a. **Fonction d'envoi des données** : pour l'envoi des données (température) à partir des capteurs vers la station de base :

```
void sendDATA(uint16_t dataa){
    uint8_t next;
    uint16_t data2=dataa;
      AODV_pkt_t* p = (AODV_pkt_t*)call Packet.getPayload(&pkt,sizeof(AODV_pkt_t));
      p->dest      = BS;
      p->src       = TOS_NODE_ID;
      p->type       = DATA1;
      next= get_next_hop(BS);
      if(next==INVALID_NODE_ID){
      return;
      }
     dt[0]=id;
     dt[1]=seqA++;
     dt[2]=x;
     dt[3]=y;
     dt[4]=z;
     dt[5]=data2;
      printf("envoi temperature= %u du capteur N %u dont la posistion est %u, %u,%u
              avec seq=%u\n",dt[5],dt[0],dt[2],dt[3],dt[4], dt[1]);
      RC4(); // pour generer le tableau pseoudo aleatoire
      encrypt();// crypter le
      p->data1[0]=dt[0];
      p->data1[1]=dt[1];
      p->data1[2]=dt[2];
      p->data1[3]=dt[3];
      p->data1[4]=dt[4];
      p->data1[5]=dt[5];
      printf("chiffres : temperature= %u du capteur N %u dont la posistion %u, %u,%u
              avec seq=%u\n",dt[5],dt[0],dt[2],dt[3],dt[4], dt[1]);
      if (call SendMsg.send (next ,&pkt, sizeof (AODV_pkt_t)  )==SUCCESS){
      atomic {
                free=0;
              }
      }
}
```

Figure III-21 : algorithme sendDATA.

b. **Fonction de réception des données par la station de base** :

```
void recvDATA(AODV_pkt_t* p){
    uint8_t vall;
    uint8_t dest;
    uint8_t me=TOS_NODE_ID ;
    dest=p->dest;
  if(dest!=me) {
    forward_DATA(p);
    //printf("forward_DATA vers: %u\n");
    }
    if(dest==me) {
        dt[0]=p->data1[0];
        dt[1]=p->data1[1];
        dt[2]=p->data1[2];
        dt[3]=p->data1[3];
        dt[4]=p->data1[4];
        dt[5]=p->data1[5];
    RC4();
    decrypt();
         printf("DATA chiffré receptionnée de la src N : %u dont la position est %u, %u, %u est de : %u
                avec seq = %u\n",p->data1[0],p->data1[2],p->data1[3],p->data1[4],p->data1[5],p->data1[1]);
         printf("en claire : Data de la src N : %u dont la position est %u, %u, %u est de : %u
                avec seq = %u\n",dt[0],dt[2],dt[3], dt[4],dt[5],dt[1]);
    }
}
```

Figure III-22 : algorithme recvDATA.

III.6.1.4. Fonctions de manipulation des messages HELLO :

Utilisés seulement dans l'algorithme AODV pour l'envoi et la réception des messages Hello entre chaque nœud et ses voisins.

```
void sendHELLO() {
      AODV_pkt_t* p= (AODV_pkt_t*)call Packet.getPayload(&pkt,sizeof(AODV_pkt_t));
      p->seq       = rreq_seq_++;
      p->dest      = BS;
      p->src       = TOS_NODE_ID;
      p->hop       = 1;
      p->prv       =TOS_NODE_ID;
      p->type      =HELLO;
      if (TOS_NODE_ID!=BS){
        printf("sendHELLO': %u\n");
        }
      add_rreq_cache( p->seq, p->dest, p->src, p->hop );
      if (call SendMsg.send (AM_BROADCAST_ADDR ,&pkt, sizeof ( AODV_pkt_t)  )==SUCCESS){
  }
//*****************************************************************************************
void recvHELLO(AODV_pkt_t* p) {
    uint8_t dest;
    uint8_t hop;
    uint8_t prv;
    uint8_t src;
  dest=  p->dest;
  hop=   p->hop;
  prv=   p->prv;
  src=   p->src;
   // add_route_table(p->seq,p->src,p->prv,p->hop) ;
    mise_a_jour(dest, prv);
    free=0;
    if (TOS_NODE_ID!=BS){
        printf("recvHELLO': %u\n",p->src);
        }
}
```

Figure III-23 : Algorithme des messages HELLO

III.6.1.5. Fonctions de mise à jour de la table de routage :

Utilisées pour l'ajout d'une ligne à la table de routage des capteurs, et pour la mise à jours de la table de routage, par la décrémentation de la valeur de TTL fixé initialement à 10.

III.6.2. Programmation :

Pour le AODV, notre package TinyOS contient les fichiers Nesc suivants :

- ✓ Le fichier de configuration AODVAppC.nc.
- ✓ Le fichier de module AODV.nc.
- ✓ Le fichier header AODV.h.

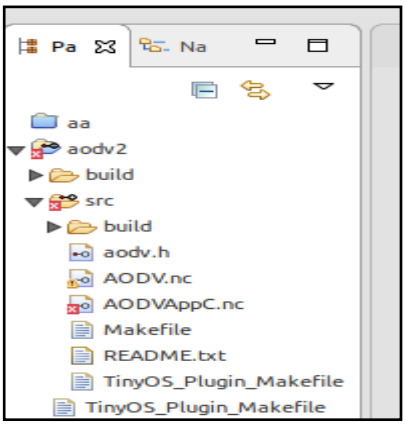


Figure III-24 : Package AODV

Pour le OAODV, notre package TinyOS contient les fichiers Nesc suivants :

- ✓ Le fichier de configuration AODV_OPAppC.nc.
- ✓ Le fichier de module AODV_OP.nc.
- ✓ Le fichier header AODV.h.

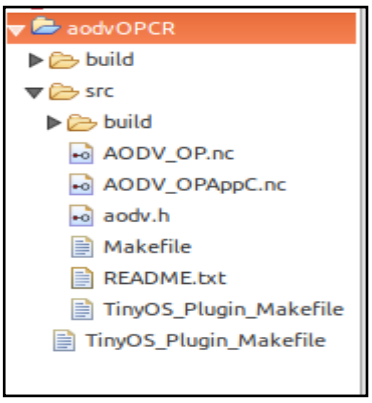


Figure III-25 : Package OAODV

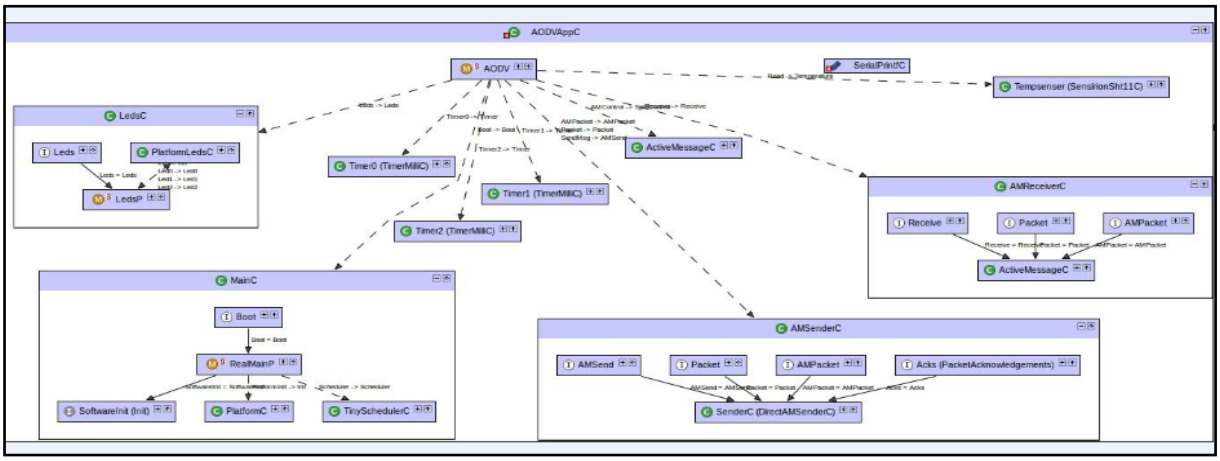


Figure III-26 : Connexion des composants.

III.7. Implémentation des algorithmes de cryptage (RC4 et ECDSA):

La solution de sécurité proposée consiste à l'utilisation d'un algorithme de chiffrement symétrique RC4 et de la signature ECDSA.

III.7.1. Algorithme RC4 :

Nous avons procédé à l'implémentation de cet algorithme dans tous les capteurs, en enregistrant la clé privé dans chaque capteur lors de l'opération de configuration, une partie de la trame data sera cryptés et décryptés, cette partie contient six (06) champs, chacun est composé de deux (02) octets, elle contient principalement la température mesuré, les

coordonnées GPS (latitude, longitude et l'altitude), l'identifiant du capteur et la séquence de l'envoi de la trame data, les fonctions utilisées sont **:**

Fonction d'initialisation : permet d'initialiser le vecteur S et le vecteur temporaire, on utilisant la clé de chiffrement. Elle permet aussi de réaliser les permutations sur le vecteur S.

Fonction de génération de flux : elle consiste à choisir une cellule du vecteur S on utilisant une méthode basée sur le numéro du champ de la donnée et du contenu du vecteur S.

```
void RC4()  {
uint8_t ii;
uint8_t i;
uint8_t j;
for ( i =0; i < 255; i++) {
T[i]=0;
S[i]=0;
}
for ( i =0; i < 255; i++) {
S[i] = i;
T[i] = key[i % keylen];
}
 j = 0;
for ( i =0; i < 255; i++) {
j = (j + S[i] + T[i]) % 255;
S[i] ^= S[j];
S[j] ^= S[i];
S[i] ^= S[j];
}
}
```

Figure III-27 : Fonction d'initialisation et de génération de flux

Fonction de cryptage et de décryptage : elle permet de réaliser la fonction XOR entre la valeur de la température chiffrée ou claire et la valeur de flux de chiffrement.

```
void encrypt() {
uint8_t i = 0, j = 0, k, t, counter;
for ( counter = 0; counter < tmsg; counter++) {
i = (i + 1) % 255;
j = (j + S[i]) % 255;
S[i] ^= S[j];
S[j] ^= S[i];
S[i] ^= S[j];
t = (S[i] + S[j]) % 255;
k = S[t];
dt[counter] = dt[counter] ^ k;
}}

void decrypt() {
uint8_t i = 0, j = 0, k, t, counter;
for ( counter = 0; counter < tmsg; counter++) {
i = (i + 1) % 255;
j = (j + S[i]) % 255;
S[i] ^= S[j];
S[j] ^= S[i];
S[i] ^= S[j];
t = (S[i] + S[j]) % 255;
k = S[t];
dt[counter] = dt[counter] ^ k;
}}
```

Figure III-28 : Fonction de chiffrement et de déchiffrement.

