République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Batna 2
Faculté des Mathématiques et d'Informatique

Thèse

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

Mécanismes de sécurité pour l'intégration des RCSFs à l'IoT (Internet of Things)

Présentée Par

Somia SAHRAOUI

Soutenue le : 09/11/2016

Membres du jury :

Président :	*ZIDANI Abdelmadjid*	*Prof. Université de Batna 2*
Rapporteur :	*BILAMI Azeddine*	*Prof. Université de Batna 2*
Examinateurs :	*CHIKHI Salim*	*Prof. Université de Constantine 2*
	CHAOUI Allaoua	*Prof. Université de Constantine 2*
	BACHIR Abdelmalik	*Prof. Université de Biskra*
	SEGHIR Rachid	*Dr. Université de Batna 2*

Remerciements

~~~~~~

Je tiens d'abord à remercier le Bon Dieu pour me munir du courage, de la force et de la patience tout au long de mon parcours.

J'exprime mes sincères remerciements et toute ma gratitude à mon directeur de thèse Professeur Bilami Azeddine pour son excellente qualité d'encadrement et pour les orientations et les consignes pertinentes qu'il m'a accordées, et qui m'ont été très utiles durant les différentes phases de réalisation de cette thèse. Je le remercie également pour sa précieuse disponibilité et pour me permettre de partager ses connaissances et sa grande expérience dans mon domaine de recherche.

Je remercie vivement les honorables membres du jury qui ont accepté d'évaluer mon travail. Je remercie Professeur Zidani Abdelmadjid qui a bien voulu présider le jury. Je remercie également les examinateurs : Professeur Chaoui Allaoua, Professeur Chikhi Salim, Professeur Bachir Abdelmalik et Docteur Seghir Rachid pour l'intérêt qu'ils ont porté à mon travail.

Je voudrais remercier mes chers parents et mes frères pour leur accompagnement et leurs encouragements permanents. Ils n'ont préservé aucun effort pour m'aider à mener à succès mes études.

Je remercie Docteur Bouam Souheila pour son amitié et ses encouragements. Je la remercie également pour ses efforts administratifs. Je remercie Docteur Sedrati Maamar pour ses encouragements.

Je n'oublierais pas de remercier mes collègues les doctorants du laboratoire LaSTIC, à leur tête les membres de l'équipe IoT.

Je remercie sincèrement tous les gens honnêtes et justes dont j'ai eu l'honneur de connaitre dans ma vie. J'espère qu'ils se reconnaissent en lisant ces mots.

~~~~~~

ملخص :

إنترنت الأشياء (IoT) هو نموذج واعد يقوم على تمديد الربط بشبكة الإنترنت ليشمل أنواع مختلفة من الأشياء الذكية غير الحواسيب والهواتف النقالة، مما يسمح بتطوير أسلوب الحياة و تحسين جودة الخدمات في عدة مجالات. شبكات الاستشعار اللاسلكية (RCSFs) باعتبارها عنصرا حيويا في إنترنت الأشياء، تسمح بتمثيل الخصائص الديناميكية للعالم الحقيقي في العالم الافتراضي للإنترنت. على هاذ الأساس تم إستحداث نسخة مضغوطة من البروتوكول IPv6 لإنترنت الأشياء، مما يمكن من عنونة الآلاف أو حتى الملايين من أجهزة الإستشعارالمتصلة بشبكة الأنترنت. و عليه تصبح هاته الأخيرة مضيفات IP حقيقية و تدخل تطبيقاتها ضمن خدمات الويب. في الواقع، نضج إنترنت الأشياء و نجاحه يعتمد مما لا شك فيه على أمن الاتصالات وحماية خصوصية المستخدمين. و لكن الإختلاف التكنولوجي والمادي، و كذلك الطبيعة غير المتكافئة التي تميز الاتصالات بين عقد الاستشعار و المحطات العادية المتصلة بالإنترنت، تجعل من تأمين إنترنت الأشياء إشكالية كبيرة. في هذا السياق، تم إقتراح العديد من الحلول لتوحيد إستراتيجيات تامين شبكات الاستشعار المندمجة في الإنترنت. في هذا العمل، نقترح حلين لحماية الاتصالات مع عقد الاستشعار المتصلة بشبكة الإنترنت. يتمثل الحل الأول في إنشاء فعال للروابط الأمنية من طرف إلى طرف إعتمادا على البروتوكول HIP الذي يحمل إيجابيات كثيرة لتطبيقات إنترنت الأشياء . هذا الحل هو أول من جمع بين أول نموذج 6LoWPAN لضغط رسائل HIP مع توزيع آمن للحمولة الأمنية المدرجة في البروتوكول HIP . بالنسبة للحل الثاني، تم إقتراح نظام انتقائي و لا متماثل لسلامة التفاعلات بين الإنسان المستخدم مع الأشياء الذكية في إنترنت الأشياء. يأخذ الحل بعين الاعتبار القيود المفروضة على شبكات الاستشعار اللاسلكية و كذلك التباين بين أجهزة الاستشعار و غيرها خلال تامين هذا النوع من الاتصالات من نهاية إلى أخرى، مع تقليل ملموس من تأثير هجمات الحرمان من الخدمة (DoS) على شبكات الاستشعار المدمجة في إنترنت الأشياء.

كلمات البحث: شبكات الاستشعار اللاسلكية، الأمن، انترنت الأشياء، الإنترنت.

