Routage et sécurité à basse consommation d'énergie dans les réseaux de capteurs sans fil

Mohamed Kasraoui

Normandie Université

THESE

Pour obtenir le grade de Docteur

opéré par l'Université de Rouen

Spécialité (Informatique)

Routage et sécurité à basse consommation d'énergie dans les réseaux de capteurs sans fil

Présentée et soutenue publiquement par
Mohamed KASRAOUI

Thèse soutenue publiquement le (03/12/2015) devant le jury composé de :		
M. Traian MUNTEAN	Professeur à Aix Marseille Université	Rapporteur
M. André-Luc BEYLOT	Professeur à l'ENSEEIHT de Toulouse	Rapporteur
Mme Houda LABIOD	Enseignant -Chercheur, (HDR) à Télécom ParisTech	Examinatrice
M. Mhamed ITMI	Maître de Conférences, (HDR) à l'INSA de Rouen	Examinateur
M. Houcine CHAFOUK	Professeur, IRSEEM / ESIGELEC, Rouen	Directeur de thèse
M. Adnane CABANI	Enseignant-chercheur, IRSEEM / ESIGELEC, Rouen	Encadrant

Thèse dirigée par Pr. Houcine CHAFOUK et Dr. Adnane CABANI, **IRSEEM (Institut de Recherche en Systèmes Electroniques EMbarqués)**

UNIVERSITÉ DE ROUEN

SPMII ECOLE DOCTORALE

irseem

Remerciements

C'est avec un grand plaisir que je réserve ces quelques lignes en signe de gratitude et de profonde reconnaissance à tous ceux qui, de près ou de loin, ont contribué à la réussite de mes travaux de thèse qui ont été menés à l'Institut de Recherche en Systèmes Electroniques Embarqués (IRSEEM) au sein du pôle IIS (Informatique-Instrumentation et Systèmes).

Je tiens à remercier M. Houcine CHAFOUK, mon directeur de thèse et Enseignant-chercheur à l'ESIGELEC, pour ses encouragements et sa confiance durant ces trois années de thèse.

Je remercie également M. Adnane CABANI, mon co-encadrant et Enseignant-chercheur à l'ESIGELEC, pour son encadrement, son soutien et ses conseils utiles. De même, je remercie M. Joseph MOUZNA, Enseignant-chercheur à l'ESIGELEC, pour sa disponibilité et son soutien importants à la concrétisation de cette recherche.

J'exprime ma reconnaissance envers Monsieur Pr. MUNTEAN Traian, Professeur des Universités, I2M, Aix-Marseille Université, et Pr. André-Luc BEYLOT, Professeur à l'ENSEEIHT de Toulouse, pour leur entière coopération et l'intérêt qu'ils ont porté à mes travaux de recherche.

Mes remerciements s'adressent également à Mme Houda LABIOD, Enseignant-chercheur, (HDR) à Télécom ParisTech, et M. Mhamed ITMI, Maître de Conférences, (HDR) à l'INSA de Rouen. Leur lecture approfondie, leurs remarques et interrogations judicieuses m'ont été très précieuses.

Enfin, j'exprime ma profonde gratitude et mon grand amour aux membres de ma famille pour leur soutien et sans lesquels je n'en serai pas là aujourd'hui. Je remercie tous les doctorants, chercheurs (Dr Wassim TROJET, Dr Halim BENHABILES, Dr Nabil AJAM, Dr Karim HAMMOUDI...) et personnels de l'IRSEEM/ESIGELEC pour leur grande gentillesse.

Lieu, le 26 octobre 2015.

Table des matières

Liste des figures

Introduction générale

.

Avec la croissance économique actuelle, la quantité des marchandises transportée par voie maritime ne cesse d'accroître. Cela génère plusieurs complications logistiques surtout dans la région Haute-Normandie qui dispose de l'un des plus grands complexe portuaire d'Europe. Cela a donné naissance au projet « passage portuaire du conteneur » financé par la région Haute-Normandie et le Fond Européen de Développement Régional. Ce projet était le centre d'intérêt de plusieurs établissements de recherche tels que : l'EDEHN, l'IRSEEM, LITIS, LMAH, LMI, etc. Ses principaux objectifs portent sur le suivi de conteneurs, l'interconnexion des systèmes d'information, la productivité, la sécurité et la sûreté, et l'intermodalité portuaire-routière, portuaire-ferroviaire et portuaire-fluviale. Le projet a été organisé autour de quatre axes de recherche. Le premier axe porte sur les nouveaux modèles d'organisation et de transport de marchandises, le deuxième porte sur la gestion des terminaux, un troisième traite le développement portuaire Haut-Normand en lien avec le développement territorial dans la Basse-Seine et le dernier axe porte sur la gestion des risques dans des systèmes distribués. Au niveau de ce dernier axe, notre équipe de recherche à l'IRSEEM (Institut de Recherche en Système Electronique EMbarqué) a montré sa motivation en proposant un sujet de thèse, intitulé « Amélioration de la Sécurité et du routage dans les réseaux de capteurs sans fil ». Notre contribution dans ce projet consiste à développer un système de gestion de données et de traçabilité des conteneurs permettant le tracking et le tracing des unités de chargement des marchandises. Pour ce faire, notre équipe a proposé une architecture d'un réseau de communication basée sur les réseaux de capteurs sans fil. Ces réseaux sont particulièrement utiles

dans des endroits disposant de peu d'infrastructures de communication et dont le déploiement est difficile ou ayant des contraintes spatiales et matérielles considérables. Malgré le gain important en flexibilité qu'ils offrent, les réseaux de capteurs sans fil (RCSFs) présentent trois problèmes incontournables. Le premier est la limitation des ressources en termes d'énergie, de mémoire et de temps de calcul. Le deuxième concerne le routage des informations collectées sur le réseau. En effet, le nombre de capteurs déployés, pouvant atteindre des milliers de nœuds, présente une contrainte forte dans la gestion des routes et le passage à l'échelle dans le réseau. Le troisième est la nature des RCSFs elle-même qui fait du réseau un milieu favorable et vulnérable à des attaques à savoir la falsification, la modification, le déni de service, etc.

Notre thèse a été proposée pour remédier à ces problèmes. En effet, Dans nos travaux de recherches, nous avons abordé deux aspects : le routage et la sécurité de bout en bout. Concernant le premier aspect, nous avons proposé une amélioration d'un protocole de routage hiérarchique ZBR-M afin d'optimiser l'acheminement des données d'une source vers une destination utilisant la technologie ZigBee. Notre choix du ZBR-M est motivé par les résultats de simulations concluants de ce protocole relativement aux autres protocoles de routage ZigBee existants. En effet, le routage hiérarchique de base présente de meilleurs délais et taux de délivrance permettant ainsi une disponibilité de service indépendamment du nombre de nœuds dans le réseau.

Le manuscrit est structuré de la manière suivante :

Dans le chapitre 1, nous commençons, dans une première section, par une présentation générale des réseaux de capteurs sans fil à savoir leurs architectures, leurs caractéristiques, leurs protocoles de communication ainsi que leur fonctionnement général. Une deuxième section est consacrée à la présentation des différents protocoles de routage existants destinés aux RCSFs. Enfin, une troisième section est consacrée à l'état de l'art des différents systèmes cryptographiques utilisés et des solutions de sécurité existantes dans la littérature.

Le chapitre 2 décrit nos travaux sur le routage dans les RCSFs. D'abord, nous commençons par une présentation vulgarisée du standard ZigBee. Par la suite, nous étudions, dans une deuxième section, les différents protocoles de routage proposés par ce standard ainsi qu'une analyse de performance et une étude comparative par simulation. Enfin une dernière section du chapitre porte sur l'étude et l'analyse des performances d'un nouveau protocole de routage hiérarchique, appelé ZBR-M.

Le chapitre 3 est dédié aux problèmes de la sécurité dans les RCSFs où nous avons ciblé plus la sécurité de bout en bout en tenant compte de la consommation énergétique. Nous commençons ce chapitre par une étude comparative afin de justifier notre choix de l'IPSec comme meilleur candidat pour la sécurité de bout en bout dans les RCSFs. Puis, nous présentons les résultats de simulations de l'IKEv2, une primitive de l'IPSec. Suite à la proposition d'une nouvelle approche appelé CKES, nous réalisons, dans la dernière section du chapitre, une comparaison avec l'IKEv2 afin d'illustrer les performances du CKES.

Dans le dernier chapitre, nous rappelons d'abord les méthodes formelles utilisées pour valider des protocoles de sécurité. Nous présentons ensuite les résultats de la validation formelle du protocole CKES en faisant appel à l'outil AVISPA.

Etat de l'art

1

Etat de l'art

Sommaire

Un capteur est un dispositif transformant l'état d'une grandeur physique observée en une grandeur utilisable [Devalan]. Les capteurs sont des éléments de base des systèmes d'acquisition des données, i.e. les procédés permettant de récolter des informations afin d'analyser un phénomène. Ces informations sont acheminées sur un support de transmission, essentiellement filaire, pour arriver au poste de traitement. La grande diversité de données recueillies nécessite, dans certains cas, l'utilisation de nombreux capteurs dont chacun

collecte une donnée particulière. Le nombre de supports de transmission filaires est devenu rapidement conséquent, d'où il fallait penser à les dématérialiser en faisant appel à la technologie sans fil. Ainsi, les avancées remarquables dans le domaine des télécommunications ont permis de supprimer les liaisons filaires de transmission fortement encombrantes en les substituant par des supports de communication sans fil. Ces réseaux sont particulièrement utiles dans des endroits disposant de peu d'infrastructures de communication et dont le déploiement est difficile ou ayant des contraintes spatiales et matérielles considérables. Malgré le gain important en flexibilité qu'il offre, les réseaux de capteurs sans fil (RCSFs) présentent trois problèmes incontournables. Le premier est la limitation des ressources en termes d'énergie, de mémoire et de temps de calcul. Le deuxième concerne le routage des informations collectées sur le réseau. En effet, le nombre de capteurs déployés, pouvant atteindre des milliers de nœuds, présente une contrainte forte dans la gestion des routes et le passage à l'échelle dans le réseau. Le troisième est la nature des RCSFs elle-même qui fait du réseau un milieu favorable et vulnérable à des attaques à savoir la falsification, la modification, le déni de service, etc. Par conséquent, la sécurisation des données transmises dans un réseau ouvert et la protection contre différents types d'attaques présentent un véritable défi pour les RCSFs. Une panoplie de protocoles a été créée afin de remédier aux problèmes susmentionnés. Afin de capitaliser les travaux de recherche existants sur la thématique de notre thèse, nous allons diviser ce premier chapitre en trois sections. Une première section introduira les réseaux de capteurs sans fil en présentant leurs différentes architectures, leurs caractéristiques, leurs protocoles de communication ainsi que leur fonctionnement général. La deuxième section détaillera les différents protocoles de routage destinés aux RCSFs classés selon le

critère « paradigme de communication ». La troisième section expliquera l'enjeu de la sécurité, les différents systèmes cryptographiques utilisés et les solutions de sécurité existantes dans la littérature.

1.1 Les réseaux de capteurs sans fil

Les réseaux de capteurs sans fil (RCSFs) sont des systèmes distribués à grande échelle qui permettent de mettre en communication un grand nombre de nœuds (capteurs). Chaque nœud capteur est considéré comme une unité électronique capable de mesurer une grandeur physique, effectuer un traitement et communiquer sans fil avec un autre nœud capteur [Devalan]. Dans la figure 1.1, on présente une architecture basique d'un nœud capteur qui est composée des éléments suivants [Angel et al.] :

- Un ou plusieurs capteurs de grandeurs physiques (température, humidité, luminosité, GPS, etc) forme l'unité de captage. Cette dernière est connectée à un microcontrôleur via un convertisseur analogique-numérique (CAN).

- Un microcontrôleur, sur lequel, les capteurs et l'antenne sont reliés. Il est doté également d'une partie logicielle exécutant les différentes applications liées aux capteurs et d'une pile protocolaire qui exécute les protocoles réseaux.

- Une antenne permettant aux nœuds de communiquer.

- Une batterie (unité d'énergie) qui est souvent la seule source d'énergie.

L'ensemble de ces éléments forme une architecture complète d'un nœud capteur capable de mesurer, traiter et communiquer avec le reste des nœuds dans le réseau.

A l'aide d'une station de base ou d'un coordinateur, ces capteurs forment alors les nœuds du réseau. Afin de donner une vision globale sur les RCSFs, nous présentons, dans la figure 1.2, l'architecture conventionnelle d'un réseau de capteurs sans fil [Akyildiz et al.]. Cette architecture est présentée par un grand nombre de capteurs capables

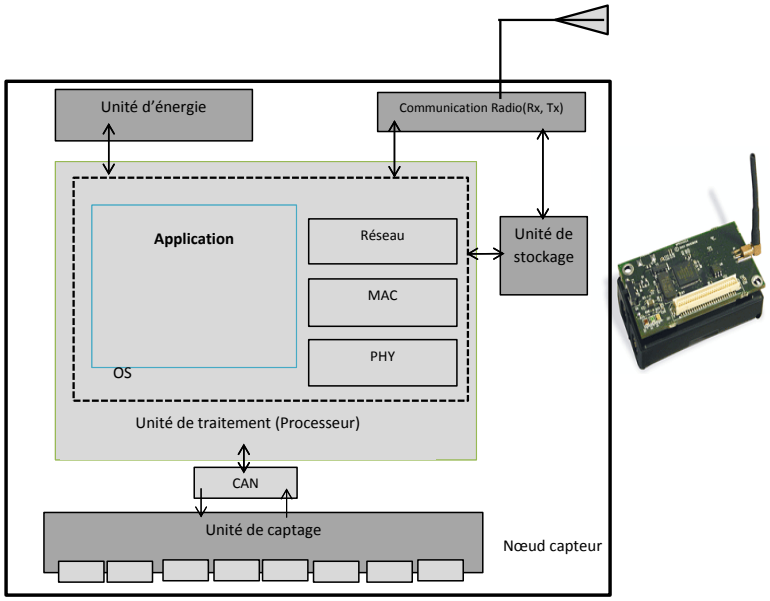

FIGURE 1.1 – *Architecture d'un nœud capteur*

de couvrir une zone géographique et de transmettre les informations collectées, via des nœuds intermédiaires, à une station de base (BS). Les nœuds des RCSFs peuvent communiquer d'une manière autonome et s'auto-organiser sans avoir besoin d'une infrastructure. Ils peuvent également se connecter au réseau Internet via des passerelles ou souvent via la BS.

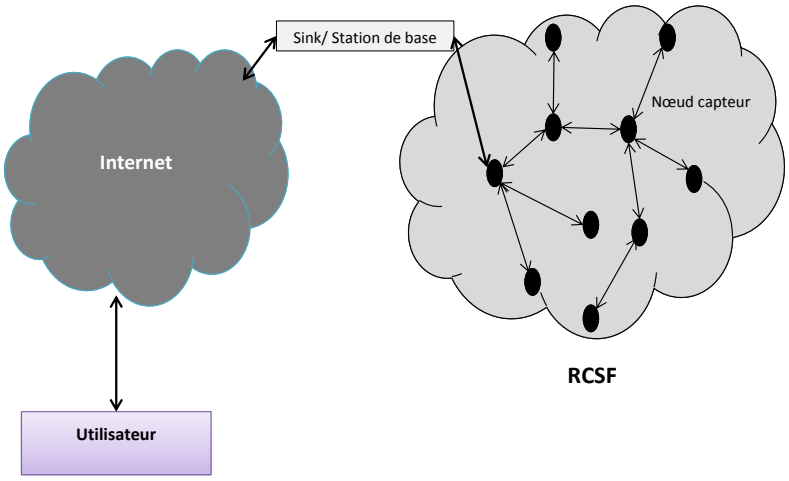

FIGURE 1.2 – *Architecture d'un réseau de capteurs sans fil*

L'architecture d'un réseau de capteurs sans fil peut être présentée sur deux échelles [Al-karaki et Kamal]. Une première architecture dite plate où tous les nœuds ont les mêmes ressources (batterie, mémoire, processeur) et participent à la constitution des routes menant vers la station de base. Une deuxième architecture dite hiérarchique où un ensemble de nœuds capteurs plus puissants sont introduits dans le réseau. Leur rôle est d'acheminer les données dans le réseau et de réduire la charge sur le reste des nœuds.

1.1.1 Caractéristiques des réseaux de capteurs sans fil

Les nœuds capteurs se caractérisent généralement par des faibles capacités de calcul, de mémoire et d'énergie. Ceci rend leur puissance de transmission faible et leur débit très modeste. Tous les protocoles de communication (routage, sécurité, mobilité) doivent prendre en considération les caractéristiques des réseaux de capteurs sans fil citées ci-dessous :

- Faible consommation : Les RCSFs sont caractérisés par leur faible consommation énergétique qui les rend peu énergivore par rapport aux autres types de réseaux sans fil. Ceci permet d'avoir une durée de vie très longue, dépendamment des exigences des applications (temps réel, périodicité, routage dynamique, sécurité...), qui peut atteindre parfois quelques années.

- Echelle du réseau : Les RCSFs sont composés de centaines, voire de milliers de nœud capteurs. Il est, donc, nécessaire de tester les performances et le taux de surcharge des protocoles de communication dans les réseaux de capteurs à grande échelle.

- Fiabilité : Les RCSFs disposent d'un système d'auto-organisation du réseau qui se lance lors d'un mauvais fonctionnement ou d'une panne

d'un ensemble de nœuds. Ainsi, le réseau garde son fonctionnement habituel sans aucun impact sur le système global.

- Faible coût : La simplicité, la petite taille et les ressources limitées des RCSFs, réduisent le coût des nœuds par rapport aux autres technologies sans fil.

- Connectivité : La connectivité dans les RCSFs dépend essentiellement de l'existence des routes entre un émetteur et un récepteur. Elle est généralement affectée par le changement de la topologie causé par la mobilité, la défaillance des nœuds, les attaques, etc. Pour remédier à ce problème, les nœuds coopèrent ensemble et essayent d'assurer la connectivité en permanence. Ces caractéristiques ont suscité l'intérêt de plusieurs laboratoires de recherche et d'entreprises (R&D). Ces derniers ont déployé des RCSFs pour fournir des services et des applications temps réels dans divers domaines.

1.1.2 Domaines d'application

Les RCSFs ont été utilisés dans différents domaines. Le domaine militaire était parmi les premiers intéressés par la mise en œuvre des RCSFs. Grâce au déploiement rapide, au coût réduit, à l'auto-organisation et à la tolérance aux pannes, les RCSFs offrent aux militaires des informations précieuses sur l'évolution des combats dans le champ de bataille telles que les mouvements des ennemis, les caractéristiques des terrains difficiles d'accès, etc. Par ailleurs, les RCSFs ont été déployés dans des endroits inaccessibles ou difficiles d'accès par les humains. Les RCSFs ont fait preuve de leur efficacité même dans des conditions environnementales défavorables et très difficiles telles que les volcans, les grandes forêts, les océans, les régions polaires, etc. Plusieurs projets européens ont été lancés, sur la thématique des

RCSFs dans le but d'assurer la surveillance environnementale. A titre d'exemple, le projet TowerPower [TowerPower (2014)] vise à développer un système de contrôle à distance du vieillissement des structures des éoliennes. Egalement, le projet européen ENORASIS [ENORASIS (2014)] développe des outils de prédiction et d'optimisation dans le secteur agricole... Les RCSFs ont montré une forte présence dans l'industrie [Elahi et Gschwender (2009)]. En effet, cette dernière s'est intéressée au potentiel des capteurs afin de diminuer le coût de contrôle et de maintenance grâce à la supervision en temps réel des usines et la traçabilité des chaînes d'approvisionnement. Le domaine médical, de son côté, a inclus l'usage des RCSFs pour une multitude d'applications [Eisenman et al. (2007)][Wood et al. (2006)]. Parmi ces applications, on cite la télésurveillance des signes vitaux des patients âgés[Boudra (2014)].

1.1.3 Protocoles de communication

La pile protocolaire utilisée par les nœud capteurs est présentée dans la figure 1.3. Cette pile offre un ensemble de services permettant la communication et la structuration des différents nœuds dans le réseau. Elle comprend la couche physique, la couche de liaison de données, la couche réseau, la couche transport, la couche application, le plan de gestion de l'énergie, le plan de gestion de la mobilité et le plan de la gestion des tâches. La couche physique gère la modulation, la transmission et la réception des données sur le médium radio. Elle dessert également des fonctionnalités à la couche supérieure à savoir l'alimentation du module radio, la mesure des puissances des signaux reçus et la régulation des puissances d'émission. La couche de liaison de données [Demirkol et al. (2006)] assure le contrôle et la gestion d'accès au support physique. Elle gère également les communications entre les

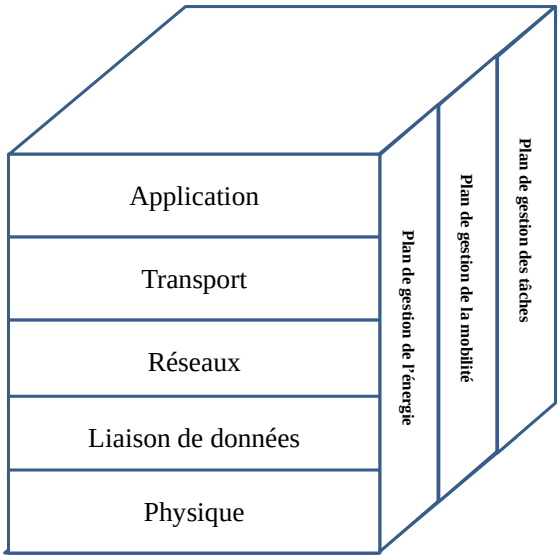

FIGURE 1.3 – *Pile protocolaire*

nœuds voisins en se basant sur un système d'adressage physique. Tout cela est fait grâce à un ensemble de protocoles dédiés aux RCSFs à savoir WiseMAC [Enz et al. (2004)], S-MAC [Ye et al. (2004)], DE-MAC [Kalidindi et al. (2003)]. La couche réseau s'occupe de l'acheminement des données entre tous les nœuds du réseau grâce à des protocoles de routage tels que SPIN [Heinzelman et al. (b)], TEEN [Manjeshwar et Agrawal (b)], et LEACH [Heinzelman et al. (a)], etc. La couche transport permet de gérer le flux de données, des éventuelles erreurs de transmission et la segmentation des données délivrées par la couche application. Trois plans s'ajoutent à la pile protocolaire : le plan de gestion d'énergie qui gère la manière avec laquelle un nœud capteur maintient sa puissance, le plan de gestion de la mobilité qui suit en temps réel le mouvement de nœuds capteurs afin de maintenir la topologie du réseau et le plan de gestion des tâches qui surveille tous les événements réalisés. Ces plans de gestion sont indispensables afin que les nœuds de capteurs puissent coopérer d'une manière efficace, acheminer les données dans un réseau de capteurs mobiles et partager les ressources entre les nœuds.

1.1.4 Consommation d'énergie dans les RCSFs

Selon Raghunathan et al [Raghunathan et al. (a)], la consommation énergétique dépend de trois facteurs : le traitement des données, la communication et la détection. Ainsi, le modèle de consommation d'énergie peut être défini comme suit :

- Energie de traitement de données : Cette énergie dépend essentiellement des opérations exécutées lors de l'activation de l'unité de traitement de données (Microcontrôleur (MCU), processeur...). Afin de minimiser la consommation énergétique au niveau des MCUs, trois modes d'opérations ont été définis : Active, Idle (Veille) et Sleep (Sommeil). Le premier désigne le mode de fonctionnement normal. Le deuxième mode met le CPU en arrêt tout en gardant en marche les périphériques nécessaires. Le dernier mode c'est le plus économe parmi les trois modes où le CPU et tous ces périphériques se mettent en arrêt. Le tableau 1.1 donne un aperçu sur la consommation d'énergie approximative des trois modes de fonctionnement sur différentes plateformes.

TABLE 1.1 – *Consommation énergétique des trois modes*

MCU	**Mode Active**	**Mode Idle**	**Mode Sommeil**
MSP430[MSP430]	3mW	98μW	15μW
CC2420[Jurdak et al.]	85.7 μW	42.3 μW	1.035 μW

- Energie du traitement radio : Cette énergie dépend principalement des données à transmettre ou à recevoir lors de l'activation de l'unité radio. On désigne 4 modes d'opération pour les modules de communication radio : transmission, réception, idle et sommeil. Selon [Xu et al. (a)], la consommation énergique durant le mode "Idle" est légèrement inférieure de celle du mode réception. Cela est dû à l'écoute du canal durant le mode Idle pour une éventuelle réception des données.

- Energie de détection d'événement : Cette énergie dépend d'une

part des données collectées lors de l'activation de l'unité d'acquisition et d'autre part des types de capteurs. Selon [Fraden], on peut classifier les capteurs, en se basant sur la consommation énergétique, en deux types : actif ou passif. Les capteurs passifs à savoir, les Tag RFID, les SAW..., ne disposent pas d'une source d'énergie embarquée et sont économes en énergie. Les capteurs actifs sont plus énergivores et nécessitent l'intégration d'une source d'alimentation.

L'architecture matérielle d'un nœud capteur n'est pas la seule source de dissipation d'énergie dans le réseau de capteurs sans fil. Toutefois, il ne faut pas négliger les liens de communication radio qui ont une grande influence sur la consommation énergétique des nœud capteurs. En effet, Raghunathan [Raghunathan et al. (b)] a montré, par expérimentation, que le module du traitement radio est le module le plus énergivore dans un nœud capteur. Le coût énergétique pour la transmission d'un bit d'information est calculé par l'équation suivante (1.1) :

$$E_{bit} = \frac{E_{start}}{L} + \frac{P_{elec} + P_{RF}(M)}{Rs * log_2(M)}(1 + \frac{H}{L}) \tag{1.1}$$

L : la taille du paquet, H : la taille de l'entête, E_{start} : le surcoût lié à l'activation du module radio, P_{elec} : Puissance consommée du circuit électronique liée aux fonctionnalités de filtrage, modulation, conversion,etc, P_{RF} est : la puissance délivrée par l'amplificateur de signal, R_s : le débit symbole et M est le nombre des symboles,

1.2 Réseaux de capteurs et Internet des Objets

La norme IEEE 802.15.4 a fixé une taille maximale d'une trame circulant dans RCSF à 127 octets. Une trame de cette taille ne peut jamais supporter un paquet IP de taille 1028 octets. Ainsi, toutes les données échangées dans un RCSF ne sont accessibles aux autres réseaux,

principalement basés sur l'IP, que par l'intermédiaire d'une passerelle. Afin d'assurer la connectivité entre les RCSFs et les réseaux Internet, plusieurs travaux de recherche ont été élaborés. Une première solution a été proposée par Dunkels dans [ETCPA]. Cette solution (architecture) repose sur le développement de deux versions de la pile protocolaire compatibles avec les RCSFs, une minimale « micro IP (μIP) » et l'autre légère « lightweight IP (lwIP) ». Ces deux versions ont été testées et implémentées sur des plateformes (open source) de 8 bits et 16 bits [ETCPA] et ont montré une utilisation efficace de la RAM qui a atteint quelques dizaines de kilo octets. Une autre solution (architecture) appelée IPSense a été proposée par Camilo[Camilo et al.]. A l'aide d'un système d'adressage IP et l'intégration de l'algorithme de colonies de fourmis, cette architecture a fait preuve d'une efficacité vis-à-vis la consommation énergétique, la connectivité ainsi que le routage. D'un autre côté, le groupe IETF (Internet Engineering Task Force) [IETF] a lancé, de sa part, une nouvelle solution (architecture) qui vise l'utilisation de l'IPv6 et l'adapter aux RCSFs en tenant compte de leurs contraintes. Ce groupe a publié et standardisé les premiers réseaux sans fil basés sur l'IPv6 appelés 6LoWPANs (IPv6 over Low-Power Wireless Personal Area Networks) [6LOWPAN]. Grâce à une couche d'adaptation 6LoWPAN, la connectivité entre les réseaux IP (IPv6) et les systèmes à faibles ressources est devenue possible. Sur la figure 1.4, nous présentons l'architecture réseaux de 6LoWPAN. Cette architecture est formée par un ensemble de nœuds contraints (débit, mémoire, énergie...) qui sont compatibles avec le standard IEEE 802.15.4. On peut distinguer trois catégories de nœuds : les routeurs de bordure (6LBR) qui présentent des points d'accès à Internet, les routeurs (6LR) et les hôtes (ED). La couche d'adaptation de 6LoWPAN est chargée de réduire la taille des en-têtes en utilisant des techniques de

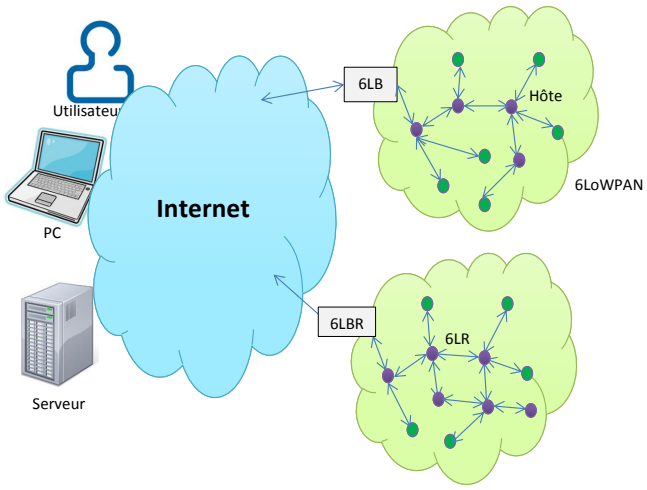

FIGURE 1.4 – *Architecture 6LoWPAN*

fragmentation et de compression [Montenegro et al.].Cette compression est nécessaire pour la transmission des données sur le réseau de capteurs (IEEE 802.15.4). En effet, la transmission d'une trame contenant un paquet IPv6 de 1280 octets de longueur n'est pas possible dans un réseau de capteurs dont le MTU (maximum transmission unit) est de 127 octets. Afin de fournir une connectivité entre les réseaux de capteurs sans fil et les réseaux IP, deux architectures possibles ont été proposées : une première qui est basée sur un proxy et une deuxième qui se base sur la pile IP et devrait être implémentée au niveau des nœuds capteurs.

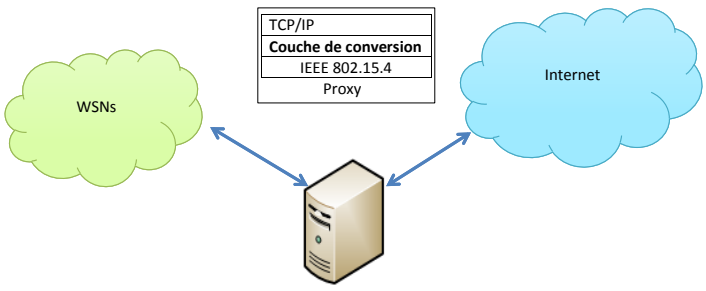

FIGURE 1.5 – *Architecture basée sur un proxy*

La première architecture (Figure 1.5) repose sur un proxy (serveur)

qui fournit une liaison entre les RCSFs et le réseau Internet. Le proxy joue le rôle d'un routeur pour rendre tous les services des RCSFs accessibles depuis Internet. L'avantage majeur de cette architecture est qu'elle est applicable à tout type de RCSF et ne nécessite aucune modification des piles protocolaires au niveau des nœuds capteurs.

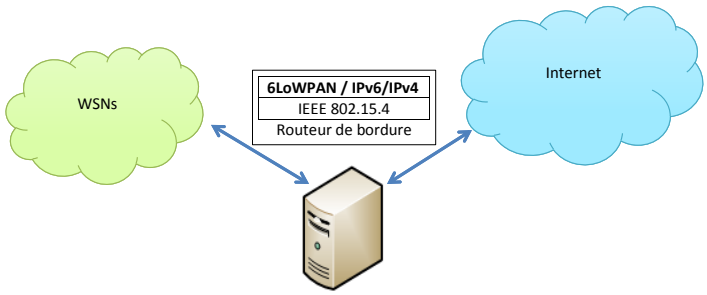

FIGURE 1.6 – *Architecture basée sur la pile IP*

La deuxième architecture (Figure 1.6) utilise le routage IP pour acheminer les données depuis les nœuds capteurs vers les réseaux externes. Tous les paquets circulant au sein des réseaux de capteurs doivent être routables.

1.3 Routage dans les réseaux de capteurs sans fil

Dans cette section, nous présentons un état de l'art des protocoles de routage traités dans la littérature. La transmission de l'information dans un réseau de capteurs peut se faire de deux manières : Soit l'envoi direct qui n'est possible que lorsque les nœuds sont suffisamment proches les uns des autres, soit l'envoi par routage de données.

1.3.1 Principaux protocoles de routage pour les RCSFs

Vue la complexité et la variété des protocles de routage dédiés aux RCSFs, plusieurs chercheurs ont proposé de les classifier selon différents critères à savoir la structure du réseau [Al-karaki et Kamal], l'établissement de la route [WASN], le paradigme de communication

[Niculescu et America], le fonctionnement du protocole [Al-karaki et Kamal]...

Dans cette section, nous allons utiliser une classification en trois catégories selon le pragmatique de communication : hiérarchique, basé sur la géolocalisation et centré sur les données.