III.7.2. Algorithme ECDSA :

Nous avons utilisé dans notre solution un algorithme asymétrique basé sur les courbes elliptiques (ECDSA) détaillé dans le chapitre 03. Notre implémentation de cet algorithme est basée sur la version 2.0 de la bibliothèque TinyECC [31], configurable pour les opérations d'ECC dans les RCSF. Nous avons choisi secp160r1 comme paramètres de la courbe elliptique déjà implémentée dans la librairie TinyECC.

Notre proposition consiste à l'implémentation de la signature ECDSA dans tous les capteurs, après création des paires de clés privé et publique pour l'ensemble des capteurs, nous procédant à l'enregistrement de tous les clés publiques dans tous les capteurs lors de leur configuration.

Seule les requêtes RREQ pour le protocole OAODV et les requêtes RREQ et RREP pour le protocole AODV qui seront signés, afin de vérifier l'authentification des capteurs qui retransmettre ces requêtes, et donc assurer le non débordement du chemin à la station de base vers une destination d'écoute n'appartienne pas à notre réseau.

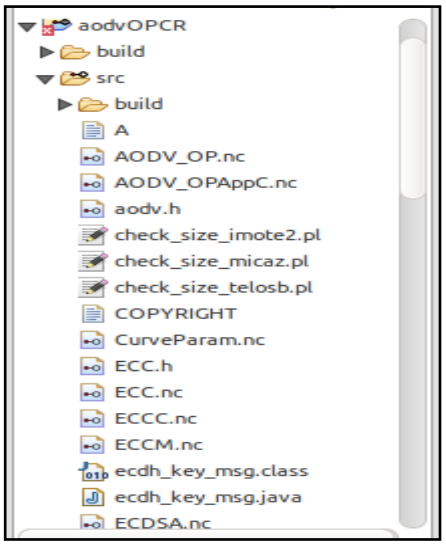


Figure III-29 : Package de protocole AODV avec la bibliothèque TinyECC.

III.8. Conclusion :

Dans ce chapitre, nous avons présenté les différents systèmes, outilles et moyens de développement et d'implémentation exploités dans notre projet, nous avons vu les principaux composants, interfaces et fonctions utilisés pour l'implémentation des protocoles de routage AODV et AODV optimisé et des algorithmes de chiffrement symétrique RC4 et asymétrique ECDSA.

CHAPITRE IV : IMPLEMENTATION ET RESULTATS

IV.1. Introduction :

Le but de notre travail est d'évaluer le protocole de routage adaptés au RCSF, appelé OAODV, en le comparant avec le protocole de routage standard AODV, toute en assurant la sécurité des communications radio, à travers une solution basée sur la cryptographie symétrique et asymétrique.

Dans ce chapitre, nous allons présenter les différents tests effectués relatifs à la mesure des performances des protocoles de routage AODV et AODV optimisé et les algorithmes de chiffrement symétrique RC4 et asymétrique ECC implémentés, en analysant les résultats obtenus. Nous allons exposer notre application web développé pour le contrôle de la température dans un environnement industrielle.

IV.2. Test et résultats :

IV.2.1. Paramètres de test :

Les paramètres de test ont été choisit pour permettre une bonne visualisation de fonctionnement des capteurs et pour faciliter aussi la comparaison entre les deux protocoles de routages AODV et OAODV, ces paramètres sont mentionnées dans le tableau suivant :

N°	Action	Modèle I (Temps)	Modèle II (Temps)
01	Envoi de message Hello	3 S	9 S
02	Envoi des données (DATA)	2 S	10 S
03	Mise à jour de la route	1 S	3 S
04	Expiration de la route (T=10)	10 S	30 S
05	Envoi de la requête RREQ	Recherche initiale ou expiration de la route	
06	Envoi de la requête RREP (par la Station de Base)	Après réception de RREQ	
07	forward la requête RREQ (par les capteurs)	Après réception de RREQ	
08	forward la requête RREP (par les capteurs)	Après réception de RREP	
09	forward la DATA (par les capteurs)	Après réception de DATA	

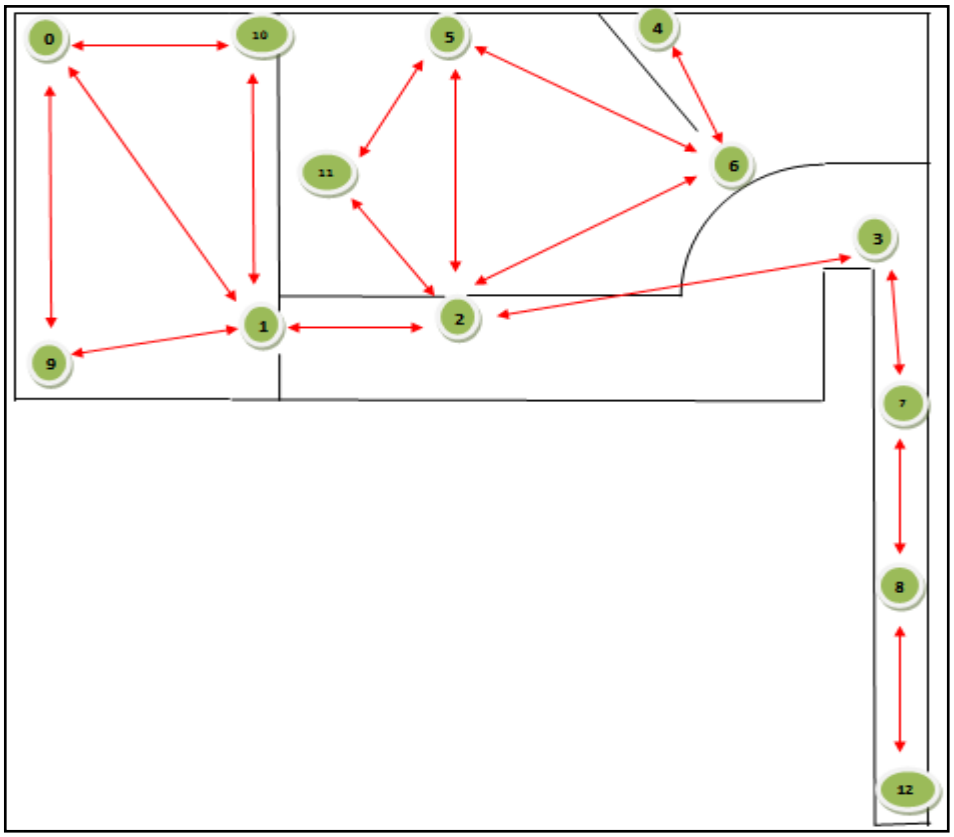
Tableau IV-1 : Paramètres du protocole AODV.

N°	Action	Modèle I (Temps)	Modèle II (Temps)
01	Envoi de requêtes RREQ (par la station de base)	3 S	9 S
02	Envoi des données (DATA)	2 S	10 S
03	Mise à jour de la route	1 S	3 S
04	Expiration de la route	10 S	30 S
05	forward la requête RREQ (par les capteurs)	Après réception de RREQ	
06	forward la DATA (par les capteurs)	Après réception de DATA	

Tableau IV-2 : Paramètres du protocole OAODV.

IV.2.2. Architecture de réseau :

Nous avons effectué les différents tests sur l'architecture de réseau présentée dans la figure suivante, en modifiant la taille de réseau allant de deux (02) capteurs jusqu'à un réseau de deuze (12) capteurs. Les nœuds sont numéroté de 0 à 12, les flèches en rouge indiquent la connexion radio entre deux capteur (couverture radio).



IV.2.3. Tests sur le fonctionnement des protocoles de routages :

Afin de vérifier le bon fonctionnement de l'algorithme AODV et OAODV, nous avons implémenté le protocole AODV et OAODV dans trois capteurs, dont un (01) est considéré comme station de base. Nous avons supervisé les messages envoyés et reçus à travers le port série. Le capteur N°2 se trouve hors la couverture de la station de base, tandis que le capteur N°1 est placé, de telle sorte qu'il peut établir des communications radio avec la station de base et avec le capteur N°2 :

IV.2.3.1. Algorithme AODV :

En analysant les messages reçu par les capteurs, on remarque que lorsque le capteur N° 1 et 2 sont mis sous tension, ils commencent à envoyer périodiquement des requêtes RREQ jusqu'a la réception d'une RREP venu de la station de base, en même temps, ils envoient des messages Hello chaque 03 s pour vérifier les nœuds de son voisinage.

Après réception du RREP, le capteur N°1 construit sa route vers la station de base, tandis que, Le capteur 2 qui se trouve hors la couverture radio de la station de base reçois la RREP par l'intermédiaire du capteur N°1, il construit alors sa route vers la station de base, en considérant le capteur N°1 comme le nœud suivant.

Les route sont mis à jour périodiquement (décrémentation de TTL chaque période), et après la réception de message Hello de la station de base la valeur de TTL est mis à10.

Si la station de base est mis hors tension, chaque capteur décrémente le TTL de sa route vers la station de base jusqu'à qu'il devient nul, il lance donc la recherche de la route à travers la requête RREQ.

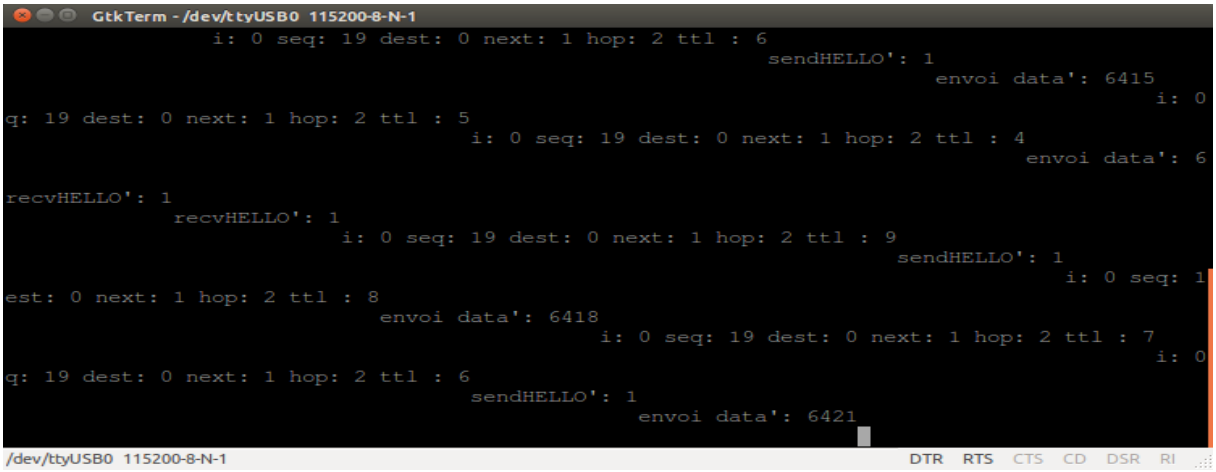


Figure IV-1 : Envoi des données par le capteur N°2.