Résumé

L'Internet des objets (IoT) est un paradigme prometteur qui étale la connexion Internet de nos jours pour interconnecter différents types d'objets intelligents, autre que les ordinateurs et les téléphones mobiles, pour un mode de vie beaucoup plus sophistiqué et une qualité de service améliorée dans différents domaines d'application. Les réseaux de capteurs sans fil (RCSFs) comme une composante vitale de l'IoT, permettent la représentation des caractéristiques dynamiques du monde réel dans le monde virtuel de l'Internet. Ainsi, le standard IPv6 (*Internet Protocol version 6*) s'est étendu en une version compressée (6LoWPAN: *IPv6 over Low power Wireless Personal Area Networks*) pour l'IoT, permettant l'adressage d'une façon unique des milliers, voire des millions de nœuds capteurs connectés à Internet. Ces derniers sont considérés comme des hôtes IP réels et, leurs applications deviennent des services web. En effet, la maturité de l'Internet des objets dépend sans aucun doute de la sécurité des communications et la protection de la vie privée des utilisateurs. Toutefois, les hétérogénéités technologiques et matérielles, ainsi que la nature asymétrique des communications entre les nœuds capteurs et les hôtes classiques de l'Internet, font de la sécurité de l'IoT, un problème crucial. Dans ce contexte, de nombreuses solutions ont été proposées pour la standardisation de la sécurité des réseaux de capteurs connectés à Internet. Dans cette thèse, nous proposons deux solutions pour la protection des communications avec les nœuds capteurs intégrés à l'IoT. La première est une solution efficace d'établissement de sécurité de bout-en-bout basée sur le protocole HIP (*Host Identity Protocol*) qui semble être avantageux pour les applications de l'IoT. Cette solution est la première à combiner le premier modèle de compression 6LoWPAN des messages HIP avec un système sécurisé de distribution des primitives sécuritaires incluses dans le standard HIP. Quant à elle, la deuxième solution proposée présente un système asymétrique et sélectif pour la sécurité des interactions humain-à-objet dans l'IoT. La solution prend en considération les contraintes des RCSFs et les hétérogénéités entre les nœuds capteurs et les hôtes ordinaires, lors de la protection de bout-en-bout de tel genre de communications, avec une atténuation considérable de l'impact des attaques par déni de service (DoS) sur les RCSFs intégrés à l'IoT.

Mots clés : réseaux de capteurs sans fil, sécurité, IoT, Internet.

Abstract

The Internet of Things (IoT) is a promising paradigm that spreads the nowadays Internet connection to interconnect several types of smart objects, other than computers and mobile phones, for a sophisticated lifestyle and an improved quality of service in various application fields. Wireless sensor networks (WSNs) as a vital component of the IoT, allow the representation of dynamic characteristics of the real world in the Internet's virtual world. Thus, the standard IPv6 (Internet Protocol version 6) is extended in a compressed version (IPv6 over Low power Wireless Personal Area Networks) for the IoT, allowing unique addressing of thousands and billions of connected sensor nodes. These last are henceforth considered as real IP hosts in the Internet, and their applications become web services. Indeed, the maturity of the Internet of Things depends on the security of communications and the protection of end-users privacy. However, technological and material heterogeneities, and the asymmetric nature of communications between sensor nodes and ordinary Internet hosts, make the security in the IoT, a crucial problem. In this context, many solutions have been proposed for the standardization of the security of the Internet-integrated sensor networks. In this work, we propose two solutions for the protection of communications with the Internet-connected sensors nodes. The first one is an effective end-to-end security establishment solution based on HIP (Host Identity Protocol) that seems advantageous IoT deployments. This solution is the first that combines the first 6LoWPAN compression model for HIP messages, with a safe secure distribution of the primitives included in HIP standard security policy. Meanwhile, the second solution presents an asymmetric and selective system for the protection of human-to-thing interactions in the IoT. The solution takes into account the constraints of WSNs and the heterogeneities between the sensors and the powerful Internet hosts while protecting such kind of communications from end to end, and reducing substantially the effect of denial of service attacks (DoS) on IoT-integrated WSNs.

Key words: wireless sensor networks, security, IoT, Internet.

Publication internationale

- S. Sahraoui, A. Bilami, Efficient HIP-based approach to ensure lightweight end-to-end security in the internet of things, Science Direct, Elsevier Computer Networks, Vol 91, pp. 26–45, 2015.

Communications internationales

- S. Sahraoui, A. Bilami, Compressed and distributed host identity protocol for end-to-end security in the IoT, In: Proceedings of 2014 Fifth International Conference on Next Generation Networks and Services (NGNS), Casablanca, Morocco, 28-30 May 2014, pp. 295 – 301. http://ieeexplore.ieee.org/document/6990267/

- S. Sahraoui, A. Bilami, Asymmetric End-to-End Security for Human-to-Thing Communications in the Internet of Things, In: Proceedings of the 4th International Symposium on Modeling and Implementation of Complex Systems (MISC 2016), Constantine, Algeria, 7-8 May 2016, pp. 249-260.
  http://link.springer.com/chapter/10.1007/978-3-319-33410-3_18

Communications nationales

- S. Sahraoui, A. Bilami, Approche pour la Sécurité de Bout-en-Bout dans l'Internet des Objets, 8 ème édition du Séminaire National d'Informatique (SNIB), Biskra, 20-21 Janvier 2015.