Routage Hiérarchique

Le routage hiérarchique est conçu pour optimiser surtout la consommation énergétique des nœud capteurs. Plusieurs techniques peuvent être utilisées pour réduire au plus le nombre de messages transmis dans le réseau et diminuer la consommation énergétique des nœud capteurs[I. Memon (2012)]. Parmi ces techniques, on cite l'agrégation des données, la fusion, la clusterisation, etc. Les protocoles de routage hiérarchique sont généralement adoptés afin de permettre au système de couvrir une zone d'intérêt plus large sans dégradation de la qualité de service. En fait, tous les nœuds savent que si le destinataire n'est pas dans leur voisinage direct, il suffit d'envoyer la requête à une passerelle qui la transmettra vers les nœuds cibles. Le protocole LEACH [Wendi R.], proposé par Heinzelman et al, est le plus connu dans cette catégorie. Il est basé sur une technique de clustérisation permettant de former des groupes de nœuds appelés « clusters ». Dans chaque cluster, un nœud maître est élu nommé Cluster Head (CH) afin de router les données vers le nœud Sink . Cela permet de réduire la charge au niveau des nœud capteurs puisque le routage sera fait au niveau des CHs qui présentent, en moyenne, 5% des nœuds dans le réseau [Wendi R.]. L'efficacité énergétique de LEACH a été prouvée par simulation en le comparant avec des algorithmes de communication directe. En revanche, LEACH ne passe pas à l'échelle et cette efficacité reste valable seulement

pour les réseaux de petite taille. Cela est dû principalement au type de clustering utilisé qui se limite au routage à un seul saut et surtout au coût énergétique lié à la formation des clusters. Une amélioration du protocole LEACH a été proposée par Lindsey et al. [Lindsey et Raghavendra] appelée PEGASIS. L'idée principale est de remplacer les clusters par des chaînes de proches voisins et de choisir un seul nœud de chaque chaîne afin de transmettre les données au nœud Sink. Afin d'équilibrer la charge entre les nœuds capteurs dans le réseau, PEGASIS se base sur l'algorithme d'ordonnancement « Round-Robin » où une période T (Round) est réservée pour chaque nœud choisi comme relais dans la chaîne. Cela permet ainsi de diminuer la consommation énergétique des nœuds relais durant la période T. Dans [Lindsey et Raghavendra], une comparaison a été également faite avec le protocole LEACH. Les résultats obtenus ont montré sa capacité de réduire la consommation énergique et donc de prolonger la durée de vie du réseau. Selon les résultats de simulation, PEGASIS peut multiplier par trois la durée de vie du réseau par rapport au protocole LEACH. Cela est réalisé grâce à la suppression du processus de clustérisation permettant ainsi l'élimination de l'overhead (la quantité de trafic de contrôle échangée).

Un deuxième protocole, appelé TEEN (Threshold sensitive Energy Efficient sensor Network protocol), plus optimisé que LEACH, a été proposé par Manjeshwar et Agrawal [Manjeshwar et Agrawal (b)]. TEEN est conçu principalement pour être sensible à des changements brusques de certains paramètres. En utilisant l'algorithme TEEN, les nœuds capteurs peuvent traiter les données d'une manière continue, tandis que la transmission des données est moins fréquente. En effet, chaque cluster-head du réseau diffuse à tous les nœuds de son cluster deux seuils, un seuil principal HT (Hard Threshold) et un seuil de

variation ST (Soft Threshold). Si la valeur mesurée par un nœud est supérieure au seuil principal, HT ,le nœud capteur, peut donc commencer l'envoi des données. Ensuite, il cesse de transmettre les mêmes données tant que la valeur mesurée n'a pas changé d'un seuil de variation ST. L'objectif du seuil principal est de réduire le nombre de transmissions en permettant aux nœuds d'envoyer uniquement quand les valeurs mesurées se trouvent dans des intervalles d'intérêt. Le seuil de variation permet également de réduire davantage le nombre de transmissions et d'éviter l'envoi des valeurs avec des changements mineurs. Par conséquent, la transmission des données devient moins fréquente et la consommation d'énergie devient moins importante que celle estimée en utilisant le protocole LEACH. Néanmoins, TEEN présente un seul inconvénient par rapport à la mise à jour des seuils SH et SV. Si un nœud ne reçoit pas les seuils, il ne pourra plus communiquer avec le Sink et il sera donc écarté du réseau. Afin de pallier à ce problème, une extension APTEEN (Adaptive Threshold-sensitive Energy Efficient sensor Network protocol) du protocole TEEN a été proposée par Manjeshwar [Manjeshwar et Agrawal (a)]. APTEEN est une solution hybride et flexible qui, selon le type d'application, change les valeurs HT (Hard Threshold) et ST (Soft Threshold) ainsi que la périodicité de transmission afin de contrôler plus la consommation énergétique.

Routage basés sur la localisation

Toujours dans le but d'optimiser la consommation énergétique, la connaissance des positions géographiques des différents nœuds dans le réseau permet de calculer facilement les coûts des itinéraires possibles reliant un émetteur à un récepteur. Les protocoles de routage géographique utilisent donc ces informations de localisation afin de

faciliter l'acheminement des données, réduire le coût du routage, simplifier la gestion et optimiser plus la consommation énergétique. L'inconvénient de ces protocoles de routage est que chaque nœud doit connaître les emplacements des autres nœuds. La connaissance de ces informations nécessite un module de géolocalisation embarqué dans chaque nœud, ou bien, l'implémentation des techniques de localisation basées sur les propriétés de signaux envoyés ou reçus dans le réseau. Sans avoir besoin ni de connaître la topologie globale du réseau ni d'avoir une table de routage, un nœud a besoin seulement de localiser ses voisins pour choisir le prochain saut et router les données. Ces dernières contiennent ainsi les informations de localisation du nœud source et celles du nœud destinataire, et via les nœuds intermédiaires , les décisions d'acheminement des données sont prises. Afin d'optimiser au plus la durée de vie du réseau, d'autres métriques peuvent être prises en considération pour les décisions de routage à savoir le taux de charge, l'énergie résiduelle et la congestion. La figure 1.7 illustre le principe de fonctionnement d'un routage géographique basique. Le nœud N_s désigne le nœud source, les nœuds N_i désignent les nœuds voisins de N_s et N_d désigne la destination. Tout d'abord, Ns calcule les différentes distances qui séparent tous ses voisins N_i du nœud destinataire N_d et les compare avec celle qui le sépare avec N_d. Tous les nœuds voisins ayant une distance inférieure à d(N_s, N_d) représentent ainsi des liens de routage (LR) pour atteindre la destination.

$$LR = \{N_i | d(N_i, N_d) < d(N_s, N_d)\} \tag{1.2}$$

Ensuite, N_s doit choisir un seul nœud de cet ensemble comme prochain saut. Pour cela, il compare toutes les distances et choisit la plus courte qui désigne le nœud le plus proche à la destination. L'opération est répétée de la même façon jusqu'à atteindre Nd.

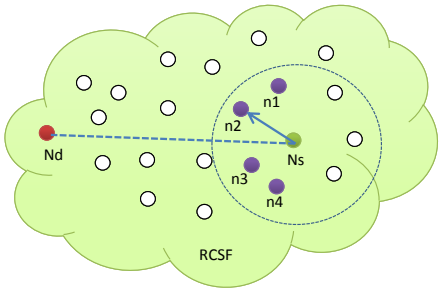

FIGURE 1.7 – *Routage géographique*

En 2000, Brad Karp et al. [Karp et Kung] ont proposé un protocole de routage géographique appelé GPSR (Greedy Perimeter Stateless Routing). Il est basé sur la combinaison de deux méthodes de routage pour acheminer les paquets dans un réseau de capteurs sans fil. La première méthode est appelée « transmission Gloutonne » (greedy forwarding, en anglais) et la deuxième est appelée « Transmission de Périmètre » (perimeter forwarding, en anglais). La « transmission Gloutonne » est utilisée comme une méthode principale pour acheminer les données dans le réseau. Le principe de routage de cette méthode est de choisir toujours le voisin le plus proche de la destination. Cependant, cette méthode devient inefficace quand on est dans une zone contenant un obstacle, appelé aussi minimum local (Figure 1.8). Dans ce cas, Les nœuds exécutant GPSR bascule à la méthode «Transmission de Périmètre » afin d'éviter cet obstacle et le contourner dans le but de trouver un nœud plus proche de la destination.

En effet, un nœud obstacle ou un minimum local se forme quand un nœud n'a pas de voisin proche via lequel il peut atteindre la destination. La figure 1.8 présente un scénario où le routage est effectué avec la combinaison des deux méthodes. Le nœud N_{ML} se trouve comme un minimum local où la destination reste inaccessible avec la méthode « transmission gloutonne ». En appliquant la deuxième méthode «

Transmission de Périmètre », le nœud N_{ML} transfère les paquets sur les facettes croisées par la ligne (N_{ML}, N_d). Dès qu'un nœud se trouve à une distance de la destination plus proche que la distance [N_{ML},N_d], il retourne en mode «transmission gloutonne » pour atteindre la destination. C'est le cas du noeud N_i et son voisin N_j qui est plus proche de la destination que le noeud N_{ML}. Par la suite, le choix du chemin entre le noeud N_j et la destination N_d est realisé en appliquant la méthode « transmission gloutonne ».

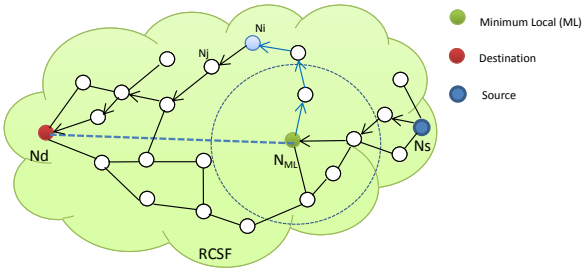

FIGURE 1.8 – *Protocole de routage GPSR*

Un nouveau protocole de routage géographique, similaire à GPSR, a été proposé par Yan Yu [Yu et al.]. Le nom de ce protocole est GEAR (Geographic and Energy Aware Routing). Son concept repose sur le découpage du réseau en plusieurs régions et d'exploiter les informations de localisation pour restreindre le nombre d'intérêts dans la diffusion dirigée [Intanagonwiwat et al.] en ciblant quelques régions au lieu d'inonder tout le réseau avec l'ensemble des données. Avec GEAR, chaque nœud connaît le coût pour atteindre la région destinatrice en passant par ses voisins. Ce coût est divisé en deux parties : un coût estimé et un coût d'apprentissage (équation 1.3). Le coût estimé est calculé en fonction de l'énergie résiduelle et la distance entre le nœud voisin et le nœud destinataire, en assumant qu'il n'y a aucun minimum local. Tant dis que le coût d'apprentissage est un raffinement du coût estimé qu'un nœud dépense pour le routage autour des trous dans le réseau. S'il n'y a pas de

trous, le coût estimé est égal au coût d'apprentissage.

$$C(N_i, R) = x * d(N_i, R) + (1 - x) * e(Ni) \tag{1.3}$$

où N_i est le voisin i, R est la région destinatrice, d est la distance, e est l'énergie résiduelle.

Une autre proposition, appelée MECN (Minimum Energy Communication Network), a été réalisée par Rodoplu et al. [Rodoplu et Meng]. L'idée de MECN est d'identifier, pour chaque nœud, un sous réseau de transmission qui représente sa passerelle afin d'atteindre, avec le moindre coût d'énergie, la destination dans le réseau. Le sous réseau sera composé d'un nombre optimal de nœuds qui nécessitent moins d'énergie pour transmettre les données entre l'émetteur et le Sink. Le principe de fonctionnement de MECN est comme suit : Soit N_i, N_r, N_j trois nœuds capteurs représentant respectivement un nœud émetteur, un nœud relais et un nœud récepteur. Pour déterminer la région relais (N_i, N_r), MECN s'appuie sur l'équation suivante (1.4) :

$$R_{i \to r} = \{(x, y) | P_{i \to r \to (x,y)} < P_{i \to (x,y)}\} \tag{1.4}$$

Où Pi->r->x,y est la puissance nécessaire pour la transmission entre 'i' et (x,y) via le nœud 'r'. Ainsi, chaque nœud N_j appartenant à la région R_{ir}, le chemin de transmission via le nœud Nr (Ni->Nr->Nj) est plus économe que le chemin direct entre Ni et Nj. SMECN (Small MECN), est une extension de MECN, qui crée un sous-graphe avec un minimum d'énergie. Le sous-réseau construit par SMECN est probablement plus petit (en termes de nombre d'arcs) que celui construit par MECN. Toujours à base d'un système de localisation des nœuds, Ya Xu et al. ont proposé le protocole GAF (Geographic Adaptive Fidelity) [Xu et al. (b)] à basse consommation d'énergie. L'idée de GAF est de mettre en veille, temporairement, des nœuds du réseau qui ne sont pas utiles pour

un transfert donné. En effet, GAF met en place des grilles virtuelles (nœuds) de taille constante de telle sorte que chaque nœud d'une grille puisse atteindre tous les nœuds de la grille voisine. Une fois les nœuds émetteur et récepteur sont identifiés et localisés, GAF désigne un nœud actif de chaque grille pour effectuer toutes les opérations de routage. Il existe trois modes dans GAF. Le mode en veille, quand le module de communication est éteint. Le mode de découverte, pour mettre à jour la liste des voisins et le mode actif pour effectuer les fonctionnalités du routage. Afin d'équilibrer la consommation énergétique, tous les nœuds d'une grille doivent basculer d'un mode à un autre.

Routage centré sur les données

Dans de nombreuses applications de réseaux de capteurs, vu le nombre élevé de nœuds déployés, il n'est pas possible d'attribuer des identificateurs globaux à chaque nœud. Cette absence d'un schéma d'adressage global avec le déploiement aléatoire de nœuds de capteurs fait qu'il est difficile de sélectionner un ensemble spécifique de nœuds de capteurs à interroger. Par conséquent, plusieurs chercheurs ont axé leurs travaux sur le développement des protocoles de routage centrés sur les données. SPIN (Sensor Protocol for Information via Negotiation) [Heinzelman et al. (b)] est le premier protocole de routage qui a été proposé dans cette catégorie. Il a été conçu pour réduire les données redondantes et économiser plus de l'énergie par rapport aux techniques de transmission classique à savoir l'inondation des données. En effet, SPIN [Heinzelman et al. (b)] consiste à proposer de nouveaux types de messages permettant d'optimiser l'envoi des données utiles dans le réseau. Le premier type de messages est 'ADV' qui représente des métadonnées décrivant les données en question. Le deuxième type est 'REQ' qui

représente une demande pour des données précises et le troisième type de message est 'DATA' qui représente les données utiles. Avec SPIN, un émetteur doit diffuser à tous ses voisins un message de type ADV avant d'envoyer les données utiles 'DATA'(Figure 1.9). En lisant les métadonnées ADV, les nœuds voisins consultent leurs bases d'intérêt. Ceux qui sont intéressés par le descriptif, ils envoient un message de type REQ à l'émetteur de l'ADV. Une fois le nœud émetteur reçoit les messages REQ, il transmet donc le message DATA à tous les nœuds intéressés.

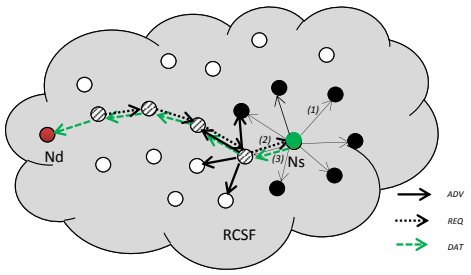

FIGURE 1.9 – *Protocole de routage SPIN*

SPIN est considéré comme un algorithme de routage très simple à mettre en œuvre. Il ne nécessite ni un système d'adressage ni des tables de routage prédéfinies. Cela a rendu le protocole SPIN, d'une part, plus économe en énergie, et d'autre part plus attractif. Néanmoins, la livraison des données n'est pas garantie avec SPIN. En effet, ce protocole se base sur les intérêts des nœuds pour transmettre les données entre un nœud source et un nœud intéressé. Si les nœuds intermédiaires ne sont pas intéressés par les données, ces dernières ne seront jamais délivrées. Dans d'autres concepts, contrairement à celui de SPIN, de nouveaux mécanismes d'interrogation de données ont été abordés. C'est le cas du protocole DD (Directed Diffusion) [Intanagonwiwat et al.]. En effet, le nœud collecteur (Sink) est celui qui interroge les capteurs (Figure 1.10) si des données spécifiques sont prêtes à être émises alors qu'avec le protocole

SPIN, c'est le nœud source qui interroge les nœuds sur ses intérêts en annonçant les données disponibles. La diffusion dirigée comporte trois phases : l'inondation d'intérêt, la propagation des données (phase d'exploration) et le renforcement positif. Pendant la première phase, le nœud Sink commence à diffuser ses intérêts à travers tous ses voisins. Un intérêt est une requête définie par une liste de pairs attribut-valeur, à savoir la position, la périodicité, la durée, le type des données, etc. Cette requête spécifie toutes les données auxquelles le nœud Sink est intéressé. Par la suite, chaque nœud recevant une requête (intérêt) garde une copie dans sa mémoire cache qui sera exploitée ultérieurement. Il définit, après, un gradient associé à cet intérêt qui sert à identifier les voisins depuis lesquels la requête est reçue. Le gradient contient deux champs : une valeur et une direction. Le champ valeur correspond au degré du renforcement du gradient (3ème phase) et le champ direction désigne le chemin vers le nœud Sink(Figure 1.10).

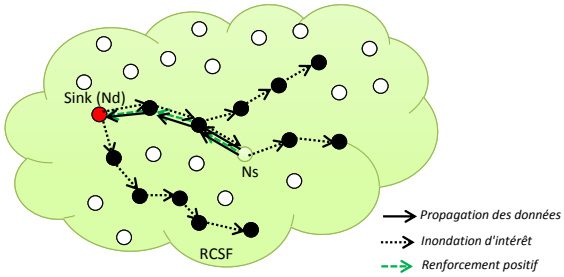

FIGURE 1.10 – *Protocole de routage DD*

Pour chaque réception d'un intérêt, le nœud met à jour son cache en comparant l'intérêt reçu avec tous les intérêts sauvegardés. Si l'intérêt existe avec un gradient différent, dans ce cas, il ajoute un nouveau gradient sinon il met à jour les valeurs des attributs. S'il ne trouve pas l'intérêt, il crée une nouvelle entrée avec le gradient correspondant. Une fois la requête est inondée dans le réseau, on passe à la deuxième phase où les nœuds ciblés deviennent des sources de données pour l'intérêt reçu. Dans

ce cas, ils commencent à envoyer les données utiles (réponse à la requête) au nœud Sink avec un débit très faible dit débit d'exploration. Chaque nœud intermédiaire redirige, par la suite, les données vers l'émetteur de l'intérêt. Pour cela, il se base sur les informations sauvegardées dans la mémoire cache en lisant les valeurs du gradient associé à l'intérêt en question. Toutes les données utiles se propagent dans le réseau jusqu'au nœud Sink. Arrivant à cette étape, plusieurs routes sont prises afin d'acheminer les mêmes données vers le nœud Sink et par conséquent le réseau devient redondant. Pour pallier à ce problème, DD propose, durant la troisième phase, un renforcement positif qui permet d'une part la sélection d'une seule route entre la source et la destination et d'autre part le changement du débit d'exploration à un débit plus haut appelé débit de renforcement. En fait, dès la réception des données d'exploration, le nœud Sink envoie un message de type renforcement positif à un seul voisin choisi selon plusieurs critères [Intanagonwiwat et al.]. Ce message correspond réellement à un intérêt à haut débit. Il sera transmis sur un seul chemin jusqu'au nœud source. En revanche, tous les nœuds recevant ce message vont renforcer leurs gradients associés à l'intérêt renforcé afin de construire un chemin à haut débit entre la source et la destination. Toutefois, DD présente une latence très élevée due aux deux premières phases (phase d'inondation et phase d'exploration). Cela le rend inefficace pour des applications temps réel qui nécessitent un haut débit à savoir les applications de suivi médical en temps réel. Par ailleurs, une extension, appelée RR (Rumor Routing) du protocole DD a été faite par Braginsky et al. [Braginsky et Estrin]. Ce protocole vise à rendre la consommation énergétique plus efficace en réduisant l'inondation du réseau. Il permet également de trouver un compromis entre l'inondation des intérêts et la propagation des données. Lors d'une détection d'un événement, les

nœuds capteurs l'ajoutent dans leurs tables d'événements locaux et créent par la suite des agents inondant le réseau avec certaines probabilités (afin d'éviter les boucles). Chaque nœud intermédiaire maintient une table de relais locale associée à chaque intérêt en sauvegardant le prochain saut vers le Sink, celui du nœud précédant, ainsi qu'une métrique qui représente le nombre de sauts vers chaque extrémité. Quand un nœud génère une requête pour un événement, les nœuds intermédiaires qui connaissent la route, peuvent répondre à la requête en se référant à leurs tables d'événements. Dans ce cas, un seul chemin est maintenu entre la source et la destination contrairement à DD où plusieurs chemins sont établis. Ce dernier n'est pas le chemin le plus optimal permettant d'envoyer les données depuis la zone d'intérêts vers le Sink mais il est considéré comme le premier chemin trouvé entre la source et la destination. Selon les résultats de simulation [Braginsky et Estrin], le protocole RR présente un taux de délivrance des requêtes de 99.9% avec un facteur de 1/40 de messages inondés dans le réseau. Cela le rend d'une part plus fiable que d'autres protocles centrés-données (SPIN par exemple) et d'autre part moins coûteux du point de vue de la consommation énergétique et de la bande passante.

Une autre amélioration du protocole DD a été proposée dans [Raghunathan et al. (b)]. C'est le protocole GBR (Routage à Base de Gradient). Avec ce protocole, chaque nœud doit sauvegarder sa « hauteur » dans le but de réduire le nombre des chemins multiples. Cette hauteur peut être définie par le nombre de sauts entre le nœud et le Sink. En effet, lors de la phase d'inondation, le Sink diffuse l'intérêt avec une hauteur égale à 0. Tout nœud recevant l'intérêt incrémente la hauteur associée et il met à jour la valeur de son gradient qui est la différence entre sa hauteur et celle du nœud voisin émetteur de l'intérêt (Gradient sur le

lien). Lors de la deuxième phase d'exploration, les nœuds intermédiaires acheminent les données seulement sur les liens du plus grand gradient. En cas d'égalité de gradient, GBR applique un schéma stochastique en choisissant au hasard un nœud voisin. Une autre technique peut être appliquée lorsque le niveau de l'énergie résiduelle baisse au-dessous d'un seuil. Dans ce cas, le nœud utilise le schéma à base d'énergie en augmentant sa hauteur. Cela permet de diminuer le gradient des liens entre le nœud et ses voisins et de réduire sa charge pour la transmission des données utiles. En outre, les chercheurs dans [Chu et al.] ont proposé une version du protocole DD plus générique. Cette version se repose sur deux techniques : IDSQ (information-driven sensor querying) qui permet d'optimiser la sélection des capteurs interrogés dans le réseau et CADR (constrained anisotropic diffusion routing) qui permet à son tour de minimiser le coût d'acheminement des données. L'idée de IDSQ est d'introduire, au niveau des nœuds, des fonctions de mesure et d'estimation de l'utilité de l'information. Cela permet aux nœuds interrogateurs d'avoir un moyen de sélection dynamique et optimale des meilleurs nœuds capteurs fournissant les informations les plus utiles dans le réseau. Ainsi, plus le nombre de nœuds interrogés diminue plus le coût énergétique, lié à l'inondation des intérêts, diminue. Avec le CDAR, chaque nœud dans le réseau peut évaluer l'objectif information/coût et déterminer la valeur de son gradient, gain de l'information, pour acheminer les données jusqu'au nœud Sink. L'inconvénient du protocole CDAR consiste au fait que les données sont transmises de chaque nœud capteurs dans la région de déploiement avec une redondance importante. Cette redondance est très pénalisante en termes de la consommation d'énergie.

1.3.2 Synthèse

Dans le tableau récapitulatif ci-dessous 1.2, nous présentons une synthèse des protocles de routage traités dans ce chapitre.

TABLE 1.2 – *Récapitulation des protocles de routage*

Catégorie	Protocole	Energie	Passage à l'échelle	Mobilité
Routage Centré sur données	DD	● ● ○	● ● ○	● ○ ○
	RR	● ○ ○	● ● ●	○ ○ ○
	GBR	● ○ ○	● ● ○	● ○ ○
	CDAR	● ● ○	● ● ○	● ○ ○
Routage hiérarchique	LEACH	● ○ ○	● ● ●	● ● ○
	PEAGAS	● ● ○	● ● ●	● ● ○
	TEEN	● ○ ○	● ● ●	● ● ○
	APTEEN	● ○ ○	● ● ●	● ● ●
Routage géographique	GPSR	● ● ○	● ● ○	● ● ○
	GEAR	● ● ○	● ● ○	● ● ○
	MECN	● ○ ○	● ● ○	● ○ ○
	GAF	● ● ○	● ● ○	● ● ○

Cette étude bibliographique nous a permis de connaître mieux l'aspect routage dans les réseaux de capteurs sans fil et d'avoir une vision critique sur l'ensemble des techniques et protocoles de routage en déterminant les avantages et les inconvénients de chacun. Pour les protocles de routage géographique, chaque nœud doit connaître la position de tous ses voisins, généralement à un seul saut, ainsi que la position où la zone dans laquelle se trouve la destination. Cela réduit, non seulement la charge et l'inondation du réseau, mais aussi les coûts dû à la communication. Plusieurs méthodes de communication ont introduit des fonctions de localisation des nœuds mobiles afin d'avoir des meilleurs performances. A titre d'exemple, le II-routage, présenté dans [T. Muntean (2000)], repose sur les informations de localisation afin de trouver un compromis entre la longueur des chemins parcourus et le coût supplémentaire induit par les messages de contrôles. Cette méthode a montré de bonnes performances par rapport aux autres méthodes dans les systèmes distribués à migration.

Néanmoins, la détermination des différentes positions des voisins ajoute un coût de calcul supplémentaire pour chaque nœud capteur ainsi qu'un coût énergétique non négligeable. Par ailleurs, le minimum local présente, à son tour, un second problème pour les protocles de routage géographique qui se trouvent face à d'autres contraintes à surmonter. En effet, les nœuds représentant des minimums locaux, doivent gérer le routage des données soit en basculant à un autre mode de routage géographique (ex : perimeter forwarding) soit en acheminement les données vers toute la zone destinatrice par inondation. Dans tous les cas, un coût additionnel est ajouté à un ensemble de nœuds ce qui par conséquence diminue la durée de vie du réseau. Quant aux protocles de routage hiérarchique, la formation et la mise à jour des clusters dans le réseau de capteurs présentent un coût rédhibitoire pour ce type de protocole. Ce coût est lié surtout à la consommation énergétique quand il s'agit d'un réseau à forte mobilité. En revanche, le délai de transmission, l'agrégation des données et le passage à l'échelle sont plus optimisés comparant aux protocles d'une autre catégorie. En effet, l'organisation et le bon partitionnement permettent de réduire le contrôle des données afin de réaliser l'acheminement entre la source et la destination. Quant au routage centré sur les données, les protocoles de cette catégorie sont connus par leur simplicité dans l'acheminement des données dans le réseau. Le choix du prochain saut est effectué seulement par le nœud d'une manière individuelle et décentralisée. Toutefois, le concept de cette catégorie se base sur le multi-chemin entre une source et une destination provoquant ainsi un surcoût de communication dans le réseau. De plus, les protocoles de routage centré sur les données ne supportent pas les nœuds mobiles puisqu'ils sont conçus pour un environnement statique. Afin de pallier aux différentes problématiques, plusieurs auteurs ont

proposé des protocoles de routage hybrides [Pinto et H, Abdulla et al.] en combinant deux ou plusieurs stratégies classiques. Ces solutions hydrides sont conçues pour offrir un routage plus flexible et peu énergivore en preneant en compte plusieurs paramètres à savoir la qualité de service, la densité du réseau, la plateforme et la nature de l'application.

1.4 Sécurité dans les réseaux de capteurs sans fil

Dans cette section, nous allons commencer par une brève introduction sur la cryptographie d'une manière générale. Nous allons détailler, par la suite, quelques solutions de sécurité pour le routage dans les RCSFs ainsi que les différentes attaques et menaces qui existent.

1.4.1 Les systèmes cryptographiques

La cryptographie est la technique la plus utilisée dans la plupart des mécanismes de sécurité. Elle est basée essentiellement sur des algorithmes de chiffrement qui permettent de répondre aux exigences fondamentales de la sécurité pour les systèmes communicants. Les méthodes cryptographiques sont parmi les solutions les plus sûres qui répondent à l'ensemble des problématiques liées à la sécurité des données dans les réseaux filaires et les réseaux sans fil traditionnels (disposant d'une capacité de calcul et de mémoire conséquente). Il existe deux principaux types de systèmes cryptographiques : la cryptographie symétrique et la cryptographie asymétrique, appelée souvent cryptographie à clé publique.

Cryptographie symétrique

La cryptographie symétrique se base sur le partage d'une seule clé de chiffrement appelée «secrète». Grâce à cette clé, l'expéditeur peut chiffrer le texte avant de l'envoyer au destinataire. Ce dernier, à son tour, pourrait

déchiffrer le texte en utilisant le même algorithme de chiffrement et la clé secrète partagée auparavant.

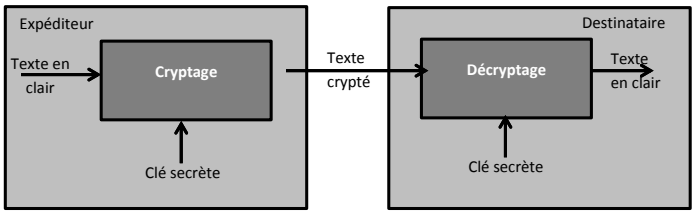

FIGURE 1.11 – *Principe du cryptage symétrique*

La figure 1.11 illustre un échange de données basé sur la cryptographie symétriques. Dans la cryptographie symétrique, nous distinguons deux types d'algorithmes. Le premier type est le chiffrement par blocs (ex : DES, AES, Skipjack , 3DES, CAST, Blowfish...) qui prend 'n bits' en entrée et restitue 'n bits' chiffrés, et le deuxième type, c'est le chiffrement par flux (ex : RC4, E0, Py...) qui crypte les données bit par bit.

Cryptographie asymétrique

La cryptographie asymétrique se base sur l'utilisation d'une paire de clés de chiffrement (clé privée / clé publique). La clé privée est gardée tout le temps par son propriétaire tandis que la clé publique est accessible par tout le monde. Dans les systèmes de cryptographie asymétrique, les clés de chiffrement et de déchiffrement sont distinctes et inductibles. Si on chiffre un message avec une clé publique d'un propriétaire, dans ce cas, on ne peut le déchiffrer qu'avec sa clé privée. Ce type de chiffrement permet alors d'assurer la confidentialité entre l'expéditeur et le destinataire. De plus, si un propriétaire chiffre un message avec sa clé privée, dans ce cas, n'importe qui possédant la clé publique pourra le déchiffrer. Cette façon de chiffrement permet de placer des signatures numériques dans un message et permet ainsi l'authentification de l'expéditeur. Généralement, les algorithmes de chiffrement asymétriques sont plus lents que les

algorithmes de chiffrement symétriques. Ils sont souvent utilisés pour chiffrer une clé de session secrète. Nous illustrons dans la Figure 1.12 un exemple de chiffrement asymétrique. Soient Kpu la clé publique et Kpr la clé privée d'un destinataire D, F la fonction de chiffrement et G la fonction de déchiffrement. Une fois le destinataire diffuse sa clé publique dans le réseau, l'émetteur envoie le message M chiffré tel que : $M_k = F(\ K_{Pu},\ M)$ où M_k est le message chiffré. À la réception du message, ce dernier sera déchiffré avec la clé privée du récepteur : $M = G(\ K_{pr}, M_k\ )$.

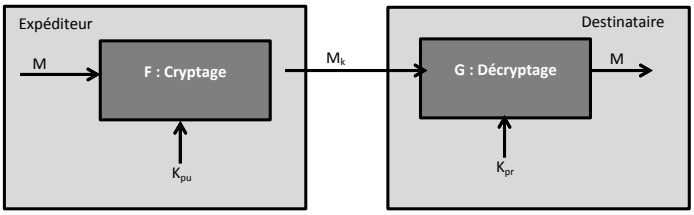

FIGURE 1.12 – *Principe du cryptage asymétrique*

Bien que les algorithmes de chiffrement asymétriques soient plus lents que la plupart des algorithmes symétriques, ces derniers sont plus efficaces[Sahingoz (2013)] en termes d'adaptation aux systèmes à grande échelle et de flexibilité d'authentification.

1.4.2 Exemple d'algorithmes cryptographiques

Diffie-Hellman

Diffie-Hellman est une méthode cryptographique proposée, en 1976, par Whitfield Diffie et Martin Hellman [Diffie et Hellman]. A l'aide des fonctions inversibles et le problème du logarithme discret (DLP), Diffie et Hellman ont démontré la possibilité d'utiliser un couple de clés au lieu d'une seule clé partagée. La clé publique est connue par tout le monde, par contre la clé privée est gardée secrète par son propriétaire. Avec DH, il est impossible de déterminer cette clé privée à partir de la clé publique. La personne possédant la clé publique peut effectuer le chiffrement mais

seule celle qui possède la clé privée pourrait faire l'opération inverse et déchiffrer les informations. Cette méthode se base sur la difficulté du problème suivant : Soient 'a' un générateur d'un groupe fini 'G' et les valeurs 'a' et 'b' tels que a= ax et b= ay. Le problème du Diffie-Hellman consiste à déterminer la valeur axy (où x et y sont des entiers aléatoires). Il est presque impossible de calculer cette valeur en connaissant seulement 'a' et 'b'.

Méthode RSA

En 1978, Adi Shamir et Leonard Adleman [Rivest et al.] ont créé la méthode de cryptographie RSA. Cette méthode est considérée parmi les systèmes cryptographiques à clé publique les plus utilisés dans les réseaux de communication. Elle repose sur la facilité de la multiplication de deux grands nombre premiers dont la factorisation de leur produit est difficile. Par exemple, pour deux nombres premiers p et q, il est facile de calculer leur produit n = p*q. Alors que la factorisation de 'n' pour retrouver p et q, en ne connaissant que 'n' est presque impossible. Le calcul des clés RSA s'effectue à l'aide d'un générateur de clé qui consiste à générer quatre nombres p, q, e et d de la façon suivante :

- p et q sont deux nombres premiers distincts générés par un algorithme de test de primalité déterministe ou probabiliste.