Les deux capteurs dont leurs ID est respectivement 1 et 2 envoient les données vers la station de base, comme le montre la figure suivante, qui représente les messages reçus au niveau de la station de base.

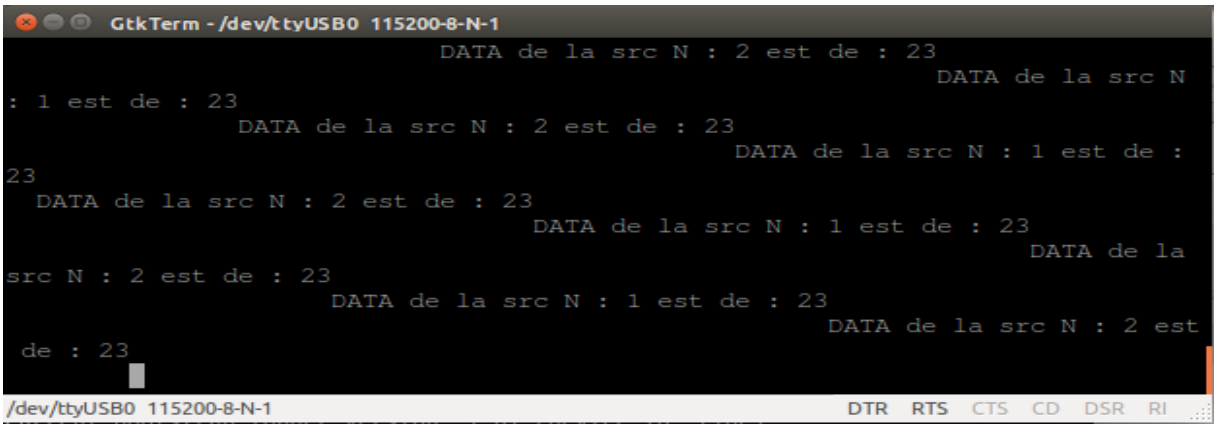


Figure IV-2 : réception des données au niveau de la station de base.

IV.2.3.2. Algorithme AODV Optimisé :

Après de la réception de la requête RREQ envoyée périodiquement par la station de base, le capteur N°1 construit sa route vers la station de base, et retransmit en même temps la requête RREQ envoyés par la station de base. Le capteur N°2 construit sa route vers la station de base, en considérant le capteur N°1 comme le nœud suivant.

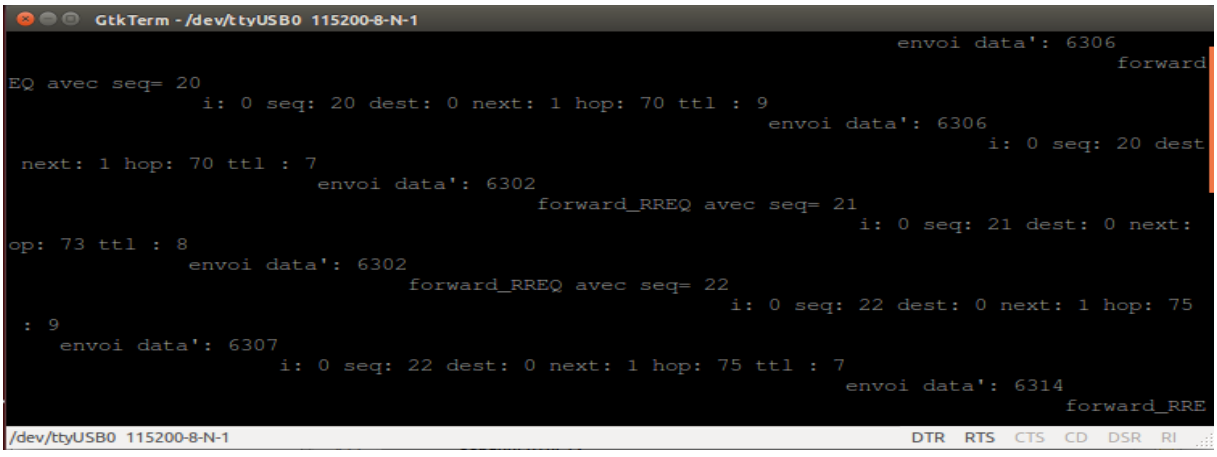


Figure IV-3 : élaboration de la route par le capteur N°2.

Après élaboration des routes, les capteurs procèdent à l'envoi des données vers la station de base.

Les routes vers la station de base sont mise à jour par décrémentation de la valeur de TTL, cette valeur est rendu à la valeur de dix (10) à chaque réception de RREQ envoyés par la station de base. Si la station de base est mis hors tension, la valeur de TTL est décrémentée

jusqu'à la valeur zéro, pour la quelle, la route est supprimé, et les capteurs arrêtent l'envoi des données.

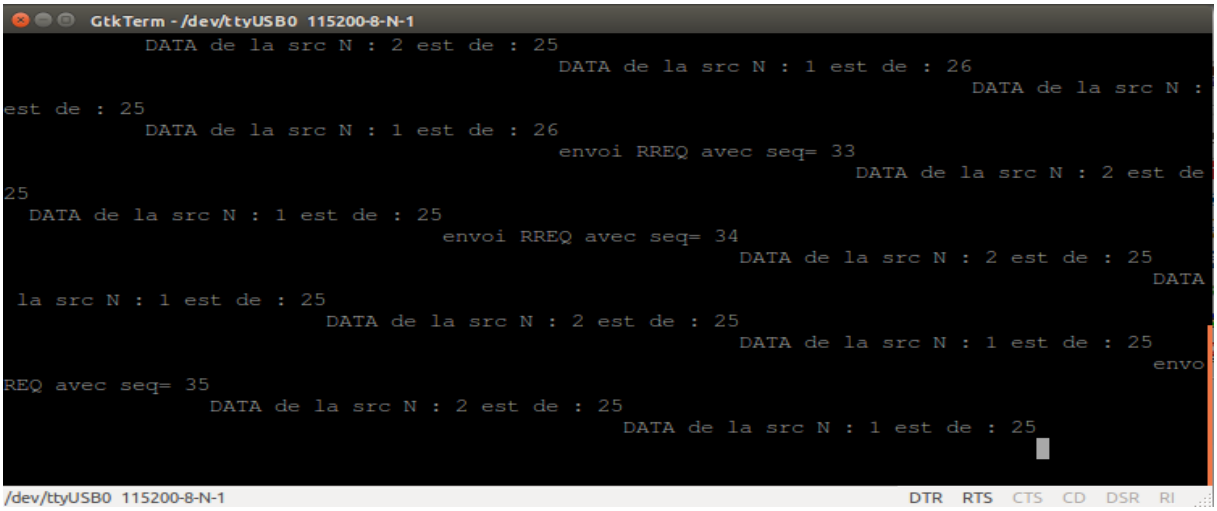


Figure IV-4 : réception des données au niveau de la station de base.

IV.2.4. Tests sur le taux de perte des paquets :

IV.2.4.1. Déroulement de test :

L'objectif de ce teste est de mesurer les performances de l'algorithme AODV Optimisé et AODV, à travers la mesure de taux de perte des paquets émises.

Les différents capteurs de réseau envoient leurs paquets indexés à la station de base, cette dernière affichent les paquets reçus sur l'interface du port série, les paquets non affichées sont considérés comme des paquets perdus. Les résultats obtenus pour les différentes tailles de réseau sont présentées dans le tableau suivant :

Résultats avec Modèle I						
Nombre de capteurs		**2**	**3**	**6**	**9**	**12**
Nombre total des paquets		2*150	3*150	6*150	9*150	12*150
AODV Optimisé	**paquets perdus**	0	0	84	540	810
	% des paquets perdus	0 %	0 %	9,33 %	40 %	45,50 %
AODV	**paquets perdus**	11	19	110	Blocage de réseau	
	% des paquets perdus	3.67	4.22 %	12,23 %		

Tableau IV-3 : Pertes des paquets pour le modèle I

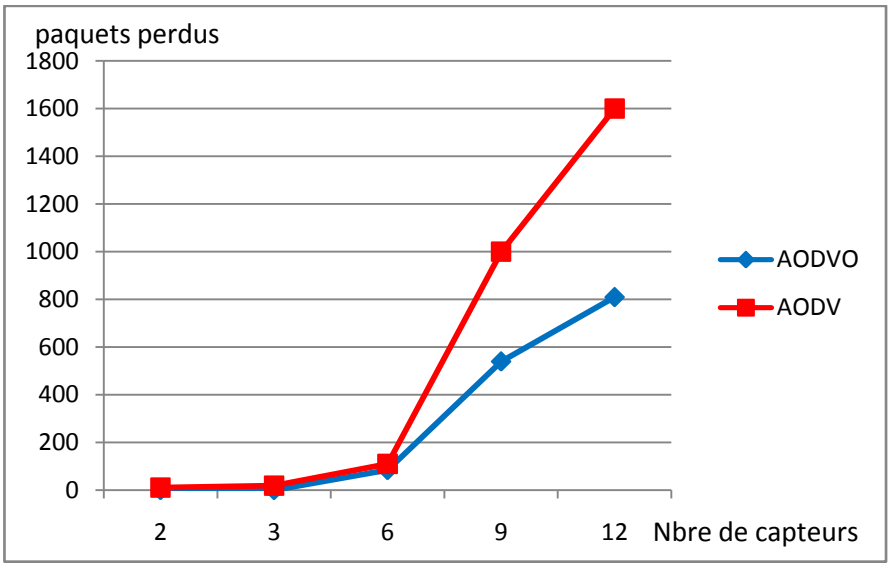


Figure IV-5 : Pertes des paquets pour le modèle I

Résultats avec Modèle II						
Nombre de capteurs		**2**	**3**	**6**	**9**	**12**
Nombre total des paquets		2*50	3*50	6*50	9*50	12*50
AODV Optimisé	**paquets perdus**	0	0	25	140	214
	% des paquets perdus	0 %	0 %	8.33 %	31.11 %	35.66
AODV	**paquets perdus**	0	1	32		Blocage de réseau
	% des paquets perdus	0 %	0.67 %	10.66 %		