Table des matières

Chapitre 2. La sécurité dans les RCSFs

Chapitre 3. Généralités sur l'Internet des objets

Chapitre 4. L'intégration des réseaux de capteurs sans fils à l'Internet des objets

Chapitre 5. La sécurité de l'intégration des RCSFs à l'IoT : état de l'art

Partie 2 : Contribution

Chapitre 6. Solutions proposées pour la sécurité de l'intégration des RCSFs à l'IoT.

Chapitre 7. Evaluation de performances des solutions proposées.

Liste des figures

Liste des tableaux

Glossaire des acronymes

- **RCSF** Réseau de Capteurs Sans Fil
- **WSN** Wireless Sensor Network
- **IoT** Internet of Things
- **IoE** Internet of Everything
- **Cooja** Contiki os java simulator
- **uIP** ubiquitous Internet Protocol
- **SMAC** Self-organizing Medium Access Control for Sensor networks
- **EAR** Eavesdrop And Register
- **LEACH** Low-Energy Adaptive Clustering Hierarchy
- **HEEP** Hybrid Energy Efficient Protocol
- **TCP** Transmission Control Protocol
- **UDP** User Datagram Protocol
- **IPv*x*** Internet Protocol version *x*
- **WMSN** Wireless Multimedia Sensor Networks
- **LRWPAN** Low Rate Wireless Personal Area Networks
- **FHSS** Frequency Hopping Spread Spectrum
- **PIV** Program Integrity Verification
- **CSMA/CA** Carrier Sense Multiple Access with Collision Avoidance
- **DoS** Denial of Service
- **DDoS** Distributed Denial of Service
- **DSA** Digital Signature Algorithm
- **ECC** Elliptic Curve Cryptography
- **WEP** Wired Equivalent Privacy
- **DES** Data Encryption Standard
- **AES** Advanced Encryption Standard
- **CBC** Cipher block chaining
- **SDI** Système de Détection d'Intrusion
- **IDS** Intrusion Detection System
- **TinySec** Tiny Security
- **MAC** Message Authentication Code
- **SPINS** Security Protocols for Sensor Networks
- **SNEP** Secure Network Encryption Protocol
- **µTESLA** Timed, Efficient, Streaming, Streaming, Loss-tolerant Authentication Protocol
- **RPK** Random Pair-wise Keys
- **RFID** Radio Frequency IDentification
- **SDN** Software-Defined Networking
- **WoT** Web of Things
- **GPS** Global Positioning System
- **H2T** Human to Thing
- **T2T** Thing to Thing
- **T2H** Thing to Human

- **M2M** Machine to Machine
- **ERCIT** European Research Cluster on the Internet of Things
- **SOA** Service Oriented Architecture
- **NITRD** Networking and Information Technology Research and Development
- **NASA** National Aeronautics and Space Administration
- **DARPA** Defense Advanced Research Projects Agency
- **GSN** Global Sensor Networks
- **IETF** Internet Engineering Task Force
- **6LoWPAN** IPv6 over Low power Wireless Personal Area Networks
- **RoLL** Routing over Low power and Lossy networks
- **CoRE** Constrained RESTful Environments
- **MTU** Maximum Transmission Unit
- **FFD** Full Function Device
- **RFD** Reduced Function Device
- **EUI-64** IEEE Unique Identifier
- **ISM** Industriel, Scientifique et Médical
- **TTL** Time To Live
- **RPL** Routing Protocol for Low power and lossy networks
- **6BR** 6LoWPAN Border Router
- **DODAG** Destination Oriented Directed Acyclic Graph
- **DIO** DODAG Information Object
- **DIS** DODAG Information Solicitation
- **DAO** DODAG Destination Advertisement Object
- **DAO-ACK** DAO Acknowledgement
- **CoAP** Constrained Application Protocol
- **REST** Representational State Transfer
- **HTTP** Hyper Text Transfer Protocol
- **URI** Uniform Resource Identifier
- **MQTT** Message Queuing Telemetry Transport
- **Thingbot** Botnet d'objets intelligents
- **ICMPv6** Internet Control Message Protocole version 6
- **PKC** Public Key Cryptography
- **ECDH** Elliptic Curve Diffie-Hellman
- **IPsec** IP sécurité
- **AH** Authentication Header
- **ESP** Encapsulating Security Payload
- **IKE** Internet Key Exchange
- **HIP** Host Identity Protocol
- **HIP-BEX** HIP Base EXchange
- **HIP-DEX** HIP Diet EXchange
- **TLS** Transport Layer Security
- **DTLS** Datagram Trasport Layer Security
- **AMIKEY** Adapted Multimedia Internet Keying
- **SA** Security Association
- **HI** Host Identifier

- **LHIP** Lightweight HIP
- **HIT** Host Identity Tag
- **CHIP** Compressed Host Identity Protocol
- **DHIP** Distributed Host Identity Protocol
- **CD-HIP** Compressed and Distributed Host Identity Protocol
- **ADSL** Asymmetric Digital Subscriber Line
- **Energest** *Energy estimation*
- **RTT** Round Trip Time

Introduction générale

Les réseaux de capteurs sans fil ont connu un succès remarquable durant les dernières décennies, en raison de leur capacité à collecter efficacement différents types d'informations concernant l'environnement du captage (un lieu ou un objet) et de les faire communiquer pour aider à la détection précoce des évènements, la supervision à distance du fonctionnement des systèmes et même la prévision des phénomènes. Cependant, les nœuds capteurs qui composent ce type particulier de réseaux, sont soumis à des contraintes sévères, spécialement celles qui sont relatives aux limitations de ressources de traitement, de stockage mémoire et surtout d'énergie. Telles doivent être absolument respectées par les solutions destinées aux réseaux de capteurs, tout en garantissant un bon fonctionnement du réseau.