- soient n= p*q et f (n) = (p-1)*(q-1).

- choisir e, tel que e est premier avec f (n).

- choisir d, tel que e*d = 1 mod f (n).

Une fois les quatre nombres sont créés, les clés publiques et privées seront constituées respectivement du couple (n,e) et du couple (n,d). Pour rendre le chiffrement RSA plus efficace, il faut avoir une factorisation très difficile à calculer en augmentant le nombre n.

Chiffrement El Gamal

Le chiffrement El-Gamal est créé en 1982 par Taher ElGamal [El Gamal]. Il est dérivé du RSA mais il propose un système plus complexe et se base toujours sur le problème du logarithme discret. Pour utiliser le chiffrement d'ElGamal, il faut d'abord choisir un grand nombre premier p et un générateur g du groupe multiplicatif dans Z /p Z. La clé privée est un nombre entier aléatoire X tel que $1 \leq X \leq (p-2)$ tandis que la clé publique Y est déterminée par l'équation : $Y = g^X \mod p$. Pour chiffrer un message M, on calcule le couple de nombres $(y1, y2)$ selon l'équation (1.5) sachant que le nombre k est choisi aléatoirement tel que $1 \leq k \leq (p-2)$:

$$MY = (y1, y2) = (y1 = g^k \mod p, y2 = MY^k \mod p) \tag{1.5}$$

Pour retrouver le message à partir du couple $(y1, y2)$, connaissant X, On calcule M tel que :

$$M = \frac{y2}{y1 * X} \mod p \tag{1.6}$$

Fonctions de hachage

Généralement, pour une communication à travers un canal peu sûr, le destinataire voudrait toujours s'assurer que le message arrive de son expéditeur sans aucune altération pendant le transfert. Pour résoudre ce type de problème, des fonctions de hachage à sens unique (ex : SHA-1, SHA-2, MD5...) sont proposées. Ces fonctions calculent une empreinte de données permettant à identifier rapidement l'expéditeur de ces données. Par définition, une fonction de hachage est une fonction $f : A \rightarrow B$ à sens unique et irréversible, i.e. il est facile de calculer $f(x), \forall x \in A$, complexité polynômiale, et il est difficile, complexité exponentielle, de

retrouver x à partir de $f(x)$. Avec cette fonction, on peut générer (produire) une empreinte unique d'un texte où un adversaire se trouve incapable d'obtenir le texte en connaissant son empreinte. Plusieurs mécanismes de sécurité reposent sur les fonctions de hachage en profitant de leurs spécificités. Parmi ces mécanismes, on cite l'authentification, l'intégrité et la signature. Afin d'évaluer et comparer les performances des fonctions de hachage, un taux de collision est défini en fonction de nombre de messages différents ayant deux empreintes identiques (ex : SHA1, collision en 263 opérations pour un message de 264 bit).

Chaînes de clés à sens unique

Inspirée de la fonction de hachage, la chaîne à sens unique est une primitive souvent utilisée par les systèmes de gestion de clés. Elle est déterminée en appliquant successivement une fonction à sens unique à une valeur initiale. Généralement, les systèmes de gestion de clés exigent le changement fréquent des différentes clés afin de résister contre les attaques exhaustives. Pour cela, cette chaîne est utilisée afin de produire plusieurs clés à chaque instant. Le principe des chaînes de clés (cf. Figure 1.13) à sens unique consiste à affecter une clé Ko initialement (Master key) nœud Master d'un réseau. Ensuite une séquence de clés est générée. La dernière clé Kn est choisie aléatoirement, puis les clés Kn-1...Ko sont générées par application successive d'une fonction de hachage à sens unique : F(Kn) = Kn-1. La dernière clé générée Ko est la première divulguée et distribuée à tous les nœuds dans un réseau. Cette chaîne permet également d'authentifier des messages en vérifiant leurs codes d'authentification de message (Message Authentication Code, MAC). Ces derniers sont considérés comme des signatures ou des codes

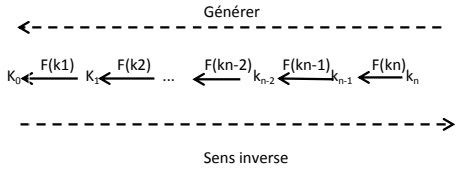

FIGURE 1.13 – *Exemple d'une chaîne de clés à sens unique*

accompagnant des données dans le but d'assurer l'intégrité et de vérifier les identités des nœuds.

Signature numérique

Dans le cas où le transfert des empreintes s'effectue sur un canal non sûr, on peut avoir une modification de ces empreintes par un intercepteur. Il faut donc trouver une méthode pour garantir que seul l'expéditeur puisse calculer l'empreinte. Une des solutions les plus sûres est d'utiliser la signature numérique. Elle a été créée dans le but de faire la preuve d'identité de l'expéditeur et de garantir l'intégrité des messages échangés dans un réseau. Généralement, elle est basée sur des fonctions de hachages et de chiffrement avec des algorithmes à clé publique. La méthode basique utilisée pour la signature numérique consiste à chiffrer l'empreinte du message comme le montre la figure 1.14.

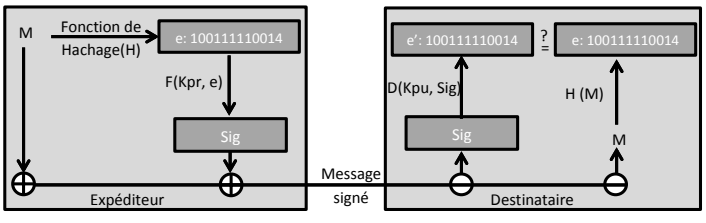

FIGURE 1.14 – *Signature numérique*

Pour la génération d'une signature numérique, il faut utiliser un algorithme de chiffrement à clé publique constitué d'une fonction de chiffrement F et une fonction de déchiffrement D, d'une clé publique

partagée sur un canal non sécurisé Kpu, une clé privée Kpr et d'une fonction de hachage H. La vérification de la signature revient à vérifier l'égalité entre l'empreinte du message reçu et la signature déchiffrée par la clé publique de l'émetteur.

1.4.3 Les exigences et les besoins de la sécurité dans les RCSFs

Comme tout autre réseau, des exigences de sécurité sont définies dans les réseaux de capteurs sans fil afin de répondre aux besoins des applications, garantir la bonne qualité de service et être ouvert aux autres réseaux. La première exigence de sécurité dans les RCSFs est la confidentialité des données. Elle pourrait être assurée en faisant appel à des mécanismes cryptographiques qui empêchent les nœuds n'ayant pas les autorisations nécessaires de lire ou de divulguer des messages circulant dans le réseau. La deuxième exigence est l'authentification qui permet de valider tout message circulant dans le réseau et d'authentifier les émetteurs de chaque message. Elle permet également de détecter les nœuds et les messages malicieux qui peuvent existent dans le RCSFs. La troisième exigence est l'intégrité des données qui assure la non-altération des données. A l'aide des fonctions de hachage et des algorithmes de chiffrement (symétriques ou asymétriques), chaque message dispose d'une empreinte et une signature numérique qui permettent de vérifier l'intégrité des données. Autre exigence, c'est la fraicheur des données qui doit être assurée dans les RCSFs . En effet, elle permet de garantir la récence des données et empêcher qu'un ancien message ne soit rejoué dans le réseau. A l'aide d'un système anti-rejeu, on peut vérifier le changement continu avec le temps de tous les messages circulants dans le réseau. Une des solutions les plus connues pour assurer la fraicheur des données est de rajouter un nombre aléatoire frais appelé 'nonce'

dans chaque paquet. La dernière exigence est la disponibilité des données même en présence d'un ou plusieurs nœuds malicieux dans les RCSFs. Ainsi, les nœuds capteurs doivent impérativement garantir la disponibilité de leurs ressources et services.

1.4.4 Sécurité dans les RCSFs

Vu les caractéristiques des RSCFS à savoir l'absence d'infrastructure, la communication sans fil et les ressources limitées dont ils disposent, les nœuds capteurs sont plus vulnérables aux attaques que les autres types de réseaux. Nous allons présenter un focus sur les attaques les plus connues dans les réseaux de capteurs sans fil ainsi que des solutions de sécurité proposées dans la littérature.

Vulnérabilité des protocoles de routage géographique

Comme mentionné dans la section 1.4.2, tous les protocoles de routage géographique se basent sur les identités et les informations de localisation des différents nœuds afin d'acheminer les données dans le réseau. Ainsi, l'efficacité de ces protocoles dépend fortement de la sûreté et la précision de ces données. Cependant, ces informations peuvent être altérées par des attaques telles que le Sybil, l'injection des « Bogus routes », le trou de ver, le trou noir et la retransmission sélective. Prenant l'exemple du protocole GEAR [Yu et al.] où les informations sur l'énergie résiduelle de chaque nœud sont nécessaires pour le calcul des coûts liés au routage. Si un nœud malveillant prétend avoir un niveau énergétique élevé, il peut, dans ce cas, augmenter sa chance pour recevoir le trafic (attaque Sinkhole). De plus, s'il intercepte toutes les données transmises, il peut dans ce cas refuser de transmettre tous les paquets qu'il a reçus (retransmission sélective). En outre, il peut annoncer une fausse information de localisation causant la

création d'une boucle de routages dans les réseaux basés sur les protocoles géographiques.

Vulnérabilité des protocoles de routage hiérarchique

Comme expliqué dans la section 1.4.1, la formation des clusters et la sélection des clusterheads (CH) se basent généralement sur la force du signal reçu au niveau de chaque nœud. Prenant le cas du protocole LEACH [Heinzelman et al. (a)], Si un nœud malveillant diffuse un message « HELLO » avec une grande puissance de transmission, il pourrait faire croire aux nœuds qu'il est un voisin proche. Dans ce cas, tous les nœuds le choisissent comme étant un CH, via lequel, ils transmettent ses données vers la station de base. Cette attaque est appelée « Hello Flooding » [Magotra et Kumar (2014)]. Le nœud malveillant peut encore utiliser la même technique pour lancer l'attaque « retransmission sélective » [de Meulenaer et al. (2008)] et donc filtrer les données reçues en laissant passer quelques-unes. Il peut de plus lancer l'attaque Sybil [Chen et al. (2010b)] où des identifiants multi-nœuds sont générés et revendiqués afin qu'il soit réélu encore une fois comme un Cluster-Head.

Vulnérabilité des protocoles de routage centrés données

Rappelons que le routage centré sur les données est basé sur le principe requête/intérêt et les techniques d'inondation. Cela rend considérablement difficile à un adversaire d'intercepter les intérêts ou d'empêcher leur diffusion dans le réseau. Néanmoins, une fois les nœuds sources commencent à envoyer les données, plusieurs attaques pourraient parvenir. Prenant par exemple l'attaque déni de service [Wood] dans le routage DD. Avec un simple message de renforcement négatif [Intanagonwiwat et al.], un nœud malveillant peut bloquer un canal de communication sans fil entre la source et la station de base. Ainsi, le débit

de transmission baisse et plusieurs données sont rendues inaccessibles. Un nœud malveillant peut également usurper l'identité du nœud Sink et récupérer par la suite toutes les données envoyées par les nœuds capteurs. On parle dans ce cas d'une attaque de clonage. Dans le tableau 1.3, nous présentons un récapitulatif des différentes attaques possibles selon le type de routage. De telles attaques empêchent les nœuds capteurs d'acheminer correctement les données dans le réseau et d'assurer la fiabilité et la qualité de leur service. De ce fait, des mécanismes de sécurité doivent nécessairement être appliqués aux réseaux de capteurs pour contrer ces attaques et assurer notamment le bon fonctionnement du routage.

TABLE 1.3 – *Récaputilatif sur les différentes attaques*

Protocoles de routage	**Menaces**
Géographique	Transmission sélective, Sybil
Centré sur les données	Transmission sélective, trou noir, trou de ver, Sybil, DoS
Hiérarchique	Inondation par " Hello ", Transmission sélective

1.4.5 Solutions de sécurité au niveau du routage

Plusieurs mécanismes de sécurité pourraient être adoptés par des protocles de routage afin de renforcer la sécurité au niveau de la couche réseau [Wood et al.,Wang et al.,Kaissi et al.]. Certains d'entre eux sécurisent le routage en empêchant tout type d'attaque dans le réseau et ils sont considérés donc comme des protocles totalement sécurisés. Alors que d'autres fournissent une seule couche de sécurité supplémentaire visant à contrer un ou quelques types d'attaques . Concernant les protocoles de routage hiérarchique, on trouve DAWWSEN [Kaissi et al.] qui a été conçu pour éviter l'attaque trou noir. L'idée consiste à construire un arbre hiérarchique géré par le nœud Sink. Cet arbre, formant les identifiants des différents nœuds, permet de faciliter le routage et de remplacer les données géographiques. DAWWSEN a fait preuve de son efficacité de contrer les attaques de type trou noir. Sec-LEACH [Oliveira et al.], un

autre protocole de sécurité hiérarchique basé sur LEACH a été proposé par Oliveira. Il consiste à adapter un schéma aléatoire de pré-distribution de clés [Eschenauer et Gligor] au routage hiérarchique. L'idée est de générer un grand ensemble de clés 'S' (S = (kid1, key1), (kid2, key2),....) et de désigner un identifiant unique à chaque clé. Par la suite, chaque nœud 'n_i' garde en mémoire 'm' clés choisies aléatoirement ($k_{i1}, ...k_{im}$) formant ainsi sa trousse de clés. Le nombre total de clés dans S est choisi de telle sorte que deux trousses de clé de taille 'm' auront une certaine probabilité p d'avoir au moins une clé en commun. Cette clé sera par la suite une clé symétrique partagée entre le nœud, le cluster-Head et le Sink. Concernant le routage centré sur les données, on trouve par exemple le protocole SDD [Wang et al.] qui présente une extension du protocole DD. Il repose sur des mécanismes cryptographiques pour réaliser la sécurité pendant chaque phase de routage. En effet, une version modifiée de TESLA [Liu et Ning (2004)], à l'aide d'une chaîne de clés à sens unique, a été utilisée pendant la phase d'inondation de l'intérêt afin de vérifier l'origine de tous les intérêts circulant dans le réseau (Sink) et assurer leurs intégrités. Deux champs doivent également être inclus avec les données (un champ de signature et un champ d'identité de chaque nœud source) durant la phase de propagation des données pour faire preuve de l'identité de l'expéditeur et de garantir l'intégrité des données propagées dans les réseaux. SDD a montré une bonne résistance contre tout type d'attaque au niveau de la couche 3 sauf les attaques actives menées par des nœuds malicieux lors de l'agrégation des données dans les RCSFs. Perrig et al. [Perrig et al.] ont proposé, de leur côté, une version sécurisée de SPIN appelée SPINS (Security Protocols for Sensor Networks). Elle est basée sur les deux protocles : SNEP (Sensor Network Encryption Protocol) et μTESLA [μTESLA]. Le premier permet d'assurer la confidentialité

et l'authentification des messages, tandis que le deuxième consiste à assurer « l'authentification broadcast ». Afin de prouver l'intégrité des paquets, SPINS a inclus un champ de sécurité basé sur le code MAC (Message Authentication Code). Cette version a été améliorée par Debao et al. [X. Debao et Ying] en proposant un nouveau protocole de routage appelé SSPIN (Secure Sensor Protocol for Information via Negotiation). Similairement au routage SPIN, SPPIN propose trois types de messages : 'ADV', 'REQ' et 'DATA'. Quand un nœud diffuse un ADV, un REQ ou des données, il calcule le MAC correspondant et l'ajoute au message à diffuser. Cela permet aux nœuds recevant les données de vérifier l'intégrité de ces dernières. Debao et al. ont [X. Debao et Ying] prouvé l'efficacité du protocole de routage SSPIN qui reste toutefois dépendante de l'algorithme MAC utilisé. Quant au routage géographique, plusieurs mécanismes de sécurité ont été proposés. Parmi ces mécanismes, on cite les protocoles SIGF-2 [Wood et al.], TRANS [Tanachaiwiwat et al.] et SGEAR [Wu]. SIGF-2 [Wood et al.] forme un ensemble de protocoles utilisant des approches non-déterministes pour assurer un routage sécurisé dans le RCSFs. Il suppose que chaque nœud dans le réseau peut accéder à toutes les informations de la localisation géographique de ses voisins (voisinage à 2 sauts). Le routage s'effectue en choisissant aléatoirement le prochain saut parmi les nœuds voisins les plus proches de la destination. Avant cela, chaque nœud attribue un niveau de confiance à ses voisins selon les critères suivants : le nombre d'acheminement réussi, le délai de délivrance des données, la mobilité et le nombre de massage envoyé. Une fois les valeurs de confiance sont déterminées, elles seront comparées à un seuil S. Tout nœud, ayant une valeur inférieure à S, sera rejeté de la liste des candidats pour le routage. Un autre protocole, appelé TRANS, a été proposé par Tanachaiwiwat et al.[Tanachaiwiwat et al.]. Il se base sur

un système de gestion de confiance (local) pour éviter les endroits non-sûrs et sélectionner les chemins les plus sécurisés. Il propose de plus une couche supplémentaire de sécurité qui peut être adaptée à n'importe quel protocole de routage géographique. Le même concept d'acheminent utilisé dans SIGF-v2, TRANS(Trust Routing for location-Aware sensor NetworkS) route les données en se basant sur des valeurs de confiance calculées en fonction de l'historique des échanges avec des nœuds voisins. Sauf qu'avec TRANS, les nœuds voisins les plus confiants sont capables de déchiffrer toutes les requêtes envoyées par l'expéditeur. De plus, TRANS rajoute des systèmes complémentaires pour renforcer la sécurité : deux systèmes pour la découverte et la localisation des nœuds malveillants (ETS et one-shot) et deux autres systèmes d'isolement et révocation (inondations et Blacklist Blacklist intégré).

Conclusion du chapitre

Les ressources limitées des RCSFs sont une contrainte à surmonter pour contrer ces attaques à moindre coût. D'un point de vue énergétique, il est plus judicieux d'utiliser des mécanismes cryptographiques symétriques, nécessitant moins de ressources, que des mécanismes cryptographiques à clé publique qui sont beaucoup plus complexes et énergivores. Pour cette raison, plusieurs protocoles de routage sécurisé ont implémenté des mécanismes symétriques et parfois hybrides afin d'économiser de l'énergie tout en assurant un niveau de sécurité efficace, sûr et complet.

Routage dans les réseaux de capteurs sans fil

2

Sommaire

Dans le chapitre précédent, nous avons fait un état de l'art sur les algorithmes de routage dans les réseaux de capteurs sans fil. Nous avons également réalisé une étude comparative et une classification des différents algorithmes proposés dans la littérature. Ce travail nous a permis non seulement de comprendre le principe de fonctionnement des algorithmes de routage mais aussi de capitaliser nos connaissances et d'avoir une vision critique sur l'ensemble des algorithmes. Cela nous a permis également d'identifier les différentes solutions existantes qui répondent à des problématiques que nous aborderons. Rappelons-nous la problématique, qui a été évoquée dans le premier chapitre,

concernant le routage des informations dans le réseau. Elle est liée à diverses contraintes dans la gestion des routes, le passage à l'échelle et la gestion d'énergie dans le réseau. Pour remédier à ce type de problème, plusieurs méthodes ont été proposées dans la littérature dont leur but est de concevoir des algorithmes de routage adaptés au réseau [Dehni et al. (2005),Mugwaneza et al. (1990)]. Cette adaptation consiste à mieux optimiser les ressources dans le réseau sans dégrader ses performances. Même avec la standardisation de plusieurs technologies de communication sans fil, ces problématiques persistent toujours. Parmi les principaux standards de communication sans fil (basés sur la norme IEEE 802.15.4), on trouve les standards ZigBee [ZigBee], WirelessHART [Chen et al. (2010a)] et ISA 100.11a[ISA]. ZigBee est considéré comme le premier standard industriel, proposé dans les RCSFs, vers 2004 [Culter (2005)]. Son coût, sa faible consommation et son évolutivité lui ont permis de garder une place importante sur le marché des technologies de communication sans fil. On trouve également le standard WirelessHART, basé sur le protocole HART (Highway Addressable Remote Transducer) [HART], qui a été proposé en 2007 afin de répondre aux besoins du monde industriel et ses exigences (sécurité, intégrité, disponibilité, autonomie, coût d'investissement ou de maintenance). Le standard ISA 100.11a 5 (Wireless Systems for Industrial Automation), apparu en 2007, est classé parmi les autres technologies de communication sans fil. Il est plus adapté aux applications non-critiques [Silva (2014)] telles que la surveillance du niveau d'oxygène, le contrôle et la surveillance des pipelines, etc. Il assure également l'interopérabilité des équipements (technologie de communication sans fil)contrairement aux autres technologies. Tous ces standards doivent impérativement respecter les contraintes temps réel telles que les contraintes applicatives, les contraintes de contrôle et les

contraintes de surveillance. En effet, lors d'une détection d'un événement particulier, une réaction doit être faite le plus rapidement possible. Tout standard doit également répondre aux autres exigences à savoir la flexibilité de la topologie, la robustesse de la propagation radio, la sécurité des données, etc. Dans ce chapitre, nous allons commencer par une présentation du standard ZigBee. Par la suite, nous allons étudier les différents protocoles de routage proposés par ce standard ainsi qu'une analyse de performance et une étude comparative par simulation. La dernière partie de ce chapitre sera consacrée à l'étude et l'analyse des performances d'un nouveau protocole de routage hiérarchique, que nous avons proposé, appelé ZBR-M (ZigBee Routing protocol-Modified).

2.1 La technologie ZigBee

L'alliance ZigBee, formée par un consortium d'entreprises, a proposé un standard dédié aux réseaux de capteurs sans fil à basse consommation d'énergie et à faible débit. Ce standard s'appuie sur toutes les couches de la norme IEEE 802.15.4 [IEEE802]. Il rajoute à cette dernière plusieurs autres fonctionnalités que nous allons présenter dans cette section. Allant de la couche réseau à la couche applicative, ZigBee propose ses propres protocoles en mettant en place une architecture multi-pile (cf. Figure 2.1).

Dans cette section, nous allons présenter d'abord les différentes couches de la norme IEEE 802.15.4 (couche physique et couche MAC), constituant la base de nombreux réseaux de capteurs sans fil et nous allons décrire par la suite, en détail, le standard ZigBee, l'une des technologies les plus prometteuses dans les réseaux de capteurs sans fil.

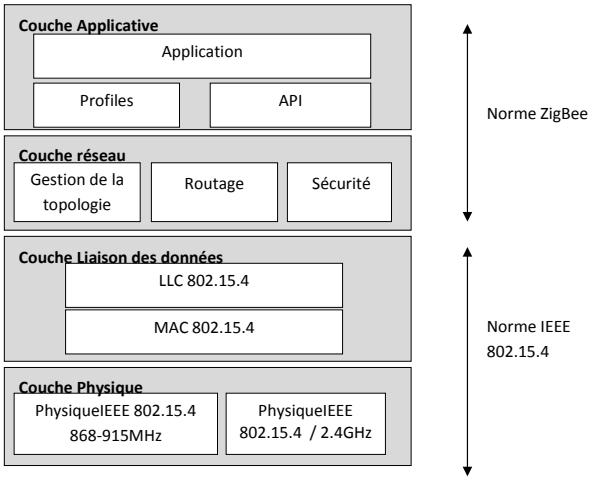

Figure 2.1 – *Pile protocolaire du standard ZigBee*

2.1.1 La norme IEEE 802.15.4

La norme IEEE 802.15.4 [Gutierrez et al. (2003)] décrit les couches basses, physique et liaison de données, dans les réseaux de capteurs sans

fil à basse consommation d'énergie. Ainsi, il est indispensable de décrire le rôle de chacune dans la chaîne de la communication sans fil.

Couche physique de l'IEEE 802.15.4

La couche physique joue un rôle important dans la gestion du support physique sur lequel toutes les transmissions de données sont faites. Elle définit notamment les techniques avec lesquelles les bits sont transmis et transformés en signaux analogiques et inversement. Parmi les principales fonctionnalités de la couche physique, nous avons l'activation et la désactivation du module radio. En effet, trois états sont définis pour un module radio : transmission, réception et sommeil. Le passage d'un état à un autre est géré par la couche physique et piloté par la couche MAC. La sélection du canal de communication est aussi l'une des fonctionnalités de la couche physique lors d'une demande de la couche MAC. De plus, la couche physique vérifie l'occupation du médium. Elle connait l'état du canal radio permettant la détection des collisions par la technique CSMA/CA de la couche supérieure (MAC). Elle indique aussi l'indicateur de la qualité d'un lien caractérisant chaque canal de transmission selon le taux d'erreurs et la vulnérabilité aux perturbations ou aux obstacles.

Couche liaison de données d'IEEE 802.15.4

Avec plus de fonctionnalités que la couche physique, la couche liaison de données de l'IEEE 802.15.4 gère principalement l'accès au médium, l'acquittement des trames et l'association des nœuds. Elle est composée de deux sous-couches : La première gère le contrôle de la liaison logique LLC (Logical Link Control) et la deuxième gère le contrôle d'accès au support MAC (Medium Access Control). La sous-couche LLC a pour rôle de contrôler la liaison logique et construire les trames de données envoyées sur le canal de communication alors que la sous-couche MAC (Medium

Acces Control), comme son nom l'indique, a pour rôle de gérer l'accès au canal avec la technique CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) mais elle permet également de gérer les « beacons » et les GTS (Guaranteed Time Slots), pour la synchronisation entre les coordinateurs et les nœuds associés (Voir section 2.1.2).

- Les trames dans IEEE 802.15.4 : Toutes les données au niveau de cette couche sont échangées sous forme de trames. Afin d'assurer correctement ces échanges, le standard IEEE 802.15.4 propose quatre types de trames : les trames « Beacon », les trames de donnée, les trames d'acquittement et les trames de contrôle : les trames Beacon sont envoyées uniquement par les nœuds coordinateurs servant à administrer le réseau. Les trames de données servent au transfert des données utiles entre les nœuds. Les trames d'acquittement servent à notifier la bonne réception des autres types de trames. Les trames de contrôle servent à effectuer des demandes spécifiques telles que l'association au réseau.

Tous les types de trames ont la même structure de données, (Figure 2.2), qui reste flexible aux types d'applications.

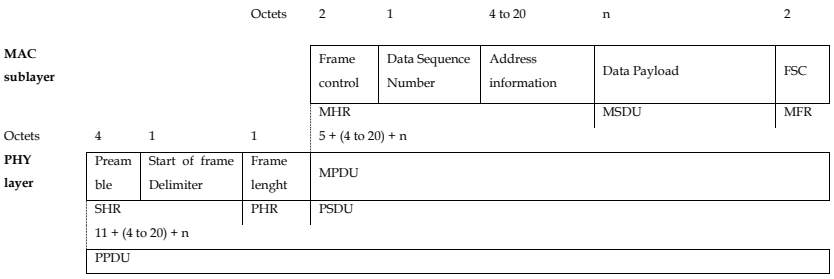

FIGURE 2.2 – *Structure d'une trame de données ZigBee*

Où : SHR : En-tête de synchronisation, PHR : En-tête physique, MHR : MAC HeadeR, MSDU : MAC Service Data Unit, MFR : MAC FooteR.

- Modes de fonctionnement dans IEEE 802.15.4 : Plusieurs techniques sont adoptées pour assurer un bon échange de données entre les nœuds

capteurs. Ces techniques définissent les modes de communication avec lesquels tous les nœuds peuvent accéder au média et partager le canal de communication physique. Deux principaux modes de communication sont définis par le standard IEEE 802.15.4. (Figure 2.3)

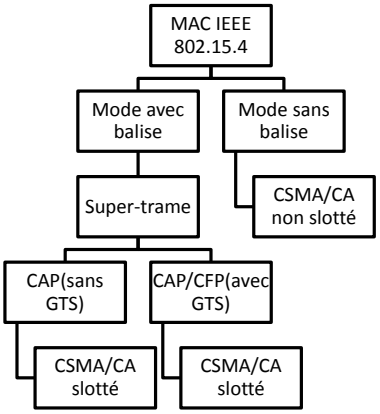

FIGURE 2.3 – *Modes de fonctionnement dans IEEE 802.15.4*

Le premier mode, appelé mode non balisé ou sans Balise, ne garantit aucune synchronisation entre les nœuds du réseau vu l'absence des trames balises. De ce fait, les nœuds doivent se mettre à une écoute permanente ou périodique du canal radio ce qui caractérise ce mode par une forte consommation d'énergie. Et afin d'éviter les collisions, ce mode est basé sur la technique CSMA/CA (non slotté). Le deuxième mode est le mode le plus important dans le standard vu ses performances en termes de débit, consommation énergétique, taux de paquets délivrés et fiabilité. Il est considéré comme un mode synchronisé puisque les nœuds doivent suivre une structure périodique appelée super-trame (Voir Figure 2.4). En effet, dès la réception d'une trame balise (beacon) envoyée par des FFDs (Full Function Device), tous les nœuds seront informés de la durée de l'espace inter-balises.

Cette structure temporelle garantie la synchronisation des nœuds communicants, la disponibilité des intervalles de temps aux nœuds

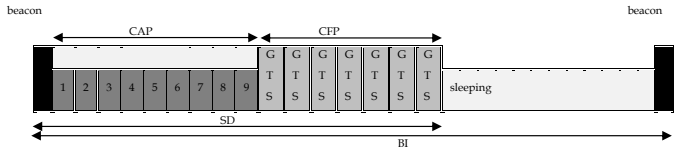

FIGURE 2.4 – *Supertrame dans IEEE 802.15.4*

communicants et la bonne distribution des temps de parole. Elle est composée de deux parties : une partie inactive (sleeping) durant laquelle aucun échange n'est possible et une partie active, composée de 16 slots, pour l'échange des données entre les nœuds capteurs. Cette partie est composée de deux sous-parties : la CAP (Contention access period), dans laquelle les accès au médium suivent le protocole CSMA/CA slotté et la CFP (Contention free period) dans laquelle un GTS (Guaranteed Time Slots) est attribué à un seul nœud pour une transmission (t-GTS) ou pour une réception (r-GTS).

2.1.2 Le standard Zigbee

Le standard ZigBee s'appuie sur l'utilisation des couches basses du standard IEEE 802.15.4. Il propose ainsi l'utilisation de sa propre couche réseau (couches hautes) qui joue un rôle très important dans la pile protocolaire. Cette couche fournit plusieurs fonctionnalités à savoir l'allocation des adresses logiques, la création de la topologie, la gestion des tables de routage ainsi que la transmission et le routage de données dans le réseau. Comme expliqué dans le premier chapitre, le routage est considéré parmi les protocoles les plus consommateurs en énergie pour un nœud capteur sans fil. Avant d'introduire les protocoles de routage utilisés dans les réseaux ZigBee, nous allons présenter en premier lieu les différentes topologies qui sont définies par le standard ZigBee ainsi que les systèmes d'adressages adoptés.

Topologie du réseau

ZigBee repose sur l'utilisation de trois types de nœuds pour constituer un réseau :

- Un coordinateur ZigBee qui est une entité FFD (Full Function Device) contenant toute la pile protocolaire nécessaire pour la communication et le traitement d'un coordinateur réseau. Il est unique dans le réseau et il est l'origine de la création du réseau.

- Un routeur ZigBee (ZR, pour ZigBee Router) qui est une entité FFD, contenant une pile protocolaire plus légère, nécessaire pour le routage des paquets sur le réseau. Il doit s'associer au coordinateur du réseau ainsi qu'aux autres routeurs. Son rôle principal est de router les données reçues selon un protocole de routage.

- Un équipement terminal ZigBee (ZED, pour ZigBee End-Device) qui représente une entité RFD (Reduced Function Devices) disposant de moins de fonctionnalités que les entités FFD, avec les quelles, il doit s'associer. L'équipement RFD n'est pas capable de router les paquets, ce qui lui permet d'économiser plus d'énergie que l'entité FFD.

Afin de créer un réseau ZigBee, un FFD est choisi comme coordinateur. Son rôle est d'initier la formation de la topologie du réseau. Le standard ZigBee prévoit deux types de topologie : une topologie en étoile et une topologie point à point (Figure 2.5). La topologie en étoile exige forcément un nœud central dans le réseau. Il s'agit d'une entité FFD qui est en liaison directe avec toutes les autres entités du réseau. Ce nœud prend en charge le relais de tous les messages échangés entre tous les nœuds du réseau. Ainsi, un nœud central doit être disposé de plus de ressources que les autres nœuds. L'inconvénient de cette topologie est que la taille du réseau reste limitée et dépend de la portée du nœud central contrairement à la topologie point à point où tous les FFD peuvent se communiquer

directement à condition qu'ils soient à portée radio. Un cas particulier de la topologie point à point est la topologie en arbre. Dans cette topologie, un nœud ne communique qu'avec son nœud père ou ses nœuds fils. Un deuxième cas particulier de la topologie point à point est la topologie en Mesh ou appelé aussi topologie maillée. Cette topologie permet de router des messages de n'importe quel nœud à n'importe quel autre sans aucune structure hiérarchique. Elle offre la fiabilité grâce à la variété des chemins à emprunter lors du routage.