Tableau IV-4 : Pertes des paquets pour le modèle II

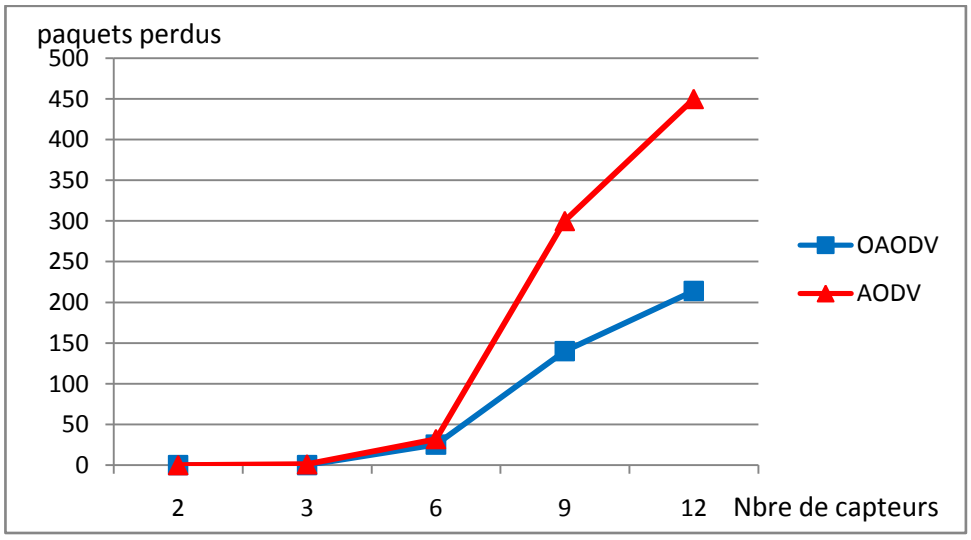


Figure IV-6 : Pertes des paquets pour le modèle II

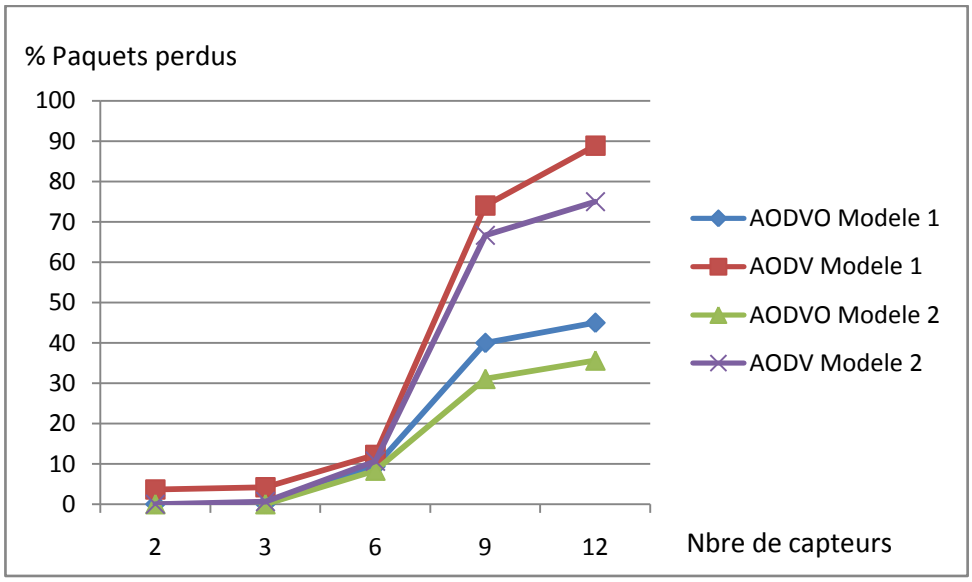


Figure IV-7 : Pertes des paquets

IV.2.4.2. Analyse des résultats :

En comparant les résultats obtenus, on peut déduire que le protocole AODV Optimisé présente moins de perte par rapport au protocole AODV. Ceci est du principalement à la diminution du nombre des messages d'établissement de la route et de mise à jours de la table pour le protocole AODV optimisé.

En augmentant la taille de réseau, les pertes des paquets envoyées en utilisant le protocole AODV optimisés augmentent et donnent des pourcentages de perte plus importants, tandis que pour le protocole AODV, on remarque un blocage de réseau lorsque le nombre des capteurs dépasse le six (06) capteurs pour le premier modèle et neuf (09) capteurs pour le deuxième modèle. Ceci revient d'une part à l'architecture difficile de réseau implanté dans un bloc contenant plusieurs obstacles, engendrant des réflexions radio, et d'autre part, à la limitation des ressources des capteurs utilisés, conduisant à la perte des paquets des données dans les filles d'attente.

En effet, les nœuds principaux dans notre architecture, utilisant le protocole AODV optimisé, sont soumis a une charge de calcule importante, à cause des opérations de routage des données. Cette charge de calcule augmente dans le cas d'utilisation du protocole AODV, à cause du traitement des messages Hello pour la mise à jour de la table de routage, et à cause des opérations de routage des données, ce qui a engendré un blocage de ces nœuds centraux.

En appliquant les paramètres du deuxième modèle, on constate une réduction des pertes des données envoyées par les deux protocoles. Cette réduction revient à la l'augmentation des

périodes de l'envoi des données et de la mise à jour de la table de routage, ce qui permettre d'avoir une légère amélioration dans la gestion des filles d'attente des capteurs centraux.

En mode réception, le capteur ne peut pas tamponner plus de 128 octets, le surplus des données est automatiquement supprimés. Dans notre cas, la longueur de la trame envoyée est de dix huit (18) octets, et sachant que une entête est ajoutée à l'émission d'une longueur de deux (02) octets contenant l'adresse source, ce qui augmente la longueur réel de la trame envoyés à vingt (20) octets. On trouve donc que le nombre maximal des trames qui peuvent être traités en même temps à la réception est de six (06) trames au maximum.

En exploitant de ces données, on trouve que pour le protocole AODV optimisé, le noud central ne peut pas recevoir les trames datas de plus de six (06) capteurs en même temps.

Tandis que pour le protocole AODV, et à cause de multiple trames envoyés data et message Hello, le noud central ne peut pas être entouré par plus de trois capteurs et recevoir en même temps leurs donnée et leurs messages Hello, ce qui est difficile à éviter dans un RCSF.

Les résultats obtenus peuvent être nettement améliorées dans un terrain dégagé sans obstacles et avec moins de nouds centraux (architecture non hiérarchique), et en réduisant la taille des trames, principalement celles des messages Hello et des requêtes RREQ.

IV.2.5. Testes sur l'effet de la taille de réseau sur la recherche de la route :

IV.2.5.1. Déroulement du test :

Ce test consiste au calcule du temps nécessaire pour l'établissement de la route en fonctions du nombre de hop (pas), en comparant les résultats obtenu pour les deux protocoles AODV et OAODV.

Pour le protocole AODV optimisé, nous avons procédé à la mesure du temps entre l'envoi de la requête RREQ par la station de base et la réception de la trame data, au niveau de la même station. L'envoi de la data a été activé automatiquement au niveau du capteur cible directement après la réception de la requête RREQ. L'envoi des trames datas a été désactivé au niveau des autres capteurs intermédiaires, afin d'éviter la perte de la première trame data envoyés par le capteur cible. Un compteur a été lancé au niveau de la station de base après l'envoi de la RREQ.

Sachant que théoriquement la route sera établie après réception du capteur cible de la requête RREQ envoyée par la station de base, on déduit que le temps estimé pour la recherche

de la route dans le protocole OAODV sera la moitié du temps mesuré. Les résultats obtenus sont mentionnées dans le tableau suivant :

Nbre de hop	**1**	**2**	**3**	**4**
Temps (ms)	15	25	44	71

Tableau IV-5 : Temps de recherche de la route (OAODV)

Dans le cas du protocole AODV, nous avons mesuré le temps entre l'envoi de la requête RREQ par le capteur cible et la réception de la requête RREP, au niveau du même capteur. L'envoi des trames datas a été désactivé au niveau des autres capteurs intermédiaires, afin d'éviter la perte de la première RREP envoyés par la station de base. Un compteur a été lancé au niveau du capteur cible après l'envoi de la RREQ. Les résultats obtenus sont mentionnées dans le tableau suivant :

Nbre de hop	**1**	**2**	**3**	**4**
Temps (ms)	26	41	70	107

Tableau IV-6 : Temps de recherche de la route (AODV)

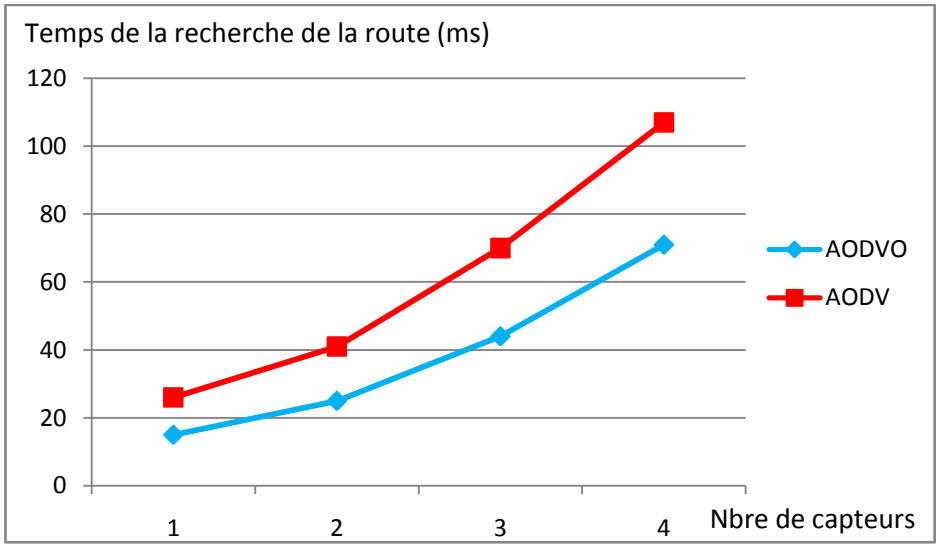


Figure IV-8 : Temps de recherche de la route pour l'AODV et OAODV

IV.2.5.2. Analyse des résultats :

Ces tests ont été déroulés dans des bonnes conditions, afin d'éviter la perte des trames data ou des requêtes RREQ ou RREP.