On assiste récemment à l'émergence de l'Internet des objets, qui est un nouveau paradigme bouleversant le domaine des réseaux de télécommunication. L'Internet des objets qui est une partie intégrante dans l'Internet du futur, consiste en une large interconnexion de toutes sortes d'objets (autre que les ordinateurs et les téléphones mobiles) dans notre entourage, à titre d'exemple les véhicules, les routes, la maison, la télévision, etc. pour un monde ambiant et intelligent. La liaison entre tels objets et l'Internet est matérialisée par des capteurs connectés à Internet et des étiquettes RFID qu'ils incorporent et qui leur (les objets) permettent de stocker, traiter des informations pertinentes et de les communiquer sur Internet.

Les réseaux de capteurs sans fil sont une technologie très importante dans l'Internet des objets. Ils permettent de présenter les caractéristiques et l'état des objets (ou environnements) dans lesquels ils sont implantés (ou déployés) comme des services web sur Internet. Les nœuds capteurs sont invités donc à jouer le rôle d'hôtes de l'Internet (souvent des serveurs web) et communiquer entre eux (dans le cadre des communications automatiques dits objet-à-objet) et avec les hôtes déjà connectés à l'Internet actuel (communications humain-à-objet). Suivant le modèle d'intégration des réseaux de capteurs à l'Internet des objets, les interactions avec les nœuds capteurs connectés peuvent être directes (le cas où les réseaux de capteurs adoptent le standard TCP/IP) ou indirecte (le cas où le réseau de capteurs est connecté par interface ou proxy). Pour une intégration flexible avec un degré élevé d'ubiquité de l'information des nœuds capteurs intégrant l'Internet, le modèle d'intégration par adoption des standards basés IP est favorisé. Pour répondre au besoin d'adressage de l'immense ensemble de capteurs intégrant l'Internet, le protocole IPv6 est utilisé.

Néanmoins, l'ouverture des réseaux de capteurs à Internet présente un problème sérieux du point de vue sécuritaire car les réseaux de capteurs connectés à Internet deviennent accessibles à distance et d'une façon ubiquitaire par n'importe quel hôte malicieux sur Internet. Les contraintes des réseaux de capteurs, ainsi que l'hétérogénéité qui caractérise les communications entre les hôtes réguliers et les nœuds capteurs dans l'Internet des objets sont les vulnérabilités majeures de l'intégration qui est à l'origine de diverses attaques, essentiellement les attaques par déni de service (DoS).

En effet, les solutions de sécurité déjà approuvées et qui ont été proposées pour l'Internet classique, ne peuvent pas être directement projetées sur les nœuds capteurs contraints. Dans ce contexte, plusieurs solutions ont récemment intensivement investigué le problème de la sécurité des réseaux de capteurs connectés à Internet [82, 85, 86, 87, 94, 95, ..]. Ces solutions ne sont pas bien adaptées car elles ne prennent pas en considération la particularité des réseaux de capteurs dans l'IoT en tant que réseaux IP(v6) particuliers et négligent parfois même la contrainte énergétique sévère des nœuds capteurs.

Dans cette thèse, nous avons proposé deux mécanismes pour la sécurité de l'intégration des réseaux de capteurs à l'Internet des objets : le premier est une solution efficace et faiblement coûteuse proposée pour l'établissement de la sécurité de bout-en-bout entre les hôtes ordinaires et les nœuds capteurs. La solution est basée sur le standard HIP (*Host Identity Protocol*) qui est assez bénéfique pour l'IoT. Le deuxième mécanisme sécuritaire proposé concerne les communications humain-à-objet. La solution protège ce type de communications de bout-en-bout, tout en considérant les hétérogénéités matérielles et technologiques entre les dispositifs impliqués, avec une atténuation significative de l'effet des attaques par déni de services qui menacent sévèrement les réseaux de capteurs intégrés à l'IoT.

La thèse est organisée en deux volets : le premier consiste en une présentation du domaine de la recherche et le second présente notre contribution. La première partie comprend cinq chapitres. Les deux premiers chapitres traitent des généralités sur les réseaux de capteurs et leur sécurité, respectivement. Le troisième chapitre est consacré à la présentation de l'Internet des objets et tout ce qui s'y rapporte. Le chapitre quatre étudie les approches d'intégration des réseaux de capteurs à l'Internet des objets. Le cinquième chapitre présente un état de l'art sur les solutions de sécurités proposées dans la littérature pour répondre à la problématique de sécurité des réseaux de capteurs dans le contexte de l'IoT.

Quant à elle, la deuxième partie de la thèse est organisée en deux chapitres: le chapitre six présente en détails les solutions proposées et le chapitre sept, et le dernier, présente et analyse les résultats d'évaluation de nos propositions.