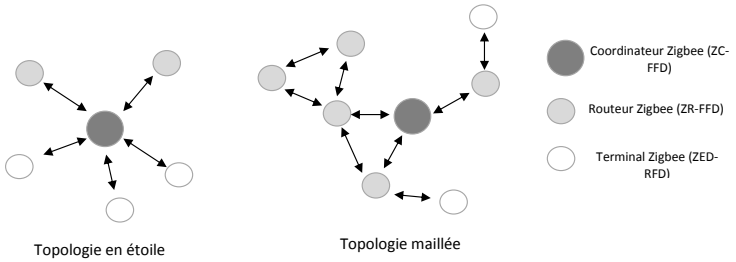

FIGURE 2.5 – *topologies du réseau ZigBee*

Adressage

L'allocation des adresses est une étape indispensable avant l'échange des données dans un réseau de capteurs sans fil. Dans les réseaux ZigBee, l'attribution des adresses s'effectue par le coordinateur du réseau. Deux mécanismes peuvent être appliqués pour attribuer les adresses logiques aux différents nœuds du réseau. Le premier mécanisme consiste à allouer les adresses d'une façon aléatoire et unique. Le coordinateur vérifie, avant l'attribution, que chaque adresse ne soit pas présente dans aucune entrée de sa table NIB (Network layer Information Base). Dans cette table, toutes les adresses déjà attribuées sont sauvegardées au fur et à mesure que l'affectation des adresses se fait. Le deuxième mécanisme consiste à attribuer hiérarchiquement et d'une manière décentralisée les adresses tout en garantissant l'unicité. Il est appelé DAAM (Distributed Address

Allocation Mechanism)[Zig (2005)]. Le coordinateur du réseau fixe trois valeurs globales pour l'allocation hiérarchique des adresses (Cm, Rm, Lm). Cm correspond au nombre maximal de fils par parent, Rm désigne le nombre maximal de routeurs par parent et Lm représente la profondeur maximale du réseau. En fonction de ces trois valeurs, chaque parent, à une profondeur 'd', calcule le nombre total des nœuds descendants qui correspond à la taille d'un sous-bloc d'adresses pouvant être allouées aux nœuds. Pour cela, il détermine la valeur Cskip(d) représentée par l'équation (2.1). Une fois le Cksip calculé, chaque nœud fils détermine son adresse en appliquant l'équation (2.2), selon son type ZED ou ZR :

$$Cskip(d) = \begin{cases} 1 + Cm(Lm - d - 1) & Rm = 1 \\ \frac{1+Cm-Rm-Cm*Rm^{Lm-d-1}}{1-Rm} & sinon \end{cases} \tag{2.1}$$

Les adresses sont allouées selon le type de la station qui s'associe :

$$\begin{cases} AdrZR = AdrP + 1 + nbR * Cskip(d) \\ AdrZED = AdrP + Rm * Cskip(d) + n \end{cases} \tag{2.2}$$

où AdrZR désigne l'adresse allouée pour un nouveau routeur, AdrP l'adresse du père, AdrZED l'adresse allouée pour un nœud terminal, nbR le nombre actuel de routeurs fils, 'n' est un entier supérieur à 1 incrémenté suite à chaque nouvelle association d'un nœud $(1 \leqslant n \leqslant (Cm - Rm))$ et 'd' représente la profondeur.

Un des avantages de ce système d'adressage est la manière avec laquelle les adresses sont attribuées. Il s'agit d'une manière distribuée où chaque nœud peut facilement déterminer son adresse, en connaissant seulement les trois paramètres globaux (CM, RM, Lm). Cela permet de minimiser le nombre de messages de contrôle échangés et réduire le coût énergétique lié au protocole de routage. De plus, ce système d'adressage arborescent permet un routage automatique sans avoir besoin de déterminer la route auparavant.

Protocoles de routage dans les réseaux ZigBee

Dépendamment de la topologie du réseau, ZigBee propose deux principaux protocoles de routage pour acheminer les données d'une source à une destination du réseau. S'il s'agit d'une topologie en arbre, dans ce cas, le routage hiérarchique sera adopté, sinon une version allégée du protocole AODV [Perkins et al. (2003)] sera utilisée où l'établissement d'une route se fait à la demande d'un nœud émetteur.

- Algorithme de routage à la demande AODV :

Par définition, AODV est un protocole de routage réactif qui reste silencieux jusqu'à ce qu'une nouvelle connexion soit nécessaire. Son avantage est de réduire le nombre de diffusions de messages par inondation, et cela en créant les routes lors d'un besoin. Ce protocole est basé sur l'utilisation de deux mécanismes : la « Découverte de route » et la « Maintenance de route ». A cause de la mobilité des nœuds dans les réseaux, les routes changent fréquemment ce qui fait que les routes, maintenues par certains nœuds, deviennent obsolètes. Pour cela, l'AODV utilise les principes des numéros de séquence afin de maintenir la pertinence des informations de routage et permettre l'utilisation des routes les plus fraiches. Le mécanisme «Découverte de route » est exécuté par les nœuds sources dans un des trois cas suivants : une nouvelle destination, une expiration de la durée de vie d'une route vers une destination ou une défaillance d'un chemin vers une destination. En effet, la source commence la diffusion d'une requête de route appelée RREQ (Route REQuest) et elle attend jusqu'à la réception d'un message de type RREP (Route Reply). Si elle ne reçoit aucun message RREP au bout d'un délai d'attente égale à NET_TRANVERSAL_TIME, elle rediffuse, dans ce cas, le message RREQ et attend une période supérieure à la première. En l'absence d'une réponse RREP, le processus peut être répété jusqu'à

RREQ_RRTRIES fois. Après avoir diffusé le RREQ, le nœud destinataire et tout nœud recevant la requête, qui possède un chemin vers la destination avec un numéro de séquence supérieur ou égal à celui reçu (RREQ), émet un RREP au nœud source. Quant au reste des nœuds recevant le RREQ, ils rediffusent le paquet RREQ (par inondation). Avant la rediffusion de RREQ, les nœuds intermédiaires incrémentent le nombre de sauts, sauvegardent les identifiants des nœuds à partir desquels les premières copies des requêtes sont reçues et gardent, également, en mémoire les adresses des nœuds sources. Toutes ces informations sont utilisées pour construire le chemin inverse et acheminer le RREP jusqu'aux nœuds sources.

Une fois la source reçoit les paquets RREP, elle peut commencer à émettre des paquets de données vers la destination. Si, ultérieurement, la source reçoit un RREP contenant un numéro de séquence supérieur ou égale à la valeur initiale avec un nombre de sauts plus petits, elle mettra à jour, dans ce cas, son information de routage vers cette destination et commencera à utiliser la meilleure route.

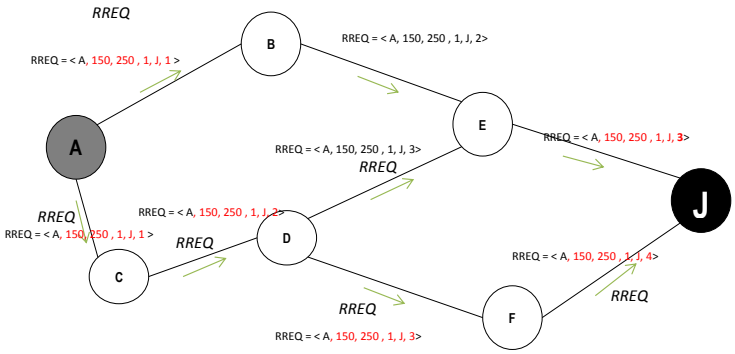

FIGURE 2.6 – *Exemple d'illustration de l'algorithme de routage à la demande AODV*

Dans la figure 2.6, un exemple de routage AODV est donné. La source 'A' désire envoyer un message vers la destination 'J'. Pour cela, elle entame la procédure de routage par une diffusion d'une requête RREQ. Cette

dernière contient cinq champs : l'ID de la source, le Broadcast ID, l'ID de la destination, numéro de séquence et le nombre e sauts qui s'incrémente au fur et à mesure. A la réception des requêtes RREQ, le nœud 'J' choisit celle qui contient le nombre minimum de sauts afin de construire le chemin inverse. Ainsi, il prend le chemin (E,B,A) pour atteindre la destination.

- Algorithme de routage hiérarchique :

Pour une topologie en arbre, le coordinateur ZigBee est le responsable du démarrage du réseau. Le réseau s'étend grâce aux routeurs ZigBee. Les données et les messages de contrôle suivent une stratégie de routage hiérarchique ainsi que le mécanisme d'allocation d'adresses DAAM (Voir Section II.b.ii). Dans ce cas, le routage est simplifié puisqu'un nœud est capable de calculer localement le chemin complet sur l'arbre pour atteindre toute destination. Ce chemin est construit comme suit (algorithme 1) :

Algorithme 1 : Description of ZBR protocol

```
if myAdd <DestAdd=myAdd+Cskip(d-1) then
    if DestAdd > myAdd+Rm*Cskip(d) then
        NextHop = DestAdd;
    else
        NextHop = myAdd + 1 + [(DestAdd − (myAdd + 1))/Cskip(d)] ∗ Cskip(d);
else
    NextHop = myParentAdd
```

Nous désignons par 'myAdd' l'adresse du routeur qui cherche à envoyer le paquet selon le routage hiérarchique, 'd' sa profondeur, 'AddP' l'adresse de son père, DestAdd l'adresse de la destination finale du paquet et NextAdd l'adresse du prochain saut. Pour commencer, le nœud source vérifie si le destinataire est l'un de ses descendants en déterminant si l'adresse destination est comprise entre AdR et AdR + Cskip(d - 1). Si ce n'est pas le cas, la source envoie les données à son père. Le nœud père, de même, envoie les données à son supérieur hiérarchique

et ainsi de suite jusqu'à arriver à un nœud parent du nœud destinataire. Le sens descendant de l'information est assuré grâce une technique de détermination du successeur basée sur l'adresse du routeur ZigBee, sa profondeur et l'adresse du nœud. En effet, pour trouver l'adresse du nœud fils vers lequel il va router le paquet, le routeur applique la formule suivante (2.3) :

$$Dest = AdR + 1 + [\frac{DestAdd - (AdR + 1)}{Cskip(d)}] * Cskip(d) \qquad (2.3)$$

Prenons le cas où Lm = 3, Rm = 2 et Cm = 4, les valeurs de Cskip selon la profondeur 'd' sont données sur le tableau 2.1 :

TABLE 2.1 – *Nombre d associées en fonction de la profondeur*

Profondeur 'd'	Nombre d'adresses associées
0	29
1	13
2	5
3	1

Comme illustré sur la figure 2.7, la source, nœud 19, désire envoyer un message vers la destination, nœud 21. D'abord, elle envoie le message à son père, nœud 15. Ce dernier, considéré comme un routeur ZigBee, trouve que l'adresse de la destination finale n'appartient à sa plage d'adresse [15-19]. Il transmet alors le paquet à son père, le nœud 14 qui, à son tour, trouve que la destination appartient pas à la plage d'adresses [14 -26]. Dans ce cas, le nœud 14 doit déterminer si la destination est l'un de ses fils, ou si le paquet doit être envoyé à un routeur fils. Donc, le nœud 14 envoie les données vers le routeur 20 qui l'achemine vers la destination 21.

2.2 Etude comparative entre les protocoles de routage AODV et ZBR

Après avoir présenté les principaux protocoles de routage utilisés dans le réseau ZigBee, il est nécessaire de comparer les performances

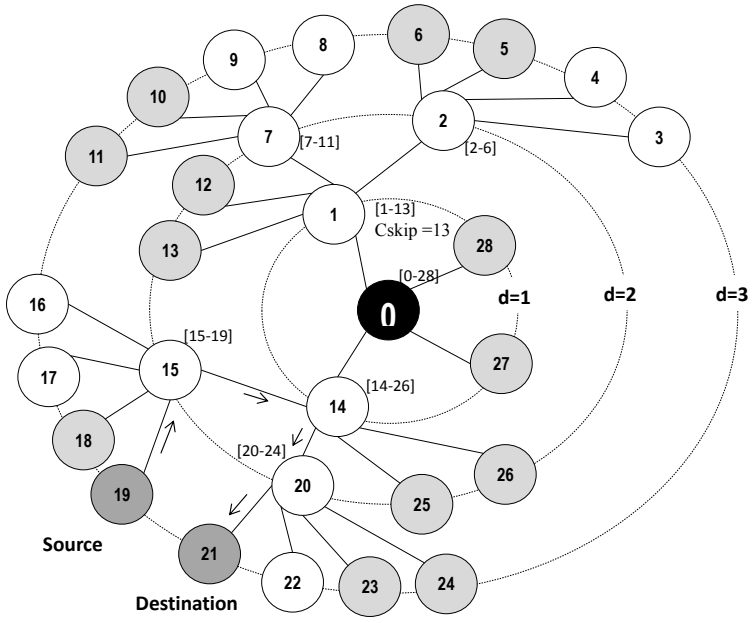

FIGURE 2.7 – *Routage hiérarchique dans les réseaux ZigBee*

de ces protocoles afin d'identifier leurs caractéristiques ainsi que leurs déficiences. Pour cela, nous avons conduit des simulations des deux protocoles AODV et ZBR. Une étude synthétique a été également faite dans le but d'analyser et améliorer les performances de ces protocoles [Kasraoui et al. (2012)]. Le plus grand réseau considéré, dans notre simulation, est constitué d'un seul coordinateur ZigBee (ZC) et 200 routeurs ZigBee (ZR). Le réseau est totalement connecté et chaque nœud n'entend que ses voisins directs. La profondeur maximale (Lm) est égale à 6 et le nombre maximal de fils par parent (Cm) est égal à 7. Une dizaine de scénarios ont été simulés. Au début, il s'agissait d'étendre le réseau en augmentant la profondeur de l'arbre et par la suite, nous avons étudié l'envoi des données vers le Sink, le nœud « 0 ». A la fin des simulations, nous calculons les moyennes des métriques de performance à savoir le délai de bout en bout, le taux de paquets délivrés, et la consommation d'énergie.

2.2.1 Etude de délai de bout en bout

Par définition, le délai de bout en bout correspond à la durée moyenne mise par les paquets pour passer des couches applicatives de la source à celles de la destination.

$$Délai = \frac{\sum_{Pi}(T_r(i) - T_e(i))}{|P_{émis}|} \tag{2.4}$$

Où : Tr(i) : l'instant de la réception du paquet Pi Te(i) : l'instant de l'émission du paquet Pi

Par rapport à ce critère, on constate que le comportement du protocole AODV, en termes de délai, est similaire à celui du routage hiérarchique seulement pour les nœuds de profondeur égale à 1 (voir figure 2.8). Pour le reste des profondeurs, nous remarquons une petite lenteur du protocole AODV par rapport à son homologue hiérarchique. Cela est dû au processus de la découverte de route qui est une phase nécessaire pour l'établissement d'un chemin où un RREQ (Route Request) et un RREP (Route Reply) doivent être échangés avant de transmettre les paquets de données.

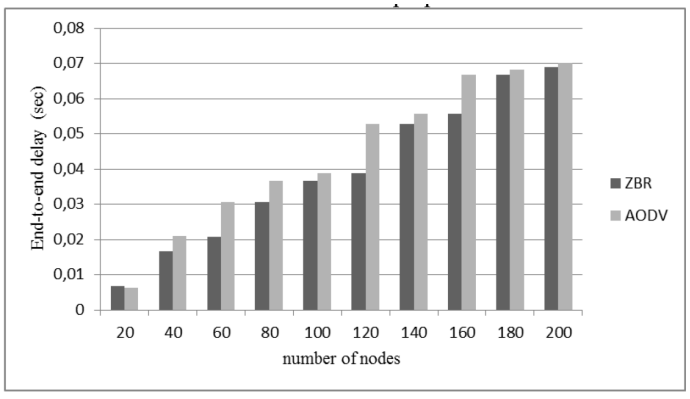

FIGURE 2.8 – *Délai de bout en bout en fonction de la profondeur de l'arbre*

En conclusion, pour un réseau de grande taille, le routage hiérarchique de base est plus performant en termes de délais de bout en bout vu que le plus court chemin, qui nous emmène vers le Sink, est celui qui suit

les liens fils-père de l'arbre, contrairement au protocole AODV qui doit découvrir les routes avant d'envoyer les données.

2.2.2 Taux de paquets délivrés (TPD)

Dans cette partie, nous étudions les performances de deux protocoles selon le taux de paquets délivrés. Ce taux correspond au rapport entre le nombre de paquets de données reçus par les destinations, pendant une période T, et le nombre de ceux qui ont été émis par les sources pendant la même période.

$$TPD = \frac{Nbdepaquetsre\ \ç\ \ us}{Nbdepaquetsreus} \tag{2.5}$$

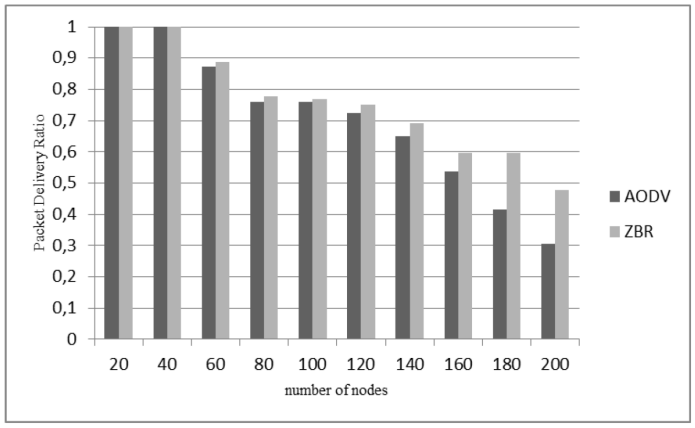

FIGURE 2.9 – *Taux de paquets délivrés en fonction de la profondeur de l'arbre*

La figure 2.9 montre que le taux de paquets délivrés diminue en fonction de la taille du réseau. Cela est dû à l'augmentation de nombre des sauts, la probabilité de collision ainsi que la surcharge des files d'attente. Cette diminution est plus significative dans le cas du protocole AODV vu le nombre de messages de contrôle supplémentaires, par rapport au routage hiérarchique, permettant l'établissement des routes entre les sources et les destinations.

2.2.3 Durée de vie du réseau (lifetime)

Le critère d'évaluation dans cette section est la durée de vie du réseau. En effet, tous les nœuds dans un réseau de capteurs sont caractérisés par une limitation en termes d'énergie. Ainsi, l'efficacité énergétique d'un protocole de routage dépend fortement de l'optimisation de la durée de vie du réseau. Selon [Luo et al. (2011)], la durée de vie du réseau est définie comme étant le temps écoulé depuis son déploiement jusqu'à ce que le réseau devienne non fonctionnel. Selon plusieurs auteurs [Chang et Tassiulas (2000)][Giridhar et Kumar (2005)][Mhatre et al. (2005)], si un nœud épuise toute son énergie, le réseau serait considéré comme étant non fonctionnel. Durant cette simulation, nous gérons, à chaque période « t », un trafic de données entre deux nœuds choisis aléatoirement. Pour les deux protocoles en question, nous avons suivi l'évolution du réseau jusqu'à la mort du premier nœud.

TABLE 2.2 – *Instant de mort du premier nœud*

Protocole	ZBR	AODV
Instant de mort (sec)	84.3946	89.4024

Le tableau 2.2 montre bien que la durée de vie d'un réseau utilisant le routage hiérarchique de base est plus courte que pour un réseau utilisant le routage à la demande. Le facteur majeur de l'épuisement d'énergie revient à la nature statique du protocole de routage hiérarchique qui ne route qu'à travers des liens père-fils. Ainsi, un seul chemin est emprunté par tout le trafic ce qui épuise rapidement l'énergie résiduelle de certains nœuds par rapport à d'autres vu qu'ils sont plus sollicités. Cela fait qu'après un certain temps, le routage vers la destination n'est plus possible. Tous les nœuds descendants du nœud mort deviennent

isolés par rapport au reste du réseau. Ceci est considéré comme une des faiblesses du routage hiérarchique de base.

2.2.4 Synthèse

Les résultats obtenus par la simulation doivent être pris comme indication pertinente sur le comportement de ces deux protocoles de routage et non pas comme une représentation exacte de leurs comportements en environnement réel. Cela est dû à plusieurs contraintes de simulation à savoir la dimension du champ des nœuds, leur répartition, le nombre de nœuds, le type de trafic et le temps de simulation. D'autre part, nous avons comparé plusieurs scénarios afin de déterminer les particularités de chaque protocole de routage dans les réseaux ZigBee. Nous avons notamment constaté que le routage hiérarchique présente dans certains scénarios des défaillances, comme par exemple le cas d'une source et une destination voisines et de père diffèrent. En revanche, les résultats de simulations ont montré que le routage hiérarchique de base présente un meilleur délai et un meilleur taux de délivrance de paquets permettant ainsi une disponibilité de service indépendamment de la taille du réseau. Le routage hiérarchique génère également moins de surcharge de contrôle que le protocole de routage AODV. Par ailleurs, le routage hiérarchique est caractérisé par sa simplicité et surtout son besoin limité en ressources (en termes d'énergie, mémoire et temps de calcul). En effet, ce type de routage ne nécessite aucun échange entre les nœuds mais, il exige, en contrepartie, une topologie stable. De plus, dans une topologie en arbre, les routes ne sont pas toujours optimales vu que le routage hiérarchique utilise les liens parent-fils et ignore les liens entre les nœuds voisins. Si on reprend l'exemple montré sur la figure 7, on remarque bien que la destination, le nœud 21, se trouve à la portée de la source, le

nœud 21, mais en appliquant le routage hiérarchique, on devrait suivre plusieurs sauts pour accéder à la destination. En tout, il faut quatre sauts pour arriver à la destination au lieu d'un seul. En résumant, selon les critères de performances analysés par simulation, le routage hiérarchique présente de meilleures performances par rapport au routage AODV. Néanmoins, plusieurs problèmes de déficience énergétique ou encore des problèmes de routage ont été identifiés dans cette section. Afin de répondre à ces problématiques, nous avons proposé une amélioration du protocole de routage hiérarchique, appelé ZBR-M, qui a été simulée par OPNET [OPNET].

2.3 Proposition d'un nouveau protocole de routage ZBR-M (ZigBee Routing protocol-Modified)

Dans cette section, nous allons introduire la contribution que nous avons faite au niveau du routage ZigBee. En effet, nous avons proposé un nouveau protocole de routage afin d'améliorer le routage hiérarchique. Grâce au module de découverte des voisins de l'entité NLME (Network Layer Management Entity), proposé par le standard ZigBee, nous avons pensé à exploiter ces données pour identifier des raccourcis qui emmènent vers la destination en diminuant le nombre de sauts. Cela permet d'éviter les liens fils-parents classiques proposés par le protocole hiérarchique et d'utiliser les liens entre les nœuds voisins à un seul saut tout en garantissant l'accès à la destination.

2.3.1 Description de l'algorithme modifié ZBR-M

Le principe de notre algorithme de routage est le suivant : Un nœud émetteur vérifie tout d'abord si la destination est l'un de ses descendants. Si oui, il l'achemine selon le routage hiérarchique de base. Si ce n'est pas le cas, il envoie une requête à ses voisins à un seul saut. Chaque voisin

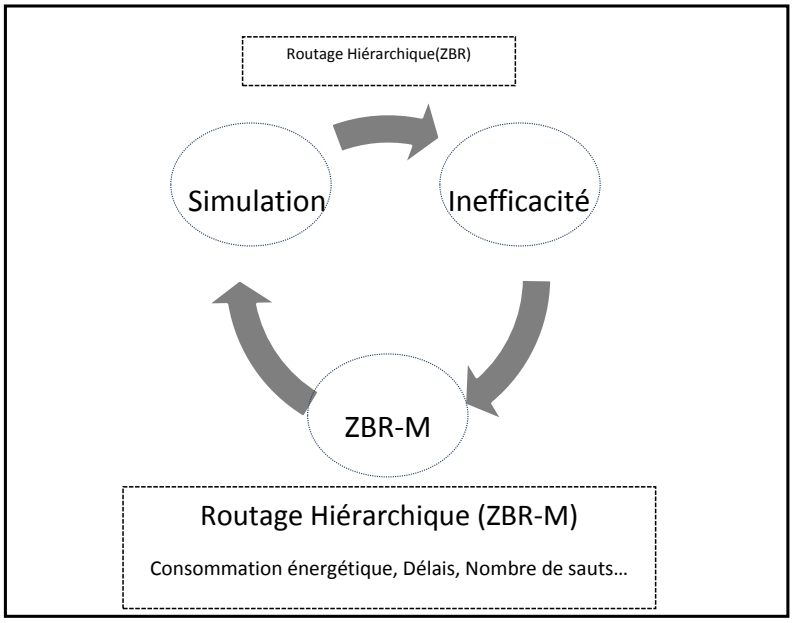

FIGURE 2.10 – *Description de l'algorithme modifié ZBR-M*

recevant le message, vérifie de même si le destinataire est l'un de ses descendants. Si oui, le voisin renvoie un accusé à l'émetteur et se charge de l'acheminement du message. Sinon, le voisin rejette le message. Au niveau du nœud émetteur, si le temporisateur expire sans rien recevoir, le message est transmis vers le père. Par ailleurs, si on applique le ZBR classique et un nœud appartenant au chemin emprunté tombe en panne, tous ses descendants ne peuvent plus acheminer leurs données vers la station de base. De plus, après avoir analysé le comportement de ce protocole, pour certains cas simulés, nous avons constaté que le paquet doit aller jusqu'au premier père commun entre l'émetteur et le destinataire pour pouvoir après descendre dans l'arbre et atteindre sa cible et cela même si les nœuds sont proches l'un de l'autre en profondeur. Ainsi, nous avons pensé à une exploration horizontale de l'arbre ce qui augmente la probabilité de trouver un chemin alternatif vers le destinataire en un minimum de sauts sans avoir obligatoirement besoin à emprunter des liens fils-père tout

en gardant le profit provenant de la simplicité du routage hiérarchique. L'algorithme proposé est le suivant :

Algorithme 2 : Description of ZBR protocol

```
if D is a descendant of node R then
    use rule given by this equation (1) to find the Next Hop
    NextHopID= myNodeID+1+
    [(FinalDstNodeID-(myNodeID+1))/Cskip(myDepth) ]
    *Cskip(myDepth)
else
    if D is a descendant of N ∈ V (x) using (2) then
    else
        Next-Hop = N
```

Étape 1 : Le nœud source vérifie si le nœud destinataire est son parent. Dans ce cas, les paquets seront remontés dans l'arbre vers le nœud parent.

Étape 2 : Le nœud source vérifie si le nœud destinataire est l'un de ses descendants. Dans ce cas, les paquets vont suivre le chemin descendant de l'arbre jusqu'à la destination.

$$myNode@ < FinalDstNode@ < myNode@ + Cskip(myDepth - 1) \quad (2.6)$$

La vérification des nœuds descendants est faite à l'aide l'inéquation(2.6). En effet, chaque nœud FFD dans le réseau gère l'attribution des adresses de tous ses descendants. Ainsi, la plage d'adresses réservée à ces nœuds est bornée entre deux variables connus par le nœud FFD.

Pour qu'un nœud ZigBee puisse trouver l'adresse du prochain saut, il utilise l'équation (2.7) :

$$NextHop@ = myNode@ + 1 + [\frac{FinalDstNode@ - (myNode@ + 1)}{Cskip(myDepth)}] * Cskip(myDepth) \quad (2.7)$$

où NextHop est l'adresse du prochain saut, Cskip (voir section 2.1), myNode@ est l'adresse du nœud routeur, myDepth est sa profondeur.

Étape 3 : Le nœud source vérifie si le nœud destinataire est l'un de ses

voisins. Dans ce cas, les paquets seront envoyés directement vers le nœud destinataire sans suivre la topologie du réseau. Cette étape est faite grâce à une table de voisins dans laquelle chaque nœud du réseau sauvegarde les adresses de tous ses voisins à un seul saut. Étape 4 : Le nœud source vérifie si le nœud destinataire est l'un des descendants de ses voisins. Pour cela, le nœud source diffuse à ses voisins une requête en mettant l'adresse en question. Tous les nœuds recevant cette requête, vérifient l'inéquation (2.6) afin de vérifier si la destination est un nœud descendant ou pas. Dans ce cas, celui qui trouve la destination dans sa plage d'adresses renvoie un message 'Reply' vers le nœud source en précisant le nombre de sauts nécessaires pour atteindre la destination. Ce dernier sert à identifier le chemin le plus court vers un nœud destinataire en choisissant le nombre minimal en cas de réception multiple de messages 'Reply'.

2.3.2 Exemple d'illustration

Prenant l'exemple d'un réseau ZigBee formé de 14 routeurs ZigBee et d'un seul coordinateur avec une topologie en arbre. Le nœud 142 est un émetteur qui cherche à transmettre des données, un message P, vers le nœud 205.

Au niveau du nœud 142, en appliquant l'algorithme ZBR-M, on trouve que le destinataire, le nœud 205, n'est pas un descendant. Dans ce cas, le nœud 142 envoie deux 'Request' vers ses nœuds voisins : Neighbors(142) = 203,204. 203 trouve que le nœud 205 est un descendant, il envoie donc un 'Reply' vers le nœud 142. 204 trouve également que le nœud 205 est un descendant, il envoie donc un 'Reply' vers le nœud 142. Dans ce cas, le nœud 142 reçoit deux 'Reply' dont les nombres de sauts est différents (2,1), il choisit alors la valeur minimale et il envoie le message P vers le nœud 204 qui est plus proche de la destination. Par ailleurs,

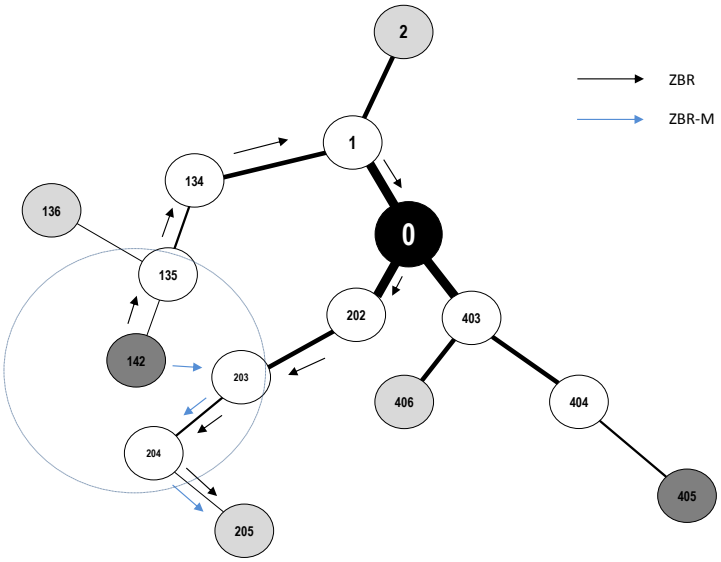

FIGURE 2.11 – *Exemple d'illustration du protocole ZBR-M*

chaque nœud routeur déclenche un temporisateur dès la diffusion d'une requête. Tout 'Reply' reçu après l'expiration de son temporisateur sera systématiquement rejeté. Les requêtes sont envoyés saut par saut pour éviter la propagation du trafic du contrôle dans des zones non concernées du réseau et donc limitation des collisions dues à l'établissement du chemin de la source vers la destination.

2.3.3 Analyse de performance du protocole ZBR-M

Dans cette section, nous présentons une étude empirique sur les performances du ZBR-M dans un réseau de communication sans fil « ZigBee ». Pour cela, nous avons entamé cette étude, par une simulation, afin de comparer les performances de notre algorithme, par rapport au protocole ZBR, selon des critères d'évaluation. Dans un premier temps, nous avons démarré la simulation avec un coordonnateur, 6 routeurs et 6 terminaux. L'ensemble des flux de données a été acheminé de différents terminaux vers le coordinateur du réseau. Dans un deuxième temps, nous avons augmenté le nombre de nœuds à 200 et nous avons analysé les pires

des cas correspondant au delais d'acheminement des donnees entre une source et une destination.

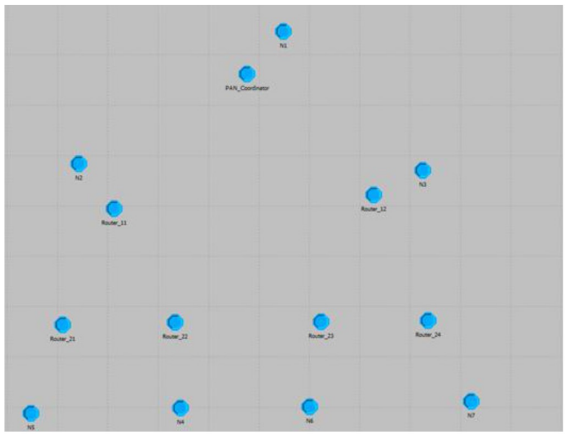

FIGURE 2.12 – *Présentation du réseau*

Au niveau de l'émission, chaque nœud dans le réseau génère un paquet de données toutes les minutes. L'envoi des données commence à un moment choisi au hasard. Les paramètres importants des simulations sont indiqués dans le tableau suivant.

TABLE 2.3 – *Paramètres de simulation*

Technologie	ZigBee
Protocole	ZBR / ZBR-M
MAC/PHY	802.15.4
Canal	canal radio
Porpagation	TwoRaYgROUND
Topologie	100*100
Nombre de noeuds	7, 200

2.3.4 Résultats de simulation

Nous avons développé le protocole de routage ZBR-M à l'aide du simulateur OPNET, tout en gardant le même réseau et les mêmes paramètres de simulation cités dans le paragraphe précédent. Cinquante simulations ont été exécutées. Pour chaque nœud choisi au hasard, le trafic de données est envoyé à partir de ce nœud vers le coordinateur du réseau. Pour les différentes simulations, nous avons commencé à travailler

sur un réseau de 7 nœuds puis sur un réseau de 200 nœuds. A la fin des simulations, on a calculé la moyenne des métriques de performance à savoir le taux de paquets délivrés, le délai de bout en bout et la consommation d'énergie. Les mesures des métriques sont montrées dans les tableaux 2.4 et 2.5 ci-dessous :

TABLE 2.4 – *Résultats de simulation pour 7 nœuds*

	ZBR	**ZBR-M**
Taux de délivrance des paquets (%)	99.87	99.88
Délai de bout en bout (ms)	8	5
Consommation énergétique (mJ)	0.449	1.003

Selon les résultats des simulations, le protocole modifié ZBR-M est plus performent que l'algorithme basique en termes de délai de bout en bout. Ce protocole permet, en exploitant les liens entre les nœuds voisins à un seul saut, de mettre fin aux retards liés au mauvais choix des routes. Néanmoins, ZBR-M induit un coût supplémentaire lié à la consommation d'énergie entrainant ainsi une réduction de la durée de vie du réseau.