On constate que le temps de recherche de la route pour le protocole OAODV est inferieur à celui du protocole AODV, environ la moitié. Ceci est due au chemin double qu'il

travers le signale radio pour la recherche de la route dans le protocole AODV, par rapport au protocole OAODV.

IV.2.6. Testes sur l'effet de la taille de réseau sur la consommation de l'énergie :

IV.2.6.1. Déroulement du teste :

L'objectif de ce teste est de comparer les performances du protocole OADOV et AODV pour la consommation de l'énergie, en fonction du nombre des capteurs composant le réseau. Pour cela nous avons procéder au calcule du nombre des requêtes de tous types (RREQ, RREP, DATA et message Hello) envoyés et reçus pour chaque capteur dans l'architecture de réseau présentée dans la figure N°. et ceci pendant une durée de 10s.

Comme nous l'avons vu précédent, la taille de la trame envoyée est de 20 octets pour tous les requêtes (RREQ, RREP, DATA et message Hello), la transmission d'un seul octet en utilisant les capteurs telosb consomme 59.2 µj, et la réception consomme 28.6 µj [4]. On trouve donc que l'énergie consommée est de 1184 µj pour l'envoi d'une trame ou requête, tandis que pour la réception, elle est de 572 µj. En obtenant l'énergie totale par l'équation suivante :

Équation IV-1 : $E_t = N_{send} * 1184 + N_{recv} * 572$

Les résultats obtenus sont présentés dans le tableau suivant :

Désignation	**Taille de réseau**		**2**	**3**	**6**	**9**	**12**
OAODV	Nbre de requête	Envoyée	10	21	66	127	176
		Reçue	10	32	154	247	538
	Energie consommée (µj)		**17560**	**43168**	**166232**	**291652**	**516120**
AODV	Nbre de requête	Envoyée	12	28	103	216	332
		Reçue	12	41	236	429	954
	Energie consommée (µj)		**21072**	**56604**	**256944**	**501132**	**938776**

Tableau IV-7 : La consommation de l'énergie en fonction du Nombre des capteurs

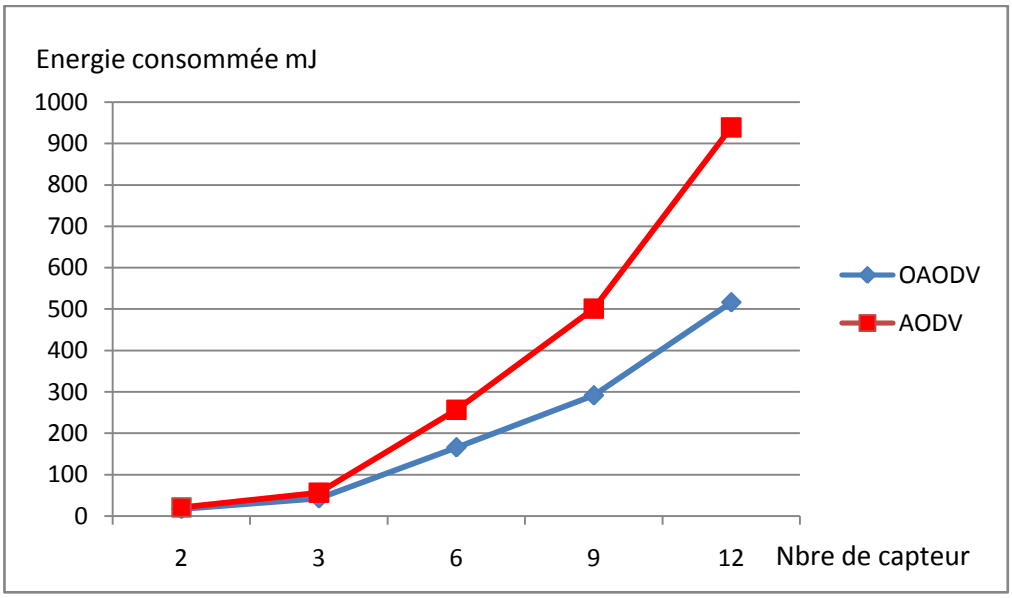


Figure IV-9 : La consommation de l'énergie en fonction du Nombre des capteurs

IV.2.6.2. Analyse des résultats :

On constate que la consommation de l'énergie est proportionnelle à la taille de réseau. Le protocole OAODV préserve l'énergie des capteurs grâce au nombre limité des requêtes envoyées et reçues par rapport au protocole AODV. Cette différence de consommation d'énergie devient plus importante avec l'augmentation de réseau et atteint presque le double pour un réseau de 12 capteurs.

IV.2.7. Tests sur le fonctionnement des algorithmes de cryptages :

IV.2.7.1. Test pour vérifier le bon fonctionnement du RC4:

Nous avons effectué un test initial pour vérifier le fonctionnement de l'algorithme RC4. Nous avons implémenté l'algorithme de chiffrement RC4 dans deux capteurs, dont un est considéré comme station de base. Les messages réceptionnés au niveau de la station de base sont présentés dans la figure suivante, le premier message présente les données reçus chiffrés, et le deuxième message affiche les données après déchiffrement :

Figure IV-6 : Opération de déchiffrement RC4.

IV.2.7.2. La mesure du temps d'exécution de l'algorithme RC4 :

Afin de mesurer le temps d'exécution de l'algorithme de chiffrement RC4 dans le protocole OAODV, nous avons procéder à la mesure du temps entre l'envoi de la requête RREQ par la station de base et la réception de la trame data, au niveau de la même station, avec et sans chiffrement RC4. L'envoi de la data a été activé automatiquement au niveau du capteur cible directement après la réception de la requête RREQ. Les résultats obtenus sont présentés dans le tableau suivant :

Nbre de hop	**1**	**2**	**3**
Temps (ms) sans chiffrement	30	50	88
Temps (ms) avec chiffrement	54	70	115
Temps de chiffrement	24	20	27

La valeur moyenne de chiffrement est de 24 ms, ce qui indique que le temps d'exécution de l'algorithme RC4 en chiffrement ou déchiffrement au niveau de chaque capteur est de 20ms, ce temps est jugé adapté au bon fonctionnement des RCSF.

Remarque :

Nous n'avons pas réussi à mettre en fonctionnement l'algorithme de chiffrement ECDSA, à cause de plusieurs erreurs signalées dans plusieurs fichiers de la bibliothèque TinyECC, lors de la compilation de cet algorithme, ce qui ne nous a pas permet d'effectuer les tests et les mesures nécessaires.

IV.3. Conception d'une application Web pour le contrôle de la température :

Pour permettre la visualisation et le suivi des données envoyées par le RCSF conçu, nous avons développé une application Web de contrôle de température, destiné à assurer la sécurité dans une unité de production ou dans un block contenant des équipements sensible et nécessitant un suivie permanant.

Nous avons choisis comme exemple, l'usine de ciments de Beni-Saf de la wilya de Aïn Témouchent, en raison de l'importance de l'opération de contrôle de température dans les cimenteries. Cette application permet de surveiller le réseau d'une façon efficace et rapide et offre des services de filtrage et d'alerte. Elle assure le contrôle à distance, et permanant de l'usine.

IV.3.1. Langage PHP :

PHP est un langage de script HTML exécuté du côté du serveur, utilisés pour réalisés des applications dynamiques. Sa syntaxe est largement inspirée du langage C, de Java et de Perl, avec des améliorations spécifiques. PHP est principalement utilisé pour rendre un site dynamique, c'est-à-dire que l'utilisateur envoie des informations au serveur qui lui renvoie les résultats modifiés en fonction de ces informations. A contraire, un site statique ne s'adapte pas aux informations fournies par un utilisateur.

IV.3.2. Base des données :

La base de données utilisée dans notre application a été crée sur le SGBD MySQL sous le nom 'température'.

IV.3.2.1. Table de la BD :

Nous avons créé une base de données nommée 'température' sur le SGBD MySQL, contenant quatre (04) tables :

Table 'capteurs' : elle contient les numéros des capteurs dans le réseau (leurs ID utilisés dans le protocole de routage), la permettions pour l'affichage sur l'application, et un champ pour des informations supplémentaires.

Table 'type_alerte' : dans cette table, nous avons inclus les différents types d'alertes possibles, à savoir :

- ✓ Le dépassement de la température (température ≥ TempMax=35°) ;
- ✓ Le démunissions de la température (température ≤ TempMin=15°).

Table 'mesures' : Cette table contient les données relatives à une mesure, à savoir la valeur de la mesure, la date et l'heur de la mesure, la position géographique du capteur, ainsi que l'ID de la table 'capteur' pour désigner le capteur effectuant la mesure, et l'ID de la table 'type_alerte' indiquant le type d'alerte.

Table 'utilisateurs' : contenant les utilisateurs et leurs mots de passes, ainsi que le droit d'administration.

IV.3.2.2. Connexion à la BD :

L'extension PDO a été choisis pour accéder à la de base de données MySQL crée. Ce moyen peut être utilisé pour se connecter aussi bien MySQL que PostgreSQL ou Oracle.

IV.3.2.3. Alimentation de la BD :

a) Réception des mesures :

Le capteur jouant le rôle de la station de base doit recevoir toutes les mesures envoyées par les capteurs de réseau, et il doit communiquer ces données à notre application. Pour cela, nous avons d'abord enrichi le programme de ce capteur par l'ajout des interfaces, composants, liens, commandes et événements nécessaires pour le fonctionnement en tant que station de base.

En deuxième étape nous avons accéder au fichier listen.java, qui se trouve dans le répertoire « /opt/tinyos-2.1.2/support/sdk/java/net/tinyos/tools», en lui apportant des modifications pour permettre l'affichage de données reçus par extraction en chaine de caractère à partir des requêtes reçus en hexadécimale.