PARTIE 1 :

Introduction au domaine de recherche

CHAPITRE 1:

Généralités sur les réseaux de capteurs sans fil

1. INTRODUCTION

La popularité des réseaux de capteurs sans fil (RCSFs) [1] ne cesse de s'accroitre dans différents domaines de la vie courante, à savoir le domaine militaire, agricole, le domaine de santé, le secteur industriel, et plain d'autres. Cela est justifié par la contribution efficace de tels réseaux à la modernisation des techniques et méthodes de surveillances et de contrôle à distance des phénomènes dans des environnements de différentes natures.

Les RCSFs se composent généralement d'un grand nombre de nœuds capteurs minuscules, stationnaires ou mobiles, souvent déployés aléatoirement dans un champ de captage. Ce dernier est généralement un milieu hostile, isolé ou difficile à contrôler, où la mission d'un nœud capteur consiste à chaque fois, de récolter, d'une façon autonome, des informations précises depuis l'environnement de déploiement. Suivant le type du nœud capteur, la donnée captée peut être la température, l'humidité, la pression, la lumière ou autres. les nœuds capteurs dans un RCSF communiquent entre eux via des liens radio pour l'acheminement des données collectées à un nœud considéré comme "point de collecte", appelé station de base ou puits. Cette dernière peut être connectée à une machine puissante, appelée gestionnaire des tâches, via Internet ou par satellite. En outre, le réseau peut être configuré de telle sorte que l'utilisateur puisse adresser ses requêtes aux capteurs en précisant l'information requise, et en ciblant les nœuds capteurs qui devraient s'y intéresser.

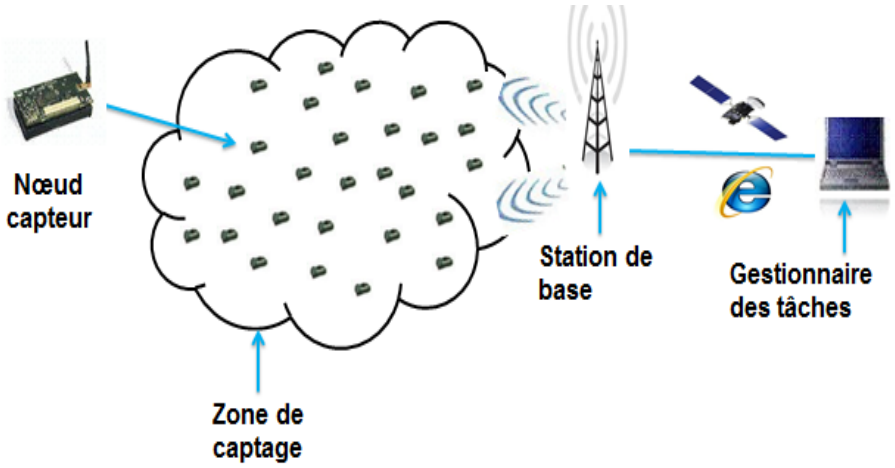


Figure 1.1. Architecture de communication d'un réseau de capteur sans fil.

En raison de leur faible coût valant juste quelques dollars, leur taille réduite et leur efficacité, les capteurs ont réussi à gagner l'appréciation de leurs utilisateurs, ce qui leur a permis d'être de plus en plus omniprésents. Ainsi, les RCSFs ont fait l'objet de nombreux projets de recherche scientifique qui ont adressé, essentiellement, l'optimisation du rendement et la sécurisation de ce type de réseaux.

Le présent chapitre est consacré à la présentation des réseaux de capteurs sans fil et leur contexte.

2. ANATOMIE DU NŒUD CAPTEUR

Un nœud capteur ordinaire comporte quatre unités de base représentées par une unité d'acquisition (dispositif de captage), une unité de traitement (un processeur), une unité de communication (un émetteur/récepteur radio) et une batterie. Le rôle de chacune des unités est défini dans les points suivants :

- **L'unité d'acquisition :** est généralement composée de deux sous-unités : les capteurs et Les convertisseurs analogique-numérique (ADCs : Analog-to-Digital Convertors). Les capteurs obtiennent des mesures numériques sur les paramètres environnementaux et les transforment en signaux analogiques. Les ADCs convertissent ces signaux analogiques en des signaux numériques.

- **L'unité de traitement :** comme le révèle son nom, cette unité est responsable de tous les traitements que doit effectuer un nœud capteur. Elle comprend deux interfaces : une interface avec l'unité d'acquisition et une autre avec le module de transmission. L'unité de traitement contrôle les procédures permettant au nœud capteur de réaliser les tâches d'acquisition et de stockage de données collectées, à travers un microcontrôleur (un simple processeur) et une mémoire limitée à quelques kilooctets.

- **L'unité de communication (*Transceiver : transmitter-receiver*) :** responsable de toutes les communications via un support de communication radio qui relie le nœud au réseau.

- **Batterie :** alimente les unités d'acquisition, de traitement et de communication.

De plus, un nœud capteur peut être équipé d'autres composants supplémentaires tels qu'un système de localisation géographique GPS (Global Position System).