TABLE 2.5 – *Résultats de simulation pour 200 nœuds*

	ZBR	**ZBR-M**
Taux de délivrance des paquets (%)	48.93	51.22
Délai de bout en bout(ms)	86.5	64.7
Consommation énergétique(mJ)	1.207	2.177

CONCLUSION DU CHAPITRE

Dans ce chapitre, nous avons pu développer des connaissances approfondies sur les réseaux de capteurs sans fil et notamment sur les réseaux ZigBee. Nous avons également étudié une panoplie de protocoles de routage proposés par le standard ZigBee. Nous avons conduit des simulations pour évaluer les performances du routage proposé par ZigBee

Alliance tout en le comparant au routage à la demande afin d'identifier les caractéristiques du routage hiérarchique ainsi que ses déficiences. Les résultats de simulations ont montré que le routage hiérarchique de base présente de meilleurs délais et taux de délivrance permettant ainsi une disponibilité de service indépendamment de la taille du réseau.

3 Sécurité à basse consommation d'énergie dans les réseaux de capteurs sans fil

Sommaire

La sécurité est considérée comme un environnement complexe surtout dans les RCSFs. Cela est dû à l'absence de la protection physique des nœuds, la limitation de ressources et l'hostilité de l'environnement de déploiement. Ces caractéristiques représentent toujours une barrière devant les RCSFs pour l'évolution et l'intégration dans l'IdO (Internet des Objets). Ce dernier, représentant le croisement d'un ensemble de technologies, nécessite des réseaux de capteurs d'être interopérables avec les autres réseaux et de contrer tout danger externe. Dans cette optique, plusieurs propositions ont été réalisées par les technologies

WirelessHART, ISA100.11.a ou ZigBee afin de rendre l'interconnexion possible et plus facile à mettre en œuvre entre les réseaux de capteurs sans fil et le réseau Internet.

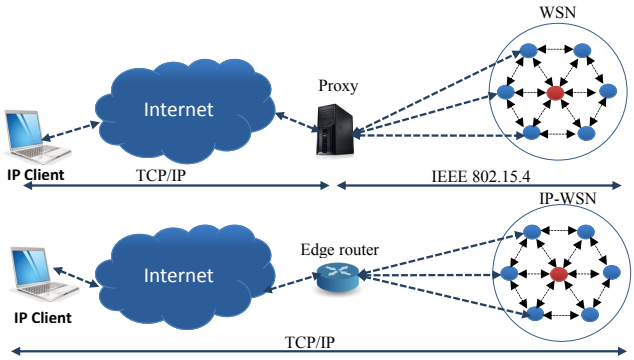

FIGURE 3.1 – *Architecture d'un RCSF basé sur l'IP*

Afin d'avoir une communication directe entre les nœuds capteurs (Sensor Node - SN) et les terminaux IPv4/6, deux architectures sont proposées [Colitti et al. (2011)][Sinniah et al. (2012)]. Comme illustré dans la figure 3.1, la première architecture consiste à mettre en place un Proxy, comme une passerelle, entre les nœuds capteurs et le réseau Internet afin de traiter toutes les données échangées. A la réception des données, en provenance de l'expéditeur, ce dernier les déchiffre, puis il les chiffre avant de les transmettre chiffrées à la destination. Cette architecture présente de nombreux inconvénients tels que l'existence d'un élément intermédiaire dans la chaîne de communication, la diminution du passage à l'échelle et la complexité de l'interopérabilité entre les RCSFs basés sur l'IP et les réseaux IP externes. La deuxième architecture, est plus évolutive et moins complexe. En effet, l'échange des données entre les réseaux de capteurs et Internet se fait par le biais d'un ou plusieurs routeurs de bordure. Dans cette architecture, la sécurité des données est assurée de bout en bout sans avoir besoin d'un Proxy.

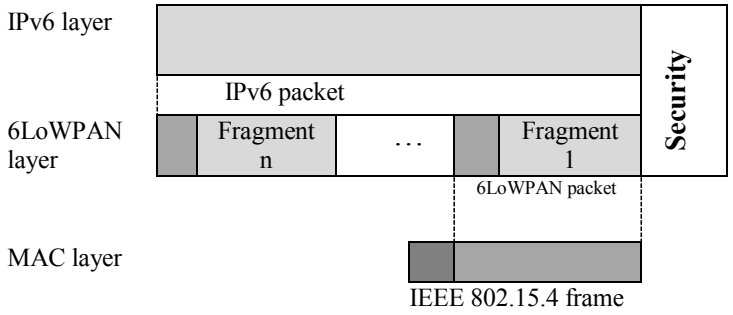

FIGURE 3.2 – *Couche d'adaptation 6LoWPAN*

Par ailleurs, le groupe IETF (Internet Engineering Task Force) a proposé une nouvelle couche d'adaptation, appelée 6LowPAN (IPv6 over Low power WPAN)(voir figure 3.2). Cette dernière, située entre la couche liaison de données et la couche réseau de la pile protocolaire, permet la transmission des paquets IPv6 via le protocole de communication IEEE 802.15.4. Cela est réalisé grâce à des mécanismes d'encapsulation et de compression permettant ainsi la fragmentation et la compression des entêtes IPv6. De plus, le groupe IETF avait comme objectif d'optimiser davantage les ressources dans les réseaux de capteurs sans fil afin d'avoir la possibilité de se connecter au réseau Internet. Outre les mécanismes d'adaptations des paquets IPv6 (compression + fragmentation), la sécurité de bout en bout est considérée comme un élément indispensable de la chaîne de communication. La sécurité de bout en bout sera donc le sujet principal dans ce chapitre. Nous avons proposé une nouvelle approche collaborative appelée CKES (Collaborative Key Exchange System) afin de sécuriser les données échangées entre les nœuds dans les RCSFs hétérogènes tout en tenant compte de leurs contraintes énergétiques. Ainsi, nous avons adapté le protocole de sécurité IPSec, conçu à la base pour sécuriser les échanges de bout en bout sur Internet, aux RCSFs de

sorte que les opérations cryptographiques énergivores soient réparties sur un ensemble de nœuds.

Dans la suite, nous allons d'abord commencer par une étude comparative afin de justifier notre choix de l'IPSec comme meilleur candidat pour la sécurité de bout en bout dans les RCSFs. Puis, nous allons présenter les résultats de simulations de l'IKEv2 après l'avoir implémenté sous NS2. Suite à la proposition d'une nouvelle approche appelé CKES, nous allons présenter une comparaison avec les résultats obtenus avec l'IKEv2 afin d'illustrer ses performances.

3.1 Choix de l'IPSec pour la sécurité de bout en bout dans les RCSFs

Nous allons commencer cette section par la présentation d'un aperçu sur les différentes approches portant sur la sécurité de bout en bout dans les réseaux de capteurs sans fil basées sur l'IP. La première initiative, a été proposée par Garanjal [Granjal et al. (2010)], appelée SIMWSN (Secure Interconnexion Model for WSN). Cette solution consiste à introduire une passerelle de sécurité entre les nœuds capteurs et les hôtes Internet (voir figure 3.3). En effet, pour chaque communication entre deux nœuds un tunnel IPSec est établi entre la passerelle et le hôte Internet, tandis que des mécanismes de sécurité basiques sont appliqués au sein du réseau de capteurs (IEEE 802.15.4). Cette solution prend en considération les règles du standard 6LoWPAN et propose une passerelle supportant les deux protocoles IPv4 et IPv6.

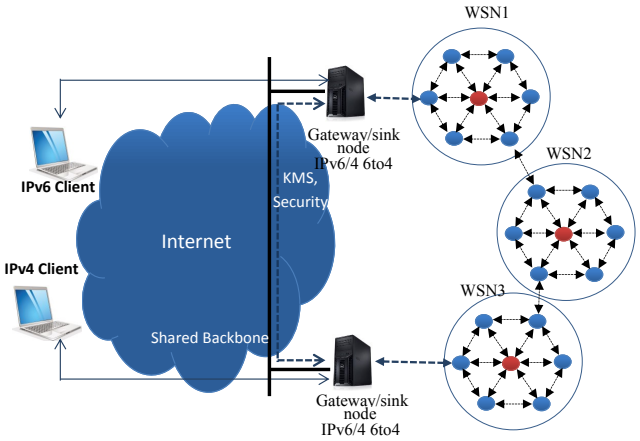

Figure 3.3 – *Architecture SIMWSN*

SIMWSN n'était qu'une proposition en phase de conception en 2010. Son implémentation a été réalisée plus tard par RAZA [Raza et al. (2011)] qui a suivi cette approche. Son idée était d'étudier et de choisir une solution efficace dans le réseau Internet afin de la déployer dans les

RCSFs. Il a implémenté une version légère de l'IPSec adaptée aux réseaux 6LoWPAN. La nouveauté de cette solution est l'utilisation des mécanismes de compression et des méthodes d'encodage pour les en-têtes AH et ESP du protocole IPSec [Hui et al.]. Par ailleurs, une implémentation d'une version simplifiée du protocole IPSec a été faite, dans [Casado et Tsigas (2009)], sur le système d'exploitation Contiki OS. Le concept de la pré-distribution des clés a été également utilisé afin d'établir des associations de sécurité entre les nœuds désirant communiquer. Cette implémentation a été réalisée avec les modes d'opération suivants : le HMAC-SHA1-96 pour le AH et l'AES-CBC pour ESP. Indépendamment de l'usage du protocole, l'IPSec nécessite entre 3.9 et 9 kilo-octets de ROM et entre 0.3 et 1.1 kilo-octets de RAM [Raza et al. (2011)]. Le même concept a été développé par Gupta [Gupta et al. (2005)] sous le nom de Sizzle. Cette solution permet une communication sécurisée basée sur le protocole SSL [Jung et al. (2009)] qui a été mise en œuvre sur toutes les passerelles. Une autre solution, plus générique, appelée SSNAIL(Security of Sensor Networks for All-IP worLd), a été proposée par W. JUNG [Jung et al. (2009)]. Ce dernier s'est inspiré du même concept que Sizzle mais aucune passerelle de sécurité n'est utilisée. Casado et al. [Casado] ont proposé la solution ContikiSec qui introduit, avec plus de flexibilité, la notion de profil en choisissant le mode le plus adéquat avec la nature de la communication. Pour récapituler, nous avons réalisé une étude comparative des différentes solutions citées auparavant (voir tableau 3.1). Cette étude vise à distinguer non seulement les points forts et faibles des différentes solutions mais aussi tous les points communs à savoir les algorithmes d'authentifications, les systèmes d'échange de clés, les tailles de clé, etc.

Selon notre étude, la proposition de Raza [Raza et al. (2011)] se trouve

TABLE 3.1 – *Protocole de sécurité de bout en bout*

	SSNAIL	**Sizzle**	**ContikiSec**	**SIMWSN**	**6LoWAN/IPSec**
Authentication	ECDSA	ECDSA	CMAC	ECDSA	AH-HMAC-SHA1-96
Key Exchange	ECDH	ECDH	————	ECDH	ISAKMP/ECDH
Confidentiality	RC4	RC4	AES-CBC	AES/CCM or 3DES	AES-CBC, AES_CTR, 3DES
Key size	160	160	128	128 ->256	128 ->256
Hashing	MD5, SHA1	MD5, SHA1	————	SHA1,SHA2	SHA1,Tigger(x3SHA1
Access Control	————	Gateway	————	Gateway	————
Layer	Transport	Transport	MAC/Network	Network	Network
Gateway	————	Yes	————	Yes	————
End-to-end Security	Yes(transport)	Yes(transport)	No	Yes(network)	Yes(network)
Attacks	MIM	MIM	Eavesdropping, Replay, DoS	Replay	MIM, Spoofing(UI), DoS, Replay, Black hole,
Network Layer	————	————	————	IPSec_TM/WSN_SM	IPSec_PAN

comme la solution la plus flexible, la plus efficace et la moins coûteuse par rapport aux autres. Toutefois, il reste quelques points critiques à étudier dans cette approche. Contrairement aux méthodes de compression appliquées aux deux primitives AH et ESP de l'IPSec, aucune adaptation ou amélioration de l'IKEv2 n'a été proposée dans le contexte des RCSFs. Ainsi, nous avons orienté nos recherches vers cette piste. Vu que les systèmes de gestion des clés sont considérés comme des piliers nécessaires pour de nombreux services de sécurité dans les RCSFs, il fallait s'ouvrir aux autres solutions proposées dans la littérature. Pour cela, nous allons présenter dans la section suivante quelques méthodes adoptées pour la gestion des clés dans les réseaux de capteurs sans fil.

3.2 Gestion des clés dans les réseaux de capteurs sans fil

Une fois le réseau est déployé, tous les capteurs commencent à faire l'échange des données utiles. Pour qu'il y ait un échange sécurisé, un établissement et un partage de clés cryptographiques devraient être effectués en amont entre les capteurs communicants. Cela nécessite une bonne gestion de clés surtout dans un environnement sans fil très vulnérable aux attaques. La solution la plus basique [Zhou et al. (2006)][Lee et Stinson (2005)][Delgosha et Fekri (2005)] est la génération et la sauvegarde des clés avant la phase de déploiement. Elle est considérée comme une solution coûteuse puisqu'elle nécessite une grande capacité de mémoire et ne passe pas l'échelle dans les réseaux de capteurs. En

effet, chaque nœud doit établir une clé correspondante à chaque capteur. Pour éviter tout cela, plusieurs systèmes de gestion des clés ont été proposés dans la littérature [Chen et al. (2009)][Çamtepe et Yener (2007)]. En se basant sur des mécanismes de sécurité, ces systèmes permettent de faciliter, voire automatiser, toutes les fonctionnalités nécessaires pour assurer un partage sécurisé des clés. Parmi ces mécanismes, on trouve les algorithmes cryptographiques à clés publique qui sont les plus utilisés par les systèmes de gestion des clés. En effet, ces algorithmes reposent sur la notion de la fonction à sens unique (voir section 1.4.1) qui est très difficile, voire impossible à inverser. Néanmoins, la cryptographie à clé publique est considérée comme une méthode très coûteuse pour la gestion de clés dans les réseaux de capteurs sans fil. Pour faire face à cela, Eschenauer et Gligo dans [Eschenauer et Gligor (2002)] ont proposé l'utilisation de la pré-distribution aléatoire des clés (RKP : random key pre-distribution). Dans cette approche, adoptée par de nombreux auteurs [Yum et Lee (2012)][Chan et al. (2003)], le nombre de clés est beaucoup plus réduit que celui de la solution basique. En effet, chaque nœud reçoit un ensemble de clés qui est choisi aléatoirement d'un grand lot de clés avant la phase de déploiement. Par la suite, tout couple de nœuds ayant une clé commune peut l'utiliser pour un échange sécurisé. Un autre concept a été proposé qui se base sur la collaboration des nœuds voisins pour la gestion de clés dans les réseaux de capteurs sans fil. A titre d'exemple, l'idée de [Zhou et al. (2006)] qui permet à toute paire de nœuds d'établir une clé unique (Unique pairewise Key), indépendamment de la densité ou de la distribution des capteurs.

Dans cette optique, nous avons décidé d'intégrer cette notion de collaboration dans le protocole IKEv2 afin optimiser la consommation énergétique dans les RCSFs(Voir tableaux 2.4 et 2.5).

3.3 Implémentation de l'IKEv2

3.3.1 Descriptif général du protocole IKEv2

La sécurité saut par saut repose sur des algorithmes appliqués seulement aux liaisons qui relient les nœuds voisins sans tenir compte de toute la chaîne de communication. Comme expliqué dans le premier chapitre, la sécurité définie par la norme IEEE 802.15.4 se focalise seulement au niveau des couches basses où les liaisons entre voisins sont les seules protégées. Quant à la sécurité de bout en bout, les protocoles et les mécanismes de sécurité sont mis en œuvre exclusivement sur chaque extrémité d'une chaîne de communication. Le but est de protéger toute la liaison entre l'émetteur et le récepteur contre tout type d'attaque. Avec le protocole de sécurité de bout en bout, IPSec, tous les canaux de sécurité sont conçus pour assurer la confidentialité, l'intégrité des données, le contrôle de données, etc. Tout cela est réalisé grâce à des associations de sécurité (SAs) qui peuvent être établies par deux entités ou plus. Le but de ces SAs est de partager des paramètres de sécurité tels que les algorithmes cryptographiques qui seront utilisés pour l'échange des données utiles, les tailles des clés, leurs durées de vie, etc. Toutes les étapes d'établissement d'un SA, entre deux points d'extrémité, sont montrées dans la figure 3.4. Chaque SA est identifié par un indice appelé SPI (Security Parameter Index) et l'adresse de l'interlocuteur. Cet indice est inscrit sur l'entête du paquet IPSec et lors d'une transmission ou réception d'un paquet (étape 4 et 10), il servira à récupérer le bon SA de la base SAD (Security Association Database) ainsi que les clés cryptographiques (étape 3 et 9). D'autre part, la négociation initiale, étapes 5 et 7, entre les deux nœuds repose sur une base de données SPD (Security Policy Database) qui définit la politique selon laquelle les données seront protégées (l'étape 5). Ensuite, pour une demande d'une nouvelle association, une négociation se fait à l'étape 7.

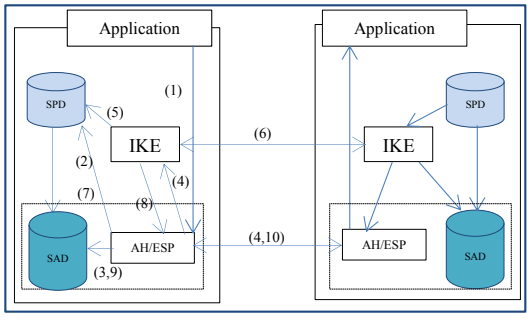

FIGURE 3.4 – *Architecture de l'IKEv2*

Dans un réseau de grande taille, il est presque impossible d'établir les SAs d'une façon manuelle. Pour résoudre cela, un protocole d'échange dynamique, appelé IKEv2[RFC5996], successeur du protocole IKE (Internet Key Exchange), a été introduit. L'établissement d'un SA s'effectue dynamiquement en trois échanges composés de paires de messages : une demande et une réponse. Tous ces messages possèdent des en-têtes du même format proposé par le protocole IKE. Une fois les SAs sont établis entre deux nœuds communicants, elles seront ensuite sauvegardées dans une base de données SAD (Security Association Database) créée pour chaque nœud. (voir figure 3.4)

Initiator		Responder
HDR, SAi1, KEi, Ni	→	
	←	HDR, Sar1, KEr, Nr, [CERTREQ]
HDR, SK {IDi, [CERT,] [CERTREQ,] [IDr,] AUTH, Sai2, TSi, TSr }	→	
	←	HDR, SK {IDr, [CERT,] AUTH, SAr2, TSi, TSr }
HDR, SK {SA, Ni, [KEi,], TSi, TSr}	→	
	←	HDR, SK {SA, Nr, [KEr,], TSi, TSr}

FIGURE 3.5 – *Principaux échanges de l'IKEv2*

L'IKEv2 se décompose en trois principaux échanges pour établir un SA : IKE_SA_INIT, IKE_AUTH et CREATE_CHILD_SA. Chaque échange consiste en une paire demande-réponse entre deux nœuds

désirant communiquer : un initiateur, qui prend l'initiative de cet échange et un répondant à cette demande. La première paire, IKE SA INIT, permet la négociation du premier IKE SA (les algorithmes cryptographiques, les clés...), l'échange des noms occasionnels (nonce) et les valeurs Diffie-Hellman. Le second échange, IKE_AUTH, permet l'authentification des messages précédents, l'échange des identités et des certificats et l'établissement du premier CHILD_SA qui sera utilisé durant les échanges AH(Authentication Header) et ESP(encapsulating Security Payload). Quant au dernier échange, il permet d'établir un SA en cas d'une demande d'une nouvelle CHILD_SA ou bien de mette à jour les deux SAs : IKE_SA et Child_SA.

Formats d'en-tête et de charge utile

Chaque message IKE commence par l'en-tête IKE, noté HDR. Suivant l'en-tête, on trouve une ou plusieurs charges utiles IKE. Le format de l'HDR est donné par la Figure 3.6. Il est composé des champs suivants :

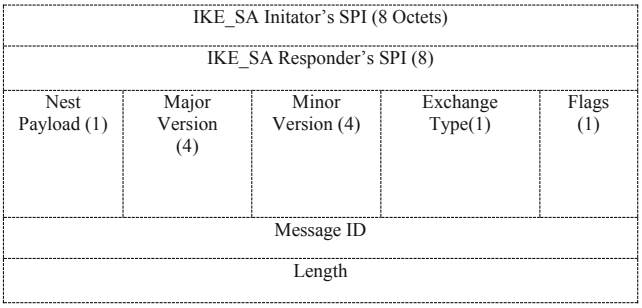

FIGURE 3.6 – *Format d'en-tête IKE*

- SPI de l'initiateur (8 octets) : Valeur choisie par l'initiateur pour identifier une association de sécurité IKE unique.

- SPI du répondant (8 octets) : Valeur choisie par le répondant pour identifier une association de sécurité IKE unique.

- Prochaine charge utile (1 octet) : Indique le type de charge utile qui suit l'en-tête.

- Version majeure (4 bits) : Indique la version majeure du protocole IKE utilisé. Pour notre cas, il s'agit de la version 2.

- Version mineure (4 bits) : Valeur par défaut 2.

- Type d'échange (1 octet) : Indique le type d'échange utilisé à savoir l'IKE_SA_INIT, IKE_AUTH et CREATE_CHILD_SA.

- Flags (1 octet) : Indique les options spécifiques qui sont établies pour le message

- Identifiant de message (4 octets) : Identifiant de message utilisé pour contrôler la retransmission des paquets perdus et faire correspondre demandes et réponses.

- Longueur (4 octets) : Longueur du message total (en-tête + charges utiles) en octets.

En-tête générique de charge utile

Chaque charge utile dans l'IKEv2 commence par un en-tête générique montré dans figure 3.7.

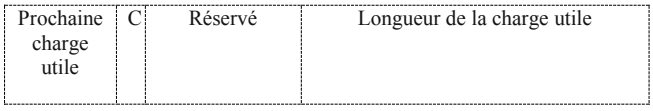

FIGURE 3.7 – *Format d'en-tête générique de la charge utile*

Le premier champ indique la prochaine charge utile sur laquelle le récepteur va pointer. Toute une liste chaînée peut être définie comportant toutes les charges utiles à savoir l'association de sécurité, l'échange de clé, les identifiants de l'initiateur et du répondant, le certificat, la demande du certificat, l'authentification et les noms occasionnels.

3.3.2 Simulation de l'IKEv2 (version légère)

Présentation succincte du simulateur

Il existe de nombreux outils qui permettent de simuler des réseaux de capteurs sans fil. Afin de choisir le bon outil qui correspond à nos besoins, nous avons réalisé une comparaison succincte entre les simulateurs les plus utilisés dans le monde de la recherche.

TABLE 3.2 – *Outils de simulation*

Simulateur	Langage de programmation	Couche transport	Modèle énergétique	Avantages	Inconvénients
NS2	C++/OTcl	UDP, TCP	Soustraction linéaire	-Simplicité d'intégration des nouveaux protocoles -Disponibilité de nombreux protocoles	-Bugs
OMNET++	C++/NED	UDP, TCP	Soustraction linéaire	-Supporter des réseaux à grande échelle -Supporter l'utilisation de la simulation parallèle	-Nombre réduit de modèles d'énergie et de routage -Pas d'implémentation du module IEEE 802.15.4
TOSSIM	nesC	———	Non supporté	-Simple configuration pour la création de la topologie réseau -Fiabilité de la simulation radio	-Aucun modèle de consommation énergétique
COOJA	Java/C	———	Non supporté	-Supporter des simulations à plusieurs niveaux	-N'est pas recommandé pour une large topologie -Supporter un nombre limité des nœuds en même temps

Dans un premier temps, nous avons sélectionné les simulateurs les plus appropriés selon trois principaux critères : gratuité, open source, citations. Le premier critère est la gratuité pour l'usage académique, le deuxième concerne la possibilité de modifier le code source du simulateur et le troisième concerne le nombre de citations dans les articles scientifiques. Parmi tous les simulateurs disponibles, On trouve : TOSSIM [Levis et al. (2003)], Omnet++ [Varga (1999)], COOJA [COOJ], NS-2 [Issariyakul et Hossain (2008)], etc. Une deuxième sélection a été faite selon trois autres critères qui se basent plus sur des critères techniques, à savoir la disponibilité des modèles (ex modèle énergétique, modèle de mobilité, modèle de transport...), la disponibilité des protocoles de communication et l'interopérabilité avec des librairies externes. Comme indiqué sur le tableau 3.2, le simulateur NS2 est beaucoup plus efficace dans le cas d'un réseau à grande échelle que le simulateur OMNET++ selon une étude comparative faite par [ksal]. De plus, en utilisant OMNeT

++, le nombre de protocles développés dans ce simulateur est inférieur à celui du NS2. Une autre étude comparative a été faite par H. L. Harsh Sundan et al. [H. L. Harsh Sundani (2011)] entre les deux simulateurs NS2 et OPNET. Le but était d'identifier le simulateur qui donne les résultats les plus proches de la réalité. Selon cette étude, le simulateur NS2 donne plus de résultats précis et concrets qu'OPNET. Après cette étude, notre choix s'est porté sur le simulateur NS2 qui est considéré comme un simulateur complet et riche en termes de fonctionnalités, protocoles, modules et librairies. Grâce à son Framework 'Mannasim' dédié aux réseaux de capteurs sans fil, nous disposons de plusieurs modules complémentaires qui nous donnent la possibilité d'évaluer les performances d'un RCSF à plusieurs niveaux [MAN]. Avec ce simulateur, nous pouvons définir des paramètres de simulation permettant de réaliser des scénarios réels et évaluer les performances des protocoles en question surtout par rapport au critère énergétique.

Intégration de l'IKEv2 dans NS2 et les librairies utilisées

NS2 est un simulateur à événements discrets écrit en C++ avec le langage interprété Tcl (Tool Command Language) pour configurer les objets.

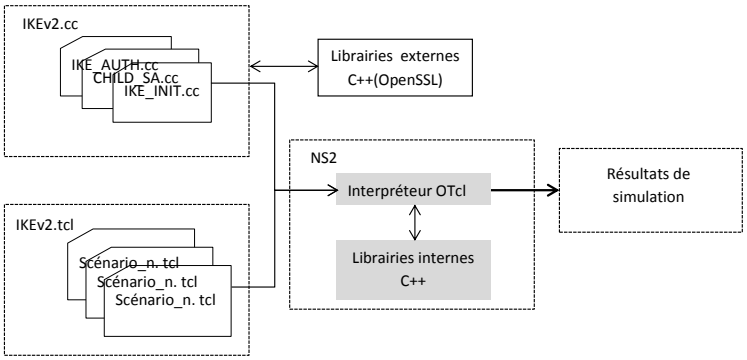

FIGURE 3.8 – *Intégration du module IKEv2 dans NS2*

Plusieurs niveaux d'abstraction sont offerts par NS2 : Routage,

propagation de paquets, modèle de trafic, modèle énergétique, application, etc. Sa distribution libre et open source, nous permet de modifier ou intégrer des modules et des protocoles afin d'évaluer leurs performances en se basant sur des métriques réseaux. Le module IKEv2 a été développé en C++ et intégré dans le simulateur NS2 comme étant une librairie externe (Voir figure 3.8). Nous avons également fait appel à des librairies externes telles que la librairie OpenSSL qui représente une boite à outils de sécurité comportant la création de clés RSA, la création de certificats, le chiffrement et le déchiffrement...

Pour réaliser une simulation, il faut définir trois principaux éléments : le réseau, les agents et le trafic. Le premier élément est lié à la topologie du réseau, les nœuds, les liens et les protocoles de routage pour l'acheminement des données entre les nœuds. Le deuxième élément consiste à définir les agents dans le réseau qui représentent les points terminaux où les paquets se construisent. Le troisième élément consiste à mettre en place un trafic réseau et les différents scénarios à tester.

Simulation

Dans cette section, nous allons présenter les résultats des simulations afin d'évaluer les performances de l'IKEv2 et voir la possibilité de l'implémenter ou l'améliorer d'un point de vue énergétique afin qu'il soit adapté aux les RCSFs. Toutes les simulations ont été réalisées sous l'outil NS2.

- Paramètres de simulation :

Pour notre simulation, nous avons créé un RCSF composé de 80 capteurs et une passerelle de sécurité unique (SG) reliant le réseau de capteurs avec des hôtes IP. Le tableau 3.3 décrit tous les paramètres de simulation utilisés.

TABLE 3.3 – *Paramètres de simulation*

Paramètres	**Valeur**
Simulateur	NS2/Mannasim
Type de trafic	UDP
Bande passante	2 Mbps
Taille du réseau	400m x 400m
Portée de communication	70 m
Protocole de la couche MAC	IEEE 802.11
Protocole de routage	AODV
Modèle de propagation	Two Ray Ground
Temps de simulation	100 s
Energie initiale	100 Joules
Puissance de transmission	0.36 Watt
Puissance de réception	0.24 Watt
Puissance de captage	0.015 Watt

En utilisant le simulateur NS2, nous sommes capables de mesurer plusieurs métriques réseaux sur tous les niveaux de la pile protocolaire. Nous avons implémenté une version légère du protocole IKEv2, décrite dans le RFC 5996, en utilisant la librairie de sécurité OpenSSL. Cette version comporte :

- Les deux phases IKE_INIT_SA et IKE_AUTH.

- DH protocole (groupe1).

- La signature RSA.

- Une seul SA_Child par IKE SA.

- DES3 comme algorithme cryptographique et SHA-1 comme une fonction de hachage.

- Résultats simulation :

Dans cette partie, nous allons étudier les trois indicateurs de performance suivants : le délai de création d'un SA, la consommation énergétique et le coût de la mémoire.

Nous avons suivi l'impact du nombre d'initiateurs sur le délai de création moyen d'un SA ainsi que sur la consommation énergétique.

Par rapport au délai, il est évident qu'il augmente en fonction du nombre d'initiateur. Toutefois, nous remarquons une augmentation

significative de 0.05 à 0.35 secondes quand le nombre d'initiateurs passe de 1 à 8.

En comparant avec les délais d'établissement des SAs entre deux machines connectées sur Internet, selon [Faigl et al.][H.Soussi et al. (2007)], le délai dans les réseaux de capteurs sans fil subit certes une latence mais aussi un surcoût énergétique énorme. En effet, la consommation énergétique augmente en fonction du nombre de nœuds dû à l'augmentation du nombre de messages échangés dans le réseau durant les deux premières phases de l'IKEv2 [Kasraoui et al. (2015b)].

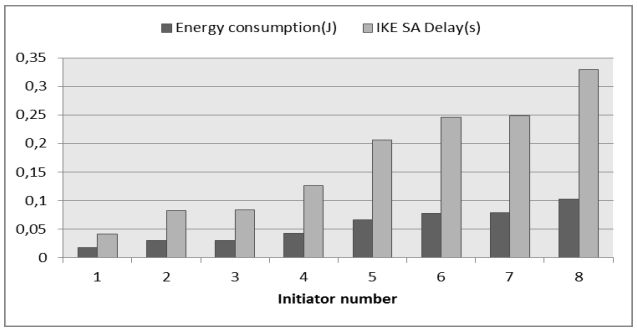

FIGURE 3.9 – *Consommation énergétique et délai d'établissement d'un SA*

En effet, l'implémentation de l'IKEv2 nécessite la sauvegarde d'un ensemble de clés de chiffrement et des associations de sécurité dans des bases de données. Ces dernières présentent un sujet tendancieux avec l'Internet des Objets (IdO). Chaque nœud (initiateur et répondant) stocke un ensemble de clés de chiffrement et un opérateur de l'authentificateur sans oublier les clés générées entre les deux premières phases (SKEYSEED, SK_d, SK_ai, SK_ar, SK_ei, SK_er, SK_pi, SK_pr), dont la taille de chacune est entre 16 et 20 octets. Le protocole IKEv2 nécessite entre environ 150 et 300 octets sans prendre en compte la base de données SAD utilisée pour sauvegarder les SA établis.

Selon, ces résultats, l'IKEv2 est considéré comme un protocole lourd

TABLE 3.4 – *Coût de la mémoire*

Elément -	Taille (octet)
	Phase 1
IKE HDR	28
SAi1 (1proposal, 4 transforms)	48
KEi	8 + Diffie-Hellman group size)
Ni	4 + random nonce size)
	Phase 2
HDR	28
SK {	
IV	16
Idi	8 + identity size (IPv4)
AUTH	8 + data authentication size (PSK, EAP...)
SAi2(1proposal, 3 transforms)	40
TSi	8 + size of traffic selector data)
TSr	8 + size of traffic selector data
Padding	1
Integrity Checksum Data	16
}	

pour les RCSFs. Nous nous sommes rendu compte qu'un seul nœud capteur est incapable de supporter un tel protocole lourd pour assurer une sécurité de bout en bout. D'où la nécessité d'adapter l'IKEv2 aux RCSF afin d'optimiser ces ressources.