Après modification, on utilise les deux commandes suivante pour afficher la requête en hexadécimale et les données extraites :

- hako@ubuntu:~$ export MOTECOM=serial@/dev/ttyUSB0:telosb
- hako@ubuntu:~$ java net.tinyos.tools.Listen

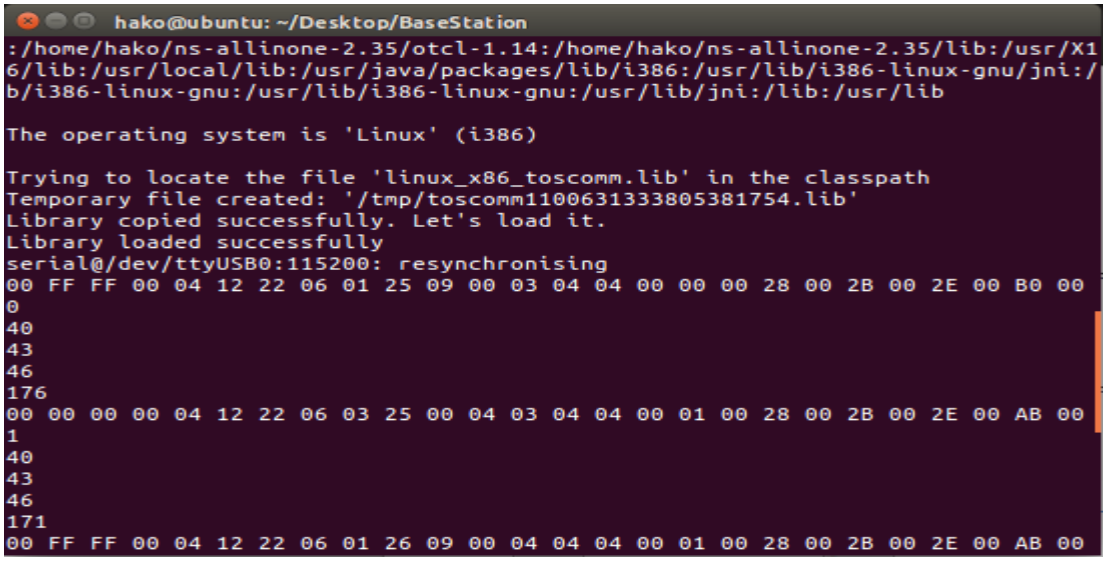


Figure IV-10 : affichage des requêtes en hexadécimale et les données extraites

a) l'envoi des mesures au serveur :

Pour alimenter la base de données, nous avons appelé un script php (add.php) à partir de fichier listen.java. Le script permet d'insérer les données dans la table mesure, après connexion au serveur par méthode PDO. Dans se script, on fixe l'heur de la mesure effectué

comme l'instant de sa réception (l'heur du système à partir de la fonction : $datetime = date("Y-m-d H:i:s")).

La sélection de type de l'alerte est effectuée au niveau du fichier listen.java en comparant la valeur de la température reçus à la température max et min.

IV.3.3. Conception de site :

Notre site est composé de 10 scripts PHP et un (01) fichier css, qui assure le design de notre site, en plus d'un dossier contenant les images utilisées dans notre site. Nous avons utilisé la classe PDO pour se connecter à la base de données 'température'.

IV.3.3.1. Page de démarrage du site (index.php)

Le fichier index est la page HTML que les utilisateurs verront en premier en accédant à notre site, les utilisateurs doit être enregistrés au préalable par l'administrateur.

L'ouverture de session est effectuée après vérification de l'enregistrement du nom d'utilisateur et la conformité du mot de passe entrée.

Figure IV-11 : Page de démarrage

IV.3.3.2. Page d'accueil (acceuil.php) :

Elle contient une introduction au site Web, un lien à l'université de Tlemcen, un lien à l'usine de ciment de Beni-Saf, et présente un menu horizontal et vertical de l'application.

Si l'utilisateur est enregistré comme administrateur, il accède à la page suivante, qui donne l'accès à la page de configuration.

Figure IV-12 : Page d'accueil pour l'administrateur

Si l'utilisateur n'est pas enregistré comme administrateur, il sera orienté vers une autre page d'accueil, qui ne donne pas l'accès à la page de configuration. Cette limitation est respectée dans toutes les autres pages de l'application.

Figure IV-13 : Page d'accueil pour un contrôleur

IV.3.3.3. Page de configuration (acceuil.php) :

Elle contient deux fenêtres, une pour la configuration des utilisateurs de l'usine, et l'autre pour la configuration des capteurs appartenant au réseau RCSF de contrôle. Seulement l'administrateur est autorisé à accéder à ces fenêtres.

a) Configuration des utilisateurs (configuration_Ut.php) :

Elle contient une table des utilisateurs enregistrés et leurs qualités (administrateur ou non). On peut ajouter un nouveau utilisateur après la saisi obligatoire de User et de password.

La suppression ou la modification d'un utilisateur est effectuée après la saisi obligatoire de User dans le champ « Rechercher par user».

Figure IV-14 : Page de configuration des utilisateurs

b) Configuration des capteurs (conf_capt.php) :

Elle contient une table des capteurs enregistrés avec un champ remarque (ex : information sur l'emplacement ou le rôle de ce capteur). L'ajouter d'un nouveau capteur nécessite la saisi obligatoire de N° de capteur, on peut choisir l'option de ne pas afficher les mesures obtenus par des capteurs utilisés comme des nœuds intermédiaires secondaires.

La suppression ou la modification d'un utilisateur est effectué après la saisi obligatoire de N° de capteur dans le champ « Rechercher par N° de capt».

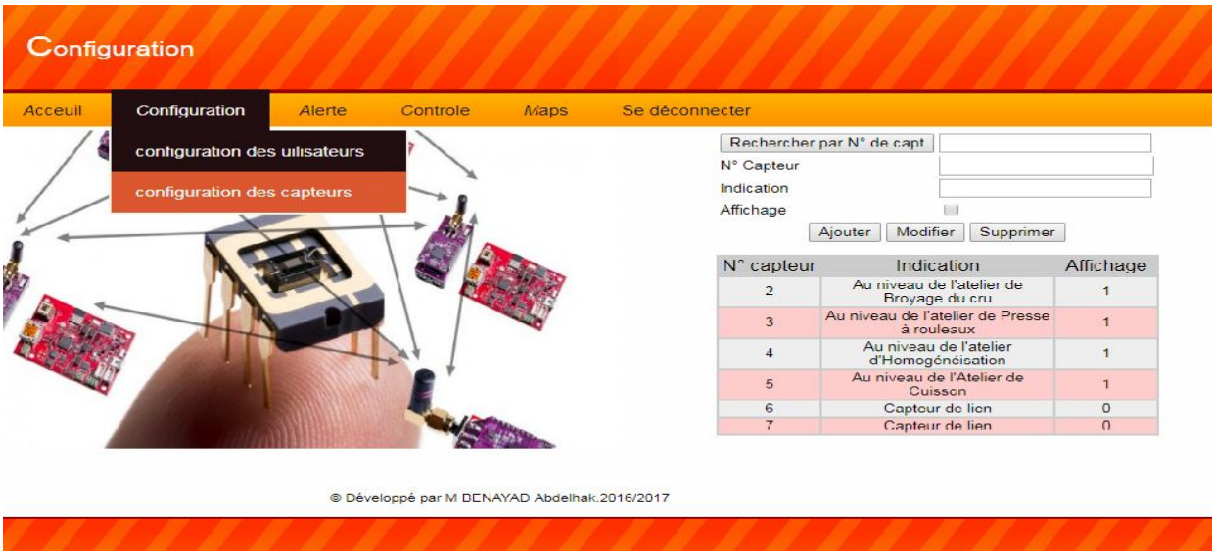


Figure IV-15 : page de configuration des capteurs

IV.3.3.4. Page des Alertes :

Elle contient deux fenêtres, destinées à l'affichage des alertes permettant aux utilisateurs un contrôle efficace et automatique.

a) Alertes de température (Alerte.php) :

Cette Fenêtre permet d'afficher d'une part, les mesures dépassants la température maximale autorisée (fixée à 35°), et d'autre part, les mesures dont la température est inferieur à la valeur minimale autorisée (fixée à 15°).

Les données relatives à l'heure de la mesure effectuée et la position du capteur effectuant cette mesure sont affichées afin de faciliter l'intervention des techniciens de maintenance sur le site.

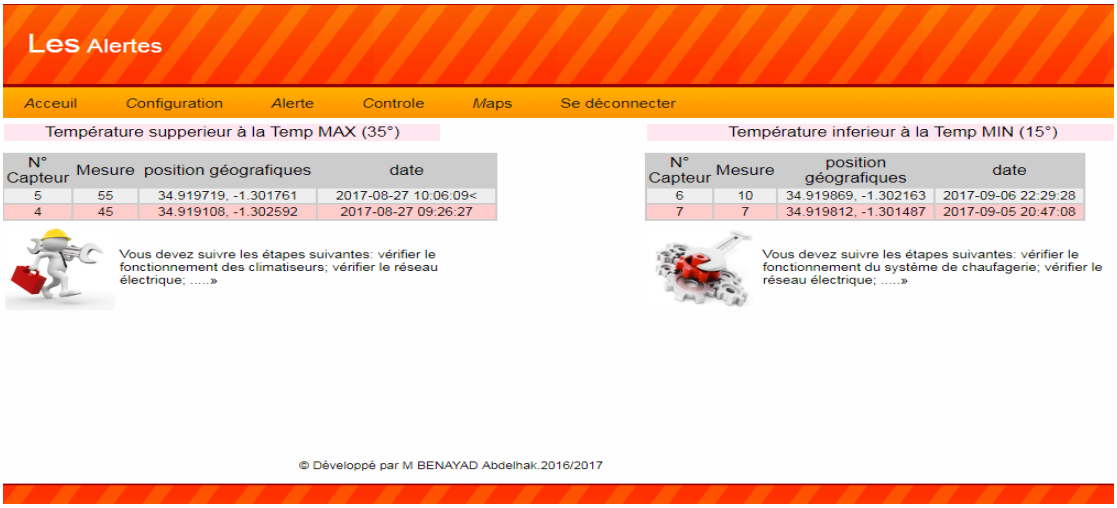


Figure IV-16 : Page des alertes de température

b) Alertes sur les pannes des capteurs (alerte_Capt.php) :

Cette Fenêtre permet d'afficher les capteurs en panne, si un capteur s'arrête d'envoyer ses mesures pendant un temps dépassant 10 mn, l'application le considère comme un capteur en panne ou sans alimentations. Elle affiche donc, sa position et l'heure de la dernière mesure envoyée.