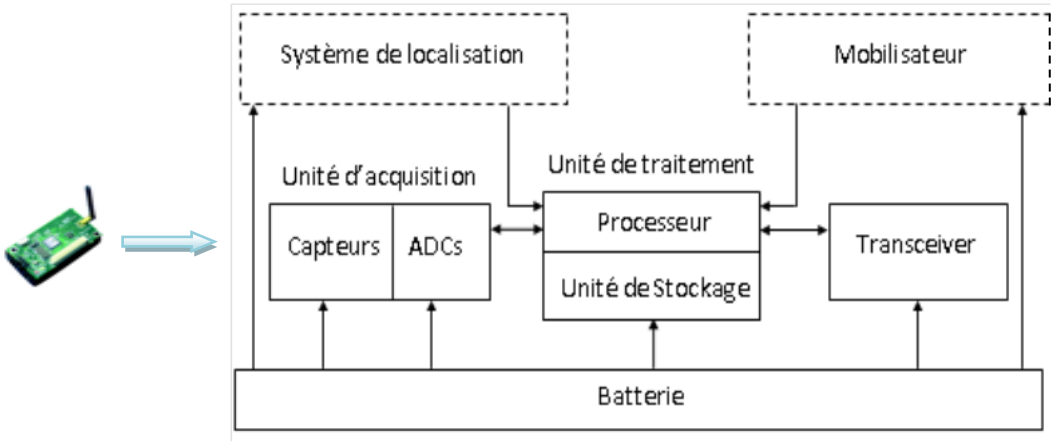
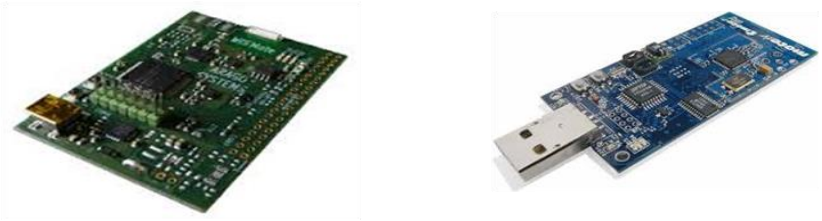


Figure 1.2. Composants du nœud capteur.

Il existe plusieurs modèles commercialisés des capteurs sans fil, parmi les plus célèbres : Tmote Sky et WiSMote (figure 1.3). Le tableau 1.1 présente les caractéristiques des deux plateformes.



Capteur WiSMote **Capteur Tmote Sky**

Figure 1.3. Les capteurs WiSMote et Tmote Sky.

Table 1.1. Caractéristiques des capteurs WiSMote et Tmote Sky.

Propriétés	WiSMote	Tmote Sky
Microcontrôleur	TI MSP430F5437x	MSP430 F1611
Fréquence d'horloge	16 MHz	3.9 MHz
RAM (Ko)	16	10
ROM (Ko)	256	48
Radio	CC2520	CC2420
Batterie	2AA	2AA

3. Caractéristiques d'un réseau de capteurs sans fil

Contrairement aux réseaux de communication sans fil ordinaires, les RCSFs ont des caractéristiques distinguées :

- **Déploiement dense :** le nombre de nœuds capteurs dans un RCSF peut être très important. il peut même atteindre des milliers, voire des millions. Le défi dans tel cas, est que le réseau soit capable de maintenir ses performances avec ce grand nombre de nœuds capteurs.

- **Déploiement aléatoire :** les nœuds capteurs sont souvent dispersés d'une manière aléatoire (à l'aide d'un avion à titre d'exemple), ce qui nécessite l'adoption des protocoles et algorithmes d''auto-organisation.

- **Environnement dur :** l'environnement dans lequel les nœuds capteurs sont dispersés peut être hostile (par exemple un champ de bataille) ou difficilement accessible (comme le fond de mer), tout dépend de l'application.

- **Durée de vie limitée :** Les nœuds capteurs sont très limités par la contrainte d'énergie, ils fonctionnent habituellement sans surveillance dans des zones éloignées. Par conséquent, le rechargement ou le remplacement de batteries s'est avéré quasi impossible.

- **Ressources limitées :** Le nœud capteur a une taille minuscule, ce facteur limite la quantité de ressources qui peuvent y être intégrées. Par conséquent, les capacités de traitement et de mémorisation sont très limitées.

- **Topologie dynamique :** La topologie des réseaux de capteurs change d'une manière fréquente. Cela peut être dû soit à la mobilité des entités du réseau, soit à la défaillance accidentelle ou causée par un épuisement de batterie des nœuds capteurs.

- **Redondance de données :** Dans le cas où les nœuds capteurs sont densément déployés dans le champ de captage, les données captées et communiquées par des multiples capteurs à proximité du même évènement détecté sont redondantes. Cela entraîne un gaspillage de ressources (énergie, bande passante et mémoire).

- **Agrégation des données :** C'est une approche bénéfique qui consiste à résumer les données au niveau des nœuds intermédiaires afin de palier le problème de redondance de données et de réduire la surcharge réseau et la consommation d'énergie induites.

- **Bande passante limitée :** Compte tenu de leur puissance limitée, les nœuds capteurs ne peuvent pas supporter des débits élevés.

- **Spécificité de l'application :** Un RCSF est mis en place spécialement pour répondre aux exigences d'une application bien déterminée. Cela a fait que la conception des RCSFs soit directement influencée par les spécificités des applications.

4. Les applications des RCSFs

Les réseaux de capteurs sans fils ont su envahir un grand nombre de domaines d'application, vu leur taille minuscule, leur coût symbolique et leur efficacité opérationnelle. Ainsi, le besoin en un suivi permanent à distance, qui est très demandé dans la majorité de ces applications [2], a encouragé encore plus l'adoption des RCSFs. Dans ce qui suit, on présente les applications éminentes des RCSFs.