3.4 Proposition d'une nouvelle approche de collaboration CKES

Nous avons consacré nos recherches sur la voie de la collaboration entre les nœuds et profiter de la caractéristique hétérogène des RCSFs. En effet, dans ces réseaux, il existe toujours un écart entre les ressources des nœuds capteurs qui dépend de la nature des applications exécutées, des plateformes utilisées et de l'architecture réseau. C'est dans cette optique de collaboration que s'articule le nouveau protocole CKES[Kasraoui et al. (2015a)]. D'ailleurs, selon [Vasilomanolakis et al. (2015)], les techniques collaboratives sont considérées comme les solutions les plus efficaces pour la sécurité dans les réseaux de capteurs sans fil. Au début, ce concept de collaboration a été proposé dans la cryptographie afin de partager

des secrets entre un ensemble de nœuds. L'idée de base était de diviser un secret et le partager avec plusieurs entités. Par la suite, Shamir et Blakley [Shamir (1979)][Blakley (1979)] ont introduit un nouveau concept de collaboration basé sur l'interpolation polynomiale et la géométrie de la projection linéaire. Leur idée consiste à distribuer un secret entre plusieurs nœuds qui peuvent, en mettant en commun les informations reçues, reconstruire le secret. En revanche, les données reçues par un seul dépositaire (nœud) ne suffisent pas pour retrouver le secret. Un seuil global pourrait être utilisé pour déterminer le nombre de nœuds à partir duquel le secret peut être récupéré. En se basant sur le théorème du reste chinois (Chinese Remainder Theorem - CRT), nous avons proposé une nouvelle approche collaborative appelée CKES (Collaborative Key Exchange System) afin de sécuriser les données échangées entre les nœuds dans les RCSFs tout en tenant compte de leurs contraintes énergétiques.

Définition : Le concept de CRT peut être présenté comme suit :

Soient $m_1, m_2, ..., m_k$ des entiers premiers entre eux, et k entiers $a_1, a_2, ..., a_k$. Le système de congruence

$$S = \begin{cases} x \equiv a_1 & (\mod m_1) \\ x \equiv a_2 & (\mod m_2) \\ . & . \\ . & . \\ . & . \\ x \equiv a_k & (\mod m_k) \end{cases} \tag{3.1}$$

admet une unique solution modulo $\prod_{i=1}^{k}(m_i)$, qui est donnée par la formule

$$X = \sum_{i=1}^{k}(a_i * M_i * y_i)(\mod M) \tag{3.2}$$

Tel que,

$$M = m_1 * m_2 * ... * m_k = \prod_{i=1}^{k}(m_i)$$

$$M_i = \frac{M}{m_i}$$

$$\text{et } y_i = M_i^{-1} (\text{mod } M)$$

Ce théorème a été appliqué dans plusieurs systèmes de sécurité à savoir la cryptographie à seuil [Desmedt (1998)], l'authentification multicast [Sadananda et al. (2013)], la signature proxy [Mambo et al. (1996)] ou le Cloud computing sécurité [Alam et K].

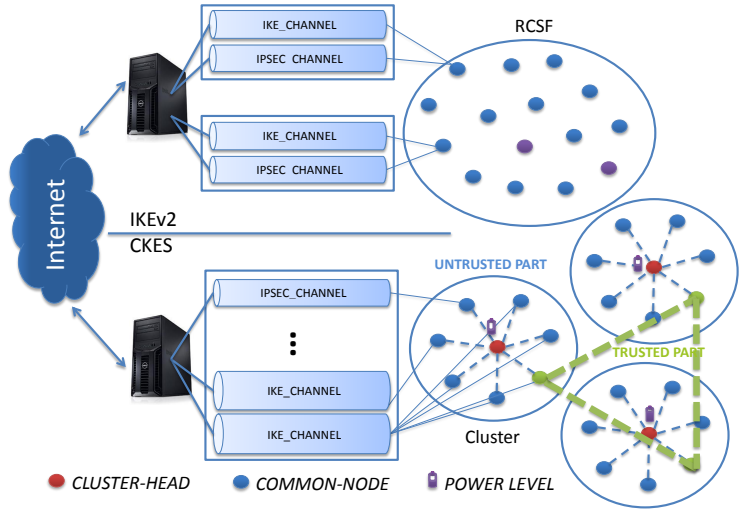

FIGURE 3.10 – *Architecture du CKES*

Dans notre cas, CKES permet l'adaptation du protocole de sécurité IPSec, conçu à la base pour sécuriser les échanges de bout en bout sur Internet, aux RCSFs de sorte que les opérations cryptographiques énergivores soient réparties sur un ensemble de nœuds.

3.4.1 Hypothèse

Dans l'approche CKES, nous avons pris en considération plusieurs hypothèses :

(i) Les nombres premiers $m_1, m_2, ..., m_n$ sont générés par la station de base (BS) et ils doivent également être co-premiers entre eux.

(ii) L'algorithme de clusterisation LEACH [Heinzelman et al. (a)] est utilisé afin d'organiser les nœuds capteurs sous forme de clusters.

(iii) Des nombres premiers sont attribués à tous les nœuds (contraints ou non).

(iv) Un système de gestion de confiance, appelé GTMS (Group-Based Trust Management Scheme)[Shaikh et al. (2009)], est utilisé dans le réseau.

(v) Les nœuds les plus confiants disposent de plus de ressources (en termes de puissance de la batterie et la capacité de mémoire). Ils prennent également en charge l'IKEv2 et les négociations entre tous les nœuds collaboratifs.

(vi) Les HTNs (Highly Trusted Nodes) peuvent demander à un CH toutes les informations concernant les niveaux de puissance de chaque nœud du même Cluster.

3.4.2 Les opérations cryptographiques les plus coûteuses

Jusqu'à nos jours, aucune implémentation de l'IKE n'a été faite dans les réseaux de capteurs sans fil. Une raison pour laquelle, nous avons creusé cet axe de recherche en espérant avoir des résultats concluants. Nous avons eu une forte conviction d'avoir contribué à un travail réaliste surtout après les études qui ont été faites par plusieurs auteurs en montrant la faisabilité de l'implémentation de l'IPSec dans les RCSFs [Varadarajan et Crosby (2014)], [Granjal et al. (2008)]. Selon une étude expérimentale [de Meulenaer et al. (2008)] faite sur une plateforme MiCA2, 359.87 mJ sont dissipés pour une signature en utilisant le protocole RSA-1024 et de 14.05 mJ pour une vérification RSA-1024. Un autre exemple a été donné dans [DHE], il montre une consommation de 1185 mJ pour un échange de clé en utilisant le protocole DH (ainsi que le calcul de la clé Master).

Par ailleurs, nous allons étudier par simulation, dans la section 3, la

TABLE 3.5 – *Coût des opérations cryptographiques*

Opération	Consommation énergétique (mJ)	Temps d'exécution(s)
RSA_Sign	359.87	12.04
Sign_Verif	14.05	0.47
DH_Exchange	1185	54.11

consommation énergétique des différentes opérations cryptographiques, utilisées par IKEv2, afin d'identifier les plus coûteuses. Selon cette étude, qui sera détaillée par la suite, nous allons confirmer que le protocole d'échange des clés Deffie-Hellman et les protocoles de signatures sont les plus coûteux en termes d'énergie. L'usage de ces protocoles pourrait alors augmenter les délais des communications ainsi que l'énergie consommée de tous les nœuds contraints. Suite à cela, nous avons pensé à externaliser les opérations cryptographiques les plus coûteuses vers les nœuds non-contraints.

3.4.3 Description du protocole CKES

L'idée de base est de solliciter des nœuds voisins disposant de plus de ressources à prendre en charge des opérations cryptographiques. Les partages se font d'une manière indirecte et sécurisée à l'aide de la technique CRT. L'algorithme CKES peut être résumé en quatorze étapes :

Etape 1 : L'initiateur A, un nœud contraint en énergie, demande à l'HTN d'ouvrir une session d'échange de clés avec le répondant (nœud B). Dans cette demande, A doit mentionner l'identité de B (B_ID) ainsi que le nombre maximal des nœuds collaboratifs (ou collaborateurs).

Etape 2 : Le HTN envoie, en multidiffusion, la demande reçue vers N nœuds (contraints ou moins contraints) qui pourraient être capables de traiter les opérations cryptographiques.

Etape 3 : Chaque nœud sollicité (CM Cluster Member) vérifie son énergie résiduelle, son indice de disponibilité ainsi que la valeur Ct (seuil

réseau). En fonction de toutes ces valeurs, le nœud décide s'il accepte ou pas la demande.

Etape 4 : Le HTN envoie les 'k' identités des nœuds collaboratifs, acceptant la demande, vers l'initiateur A ainsi que les coefficients $(y_1 * M_1, y_2 * M_2, ..., y_k * M_k)$ donnés par le CH.

Etape 5 : Le nœud 'A' génère un secret 'a' qui va être utilisé pour l'échange des clés DH et le calcul des clés Master. Ce secret est déterminé à partir de la somme des valeurs $a_1, a_2, ..., a_k$

$$\text{où } min(mi) \forall i \in [1..k] eta = \sum_{i=1}^{k} (a_i)$$

.

Etape 6 : 'A' détermine la solution X du système (S) en utilisant l'équation (3.2) mais sans appliquer le modulo M.

Etape 7 : 'A' envoie la solution X du système, l'association de sécurité SAi1 et le code d'authentification MAC vers le HTN

Etape 8 : HTN envoie, en multidiffusion, X vers tous les nœuds collaboratives et envoie un paquet IKE, qui contient l'HDR, Sai1, Ni CERT_HTN vers le répondant 'B'.

Etape 9 : Après la réception de X, chaque nœud collaborateur calcule son propre secret 'ai' (a_i= X mod $(m)_i$)) qui lui permet par la suite de calculer la partie g^{a_i} mod p de DH et l'envoyer au répondant B.

Etape 10 : Pour calculer la clé publique de A (DH_Pub), B fait le produit des différentes valeurs reçues des CMs.

$$\prod_{i=1}^{k} (g_i^a mod p) = g^{\sum_{i=1}^{k} (a_i)} mod p = g^a mod p$$

Etape 11 : Après vérification du certificat et le calcul de la clé maître g^{a*b} mod p, le nœud B envoie la valeur g^b mod p vers l'HTN ainsi que le paquet IKE (HDR, SAr1, etc).

Etape 12 : Chaque nœud collaboratif détermine sa clé maître g^{b*a_i} mod p et l'envoie vers l'initiateur en incluant la partie de l'HTN.

Etape 13 : Après la réception de la clé maître (portion), 'A' calcule le produit de toutes les valeurs reçues afin de déterminer sa clé maître.

$$\prod_{i=1}^{k}(g^{b*a_i} mod p) = g^{b*\sum_{i=1}^{k}(a_i)} mod p = g^{b*a} mod p$$

Etape 14 : Une fois la clé matîre est calculée, les deux nœuds A et B peuvent commencer le deuxième échange IKE_AUTH basé sur la SA négociée et la clé M.

Tous les détails du protocole CKES sont illustrés dans la figure 3.11. Comme expliqué auparavant, une fois les clusters formés et les CHs choisis (en appliquant LEACH), on détermine les valeurs de confiance de chaque nœud et on identifie également les HTNs en appliquant l'algorithme de gestion de confiance GTMS [Shaikh et al. (2009)].

En effet, le GTMS détermine chaque valeur de confiance en se basant sur les observations directes et indirectes. Ces dernières consistent à calculer le nombre des interactions faites avec succès ou avec échec. Les observations indirectes sont déterminées à partir des recommandations des nœuds confiants. Cette valeur est susceptible d'être changée au cours du temps. Les processus CKES commencent l'échange par une demande 'CKES message request' envoyée vers le HTN. Par la suite, le nœud collaboratif sera choisi selon son niveau énergétique et sa valeur de confiance. Le nombre minimal exigé est fixé à 3 afin de garantir un niveau de sécurité satisfaisant en utilisant le CRT.

3.4.4 Simulation

Dans cette section, nous allons présenter les résultats des simulations afin de prouver l'efficacité de CKES en le comparant avec l'IKEv2.

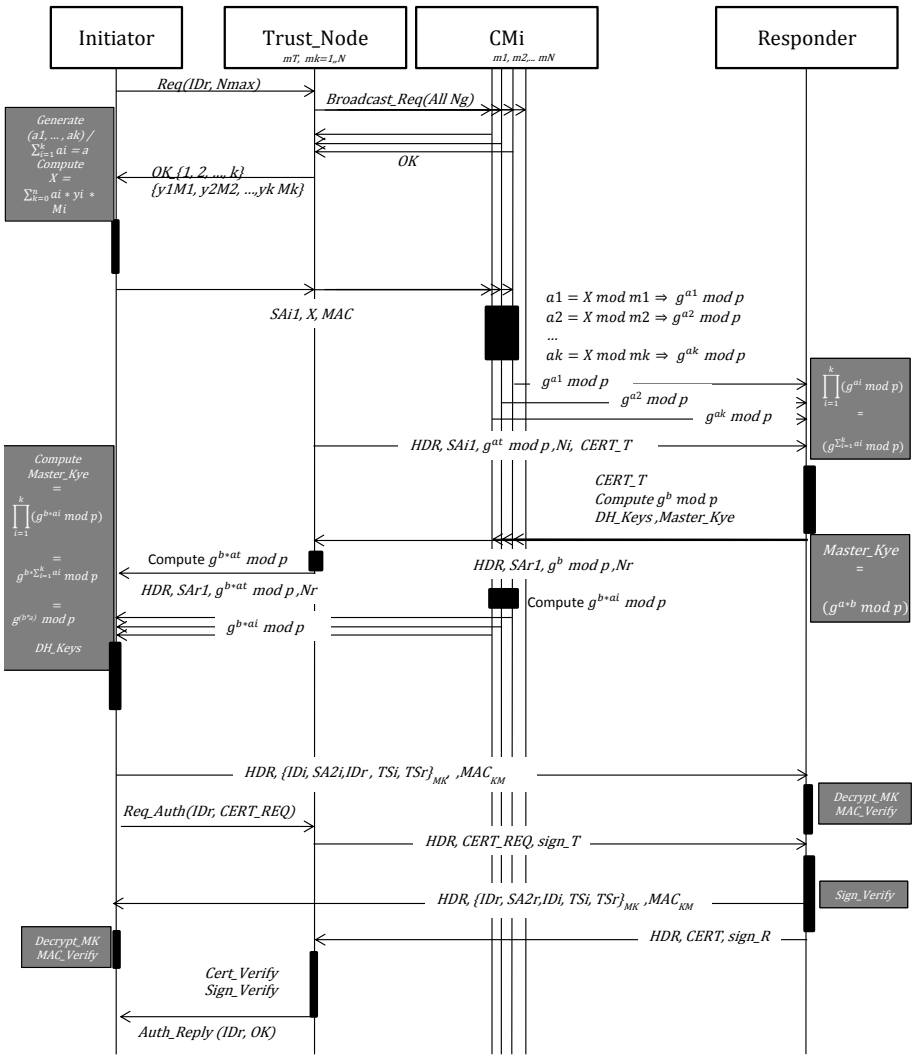

FIGURE 3.11 – *Echanges CKES*

Le modèle énergétique a été modifié de sorte que la consommation énergétique liée aux opérations cryptographiques soit prise en compte avec le coût énergétique lié à la communication. Ce modèle est basé sur une décroissance linéaire dans le calcul de l'énergie résiduelle. En effet, chaque instruction est comptée dans la décrémentation du niveau énergétique c.-à-d. pour déterminer le coût énergétique total d'une opération cryptographe, on multiple le nombre d'instructions par le coût d'une seule instruction.

Paramètres de simulation

Pour notre simulation, nous avons utilisé les mêmes paramètres que ceux utilisés pour évaluer les performances de l'IKEv2 (voir section 3.3.2). Nous rappelons ci-dessous succintement ces paramètres :

- Un RCSF composé de 80 capteurs

- Une passerelle de sécurité unique (SG) reliant le réseau de capteurs avec des hôtes IP.

- Deux catégories de nœuds : contraint et non-contraint.

Le tableau 3.6 décrit tous les paramètres de simulation utilisés.

TABLE 3.6 – *Paramètres de simulation*

Paramètres	**Valeur**
Simulateur	NS2/Mannasim
Type de trafic	UDP
Bande passante	2 Mbps
Taille du réseau	400m x 400m
Portée de communication	70 m
Protocole de la couche MAC	IEEE 802.11
Protocole de routage	AODV
Modèle de propagation	Two Ray Ground
Temps de simulation	100 s
Energie initiale	100 Joules

Résultats de simulation

- Coût de la communication :

Tout d'abord, nous avons déterminé le coût énergétique lié à la communication (aux échanges des paquets) d'un nœud contraint (l'initiateur). Ce coût est calculé en fonction du nombre de bits envoyés ou reçus durant les deux phases IKE_INIT et IKE_AUTH. Les résultats de la simulation sont montrés dans le tableau 3.7. Après la détermination de la consommation énergétique des deux protocoles CKES et IKEv2, nous remarquons bien une baisse de 20 % avec l'utilisation de CKES. Cela prouve l'efficacité énergétique vis-à-vis des nœuds contraints

TABLE 3.7 – *- Coût de la communication des protocoles IKEv2 et CKES*

IKEv2	Sent (bytes)	Recv(bytes)
IKE_INIT	124	124
IKE_AUTH	497	497
IKEv2 communication costs (mJ)	1.29	1.43
CKES communication costs (mJ)	0.89	1.24

Toutefois, en déterminant le coût énergétique lié à la consommation des données dans tout le réseau, une importante hausse est remarquable qui dépasse le double de la consommation énergétique de l'IKEv2. Cela est dû aux échanges entre les nœuds initiateurs et les nœuds voisins avant l'acheminement des paquets entre l'initiateur et le répondant. Durant les simulations des deux protocles, les mêmes paramètres réseaux, le même protocole de routage et le même algorithme de clustérisation ont été utilisés pour les deux.

TABLE 3.8 – *- Coût énergétique du réseau*

Network communication costs using IKEv2 (mJ)	Network communication costs using CKES (mJ)
42.81	96.5

- La consommation d'énergie des opérations cryptographiques :

Quant au coût lié au calcul interne des nœuds, nous avons compté chaque instruction exécutée afin de connaitre le coût total lié aux différentes opérations cryptographiques.

Contrairement à l'IKEv2, les opérations cryptographiques les plus coûteuses sont exécutées par les nœuds collaboratifs. Quelques fonctions

TABLE 3.9 – *Consommation d'énergie des opérations cryptographiques*

	IKEv2	CKES
Total Computation costs (mJ)	252.87	4.8

exécutées au niveau des nœuds contraints sont gardées tandis que toutes les autres sont déléguées aux nœuds voisins. Cela a réduit énormément la consommation énergétique des nœuds contraints ainsi que la mémoire utilisée pour la sécurité. Comme montré dans le tableau 3.9 , nous avons pu économiser environ 98% de l'énergie pour les nœuds contraints ce qui prolonge la durée de vie de ces nœuds.

- Coût Total de l'énergie :

Par la suite, nous avons comparé la consommation énergétique totale des deux protocoles. Certes le CKES consomme plus au niveau du module de communication mais au niveau global il consomme moins. Il est peu énergivore par rapport à l'IKEv2. Le tableau 3.10 montre que, globalement, le CKES consomme environ 30% de l'énergie dissipée par IKEv2.

TABLE 3.10 – *Coût énergétique total des protocoles IKEv2 et CKES*

	IKEv2	CKES
Total Communication costs (mJ)	42.81	96.5
Total Computation costs (mJ)	252.87	4.8
Total costs (mJ)	295.68	101.3

- Analyse énergétique du CKES selon la répartition de nœuds contraints :

Afin de suivre le comportement énergétique du protocole CKES par rapport à l'IKEv2, nous avons suivi la consommation énergétique en fonction de la répartition des nœuds contraints dans le réseau. A chaque valeur (répartition), nous déterminons la différence des coûts entre les deux protocoles. Comme le montre la figure 3.12, à 5 %, nous remarquons une augmentation brusque du coût énergétique de 20 mJ à 150mJ. Cette augmentation est due au nombre minimal des nœuds collaboratifs qui

est fixé à 3 (c = 3). En effet, nous avons fixé ce seuil afin de garantir un niveau de sécurité minimal et éviter tout risque permettant la divulgation du secret généré par l'initiateur. Ce risque dépend forcement du nombre des nœuds collaboratifs. Plus ce dernier augmente, plus la reconstruction du secret devient compliquée.

La figure 3.12 représente le coût énergétique de l'IKEv2 en trois couleurs indiquant le coût lié au calcul (au niveau de l'initiateur), le coût lié à la communication de ce dernier ainsi que le coût lié à tout le réseau. Les mêmes mesures ont été faites sur le protocole CKES et présentées sur la figure 3.13.

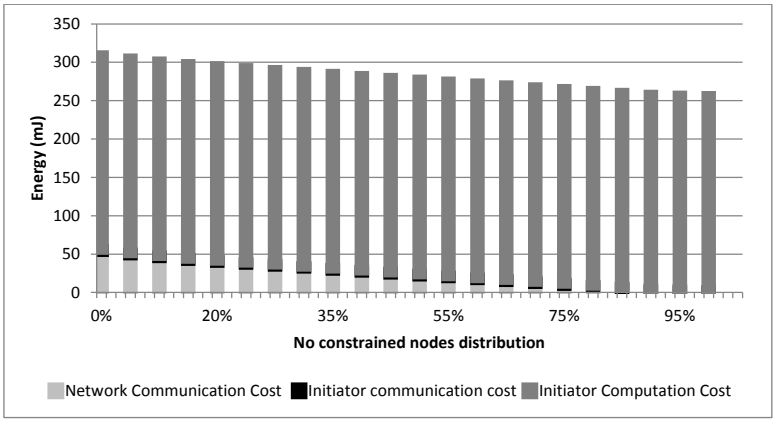

FIGURE 3.12 – *Consommation d'énergie totale de l'IKEv2*

Sur cette dernière, nous remarquons bien qu'à partir de 5%, le coût énergétique du réseau commence à baisser proportionnellement au nombre des nœuds non-contraints. Sur toutes les répartitions, le coût de la communication du CKES est quasiment le double du celui l'IKEv2. Cela est dû au nombre de messages échangés entre l'initiateur et les nœuds collaborateurs. Quand le nombre des nœuds non contraints devient plus élevé, l'initiateur aura plus de chance à trouver des nœuds voisins qui collaborent avec lui. Ainsi, le nombre de sauts diminue significativement

entre l'initiateur et les nœuds CMs, ce qui signifie (permet) une baisse de la consommation énergétique dans tout le réseau.

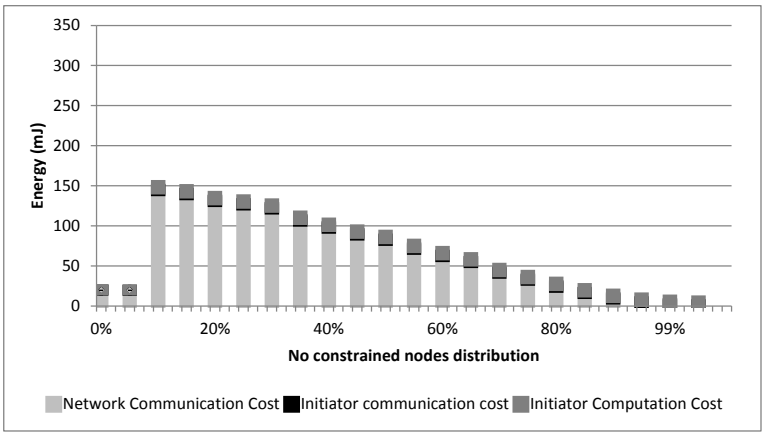

FIGURE 3.13 – *Consommation énergétique du CKES*

- Gain énergétique par rapport à l'IKEv2 :

En outre, nous avons déterminé le gain (en %) de l'énergie résiduelle de CKES par rapport à l'IKEv2. La figure 3.14 montre que les nœuds contraints peuvent économiser environ 50% de leurs énergies pour une distribution de 5% (de nœuds contraints) et environ 94% pour une distribution de 95

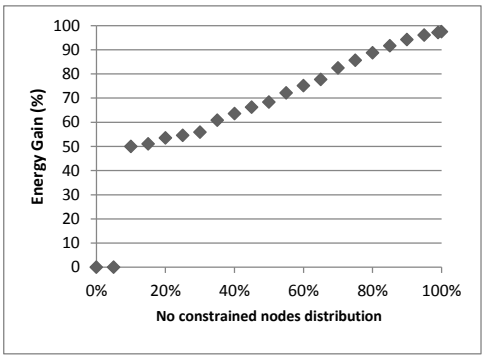

FIGURE 3.14 – *Gain énergétique du CKES*

- Durée de vie du CKES Nous avons étudié la durée de vie du réseau pour les deux protocoles. Pour cela, nous avons lancé une simulation de

80 nœuds dont 40% sont des nœuds contraints. Au début de la simulation, chaque nœud dispose de 100 Joules comme énergie initiale, notée Ei. Par la suite, nous choisissons aléatoirement un initiateur afin d'établir une seule association de sécurité, SA, avec le nœud Sink (nœud '0') durant chaque tour. Cela continue jusqu'à la mort du premier nœud dans le réseau.

TABLE 3.11 – *Durée de vie des protocoles IKEv2 et CKES*

	Network lifetime (rounds)
IKEv2	391
CKES	14367

Les résultats de simulation étaient attendus depuis l'idée de la délégation du calcul lourd qui a conduit à plus d'économies d'énergie au niveau des nœuds contraintes. Un tel déchargement des opérations de signature et de chiffrement donne aux nœuds capteurs un gain important par rapport à la solution classique IKEv2. Comme le montre le tableau 3.11, la comparaison entre IKEv2 et CKES confirme l'efficacité du système collaboratif d'échange de clés en termes de coûts de l'énergie. Cela prouve aussi la viabilité de l'approche collaborative proposée dans le contexte étudié avec des nœuds à ressources limitées. Outre le niveau de sécurité qui est à peu près équivalent à celui de l'IKEv2, le CKES introduit un délai supplémentaire pour établir des associations de sécurité SA entre les paires de nœuds, mais en contrepartie, il permet d'économiser plus d'énergie et augmente la durée de vie du réseau.

CONCLUSION DU CHAPITRE

Dans ce chapitre, nous avons commencé par une étude comparative de différentes solutions de sécurité de bout en bout dans les RCSFs. Par la suite nous avons présenté une nouvelle approche collaborative appelée CKES (Collaborative Key Exchange System) qui permet de sécuriser les données échangées entre les nœuds tout en tenant compte de leurs

contraintes énergétiques. L'outil NS2 (Network Simulator 2) a été utilisé pour valider notre approche CKES et analyser ses performances en termes d'énergie et d'autres métriques réseaux. Selon les résultats de la simulation, nous avons prouvé que le protocole CKES est plus performant que l'IKEv2 (Internet Key Exchange version 2), une composante d'IPSec, en termes de la consommation énergétique. Cette dernière est liée à la taille des données échangées au niveau de chaque nœud contraint durant les phases d'établissement des associations de sécurité. Nous avons également comparé la consommation énergétique liée au calcul des différentes opérations cryptographiques au niveau de chaque nœud contraint. Ce coût dépend de la complexité des algorithmes de sécurité utilisés ainsi que du nombre des instructions effectuées lors du calcul [Kasraoui et al. (2014b)]. Afin de prouver l'efficacité de notre système collaboratif contre les attaques externes et sa capacité à répondre aux critères fondamentaux de la sécurité (authentification, intégrité, confidentialité), nous allons réaliser une étude de vérification formelle du CKES dans le chapitre suivant.

4 Validation et vérification du CKES en utilisation les méthodes formelles

Prouver la sûreté d'un protocole de sécurité revient à prouver son efficacité d'empêcher toute perturbation au fonctionnement normal du protocole. Cela revient également à s'assurer qu'aucune attaque ne pourrait exister même en présence des intrus dans le réseau. Pour ce faire, plusieurs scenarii peuvent être testés en vérifiant qu'un intrus possédant des capacités ne peut infiltrer le réseau ni perturber son fonctionnement. Cette vérification est considérée comme une phase fastidieuse surtout quand il s'agit de tester manuellement tous les cas possibles pour trouver une attaque. Afin de remédier à ce type de problème, plusieurs outils de vérification automatique ont été proposés rendant ainsi cette phase plus rapide et plus sûre. Parmi ces outils, nous trouvons AVISPA (Automated Validation of Internet Security Protocols and Applications) [AVIS], Proverif [Prof], Scyther [Scy], etc. Tous ces outils sont basés sur des modèles permettant de valider formellement tout protocole. Dans ce chapitre, nous allons présenter une vérification formelle de l'efficacité du protocole CKES en utilisant la plateforme la plus adéquate. Dans la suite, nous commençons par un rappel des méthodes formelles. Puis, nous

présentons l'outil avec lequel nous avons réalisé toutes les spécifications. Enfin, nous présentons les résultats de la validation formelle faite sur le protocole de sécurité CKES proposé dans le deuxième chapitre.

4.1 Notions de base

Dans cette section, nous allons donner un focus sur les principes de base pour mener à bien une vérification d'un protocole donné. Nous présentons les méthodes formelles, la spécification formelle ainsi que la validation.

4.1.1 Méthodes de vérification formelle

La vérification des protocoles de sécurité consiste, en général, à prouver que tous les besoins fondamentaux de sécurité sont remplis. Pour ce faire, il existe des méthodes formelles qui sont très utilisées dans le monde académique et dans l'industrie permettant d'améliorer la qualité des systèmes logiciels et matériels. Ces méthodes sont suffisamment lisibles pour être utilisées comme un outil de documentation. Son utilisation avant le processus de conception aide à la suppression de beaucoup d'erreurs qui peuvent être découvertes ultérieurement.

4.1.2 Spécification formelles

Une spécification formelle est simplement une description du système utilisant une notation mathématique. L'avantage d'utiliser cette notation est la précision dans la description, contrairement aux langages naturels et aux diagrammes. Le processus actuel de la conception des systèmes doit profiter de la notation formelle pour faciliter la communication des idées entre les membres d'une équipe de conception. La spécification formelle est très précise, c'est-à-dire même si une telle spécification est incorrecte (inadéquate aux exigences fixées dès le début), il est facile de détecter où sont les erreurs et les corriger, contrairement aux spécifications informelles. L'utilisation d'une notation formelle augmente la compréhension des opérations des systèmes surtout durant

la première phase de la conception. Elle aide également les concepteurs à l'organisation et la clarification des idées.

4.1.3 Vérification formelle

Un avantage considérable de l'utilisation d'une spécification formelle est qu'elle va permettre de tester rapidement tous les cas de figure du déroulement d'un protocole. Deux techniques de vérification existent : le model checking [Clarke et Emerson (1982)] et la preuve de théorème [Monin (2000)].

Model checking

Il consiste à construire un modèle fini du système et à vérifier, via une recherche exhaustive dans l'espace des états, que la propriété désirée est satisfaite dans ce modèle. Il peut être utilisé pour vérifier des spécifications partielles, puis fournir des informations utiles de la cohérence du système même si le système n'est pas totalement spécifié. Aussi, il peut produire des contre-exemples qui sont typiquement subtiles dans la conception. Le problème dans le model checking est l'explosion combinatoire. Cependant, une représentation efficace des transitions d'état peut réduire la taille du système qu'on doit vérifier.

Preuve de théorème

C'est une technique qui exprime le système et ses propriétés dans des formules basées sur la logique, en termes d'axiomes et des règles d'inférence. La preuve des théorèmes est le processus chargé de trouver une preuve pour une propriété à partir des axiomes, des règles, d'éventuelles définitions et de lemmes. En contradiction avec le model checking, la preuve de théorème peut s'appliquer dans un espace infini d'états, mais les résultats de la vérification sont généralement lents et

parfois erronés. Il y a des outils qui supportent la vérification formelle. Ces outils ont pour but de démontrer de manière rigoureuse, c'est-à-dire, en utilisant des règles de calcul, qu'une proposition est vraie. Il existe plusieurs types d'outils permettant de vérifier plusieurs types de propriétés. Ces outils nécessitent deux entrées :

- Une description formelle du système

- Une expression formelle des propriétés à vérifier

4.2 Outil de vérification formelle

4.2.1 AVISPA

AVISPA est l'un des outils de vérification formelle permettant une analyse automatique d'un protocole de sécurité. Il est appelé aussi un moteur de vérification automatique. En effet, tout protocole de communication peut être testé à n'importe quelle échelle. Grâce à sa plateforme dédiée à la validation formelle, AVISPA est capable de vérifier les propriétés de sécurité.

Spécification formelle en HLPSL

A l'aide d'un langage de spécification formelle, appelé HLPSL (High Level Protocol Specification Language), les protocoles sont décrits en détail tout en offrant plusieurs niveaux d'abstraction ainsi que des interfaces dédiées aux protocoles de sécurité. La spécification se fait sous forme de rôles. Chaque rôle définit les actions d'un agent durant le déroulement du protocles de sécurité. Deux types de rôles sont définis : un rôle basique et un rôle composé. Le rôle basique représente tous les participants au protocole sous forme d'agents. Le rôle composé, tel que le rôle session ou bien le rôle environnement, représente des scenarii entre les rôles basiques. Tous ces rôles disposent de plusieurs paramètres d'entrée représentant des informations initiales, à savoir les noms des

agents, les nonces, les clés, les canaux de communications, etc. Toutes ces données sont considérées comme des informations préalables nécessaires pour réaliser la spécification du protocole de sécurité. De même, des variables locales, représentant l'état interne de chaque agent, devraient être déclarées ainsi que tous les messages échangés.

Par la suite, une partie fondamentale doit s'ajouter à chaque rôle représentant les transitions qui consistent à un ensemble d'actions exécutées suite à une réception d'un message.