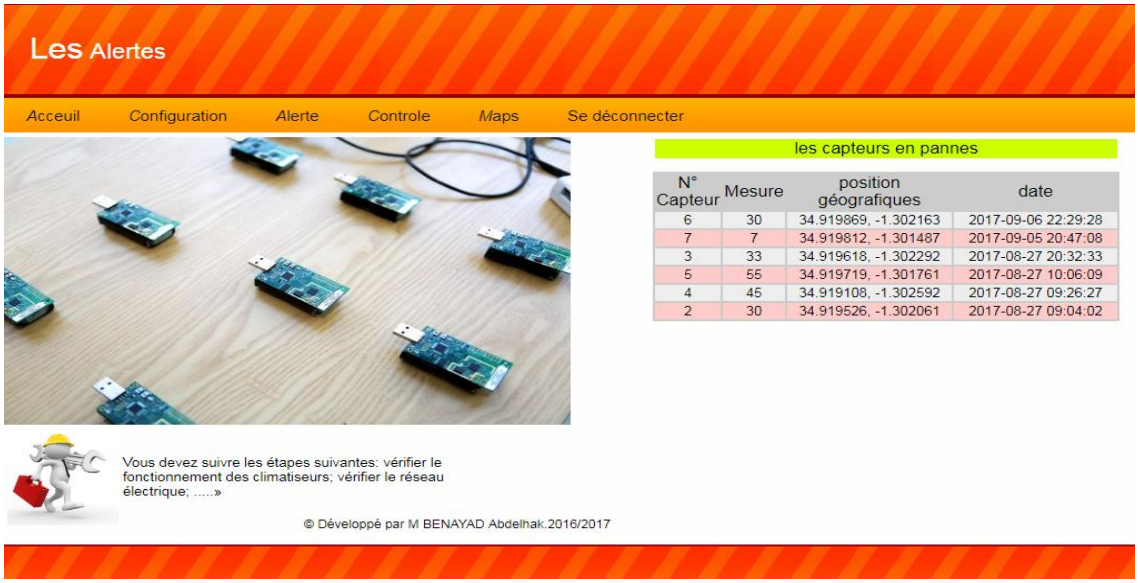


Figure IV-17 : Page des alertes sur les capteurs

IV.3.3.5. Page de Contrôle (controle.php):

Cette page permet l'affichage des dernières mesures de température effectuées par tous les capteurs autorisés pour l'affichage. Elle indique ainsi, le type d'alerte enregistré pour chaque mesure.

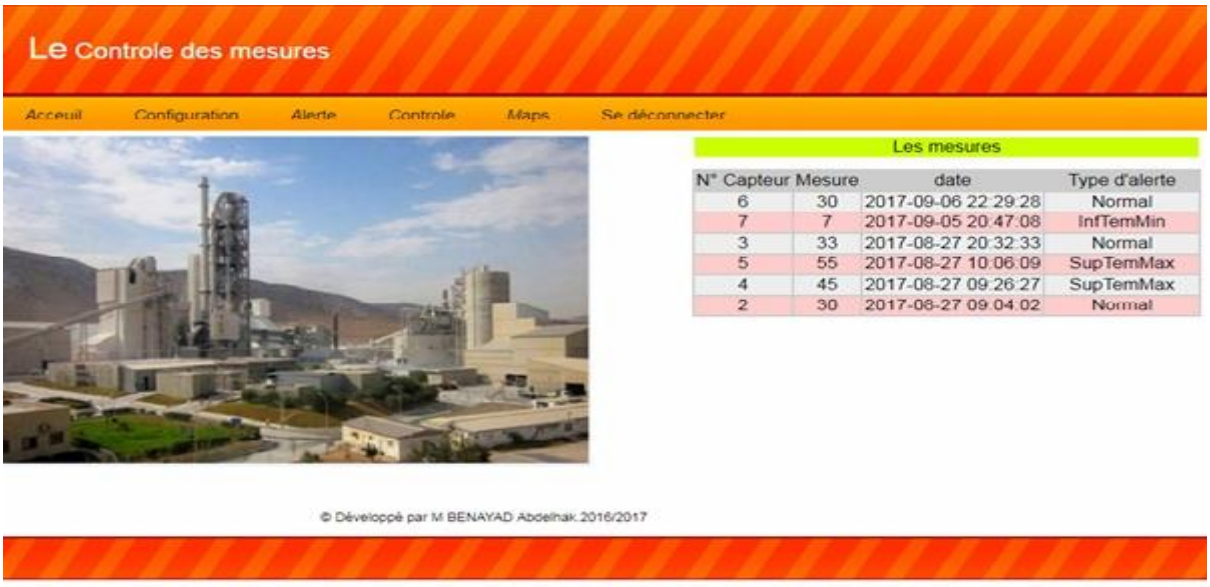


Figure IV-18 : Page de contrôle

IV.3.3.1. Page de l'affichage sur Map :

Cette page permet l'affichage des dernières mesures de température effectuées par tous les capteurs autorisés pour l'affichage. Elle indique ainsi, le type d'alerte enregistré pour chaque mesure.

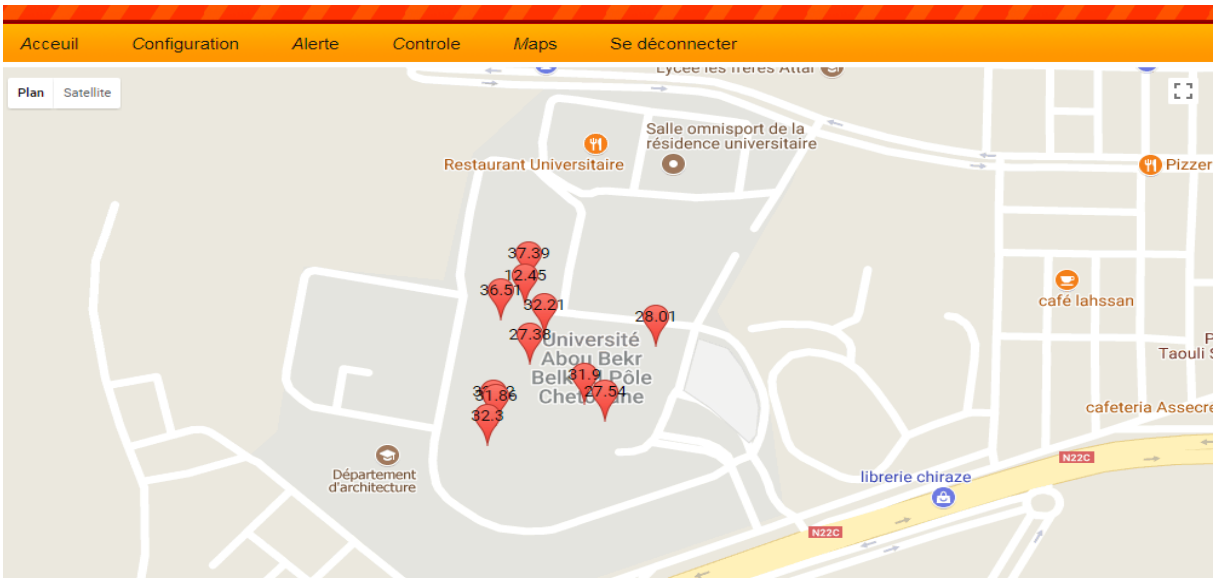


Figure IV-19 : page Map.

IV.4. Conclusion :

Dans ce chapitre, nous avons présenté les différents systèmes, outilles et moyens de développement et d'implémentation exploité dans notre projet, nous avons vu les principaux composants, interfaces et fonctions utilisés pour l'implémentation des protocoles de routage

AODV et AODV optimisé et des algorithmes de chiffrement symétrique RC4 et asymétrique ECDSA. Nous avons présenté les tests effectués et les résultats obtenus, qui montrent l'efficacité de l'algorithme AODV optimisé pour les RCSF et ses avantages pour la recherche de la route, la consommation de l'énergie et pour le routage des paquets.

L'application Web développée permet la gestion et le contrôle en temps réel, des données récoltées par un RCSF dans un environnement industriel, offrant des alertes, des états de sortie, et un positionnement sur carte géographiques des mesures effectuées.

CONCLUSION GENERALE ET PERSPECTIVES

Aussi bien dans les domaines publics, industriel, militaire que chez les particuliers, la volonté de collecter puis d'analyser des données sur l'environnement, sur des processus, sur des flux, n'a de cesse de croitre. Poussés par les progrès constants dans la miniaturisation des composants électroniques et par la conception de batteries toujours plus performantes, rendus accessibles par la standardisation des protocoles et par l'utilisation d'un matériel peu couteux dans leur fabrication, les réseaux de capteurs sans fil constituent la solution technique idéale pour répondre à ces besoins. Ils sont ainsi pleinement intégrés au processus de surveillance des sites sensibles, à la gestion du trafic urbain, ou encore à l'étude de l'environnement naturel. Les capteurs comptent aussi de plus en plus d'applications médicales, ou bien appartenant au domaine militaire.

De tels usages ne peuvent être mis en œuvre sans garantir un routage adéquat adapté aux contraintes physiques et d'application des RCSF, et une sécurité stricte de réseau, assurant la confidentialité, l'authentification, l'intégrité des communications.

Dans ce travail, nous avons essayé d'implémenter un protocole de routage adapté aux RCSF, nommé OAODV, et une solution de sécurité, basé sur l'utilisation de l'algorithme de chiffrement symétrique RC4 et la signature ECDSA.

Dans un premier temps, nous avons donné une étude sur les réseaux de capteur sans fil, portée sur les contraintes de conception et les domaines d'applications des RCSF, ainsi que leurs architectures de communication, Nous avons présenté également les contraintes du routage, les critères de performances des protocoles de routage pour les réseaux RCSF, et les principaux protocoles de routage proposé pour ces réseaux. Nous avons cités aussi, les exigences en sécurité et les vulnérabilités dans les RCSF ainsi que les solutions proposés.

Après présentation d'une étude sur le protocole AODV standard, nous avons détaillé le fonctionnement du protocole de routage proposé, nommé AODV optimisé, et les améliorations apportés. Nous avons adoptée une solution de sécurité des communications radio des RCSF, pour notre projet, basé sur l'implémentation de l'algorithme de chiffrement symétrique RC4 et asymétrique ECDSA.

Par la suite, nous avons présenté l'environnement de développement et d'implémentation exploité dans notre projet, en décrivant les composants du capteur CM5000 utilisé. Nous avons montré les principaux composants, interfaces et fonctions utilisés pour l'implémentation des protocoles de routage AODV et AODV optimisé et les algorithmes de chiffrement dans les capteurs Telosb.

En fin, nous avons décri les différents tests effectués relatifs à la mesure des performances des protocoles de routage AODV et AODV optimisé et les algorithmes de chiffrement symétrique RC4 et asymétrique ECC implémentés, en analysant les résultats obtenus. Nous avons exposé ainsi, notre application web développé pour le contrôle de la température dans un environnement industrielle.