- **Les applications militaires :** dans de tel domaine, l'utilisation des réseaux de capteurs sans fil s'avère très utile et appréciable, comme les réseaux de capteurs sont basés sur le déploiement dense et rapide et peu couteux d'un nombre très important de nœuds capteurs, la défaillance de certains de ces nœuds n'interrompra pas le bon fonctionnement du réseau. Ainsi, la surveillance des champs de bataille, ou des frontières, se fait de manière continue par les nœuds capteurs ; en outre ces derniers sont capables de détecter une variété d'évènements tels que la pression, la présence ou l'absence de certains types d'objets (agents chimiques, biologiques, ou radiations), sa position, sa vitesse, sa taille, ou encore sa direction.

- **Les applications médicales :** les RCSFs sont largement répandus et utilisés aujourd'hui dans le domaine médical; la plupart des capteurs médicaux (dits aussi capteurs corporels) ont toujours été trop coûteux et trop complexe pour être utilisés à l'extérieur des milieux cliniques. Toutefois, les récents progrès de la microélectronique et de l'informatique ont fait de nombreuses formes de capteurs médicaux accessibles aux personnes à leur domicile, lieux

de travail et autres espaces de vie. Ces capteurs peuvent être très utiles lors de la surveillance à distance d'un patient (une personne à mobilité réduite, âgée, handicapée), en déployant de manière réparti les capteurs dans l'environnement du patient. Ensuite, les informations collectées (concernant le mouvement du patient par exemple) vont être envoyées à la station de base, ce qui va permettre une intervention rapide si c'est nécessaire.

- **Les applications environnementales :** les réseaux de capteurs sans fil sont largement utilisés dans les applications environnementales en raison de leur capacité de prévenir les catastrophes naturelles (tempête, inondation, feu de forêt …), et de détecter les fuites de produits toxiques (gaz, élément radioactif …) dans des zones industrielles ; cela permet une intervention rapide et efficace des secours. Un autre exemple où l'emploi des réseaux de capteurs sans fil s'est avéré bénéfique, est l'agriculture de précision. Des informations précises sur l'état du champ (le taux et la localisation de sécheresse, la détection de l'invasion des insectes nuisibles,…) sont collectées par les capteurs et communiquées en temps réels, ce qui permet une intervention efficace ainsi qu'une optimisation des ressources d'eau.

- **Les applications industrielles :** la miniaturisation des capteurs sans fil permet de nombreuses perspectives applicatives comme celles liées au contrôle et au suivi industriel. En dispersant les capteurs dans différents emplacements (à l'intérieur ou à l'extérieur, ou encore au-dessus) d'une zone industrielle, ces derniers peuvent : détecter et contrôler des fuites de produits chimiques, diagnostiquer les dispositifs de production, contrôler la qualité des produits, contrôler des processus d'usinage, etc. ces informations peuvent être très utiles et mènent à ajuster et améliorer certains paramètres (température, ingrédient,…), pour un bon fonctionnement et une meilleur qualité de production.

- **La domotique :** Un autre type d'application dans lequel les réseaux de capteurs émergent, est la domotique. Dans cette application, le réseau de capteurs est déployé dans l'habitation. Le principe est que le réseau forme un environnement, dit pervasif où l'objectif étant de fournir toutes les informations nécessaires aux applications d'automatisation pour le confort, la sécurité et la maintenance dans la maison. Les capteurs sont incorporés dans différents dispositifs domestique (tel que : chauffage, système d'éclairage, dispositif d'incendie, alarme de détection d'intrusion, volet roulant, etc.), afin de répondre aux besoins de l'utilisateur, en lui permettant un contrôle plus aisé sur ces dispositifs localement ou à distance, par internet ou par satellite. Ce qui garantira aux habitants plus de confort, de sécurité, ainsi une facilité de maintenance, et d'un autre coté c'est l'un des moyens utilisés pour pouvoir minimiser les dépenses énergétique.

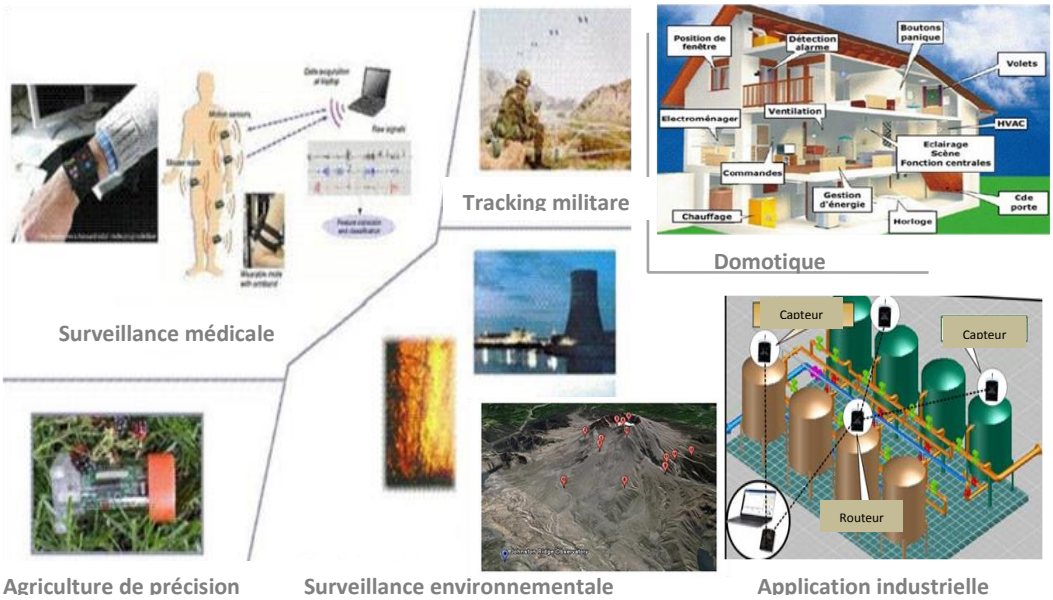


Figure 1.4. Quelques applications des réseaux de capteurs sans fil.