Après avoir défini les agents, deux principaux rôles doivent être ajoutés. Le rôle 'session' pour regrouper tous les acteurs et le rôle 'environnement' pour lancer plusieurs 'sessions' en même temps et les différents scénarii de l'intrus.

Enfin, l'exécution du protocole consiste tout simplement à lancer l'exécution de l'environnement.

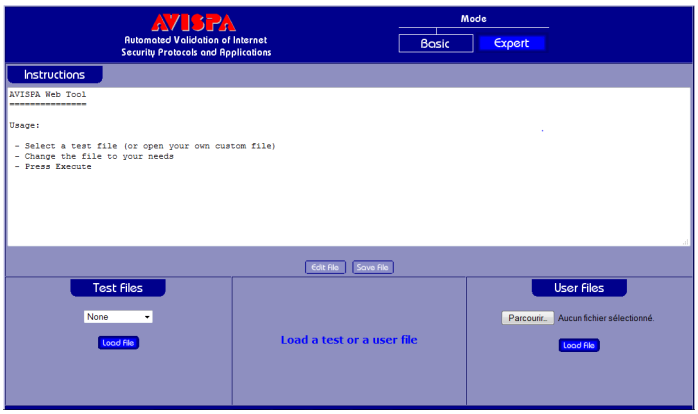

FIGURE 4.1 – *Interface web de l'outil AVISPA*

Exemple d'une spécification formelle faite avec AVISPA

Afin de simplifier les choses, nous avons pris un exemple d'une spécification formelle d'un simple protocole d'échange de clés appelé WMF (Wide Mouthed Frog)[Burrows et al. (1990)]. Il s'agit d'un protocole de sécurité basique permettant un échange sécurisé entre deux entités

A et B. Par l'intermédiaire d'un troisième intervenant, généralement un serveur, la sécurité du canal de communication entre A et B est assurée. En effet, le serveur est considéré comme un tiers de confiance à travers lui toutes les informations secrètes sont passées et tous les messages échangés avec lui sont chiffrés par une clé partagée Kas(entre A et S) ou Kbs (entre B et S). Afin d'établir une clé de session entre A et B, A génère une clé Kab et l'envoie vers le serveur chiffrée par la clé Kas. Par la suite, le serveur S, en déchiffrant le message reçu, transfère la clé de session Kab vers B après l'avoir chiffré avec la clé Kbs.

```
A -> S : {Kab}_Kas
S -> B : {Kab}_Kbs
```

FIGURE 4.2 – *Echanges WMF*

Comme le langage HLPSL est basé sur le principe de rôles, chaque agent (participant) associé à un rôle mène des actions indépendamment des autres. Ainsi, pour le cas du WMF, trois rôles sont définis, correspondant aux trois participants Alice, Bob et le Serveur. Commençant par le rôle 'Alice', nous devons préciser, en paramètres d'entrée, tous les agents participants aux échanges (protocole) tels que B et S, les clés symétriques de A à savoir la clé Kas partagée avec l'agent S ainsi que les canaux de communications SND et RCV, respectivement, pour l'émission et la réception de données. Concernant les variables locales, elles sont définies dans la section «local» où on trouve l'état initial de l'agent, représenté par un nombre naturel 'o', ainsi que la clé de session symétrique, partagée avec l'agent B.

Dans la section 'transition', nous avons défini toutes les actions prises en charge par l'agent A. Dans notre exemple, il s'agit d'une seule action qui est définie par l'envoie d'une clé Kab suite à la réception d'un message.

Prenons comme exemple le rôle du serveur, au début, son état est initié

```
role alice(A,B,S : agent,
           Kas : symmetric_key,
           SND, RCV : channel (dy))
played_by A def=
  local State: nat, Kab: symmetric_key
  init  State := 0
  transition
    ...
end role
```

FIGURE 4.3 – *Rôle d'Alice*

par 0. Puis, à la réception d'un message sur le canal RCV, contenant la clé Kab cryptée par Kas, il l'envoie chiffrée sur le canal SND et l'état de l'agent 'serveur' prend la valeur 2.

```
step1. State  = 0 /\ RCV({Kab'}_Kas) =|>
       State':= 2 /\ SND({Kab'}_Kbs)
```

FIGURE 4.4 – *Transition au niveau serveur*

En plus des rôles basiques (Alice, Bob et le seveur), nous avons également défini les rôles composés. Le premier rôle 'session' regroupe tous les rôles des agents, les canaux de communications et le déroulement global du protocole.

```
role session(A,B,S    : agent,
             Kas, Kbs : symmetric_key) def=

local SA, RA, SB, RB SS, RS: channel (dy)

composition
    alice (A, B, S, Kas, SA, RA)
/\  bob   (B, A, S, Kbs, SB, RB)
/\  server(S, A, B, Kas, Kbs, SS, RS)

end role
```

FIGURE 4.5 – *Rôle session*

Comme mentionné dans la figure 4.5, nous précisons la composition d'une session. Tous les rôles sont instanciés et doivent être exécuté en parallèle. De plus, tous les canaux de communications doivent être définis (SA, RA, SB, RB, SS, RS). Par la suite, nous définissons la spécification du rôle environnement où toutes les variables globales sont déclarées ainsi

que les différentes sessions à établir. Ce rôle permet de préciser le point auquel un intrus peut intervenir. Pour notre exemple, nous envisageons le cas où l'intrus peut jouer le rôle d'un participant légitime et remplace soit A, soit B.

```
role session(A,B,S    : agent,
             Kas, Kbs : symmetric_key) def=

local SA, RA, SB, RB SS, RS: channel (dy)

composition
    alice (A, B, S, Kas, SA, RA)
/\  bob   (B, A, S, Kbs, SB, RB)
/\  server(S, A, B, Kas, Kbs, SS, RS)

end role
```

FIGURE 4.6 – *Rôle environnement*

Une fois le fichier HLPSL est prêt, il passe en entrée de l'outil AVISPA afin d'être testé et analysé. Pour comprendre le fonctionnement interne de l'outil, nous décrivons l'architecture d'AVISPA en détails.

Architecture de l'outil

Disposant de quatre terminaux de gestion, AVISPA peut fournir quatre analyses différentes d'un même protocole. Un seul langage à haut niveau, HLPSL, utilisé comme entrée, est converti en format IF (Intermediate Format) à l'aide d'un traducteur. Le résultat obtenu est analysé et est considéré positif dans le cas où on arrive à identifier une attaque et il est considéré négatif dans le cas contraire. Ainsi, le protocole est fiable si le résultat est négatif selon l'analyse formelle. -Outils de vérification : AVISPA [Armando et al. (2005)]propose quatre outils de vérifications : OFMC, CL-AtSe, SATMC et TA4SP.

OFMC (On-the-fly Model-Checker) : il repose sur la vérification itérative en parcourant exhaustivement tous les états transitoires dans le fichier IF. CL-AtSe(L-based Attack Searcher) : repose sur les contraintes et l'élimination des redondances. Son avantage qu'il peut s'enrichir

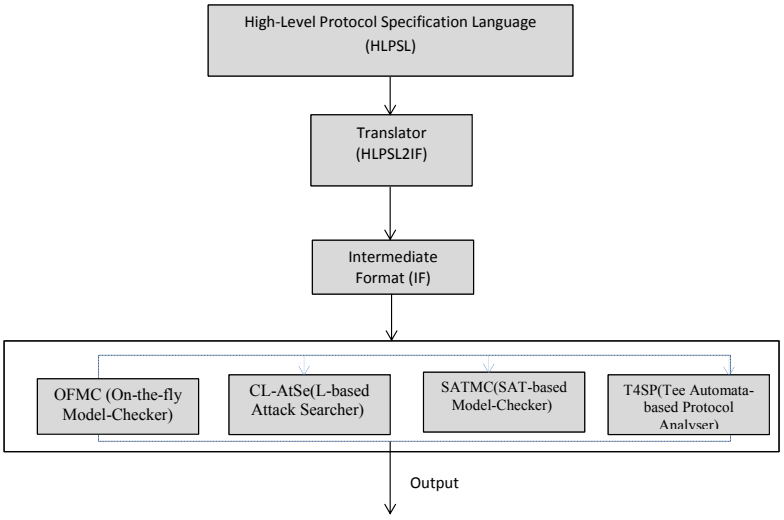

FIGURE 4.7 – *Architecture de l'outil AVISPA*

par d'autres spécifications des opérations cryptographiques grâce à sa modularité. SATMC(SAT-based Model-Checker) : permet de générer une attaque grâce à un état transitoire et recherche d'éventuelles violations causées par une attaque. T4SP(Tee Automata-based Protocol Analyser) estime les capacités d'un intrus pour connaitre la vulnérabilité d'un protocole. Toutes les propriétés de sécurité décrites dans le fichier HLPSL seront testées par ces techniques. Ces dernières permettent d'identifier les failles de sécurité du protocole mais ne garantissent jamais leurs absences. En effet, une attaque peut à tout moment apparaître dans un espace d'états qui n'a pas été exploré.

4.2.2 Outil graphique SPAN

Afin de faciliter la conception de la spécification formelle, Y. Glouch et al. [Glouche et al. (2006)] ont proposé un outil appelé SPAN (Security Animator for AVISPA). Cet outil consiste à animer les spécifications HLPSL en mettant en place des messages spécifiques de types MCS (Message Sequence Charts). Grâce à SPAN, on peut visualiser tous les messages échangés entre les agents en les affichant en ordre

chronologique. Avant la phase d'analyse du protocole, SPAN fait une compilation du fichier HLPSL en vérifiant l'existence des erreurs de spécifications non détectées par AVISPA. Il est caractérisé par une flexibilité sur le choix des messages qui pourraient être interceptés par l'intrus. De plus, il permet de reconstruire les attaques d'une manière intuitive.

Dans le cadre de droite (Figure 4.8) de l'interface SPAN, tous les messages déjà envoyés sont affichés par ligne sous forme d'un diagramme de séquence. S'il n'y a plus de transitions, c'est qu'on est à la fin du protocole ou qu'il y a une erreur dans la spécification HLPSL.

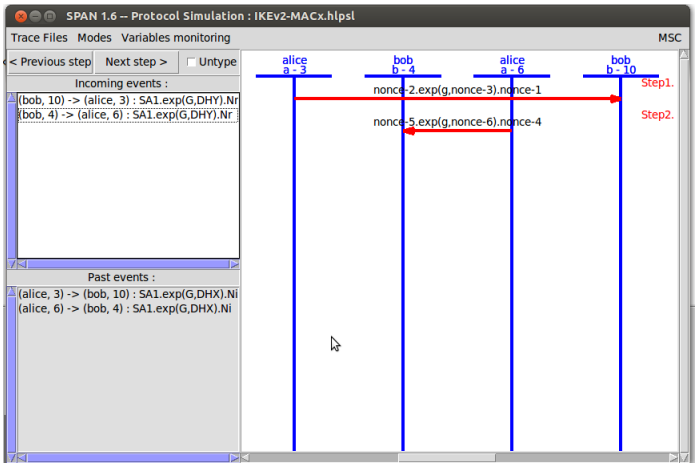

FIGURE 4.8 – *Interface SPAN*

Deux modes de simulation sont proposés par SPAN : mode normal et mode intrus. En mode normal, on n'affiche que les transitions réalisées entre les agents alors qu'en mode intrus, tous les messages peuvent être passés par l'intrus pour enrichir sa base de connaissances.

Pour une modélisation d'un protocole de sécurité, il est nécessaire de modéliser également l'intrus, c'est-à-dire de définir son comportement et ses limites. Pour cela, AVISPA/SPAN repose sur un modèle simple qui a été proposé dans [Dolev et Yao (1983)] sous le nom « modèle de Dolev-Yao ». Ce dernier est un des premiers modèles formels conçu pour

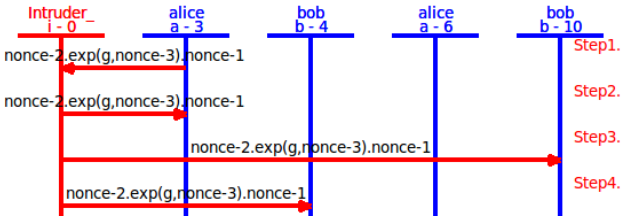

FIGURE 4.9 – *Mode Intrus*

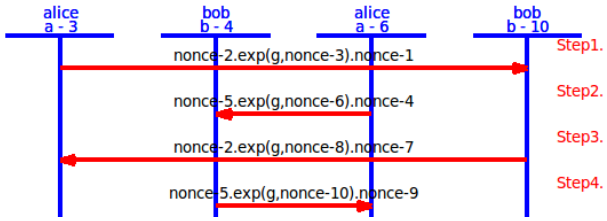

FIGURE 4.10 – *Mode Normal*

la vérification de protocoles cryptographiques. Il consiste à formaliser les capacités d'un attaquant en se basant sur l'hypothèse suivante : Un chiffrement parafait dans un réseau idéalisé. En effet, le modèle suppose que le réseau est idéal dans le sens où aucun message échangé ne peut se perdre et les messages sont tous envoyés et reçus instantanément par les agents. De plus, avec ce modèle, il n'est pas possible de prendre en compte le temps dans les capacités d'un attaquant et tous les canaux de communication sont supposés être publics. Par ailleurs, l'intrus ou l'attaquant pourrait en fonction de sa connaissance chiffrer ou déchiffrer un message dans la mesure du possible, composer ou décomposer un message, créer des valeurs aléatoires, envoyer ou intercepter des messages dans le réseau, appliquer des fonctions de hachage, etc.

4.3 FORMALISATION ET VALIDATION DU PROTOCOLE CKES

Vu la complexité du langage HLPSL et la difficulté pour simuler tout le réseau, nous avons fait quelques simplifications par rapport au cas réel. En effet, nous avons défini les principaux intervenants au protocole

comprenant l'initiateur, le répondant, un nœud collaboratif Ci et le nœud de confiance T.

Durant l'échange CKES, chaque intervenant dispose d'un lot de clés publiques (Ka, Kb, Kt et kc) ainsi qu'une fonction de hachage connue par tous les autres agents. Le groupe G de l'échange Deffie-Hellman est considéré également comme une variable publique. Avant de commencer la description du code HLPSL, nous citons, dans le tableur I, tous les symboles à utiliser.

TABLE 4.1 – *Notations*

Symbole	Interprétation
DHX	Valeur Deffie-Hellman
KE	Key Exchange - DH
SA	Security Association
KAN	Shared Key between A and N
PSK	Shared Key between A and B
KAC	Shared Key between A and C
EXP	Exponential function

Rappelons le principe du protocole CKES proposé dans le second chapitre, la figure 4.11 montre les principaux échanges du protocole.

Nous présentons dans ce qui suit les différents agents définis dans la description HLPSL de notre protocole [Kasraoui et al. (2014a)] :

- L'agent A. Il joue le rôle 'Initiateur' représenté dans la figure 4.11 dont l'objectif est d'initier un canal de communication sécurisé avec un autre agent. Ce rôle évolue de la façon suivante : La transition 1 : L'initiateur envoie à l'agent T un message demandant l'ouverture d'une session d'échange de clés avec le répondant B. Dans ce message, il précise l'identité de B ainsi que le nombre maximal des nœuds collaboratifs (ou collaborateurs). La transition 2 : Une fois qu'il a reçu les identités des nœuds collaboratifs, il génère un secret 'a' qui va être utilisé pour l'échange des clés DH et le calcul des clés Master. Après la résolution du

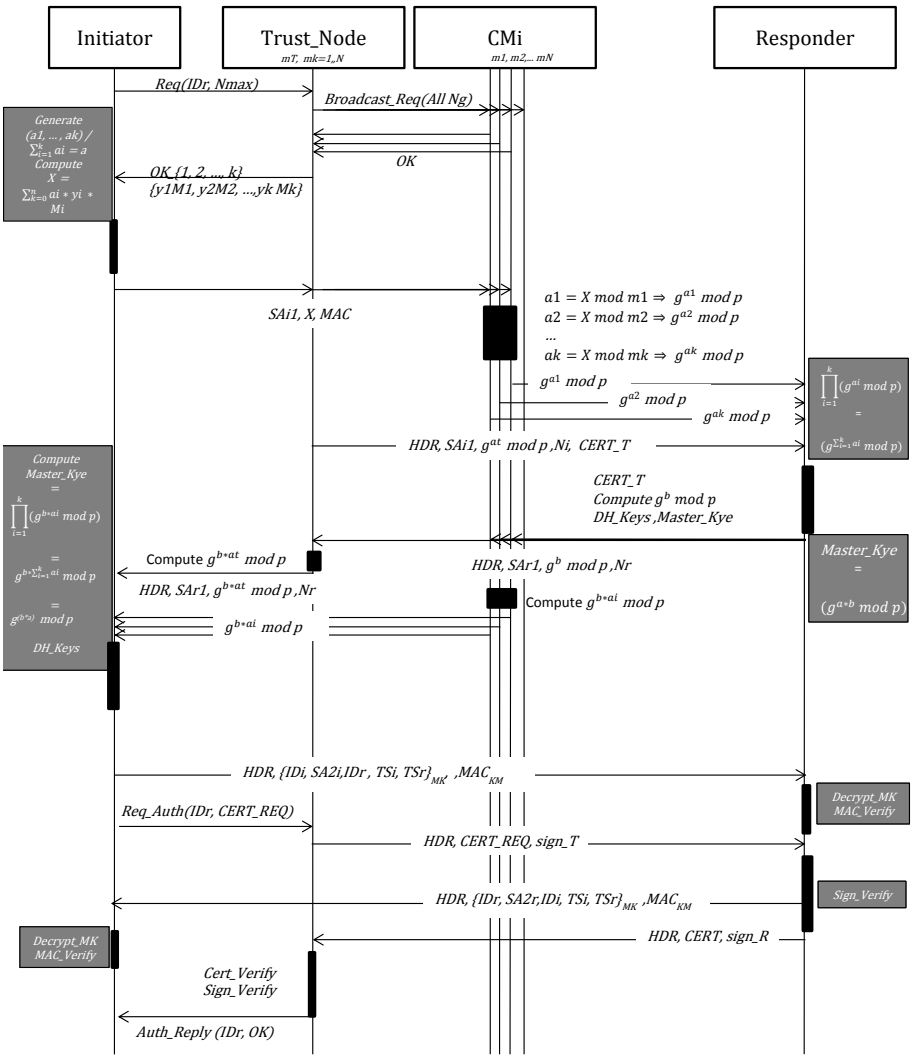

FIGURE 4.11 – *Echanges CKES*

système CRT, il envoie à l'agent T, le SAi1, le code d'authentification MAC et un message contenant X.

La transition 3 : Après la réception de la clé maîtresse (portion), l'agent 'A' calcule le produit de toutes les valeurs reçues afin de déterminer sa clé maitre.

```
%%  HLPSL:
role ch(
    A,B,N,C : agent,
    SND,RCV    : channel(dy),
    F          : function,
    PSK, KAN, KAC : symmetric_key,  %%PSK
    G      : text)
played_by A def=

  local
State: nat,
     Ni, SA1, SA2, DHX, S1, S2: text,
        Nr: text,
        KEr: message, %% more spefic: exp(text,text)
        SK: hash(text.text.text.message),
        AUTH_B: message

  const sec_a_SK : protocol_id
  init  State := 0
  transition

 1. State  = 0 /\ RCV(start) =|>
    State':= 1        /\SA1' := new()
                      /\ DHX' := new()
                      /\ Ni' := new()
                      /\ SND(SA1'.exp(G,S1').Ni' )
                  /\ SND(SA1'.exp(G,S2').Ni' )

 2. State  = 1 /\RCV(SA1.KEr'.Nr')  =|>
    State':= 2  /\ SA2' := new()
                /\ SK' := F(Ni.Nr'.SA1.exp(KEr',DHX))
                /\ SND( {A.F(PSK.SA1.exp(G,DHX).Ni.Nr').SA2'}_SK' )
             %%  /\ witness(A,B,sk2,F(Ni.Nr'.SA1.exp(KEr,DHX)))

 3. State  = 2 /\ RCV({B.F(PSK.SA1.KEr.Ni.Nr).SA2}_SK) =|>
    State':= 3 /\AUTH_B' := F(PSK.SA1.KEr.Ni.Nr)
               /\ secret(SK,sec_a_SK,{A,B}) %% ,N}
               /\ request(A,B,sk1,SK)
            /\ SND({A.SA2.SK}_KAN)
                 /\ witness(A,B,sk1,SK)
end role
```

FIGURE 4.12 – *Rôle du nœud collaboratif*

L'agent B. Il joue le rôle 'Répondant' qui répond à toute demande d'un échange IKE.

La transition 1 : B fait le produit des différentes valeurs reçues d'agents 'T' et 'Ci' afin de calculer la clé publique de A (DH_Pub),

La transition 2 : le nœud B envoie la valeur g^b mod p vers l'HTN ainsi que le paquet IKE (HDR, SAr1,Nr).

L'agent T. Il joue le rôle 'Trust' qui valide le choix de nœuds collaboratifs et participe aux processus collaboratif. La transition 1 : Dès

```
role host (
    A,B,N,C : agent,
    SND,RCV    : channel(dy),
    F          : function,
    PSK, KBN, KBC      : symmetric_key,
    G      : text)
played_by B def=

  local
     State: nat,
     Ni, SA1, SA2: text,
        Nr, DHY, MN, MB, S1, S2 : text,
        SK: hash(text.text.text.message),
        KEi: message,
        AUTH_A : message
  const sec_b_SK : protocol_id

  init  State := 0

  transition

1. State=0 /\ RCV(SA1'.exp(G,S1').Ni') =|>
           State':=1

2. State = 1  /\ RCV( SA1'.exp(G,S2').Ni' ) =|>


   State':= 2         /\ DHY' := new()
                      /\ Nr' := new()
                      /\ SND(SA1'.exp(G,DHY').Nr')
                      /\ SK' := F(Ni'.Nr'.SA1'.exp(KEi',DHY'))

  3. State = 2 /\ RCV( {A.F(PSK.SA1.KEi.Ni.Nr).SA2'}_SK ) =|>
     State':= 3 /\ AUTH_A' := F(PSK.SA1.KEi.Ni.Nr)
           /\ SND( {B.F(PSK.SA1.exp(G,DHY).Ni.Nr).SA2'}_SK )
                 /\ witness(B,A,sk2,SK)
           %%    /\ secret(SK,sec_b_SK,{A,B})
           %%    /\ request(B,A,sk2,SK)

  4. State = 3 /\ RCV({N.MN'}_SK) =|>  %% _SK

     State':= 4        /\ MB':=new()
                 %% /\ witness(B,N,sk3,SK)
                  /\ SND({B.MB'}_SK)  %% SK
                  /\ secret(SK,sec_b_SK,{A,B}) %% N
end role
```

FIGURE 4.13 – *Rôle du nœud Répondant*

la réception d'une demande CKES, L'agent T envoie, en multidiffusion, cette demande vers N nœuds qui pourraient être capables de traiter les opérations cryptographiques (dans cet exemple, nous avons pris un seul nœud collaboratif) La transition 2 : Suite au retour du nœud collaboratif 'Ci', l'agent 'T'envoie les 'k' identités des nœuds collaboratifs, acceptant la demande, vers l'initiateur A ainsi que les coefficients (y1*M1,y2*M2, ..., ykMk) donnés par le CH. La transition 3 : l'agent 'T'envoie, en multidiffusion, la valeur X vers tous les nœuds collaboratifs et envoie un paquet IKE, qui contient l'HDR, Sai1, Ni CERT_HTN vers le répondant 'B'. La transition 4 : 'T' détermine sa clé maitre g^{b*a_t} mod p et l'envoie vers l'initiateur.

L'agent Ci. Il joue le rôle 'Collaborative nœud' qui est considéré comme un nœud confiant La transition 1 : L'agent accepte la demande CKES en envoyant un message REP à l'agent 'T'. La transition 2 : Après la réception de X, chaque Ci calcule son propre secret 'ai' $(a_i = Xmod(m_i))$ qui lui permet par la suite de calculer la partie de DH et l'envoyer au répondant B. La transition 3 : L'agent Ci détermine sa clé maîtresse g^{b*a_i} mod p et l'envoie vers l'initiateur en incluant la partie de l'HTN. Enfin, nous définissons le rôle 'environnement' où le rôle 'intrus' s'ajoute. Nous indiquons toutes les données initialement connues par l'intrus à savoir les autres identités des autres intervenants, les clés publiques, etc. Par la suite, nous définissons toutes les propriétés de sécurité à vérifier. Cela se fait grâce aux prédicats suivant : secret, request witness...

Selon l'analyse réalisée par SPAN, tous les résultats donnent un 'SAFE', c'est-à-dire, le protocole mis en question est considéré comme un protocole sûr. Aucune faille de sécurité n'a été détectée.

```
role tn(
   A,B,N,C : agent,
    SND,RCV    : channel(dy),
    F          : function,
    KAN, KBN   : symmetric_key,
    G      : text)
played_by N def=

  local
     Ni, Nr, SA1, SA2: text,
        SK: hash(text.text.text.message),
        State: nat,
        MN,MB,S1 : text
  init
    State := 0

  transition

   1. State=0 /\ RCV(SA1'.S1'.Ni') =|>

      State':=1
           /\ S1':=new()
           /\ SND(SA1'.exp(G,S1').Ni')


  2. State = 1  /\ RCV({A.SA2'.SK'}_KAN) =|>

    State':= 2
                  /\ MN':=new()
                  /\ SND({N.MN'}_SK')    %%_SK'

 3. State = 2  /\ RCV({B.MB'}_SK) =|>

     State':= 3
                  /\ MN':=new()
                 /\ SND({N.MN'}_KBN)    %%_SK'
end role
```

FIGURE 4.14 – *Rôle du nœud Trust*

```
role environment()
def=

  const
    a,b,n,c       : agent,
    f             : function,
    sk1,sk2,sk3     : protocol_id,
    kab,kna,kbn,kac, kbc, kai,kbi, kni  : symmetric_key,
    g         : text

  intruder_knowledge = {a,b,n,c,f,kai,g}

composition
     session(a,b,n,c,f,kab,kna,kbn,kac,kbc,g)
%%  /\ session(n,a,b,f,kna,kbn,g)
 %% /\ session(b,a,n,f,kbn,kab,g)
%%  /\ session(i,a,n,f,kni,kna,kai,g)

end role
```

FIGURE 4.15 – *Rôle Environnement*

```
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/mkasraoui/span/testsuite/results/CKES.if
GOAL  as specified
BACKEND  OFMC
STATISTICS
  TIME 400 ms
  parseTime 0 ms
  visitedNodes: 428 nodes
  depth: 6 plies
```

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/mkasraoui/span/testsuite/results/CKES.if