Les résultats fournis pour l'implémentation du protocole de routage AODV optimisé, montrent l'efficacité de ce protocole pour ce gendre de réseau, en enregistrant un temps minimale pour la recherche de la route, une consommation réduite de l'énergie et un pourcentage faible pour la perte des paquets routés, en comparant avec les résultats obtenus avec le protocole AODV. Ce qui permet d'augmenter la durée de vie de réseau et assurer une exploitation raisonnable des ressources des capteurs, et garantir un routage avec un faible taux de perte.

Le chiffrement symétrique proposé utilisant l'algorithme RC4 a été validée à travers les tests concluants effectués, assurant la confidentialité, et l'intégrité des communications, enregistrant une faible consommation d'énergie. La solution de sécurité proposée initialement n'a pas été vérifiée, à causse du non réussite à l'implémentation de la signature ECDSA.

L'application Web développée est un exemple de gestion et de contrôle en temps réel, des données récoltées par un RCSF dans un environnement industriel, offrant des alertes, des états de sortie, et un positionnement sur carte géographiques des mesures effectuées.

Enfin, comme perspectives nous envisageons d'améliorer les performances de notre protocole de routage en réduisant la taille des trames, principalement celles des requêtes RREQ. Afin d'améliorer la solution de sécurité proposée, nous proposons d'implémenter la signature ECDSA et l'algorithme ECDH pour l'échange des clés entre les capteurs, sur des réseaux de capteurs sans fils à grand échelle.

BIBLIOGRAPHIE

[1] Mr Ismail MANSOUR, 'Contribution à la sécuritée des communications des réseaux de capteurs sans fil'. Thèse de Doctorat Spécialité Informatique, UNIVERSITÉ BLAISE PASCAL – CLERMONT II , Octobre 2010.

[2] Yaser Yousef, Routage pour la gestion de l'énergie dans les réseaux de capteurs sans fil, Thèse de Doctorat Spécialité informatique, université de haute alsace, 2010.

[3] Mr SAHRAOUI belkheyr, Etude d'un protocole de routage basé sur les colonies de Fourmis dans les réseaux de capteurs sans fil, Mémoire de fin d'études Pour l'obtention du diplôme de Master en Informatique ,Université Abou Bakr Belkaid– Tlemcen,2012-2013.

[4] BENAMAR KADRI, MOHAMMED FEHAM, ABDELLAH MHAMMED Secured and Optimized AODV for Wireless Sensor Networks, International Journal of Information Technology and Computer Science(IJITCS), Mai 2013.

[5] MOHAMMED BELBACHIR , Stratégie de tolérance aux pannes pour un routage efficace dans les réseaux de capteurs ,Mémoire de fin d'études Pour l'obtention du diplôme de Master en Informatique ,Université Abou Bakr Belkaid– Tlemcen,2014.

[6] BENINE Safa, 'le routage multi-chemin dans les réseaux de capteurs sans fil'. Mémoire de fin d'études Pour l'obtention du diplôme de Master en Informatique , Université Echahide Hamma Lakhdar El-oued,2014-2015.

[7] HADJ ADDA Asmaa, BENALLAL Wafaa, 'Mise en place d'un schéma de routage pour la tolérance aux pannes dans les RCSF'. Mémoire de fin d'études Pour l'obtention du diplôme de Master en Informatique ,Université Abou Bakr Belkaid– Tlemcen,2014-2015.

[8] RAMDANI MOHAMED, ' Problèmes De Sécurité Dans Les Réseaux De Capteurs Avec Prise En Charge De L'énergie'. Mémoire de fin d'études Pour l'obtention du diplôme de Magister en Informatique, Université de SAAD DAHLAB DE BLIDA ,2012-2013.

[9] Mr SAHRAOUI belkheyr, Etude d'un protocole de routage basé sur les colonies de Fourmis dans les réseaux de capteurs sans fil, Mémoire de fin d'études Pour l'obtention du diplôme de Master en Informatique ,Université Abou Bakr Belkaid– Tlemcen,2012-2013

[10] E.M Royer and C-K Toh. "A review of current routing protocols for ad-hoc mobile wireless networks" . IEEE Personal Communications, Apr. 1999.

[11] C. E. Perkins, E. E. Royer, S. R. Das, and M. K. Marina, "Performance comparison of two on-demand routing protocols for ad hoc networks,"In IEEE Personal Communications, Feb 2001, vol. 8, pp. 16–28.

[12] Z.Haas et M. Pearlman, "The performance of query control schemes for the Zone Routing Protocol",IEEE selected area in communication , Août 1998.

[13] Khaled BOUCHAKOUR, Routage hiérarchique sur les réseaux de capteurs sans fil: Protocole khlch (K-hop Layered Clustering Hierarchy), Memoirre Présenté pour l'obtention d'un diplôme de magister en informatique, Ecole Nationale Supérieure de l'Informatique (ESI),2012.

[14] YOUCEF ZIANI, Etude comparative de méthodes de routage dans les Réseaux de capteurs sans fil pour le domaine Résidentiel, mémoire présenté à l'université duQuébec à trois-riviéres, juin 2013.

[15] RAMDANI MOHAMED, Problèmes de sécurité dans les Réseaux de capteurs avec prise en charge de l'énergie, mémoire de magister, université de saad dahlab de Blida, Novembre 2013.

[16] Wassim Znaidi, « Modélisation formelle de réseaux de capteurs à partir de TinyOS», Projet de fin d'études, Ecole Polytechnique de Tunisie, 2006.

[17] BRAHIM Nacera, Routage multichemin securisé pour un réseau de capteur sans fil video, attaque wormhole: etude et contre mesure, mémoire de magister, université d'Oran, Mai 2012.

[18] YACINE Younes, Minimisation d'énergie dans un réseau de capteurs, mémoire de magister, université Mouloud Mammeri de Tizi-Ouazou, Septembre 2012.

[19] N. Mejri, F. Kamoum, "Algorithme de Routage Hiérarchique MHEED à Plusieurs Sauts pour Les Grands Réseaux de Capteurs ". In SETIT 2007, 4th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications. TUNISIA. Mars 2007.

[20] M. Achir and L. Ouvry. "A routing protocol for wireless ad-hoc sensor networks : Multi-Path Source Routing Protocol (MPSR)", ICN'05 : 4th International Conference on Networking (IEEE), Ile de la Réunion. Avril 2005.

[21] C. B. Abbas, R. González, N. Cardenas, L. J. G. Villalba, "A proposal of a wireless sensor network routing protocol", Springer Science and Business Media Telecommunication Systems. pp. 61–68. March 2008.

[22] D. Braginsky, D. Estrin, "Rumor Routing Algorithm for Sensor Networks", 1st Workshop. Sensor Networks and Apps., Atlanta, GA. 2002.

[23] Mr Samir ATHMANI, 'Protocole de sécurité Pour les Réseaux de capteurs Sans Fil'. Mémoire de fin d'études Pour l'obtention du diplôme de Magister en Informatique, Université Hadj Lakhder Batna , Juillet 2012.

[24] Mr ZNAIDI Wassim, 'Quelques propositions de solutions pour la sécurité des réseaux de capteurs sans fil'. Thèse de Doctorat Spécialité Informatique et Mathématiques, L'Institut National des Sciences Appliquées de Lyon , Octobre 2010.

[26] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures", In Proc. 1st IEEE Int‟l. Wksp. Sensor Network Protocols and Applications (SNPA‟03), pp. 293-315, 2003.

[27] Mr Edgard Haddad, 'Détection de la retransmission sélective sur les réseaux de capteurs'. Mémoire présenté en vue de l'obtention du grade de Maître es sciences en informatique, Université de Montréal , Avril 2011.

[28] Mr Thomas Plantard, 'Arithmétique modulaire pour la cryptographie '. Thèse de Doctorat Spécialité Informatique, Université de Montpellier II.

[29] Mr Ismail MANSOUR, 'Contribution à la sécurité des communications des réseaux de capteurs sans fil '. Thèse de Doctorat Spécialité Informatique, Université blaise Pascal Clermont II.

[30] Mr Aqeel, Mr Kuldip, Mr Sandeep, 'Implementation of Elliptic Curve Digital Signature Algorithm '. International Journal of Computer Applications (0975 – 8887),Volume 2 – No.2, May 2010.

[31] TinyECC version 2.0 (02/03/2011), « A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks ». [En ligne]. Disponible sur: http://discovery.csc.ncsu.edu/software/TinyECC/. [Consulté le: 07-avr-2013].

RESUME

Un réseau de capteurs sans fil (RCSF) est un réseau ad hoc, composé d'un ensemble de nœuds généralement dédiés à la collecte d'information, capables de communiquer entre eux afin de réaliser des tâches diverses. Dans ce travail, nous avons essayé d'implémenter un protocole de routage adapté aux RCSF, nommé OAODV, et une solution de sécurité, basé sur l'utilisation de l'algorithme de chiffrement symétrique RC4 et la signature ECDSA.

Les résultats obtenus suite aux testes pratiques effectués, montrent l'efficacité du protocole de routage AODV optimisé pour ce gendre de réseau, en enregistrant un temps minimale pour la recherche de la route et un pourcentage faible pour la perte des paquets routés avec une consommation réduite de l'énergie, en comparant avec les résultats obtenus avec le protocole AODV. Le chiffrement symétrique proposé utilisant l'algorithme RC4 a été validée à travers les tests concluants effectués, assurant la confidentialité, et l'intégrité des communications, enregistrant une faible consommation d'énergie.

Mots-clés : Réseaux de capteurs sans fil, RCSF, protocole AODV,OAODV.

ABSTRACT

A Wireless Sensors Network (WSN) is an ad hoc network, composed of a set of nodes generally dedicated to the collection of information, able to communicate with each other in order to carry out various tasks. In this work, we tried to implement an WSN adapted routing protocol, named OAODV, and a security solution, based on the use of the RC4 symmetric encryption algorithm and the ECDSA signature.

The results obtained from the practical tests carried out show the efficiency of the AODV routing protocol optimized for this network by recording a minimum time for the search of the road and a low percentage for the loss of the routed packets with a reduced consumption of the energy, comparing with the results obtained with the AODV protocol. The proposed symmetric encryption using the RC4 algorithm was validated through the successful tests performed, ensuring confidentiality, and integrity of communications, recording low power consumption.

Keywords: Wireless Sensor Networks, WSN, AODV protocol, OAODV.