GOAL
  As specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 10 states
  Reachable  : 4 states
  Translation: 0.06 seconds
  Computation: 0.00 seconds
```

FIGURE 4.16 – *Résultats de la vérification formelle*

Conclusion du chapitre

Dans ce chapitre, nous avons introduit des notions de base sur les méthodes formelles, puis nous avons spécifié formellement notre approche CKES présentée dans le second chapitre. La solution a été par la suite validée par l'outil AVISPA / SPAN en montrant sa fiabilité et son efficacité à contrer des attaques.

Conclusion générale et perspectives

Dans ces dernières années, les réseaux de capteurs sans fil ont connu un énorme progrès et une remarquable évolution permettant ainsi de les exploiter et généraliser leurs utilisations dans la plupart des domaines. Certaines caractéristiques des RCSFs telles que la mobilité, la simple configuration ou l'interopérabilité leurs permettent d'être plus efficaces relativement à d'autres technologies sur plusieurs points. Mais certaines autres caractéristiques à savoir l'absence de la protection physique, l'environnement de déploiement hostile et la limitation de ressources, posent des véritables contraintes impactant l'évolution des RCSFs.

Dans cette thèse, nous avons focalisé notre recherche sur l'aspect routage et sécurité tout en tenant compte de la consommation énergétique. Nos contributions répondent à des problématiques qui n'étaient pas encore résolues par la communauté scientifique.

Concernant le routage, nous avons traité des problématiques liées au passage à l'échelle des protocoles de routage (scalability) et l'optimisation des chemins de routage. Notre contribution consiste à proposer le protocole ZBR-M qui est un protocole de routage hiérarchique permettant d'identifier des raccourcis vers la destination avec le moindre nombre de sauts. Cela a permis d'éviter les liens fils-parents classiques proposés par le protocole hiérarchique de base et d'utiliser les liens entre les nœuds voisins à un seul saut tout en garantissant l'accès à la destination. Par rapport à la sécurité, la majeure difficulté était au niveau de l'adaptation de l'IKEv2 aux réseaux de capteurs sachant qu'il est considéré comme un protocole très énergivore. Notre contribution consiste à adapter le protocole de sécurité IKEv2, aux RCSFs. Dans le premier chapitre, nous avons réalisé un état de l'art. Nous avons commencé par l'introduction

des réseaux de capteurs sans fil (RCSFs) au sens large du terme, ensuite nous avons présenté les travaux existants sur le routage, la sécurité et la gestion des clés dans ce type de réseaux. Nous avons effectué des études comparatives des différentes solutions existantes et nous avons identifié les contraintes à surmonter dans les RCSFs. En effet, Les ressources limitées des RCSFs présentent une contrainte à surmonter pour contrer les attaques sur ces réseaux à moindre coût. D'un point de vue énergétique, il est plus judicieux d'utiliser des mécanismes cryptographiques symétriques, nécessitant moins de ressources, que des mécanismes cryptographiques à clé publique qui sont beaucoup plus complexes et énergivores. Pour cette raison, plusieurs protocoles de routage sécurisés ont implémenté des mécanismes symétriques et parfois hybrides afin d'économiser de l'énergie tout en assurant un niveau de sécurité efficace, sûr et complet. Cette étape d'étude bibliographique bien qu'elle soit longue et fastidieuse est une étape primordiale qui nous a aidé par la suite à nous aiguiller vers le bon choix et assurer la réussite de nos travaux.

Dans le chapitre 2, nous avons pu développer des connaissances approfondies sur les réseaux de capteurs sans fil et notamment sur les réseaux ZigBee. Nous avons également étudié une panoplie de protocoles de routage proposés par le standard ZigBee. Nous avons conduit des simulations pour évaluer les performances du routage proposé par ZigBee Alliance tout en le comparant au routage à la demande afin d'identifier les caractéristiques du routage hiérarchique ainsi que ses déficiences. Les résultats de simulations ont montré que le routage hiérarchique de base présente de meilleurs délais et taux de délivrance permettant ainsi une disponibilité de service indépendamment de la taille du réseau. Cependant ce type de routage ne tient pas pour une longue durée à cause de sa nature statique. De même, en tenant compte de l'overhead, le routage hiérarchique en génère moins que le protocole de routage AODV. Dans le chapitre 3, nous avons commencé par une étude comparative de différentes solutions de sécurité de bout en bout dans les RCSFs. Par la suite nous avons présenté une nouvelle approche

collaborative appelée CKES (Collaborative Key Exchange System) qui permet de sécuriser les données échangées entre les nœuds tout en tenant compte de leurs contraintes énergétiques. Nous avons utilisé l'outil NS2 (Network Simulator 2) pour valider notre approche CKES et analyser ses performances en termes d'énergie et d'autres métriques réseaux. Selon les résultats de la simulation, nous avons prouvé que le protocole CKES est plus performant que l'IKEv2 (Internet Key Exchange version 2), une composante d'IPSec, en termes de la consommation énergétique. Cette dernière est liée à la taille des données échangées au niveau de chaque nœud contraint durant les phases d'établissement des associations de sécurité. Nous avons également comparé la consommation énergétique liée au calcul des différentes opérations cryptographiques au niveau de chaque nœud contraint. Ce coût dépend de la complexité des algorithmes de sécurité utilisés ainsi que du nombre des instructions effectuées lors du calcul. Afin de prouver l'efficacité de notre système collaboratif contre les attaques externes et sa capacité à répondre aux critères fondamentaux de la sécurité (authentification, intégrité, confidentialité), le dernier chapitre a été consacré à la validation formelle du protocole. En effet, nous avons spécifié formellement le protocole CKES, ensuite nous l'avons validé en utilisant l'outil SPAN/AVISPA. Cette validation a été faite en utilisant seulement deux nœuds collaboratifs. Mais cela n'empêche pas que les résultats soient les mêmes avec plusieurs nœuds collaboratifs. Les résultats de la validation ont prouvé la fiabilité et l'efficacité du protocole CKES à contrer tout type d'attaque.

Perspectives

Dans la continuité de ces travaux, nous envisageons d'optimiser la consommation énergétique du protocole de routage ZBR-M. L'idée proposée consiste à définir périodiquement des passerelles pertinentes qui dépendraient du trafic de données. Ces passerelles formeraient ainsi des raccourcis pour transiter le trafic en suivant les chemins les plus optimaux. Pour cela, nous comptons utiliser des techniques d'apprentissage pour identifier ces passerelles dans une phase hors-ligne (offline). Une fois la phase d'apprentissage faite, le coordinateur serait capable d'identifier

ultérieurement ces passerelles en fonction de l'état de chaque nœud dans le réseau. Le choix de passerelles dépend en effet de l'énergie résiduelle de chaque nœud, la charge ainsi que le temps du traitement. Enfin, il serait intéressant d'implémenter nos approches sur une plateforme réelle. Plusieurs tests expérimentaux devraient être réalisés afin de comparer les résultats de simulation à ceux obtenus expérimentalement.

Bibliographie

ZigBee Specification, 2005. URL http://www.zigbee.org/en/index.asp. (Cité page 60.)

Ahmed E. A. A. Abdulla, Hiroki Nishiyama, et Nei Kato. Extending the lifetime of wireless sensor networks : A hybrid routing algorithm. *Comput. Commun.*, 35(9) :1056–1063, Mai . ISSN 0140-3664. URL http://dx.doi.org/10.1016/j.comcom.2011.10.001. (Cité page 35.)

I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, et E. Cayirci. Wireless sensor networks : A survey. *Comput. Netw.*, 38(4) :393–422, Mars . ISSN 1389-1286. URL http://dx.doi.org/10.1016/S1389-1286(01)00302-4. (Cité page 9.)

Jamal N. Al-karaki et Ahmed E. Kamal. Routing techniques in wireless sensor networks : A survey. *IEEE Wireless Communications*, 11 :6–28, 2004. (Cité pages 11, 19 et 20.)

Md Kausar Alam et Sharmila Banu K. An approach secret sharing algorithm in cloud computing security over single to multi clouds. (Cité page 98.)

Miguel Angel, Erazo Villegas, Seok Yee Tang, et Yi Qian. Wireless sensor network communication architecture for wide-area large scale soil moisture estimation and wetlands monitoring network communications infrastructure group. (Cité page 9.)

A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P. C. Heám, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santiago, M. Turuani, L. Viganò, et L. Vigneron. The avispa tool for the automated validation of internet security protocols and applications.

Dans *Proceedings of the 17th International Conference on Computer Aided Verification*, CAV'05, pages 281–285, Berlin, Heidelberg, 2005. Springer-Verlag. ISBN 3-540-27231-3, 978-3-540-27231-1. URL http://dx.doi.org/10.1007/11513988_27. (Cité page 119.)

AVIS. Avis. URL http://www.avispa-project.org/. (Cité page 111.)

G.R. Blakley. Safeguarding cryptographic keys. Dans *Proceedings of the 1979 AFIPS National Computer Conference*, pages 313–317, Monval, NJ, USA, 1979. AFIPS Press. (Cité page 97.)

Boudra. Un prototype de systÃ¨me de tÃ©lÃ©surveillance mÃ©dicale basÃ© sur les capteurs et les rÃ©seaux de capteurs sans fil. 2014. URL http://www.archipel.uqam.ca/6035/. (Cité page 13.)

David Braginsky et Deborah Estrin. Rumor routing algorthim for sensor networks. Dans *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA ,2002, pages 22–31, New York, NY, USA. ACM. ISBN 1-58113-589-0. URL http://doi.acm.org/10.1145/570738.570742. (Cité pages 30 et 31.)

Michael Burrows, Martin Abadi, et Roger Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8(1) :18–36, Février 1990. ISSN 0734-2071. URL http://doi.acm.org/10.1145/77648.77649. (Cité page 116.)

Tiago Camilo, Jorge SÃ¡ Silva, et O Boavida. Some notes and proposals on the use of ip-based approaches in wireless sensor networks. (Cité page 17.)

Er Casado. Contikisec : A secure network layer for wireless sensor networks under the contiki operating system. (Cité page 84.)

Lander Casado et Philippas Tsigas. Contikisec : A secure network layer for wireless sensor networks under the contiki operating system. Dans Audun JÃ¸sang, Torleiv Maseng, et SveinJohan Knapskog, éditeurs, *Identity and Privacy in the Internet Age*, volume 5838 de *Lecture*

Notes in Computer Science, pages 133–147. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-04765-7. URL http://dx.doi.org/10.1007/978-3-642-04766-4_10. (Cité page 84.)

Seyit A. Çamtepe et Bülent Yener. Combinatorial design of key distribution mechanisms for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 15(2) :346–358, Avril 2007. ISSN 1063-6692. URL http://dx.doi.org/10.1109/TNET.2007.892879. (Cité page 86.)

Haowen Chan, A. Perrig, et D. Song. Random key predistribution schemes for sensor networks. Dans *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, pages 197–213, May 2003. (Cité page 86.)

Jae-Hwan Chang et L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. Dans *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 22–31 vol.1, 2000. (Cité page 68.)

Deji Chen, Mark Nixon, et Aloysius Mok. *WirelessHART : Real-Time Mesh Network for Industrial Automation*. Springer Publishing Company, Incorporated, 1st édition, 2010a. ISBN 1441960465, 9781441960467. (Cité page 50.)

Shanshan Chen, Geng Yang, et Shengshou Chen. A security routing mechanism against sybil attack for wireless sensor networks. Dans *Communications and Mobile Computing (CMC), 2010 International Conference on*, volume 1, pages 142–146, April 2010b. (Cité page 44.)

Xiangqian Chen, Kia Makki, Kang Yen, et N. Pissinou. Sensor network security : a survey. *Communications Surveys Tutorials, IEEE*, 11(2) :52–73, Second 2009. ISSN 1553-877X. (Cité page 86.)

Maurice Chu, Horst Haussecker, Feng Zhao, Maurice Chu, Horst Haussecker, et Feng Zhao. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *International Journal of High Performance Computing Applications,2002*, 16. (Cité page 32.)

Edmund M. Clarke et E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. Dans *Logic of Programs, Workshop*, pages 52–71, London, UK, UK, 1982. Springer-Verlag. ISBN 3-540-11212-X. URL http://dl.acm.org/citation.cfm?id=648063.747438. (Cité page 114.)

W. Colitti, K. Steenhaut, N. De Caro, B. Buta, et V. Dobrota. Rest enabled wireless sensor networks for seamless integration with web applications. Dans *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, pages 867–872, Oct 2011. (Cité page 80.)

COOJ. Cooj. URL http://anrg.usc.edu/contiki/index.php/. (Cité page 91.)

T Culter. Deploying zigbee in existing industrial automation networks. Dans *Industrial Embedded System Resource Guide, Networking : Technology*, pages 34–36, 2005. (Cité page 50.)

G. de Meulenaer, F. Gosset, O.-X. Standaert, et O. Pereira. On the energy cost of communication and cryptography in wireless sensor networks. Dans *Networking and Communications, 2008. WIMOB '08. IEEE International Conference on Wireless and Mobile Computing,*, pages 580–585, Oct 2008. (Cité pages 44 et 99.)

LahcÃ¨ne Dehni, YounÃ¨s Bennani, et Francine Krief. Lea2c : Low energy adaptive connectionist clustering for wireless sensor networks. Dans Thomas Magedanz, Ahmed Karmouch, Samuel Pierre, et Iakovos Venieris, éditeurs, *Mobility Aware Technologies and Applications*, volume 3744 de *Lecture Notes in Computer Science*, pages 405–415. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-29410-8. URL http://dx.doi.org/10.1007/11569510_39. (Cité page 50.)

F. Delgosha et F. Fekri. Key pre-distribution in wireless sensor networks using multivariate polynomials. Dans *Sensor and Ad Hoc Communications and Networks, 2005. IEEE SECON 2005. 2005 Second Annual IEEE Communications Society Conference on*, pages 118–129, Sept 2005. (Cité page 85.)

I. Demirkol, C. Ersoy, et F. Alagoz. Mac protocols for wireless sensor networks : A survey. *Comm. Mag.*, 44(4) :115–121, Septembre 2006. ISSN 0163-6804. URL http://dx.doi.org/10.1109/MCOM.2006.1632658. (Cité page 13.)

Yvo Desmedt. Some recent research aspects of threshold cryptography. Dans Eiji Okamoto, George Davida, et Masahiro Mambo, éditeurs, *Information Security*, volume 1396 de *Lecture Notes in Computer Science*, pages 158–173. Springer Berlin Heidelberg, 1998. ISBN 978-3-540-64382-1. URL http://dx.doi.org/10.1007/BFb0030418. (Cité page 98.)

P. Devalan. *Introduction à la mécatronique*. Ed. Techniques Ingénieur. URL http://books.google.fr/books?id=qHUv6-KhISAC. (Cité pages 5 et 9.)

DHE. Dhe. URL https://etd.ohiolink.edu/rws_etd/document/get/ysu1253597142/inline. (Cité page 99.)

W. Diffie et M. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theor.*, 22(6) :644–654, Septembre . ISSN 0018-9448. URL http://dx.doi.org/10.1109/TIT.1976.1055638. (Cité page 37.)

D. Dolev et Andrew C. Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2) :198–208, Mar 1983. ISSN 0018-9448. (Cité page 121.)

S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G-S. Ahn, et A. T. Campbell. The bikenet mobile sensing system for cyclist experience mapping. Dans *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, SenSys '07, pages 87–101, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-763-6. URL http://doi.acm.org/10.1145/1322263.1322273. (Cité page 13.)

Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. Dans *Proceedings of CRYPTO 84 on Advances in Cryptology*, pages 10–18, New York, NY, USA. Springer-Verlag New York, Inc. ISBN 0-387-15658-5. URL http://dl.acm.org/citation.cfm?id=19478.19480. (Cité page 39.)

A. Elahi et A. Gschwender. *ZigBee Wireless Sensor and Control Network*. Prentice Hall Communications Engineering and Emerging Technologies Series. Pearson Education, 2009. ISBN 9780137059409. URL http://books.google.fr/books?id=481PeRPyiEoC. (Cité page 13.)

ENORASIS. Enorasis. 2014. URL http://www.enorasis.eu. (Cité page 13.)

Christian C. Enz, Amre El-Hoiydi, Jean-Dominique Decotignie, et Vincent Peiris. Wisenet : An ultralow-power wireless sensor network solution. *IEEE Computer*, 37(8) :62–70, 2004. URL http://dblp.uni-trier.de/db/journals/computer/. (Cité page 14.)

Laurent Eschenauer et Virgil D. Gligor. A key-management scheme for distributed sensor networks. Dans *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS , 2002, pages 41–47, New York, NY, USA. ACM. ISBN 1-58113-612-9. URL http://doi.acm.org/10.1145/586110.586117. (Cité page 46.)

Laurent Eschenauer et Virgil D. Gligor. A key-management scheme for distributed sensor networks. Dans *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 41–47, New York, NY, USA, 2002. ACM. ISBN 1-58113-612-9. URL http://doi.acm.org/10.1145/586110.586117. (Cité page 86.)

ZoltÃ¡n Faigl, Stefan Lindskog, et Anna Brunstrom. A measurement study on ikev2 authentication performance in wireless networks. (Cité page 95.)

Jacob Fraden. *Handbook of Modern Sensors : Physics, Designs, and Applications (Handbook of Modern Sensors)*. SpringerVerlag. ISBN 0387007504, 2004. (Cité page 16.)

A. Giridhar et P.R. Kumar. Maximizing the functional lifetime of sensor networks. Dans *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 5–12, April 2005. (Cité page 68.)

Yann Glouche, Thomas Genet, Olivier Heen, et Olivier Courtay. A security protocol animator tool for avispa. Dans *In ARTIST-2 workshop*, 2006. (Cité page 120.)

J. Granjal, E. Monteiro, et J. Sa Silva. A secure interconnection model for ipv6 enabled wireless sensor networks. Dans *Wireless Days (WD), 2010 IFIP*, pages 1–6, Oct 2010. (Cité page 83.)

J. Granjal, R. Silva, E. Monteiro, J. Sa Silva, et F. Boavida. Why is ipsec a viable option for wireless sensor networks. Dans *Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference on*, pages 802–807, Sept 2008. (Cité page 99.)

V. Gupta, M. Millard, S. Fung, Yu. Zhu, N. Gura, H. Eberle, et S.C. Shantz. Sizzle : a standards-based end-to-end security architecture for the embedded internet. Dans *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on*, pages 247–256, March 2005. (Cité page 84.)

Jose A. Gutierrez, Edgar H. Callaway, et Raymond Barrett. *IEEE 802.15.4 Low-Rate Wireless Personal Area Networks : Enabling Wireless Sensor Networks*. IEEE Standards Office, New York, NY, USA, 2003. ISBN 0738135577. (Cité page 53.)

Mansoor Alam Prabir Bhattacharya H. L. Harsh Sundani, Vijay K. Devabhaktuni. Wireless sensor network simulators a survey and comparisons. *International Journal Of Computer Networks (IJCN)*, pages 04, 2011, 2011. (Cité page 92.)

HART. Hart. URL `http://www.hartcomm.org`. (Cité page 50.)

Wendi Rabiner Heinzelman, Anantha Chandrakasan, et Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. Dans *Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8 - Volume 8*, HICSS '00, pages 8020–, Washington, DC, USA, 2000, a. IEEE Computer Society. ISBN 0-7695-0493-0. URL

http://dl.acm.org/citation.cfm?id=820264.820485. (Cité pages 14, 44 et 99.)

Wendi Rabiner Heinzelman, Joanna Kulik, et Hari Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. Dans *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom ,1999, pages 174–185, New York, NY, USA, b. ACM. ISBN 1-58113-142-9. URL http://doi.acm.org/10.1145/313451.313529. (Cité pages 14 et 27.)

H.Soussi, M.Hussain, H.Afifi, et D.Seret. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 1 (6) :1779 – 1782, 2007. ISSN 1307-6892. URL http://waset.org/Publications?p=6. (Cité page 95.)

J. Hui, Arch Rock Corporation, et P. Thubert. Compression format for ipv6 datagrams. (Cité page 84.)

T. Muntean I. Memon. Cluster-based energy-effcient composite event detection for wireless sensor networks. Dans *Sixth International Conference on Sensor Technologies and Applications*, SENSORCOMM 2012, pages 260–269, 2012. ISBN 978-1-61208-207-3. (Cité page 20.)

IEEE802. Ieee802. URL http://www.ieee802.org/15/pub/TG4.html. (Cité page 53.)

IETF. Ietf. URL https://www.ietf.org/. (Cité page 17.)

Chalermek Intanagonwiwat, Ramesh Govindan, et Deborah Estrin. Directed diffusion : A scalable and robust communication paradigm for sensor networks. Dans *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom ,2000, pages 56–67, New York, NY, USA. ACM. ISBN 1-58113-197-6. URL http://doi.acm.org/10.1145/345910.345920. (Cité pages 25, 28, 30 et 44.)

ISA. Isa. URL http://www.isa.org. (Cité page 50.)

Teerawat Issariyakul et Ekram Hossain. *Introduction to Network Simulator NS2*. Springer Publishing Company, Incorporated, 1 édition, 2008. ISBN 0387717595, 9780387717593. (Cité page 91.)

Wooyoung Jung, Sungmin Hong, Minkeun Ha, Young-Joo Kim, et Daeyoung Kim. Ssl-based lightweight security of ip-based wireless sensor networks. Dans *Advanced Information Networking and Applications Workshops, 2009. WAINA '09. International Conference on*, pages 1112–1117, May 2009. (Cité page 84.)

Raja Jurdak, Antonio G. Ruzzelli, et Gregory M. P. O'Hare. Radio sleep mode optimization in wireless sensor networks. *IEEE Transactions on Mobile Computing*, 9(7) :955–968, Juillet . ISSN 1536-1233. URL `http://dx.doi.org/10.1109/TMC.2010.35`. (Cité page 15.)

Rouba El Kaissi, Ayman Kayssi, Ali Chehab, et Zaher Dawy. Dawwsen : a defence mechanism against wormhole attacks in wireless sensor networks howpublished = Proceedings of the 2nd International Conference on Innovations in Information Technology (IIT'05), Dubai, UAE year = 2005. (Cité page 45.)

Ramaraju Kalidindi, Rajgopal Kannan, S. Sitharama Iyengar, et Lydia Ray. Distributed energy aware mac layer protocol for wireless sensor networks. Dans Weihua Zhuang, Chi-Hsiang Yeh, Olaf Droegehorn, C.-T. Toh, et Hamid R. Arabnia, éditeurs, *International Conference on Wireless Networks*, pages 282–286. CSREA Press, 2003. ISBN 1-932415-03-3. URL `http://dblp.uni-trier.de/db/conf/icwn/`. (Cité page 14.)

Brad Karp et H. T. Kung. Gpsr : Greedy perimeter stateless routing for wireless networks. Dans *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom ,2000, pages 243–254, New York, NY, USA. ACM. ISBN 1-58113-197-6. URL `http://doi.acm.org/10.1145/345910.345953`. (Cité page 24.)

M. Kasraoui, A. Cabani, et H. Chafouk. Formal verification of wireless sensor key exchange protocol using avispa. Dans *Computer, Consumer*

and Control (IS3C), 2014 International Symposium on, pages 387–390, June 2014a. (Cité page 123.)

M. Kasraoui, A. Cabani, et H. Chafouk. Collaborative key exchange system based on chinese remainder theorem in heterogeneous wireless sensor networks. *International Journal of Distributed Sensor Networks*, in press., 2015a. ISSN 1550-1329. (Cité page 96.)

M. Kasraoui, A. Cabani, et H. Chafouk. Secure collaborative system in heterogenous wireless sensor networks. *Journal of Applied Research and Technology*, 13(2) :342 – 350, 2015b. ISSN 1665-6423. URL http://www.sciencedirect.com/science/article/pii/S1665642315000188. (Cité page 95.)

M. Kasraoui, A. Cabani, et J. Mouzna. Routage dans les rÃ©seaux de capteurs sans fil. Dans *The 1st IEEE International Conference on Logistics Operations Management (GOL 2012)*, 2012. (Cité page 65.)

Mohamed Kasraoui, Adnane Cabani, et Houcine Chafouk. Ikev2 authentication exchange model in ns-2. Dans *Proceedings of the 2014 International Symposium on Computer, Consumer and Control*, IS3C '14, pages 1074–1077, Washington, DC, USA, 2014b. IEEE Computer Society. ISBN 978-1-4799-5277-9. URL http://dx.doi.org/10.1109/IS3C.2014.280. (Cité page 110.)

M. Miran K ksal. A survey of network simulators supporting wireless networks. *Middle East Technical University, Ankara, Turkey,*, pages 10, 2008. (Cité page 91.)

Jooyoung Lee et DouglasR. Stinson. Deterministic key predistribution schemes for distributed sensor networks. Dans Helena Handschuh et M.Anwar Hasan, éditeurs, *Selected Areas in Cryptography*, volume 3357 de *Lecture Notes in Computer Science*, pages 294–307. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-24327-4. URL http://dx.doi.org/10.1007/978-3-540-30564-4_21. (Cité page 85.)

Philip Levis, Nelson Lee, Matt Welsh, et David Culler. Tossim : Accurate and scalable simulation of entire tinyos applications. Dans

Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, SenSys '03, pages 126–137, New York, NY, USA, 2003. ACM. ISBN 1-58113-707-9. URL http://doi.acm.org/10.1145/958491.958506. (Cité page 91.)

Stephanie Lindsey et Cauligi S. Raghavendra. PEGASIS : Power-efficient gathering in sensor information systems. (Cité page 21.)

Donggang Liu et Peng Ning. Multilevel μtesla : Broadcast authentication for distributed sensor networks. *ACM Trans. Embed. Comput. Syst.*, 3(4) :800–836, Novembre 2004. ISSN 1539-9087. URL http://doi.acm.org/10.1145/1027794.1027800. (Cité page 46.)

Dijun Luo, Xiaojun Zhu, Xiaobing Wu, et Guihai Chen. Maximizing lifetime for the shortest path aggregation tree in wireless sensor networks. Dans *INFOCOM, 2011 Proceedings IEEE*, pages 1566–1574, April 2011. (Cité page 68.)

S. Magotra et K. Kumar. Detection of hello flood attack on leach protocol. Dans *Advance Computing Conference (IACC), 2014 IEEE International*, pages 193–198, Feb 2014. (Cité page 44.)

M. Mambo, K. Usuda, et E. Okamoto. Proxy signatures : Delegation of the power to sign messages. Dans *IEICE Trans. Fundamentals*, volume E79-A, pages 1338–1353, Septembre 1996. (Cité page 98.)

MAN. Man. URL https://www.mannasim.dcc.ufmg.br/. (Cité page 92.)

Arati Manjeshwar et Dharma P. Agrawal. Apteen : A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. Dans *Proceedings of the 16th International Parallel and Distributed Processing Symposium*, IPDPS ,2002, pages 48–, Washington, DC, USA, a. IEEE Computer Society. ISBN 0-7695-1573-8. URL http://dl.acm.org/citation.cfm?id=645610.662036. (Cité page 22.)

Arati Manjeshwar et Dharma P. Agrawal. Teen : A routing protocol for

enhanced efficiency in wireless sensor networks. Dans *in Proc. IPDPS 2001 Workshops*, b. (Cité pages 14 et 21.)

V.P. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, et N. Shroff. A minimum cost heterogeneous sensor network with a lifetime constraint. *Mobile Computing, IEEE Transactions on*, 4(1) :4–15, Jan 2005. ISSN 1536-1233. (Cité page 68.)

J-F. Monin. *Introduction aux mÃ©thodes formelles*. CTST. HermÃ¨s, 2000. (Cité page 114.)

G. Montenegro, N. Kushalnagar, J. Hui, et D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944 (Proposed Standard). (Cité page 18.)

MSP430. Msp430. URL `http://www.ti.com/lit/an/slaa515b/slaa515b.pdf`. (Cité page 15.)

LÃ©on Mugwaneza, Traian Muntean, et Ibrahima Sakho. A deadlock free routing algorithm with network size independent buffering space. volume 457 de *Lecture Notes in Computer Science*, pages 489–501. Springer Berlin Heidelberg, 1990. ISBN 978-3-540-53065-7. (Cité page 50.)

Dragos Niculescu et Nec Laboratories America. Topics in ad hoc networks communication paradigms for sensor networks. (Cité page 20.)

Leonardo B. Oliveira, Adrian Ferreira, Marco A. Vilaça, Hao Chi Wong, Marshall Bern, Ricardo Dahab, et Antonio A. F. Loureiro. Secleach-on the security of clustered sensor networks. *Signal Process.*, 87(12) :2882–2895, Décembre . ISSN 0165-1684. URL `http://dx.doi.org/10.1016/j.sigpro.2007.05.016`. (Cité page 45.)

OPNET. Opnet. URL `http://www.riverbed.com`. (Cité page 70.)

C. Perkins, E. Royer, et S. Das. Rfc 3561 ad hoc on-demand distance vector (aodv) routing. Rapport technique, 2003. URL `http://tools.ietf.org/html/rfc3561`. (Cité page 61.)

Adrian Perrig, Robert Szewczyk, J. D. Tygar, Victor Wen, et David E. Culler. Spins : Security protocols for sensor networks. *Wirel. Netw.*, 8

(5) :521–534, Septembre . ISSN 1022-0038. URL http://dx.doi.org/10.1023/A:1016598314198. (Cité page 46.)

Jeevan L J Pinto et Manjaiah D. H. Article : Modified distributed energy efficient clustering routing protocol for wireless sensor networks. *International Journal of Computer Applications*, 81(19) :38–42, 2013, November . Full text available. (Cité page 35.)

Prof. Prof. URL http://proverif.rocq.inria.fr/. (Cité page 111.)

Vijay Raghunathan, Curt Schurgers, Sung Park, Mani Srivastava, et Barclay Shaw. Energy-aware wireless microsensor networks. Dans *IEEE Signal Processing Magazine,2002*, pages 40–50, a. (Cité page 15.)

Vijay Raghunathan, Curt Schurgers, Sung Park, Mani Srivastava, et Barclay Shaw. Energy-aware wireless microsensor networks. Dans *IEEE Signal Processing Magazine*, pages 40–50, 2002, b. (Cité pages 16 et 31.)

S. Raza, S. Duquennoy, T. Chung, D. Yazar, T. Voigt, et U. Roedig. Securing communication in 6lowpan with compressed ipsec. Dans *Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference on*, pages 1–8, June 2011. (Cité pages 83 et 84.)

RFC5996. Rfc5996. URL http://tools.ietf.org/html/rfc5996. (Cité page 88.)

R. L. Rivest, A. Shamir, et L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2) :120–126, Février . ISSN 0001-0782. URL http://doi.acm.org/10.1145/359340.359342. (Cité page 38.)

V. Rodoplu et T. H. Meng. Minimum energy mobile wireless networks. *IEEE J.Sel. A. Commun.*, 17(8) :1333–1344, Septembre . ISSN 0733-8716. URL http://dx.doi.org/10.1109/49.779917. (Cité page 26.)

Priyanka Sadananda, Wassim Trojet, et Joseph Mouzna. Article : Multicast authentication framework for hierarchical networks using chinese remainder theorem. *International Journal of Computer Applications*, 82(11) : 1–7, November 2013. Full text available. (Cité page 98.)

Ozgur Koray Sahingoz. Large scale wireless sensor networks with multi-level dynamic key management scheme. *Journal of Systems Architecture*, 59(9) :801 – 807, 2013. ISSN 1383-7621. URL http://www.sciencedirect.com/science/article/pii/S1383762113001033. (Cité page 37.)

Scy. Scy. URL http://www.cs.ox.ac.uk/people/cas.cremers/scyther/. (Cité page 111.)

R.A. Shaikh, H. Jameel, B.J. d'Auriol, Heejo Lee, Sungyoung Lee, et Young-Jae Song. Group-based trust management scheme for clustered wireless sensor networks. *Parallel and Distributed Systems, IEEE Transactions on*, 20(11) :1698–1712, Nov 2009. ISSN 1045-9219. (Cité pages 99 et 102.)

Adi Shamir. How to share a secret. *Commun. ACM*, 22(11) :612–613, Novembre 1979. ISSN 0001-0782. URL http://doi.acm.org/10.1145/359168.359176. (Cité page 97.)

F. Silva. Industrial wireless sensor networks : Applications, protocols, and standards [book news]. *Industrial Electronics Magazine, IEEE*, 8(4) :67–68, Dec 2014. ISSN 1932-4529. (Cité page 50.)

Gopinath R. Sinniah, Zeldi Suryady, Reza Khoshdelniat, Usman Sarwar, et Mazlan Abbas. Ipv6 wireless sensor network gateway design and end-to-end performance analysis. Août 2012. (Cité page 80.)

J. Garmendia T. Muntean. A routing model for mobile agents. PDCS'00 Proceedings, 2000. (Cité page 33.)

Sapon Tanachaiwiwat, Pinalkumar Dave, Rohan Bhindwale, et Ahmed Helmy. Poster abstract secure locations : Routing on trust and isolating compromised sensors in location-aware sensor networks. Dans *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys , 2003, pages 324–325, New York, NY, USA. ACM. ISBN 1-58113-707-9. URL http://doi.acm.org/10.1145/958491.958542. (Cité page 47.)

TowerPower. The towerpower project has been kicked-off. 2014. URL http://www.wlbltd.eu/node/24. (Cité page 13.)

P. Varadarajan et G. Crosby. Implementing ipsec in wireless sensor networks. Dans *New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on*, pages 1–5, March 2014. (Cité page 99.)

A. Varga. Using the omnet++ discrete event simulation system in education. *Education, IEEE Transactions on*, 42(4) :11 pp.–, Nov 1999. ISSN 0018-9359. (Cité page 91.)

Emmanouil Vasilomanolakis, Shankar Karuppayah, Max Mühlhäuser, et Mathias Fischer. Taxonomy and survey of collaborative intrusion detection. *ACM Computing Surveys*, 47(55) :33, Mai 2015. ISSN 0360-0300. (Cité page 96.)

Xiaoyun Wang, Lizhen Yang, et Kefei Chen. Sdd : Secure directed diffusion protocol for sensor networks. Dans *Proceedings of the First European Conference on Security in Ad-hoc and Sensor Networks*, ESAS , 2005, pages 205–214, Berlin, Heidelberg. Springer-Verlag. ISBN 3-540-24396-8, 978-3-540-24396-0. URL `http://dx.doi.org/10.1007/978-3-540-30496-8_17`. (Cité pages 45 et 46.)

Hari B. Wendi R., Anantha C. Energy-efficient communication protocol for wireless microsensor networks. *IEEE In Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, pages 10, 2000. (Cité page 20.)

A. Wood. Denial of service in sensor networks. *Computer*, 35. (Cité page 44.)

A. Wood, G. Virone, T. Doan, Q. Cao, L. Selavo, Y. Wu, L. Fang, Z. He, S. Lin, et J. Stankovic. Alarm-net : Wireless sensor networks for assisted-living and residential monitoring. Rapport technique, 2006. (Cité page 13.)

Anthony D. Wood, Lei Fang, John A. Stankovic, et Tian He. Sigf : A family of configurable, secure routing protocols for wireless sensor networks. Dans *In Proceedings of ACM SASN, 2006*, pages 35–48. ACM Press. (Cité pages 45 et 47.)

Haixia Zhao ; Yaowei Li ; Mingchuan Zhang ; Ruijuan Zheng ; Qingtao Wu. A new secure geographical routing protocol based on location pairwise keys in wireless sensor networks. *International Journal of Computer Science Issues (IJCSI),2002*, 10(2) :365, Septembre . (Cité page 47.)

W. Meijuan X. Debao et Z. Ying. Secure-spin : secure sensor protocol for information via negotiation for wireless sensor networks. *in Proceedings of the 1st IEEE Conference on Industrial Electronics and Applications (ICIEA '06)*. (Cité page 47.)

Ya Xu, John Heidemann, et Deborah Estrin. Geography-informed energy conservation for ad hoc routing. Dans *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom ,2001, pages 70–84, New York, NY, USA, a. ACM. ISBN 1-58113-422-3. URL `http://doi.acm.org/10.1145/381677.381685`. (Cité page 15.)

Ya Xu, John Heidemann, et Deborah Estrin. Geography-informed energy conservation for ad hoc routing. Dans *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom ,2001, pages 70–84, New York, NY, USA, b. ACM. ISBN 1-58113-422-3. URL `http://doi.acm.org/10.1145/381677.381685`. (Cité page 26.)

Wei Ye, J. Heidemann, et D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *Networking, IEEE/ACM Transactions on*, 12(3) :493–506, June 2004. ISSN 1063-6692. (Cité page 14.)

Yan Yu, Ramesh Govindan, et Deborah Estrin. Geographical and energy aware routing : a recursive data dissemination protocol for wireless sensor networks. Rapport technique, 2001. (Cité pages 25 et 43.)

Dae Hyun Yum et Pil Joong Lee. Exact formulae for resilience in random key predistribution schemes. *Wireless Communications, IEEE Transactions on*, 11(5) :1638–1642, May 2012. ISSN 1536-1276. (Cité page 86.)

Li Zhou, Jinfeng Ni, et C.V. Ravishankar. Supporting secure communication and data collection in mobile sensor networks. Dans *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12, April 2006. (Cité pages 85 et 86.)

ZigBee. Zigbee. URL `http://www.ZigBee.org`. (Cité page 50.)

Ce document a été préparé à l'aide de l'éditeur de texte GNU Emacs et du logiciel de composition typographique LaTeX 2ε.

Titre Routage et sécurité à basse consommation d'énergie pour les réseaux de capteurs sans fil

Résumé Les avancées remarquables dans le domaine des télécommunications ont permis de supprimer les liaisons filaires de transmission fortement encombrantes en les substituant par des supports de communication sans fil. Ces réseaux sont particulièrement utiles dans des endroits disposant de peu d'infrastructures de communication et dont le déploiement est difficile ou ayant des contraintes spatiales et matérielles considérables. Malgré le gain important en flexibilité qu'ils offrent, les réseaux de capteurs sans fil (RCSFs) présentent trois problèmes incontournables. Le premier est la limitation des ressources en termes d'énergie, de mémoire et de temps de calcul. Le deuxième concerne le routage des informations collectées sur le réseau. En effet, le nombre de capteurs déployés, pouvant atteindre des milliers de nœuds, présente une contrainte forte dans la gestion des routes et le passage à l'échelle dans le réseau. Le troisième est la nature des RCSFs elle-même qui fait du réseau un milieu favorable et vulnérable à des attaques à savoir la falsification, la modification de données, le déni de service, etc. Dans le cadre de cette thèse, nous avons abordé deux aspects : le routage et la sécurité de bout en bout. Concernant le premier aspect, nous avons proposé une amélioration d'un protocole de routage hiérarchique, ZBR, afin d'optimiser l'acheminement des données d'une source vers une destination utilisant la technologie ZigBee. Des simulations ont été réalisées pour évaluer les performances du routage proposé par ZigBee Alliance tout en le comparant au routage à la demande, AODV, afin d'identifier les caractéristiques du routage hiérarchique ainsi que ses déficiences. Les résultats de simulations ont montré que le routage hiérarchique de base présente de meilleurs délais et taux de délivrance permettant ainsi une disponibilité de service indépendamment de la taille du réseau. Concernant le deuxième aspect, nous avons proposé une nouvelle approche collaborative appelée CKES (Collaborative Key Exchange System) afin de sécuriser les données échangées entre les nœuds dans les RCSFs tout en tenant compte de leurs contraintes énergétiques. Ainsi, nous avons adapté le protocole

de sécurité IPSec (Internet Protocol Security), conçu à la base pour sécuriser les échanges de bout en bout sur Internet, aux RCSFs de sorte que les opérations cryptographiques énergivores soient réparties sur un ensemble de nœuds. L'outil NS2 (Network Simulator 2) a été utilisé pour valider notre approche CKES et analyser ses performances en termes d'énergie et d'autres métriques réseaux. Selon les résultats de la simulation, nous avons prouvé que le protocole CKES est plus performant que l'IKEv2 (Internet Key Exchange version 2), une composante d'IPSec, en termes de la consommation énergétique. Cette dernière est liée à la taille des données échangées au niveau de chaque nœud contraint durant les phases d'établissement des associations de sécurité. Nous avons également comparé la consommation énergétique liée au calcul des différentes opérations cryptographiques au niveau de chaque nœud contraint. Ce coût dépend de la complexité des algorithmes de sécurité utilisés ainsi que du nombre des instructions effectuées lors du calcul. Une analyse formelle a été également effectuée à l'aide de l'outil AVISPA (Automated Validation of Internet Security Protocols and Applications) afin de prouver l'efficacité de notre système collaboratif contre les attaques externes et sa capacité à répondre aux critères fondamentaux de la sécurité (authentification, intégrité, confidentialité)

Mots-clés Sécurité de bout en bout, RCSF, Routage

Title Le titre en anglais

Abstract Le résumé en anglais (≈ 1000 caractères)

Keywords Les mots-clés en anglais