5. INFRASTRUCTURE LOGICIELLE DES CAPTEURS

Dans cette section, nous allons nous intéresser à la partie logicielle des capteurs. Nous étudions d'abord les systèmes d'exploitation dédiés aux capteurs ensuite, les protocoles utilisés pour les RCSFs.

5.1. Systèmes d'exploitation pour capteurs

L'adoption des systèmes d'exploitation traditionnels pour les nœuds capteurs dans un RCSF est désapprouvée, voire impossible, à cause de l'énorme marginalisation des caractéristiques distinguées des nœuds capteurs qu'elle entraîne, surtout en ce qui concerne le faible espace mémoire, la capacité de calcul impuissante et la contrainte sérieuse d'énergie.

De ce fait, il était inévitable de développer de nouvelles solutions qui soient mieux adaptées. TinyOS (*tiny operating system*) et Contiki sont parmi les systèmes d'exploitation les plus répandus.

5.1.1. TinyOS :

Le système d'exploitation TinyOS est l'un des premiers systèmes d'exploitation conçus pour être adapté aux réseaux da capteurs sans fil. TinyOS est un système open source développé par l'université américaine de Berkeley, écrit en langage NesC (langage syntaxiquement proche du C) qui est orienté composants.

TinyOS est dirigé par les évènements (*event-driven*), c'est-à-dire que les traitements ne s'effectuent que lors d'un stimulus découlant de leurs environnement, en outre ce dernier est capable d'inséré rapidement les changements et les nouveautés liés à l'application, et/ou à la topologie du réseau. Son rôle est fondé sur l'association de composants ce qui réduit de façon remarquable la taille

du code ainsi réduire l'usage de la mémoire. Il est réputé par sa bibliothèque particulièrement complète car elle comprend les protocoles réseaux, des pilotes de capteurs ainsi que des outils d'acquisition de données. Ces derniers peuvent être utilisés directement ou adapter à une application précise.

Les principales caractéristiques de TinyOS sont résumées dans les points suivants :

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

- **Orienté évènement** : Le fonctionnement d'un système se focalise sur TinyOS s'appuie sur la gestion des évènements qui ce déclenche. Ainsi, l'activation de tâches, leur interruption ou encore la mise en veille du capteur s'effectue à l'apparition d'évènements. De plus, TinyOS n'est pas prévu pour avoir un fonctionnement temps réel.

- **Non préemptif** : TinyOS ne gère pas le mécanisme de préemption entre les tâches, mais donne la priorité aux interruptions matérielles. Ainsi, les tâches entre elles ne s'interrompent pas mais une interruption peut stopper l'exécution d'une tâche.

- **Consommation minimale d'énergie**: TinyOS a été conçu pour réduire au maximum possible la consommation énergétique des nœuds capteurs. Ainsi, lorsqu' aucune tâche n'est active, il se met automatiquement en veille.

- **Modularité** : TinyOS est défini comme étant un ensemble de composants logiciels qui peuvent être reliés ensemble (par une édition de lien statique) en un seul exécutable.

5.1.2. Contiki

Contiki est un système d'exploitation écrit en langage C, portable et *open source* pour capteurs miniatures. Contiki est spécialement conçu pour respecter les contraintes des RCSFs, en particulier, celles qui sont liées aux limitations de l'espace mémoire (il en occupe environ 32 kilooctets de ROM et 4 kilooctets de RAM).

Contiki contient un noyau événementiel, au dessus duquel les programmes d'application peuvent être chargés dynamiquement et déchargés au moment de l'exécution. Les processus Contiki utilisent le *protothreading* [128] *;* un style de programmation qui présente un bon compromis entre la programmation évènementielle et la programmation par multithreading. En plus de *protothreading*, Contiki supporte également la préemption entre les threads.

Pour la communication, Contiki implémente deux mécanismes : Rime et uIP. Le premier mécanisme consiste en une couche située juste au-dessous des applications. Telle couche fournit un ensemble d'instructions de communication. Quant à lui, le deuxième mécanisme (uIP : micro IP) est

une implémentation adaptée d'une pile protocolaire basée IP (les protocoles : TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*), IP (*Internet Protocol*), ICMP (*Internet Control Message Protocol*)). L'adoption de tel mécanisme de communication rend possible la communication directe entre un capteur et n'importe quel hôte IP.

De plus, Contiki offre d'autres fonctionnalités comme un serveur telnet, un serveur web et trois environnements de simulation: Cooja, MSPsim, Netsim. Dans le tableau ci-dessous, on compare Contiki et TinyOS suivant plusieurs critères :

Table 1.2. TinyOS et Contiki.

	Modèle de programmation	**Ordonnancement**	**Support des apps temps-réel**	**Langage**	**Simulateur**	**Shell**
TinyOS	Orienté évènements	FIFO	Non	NesC	TOSSIM	Non
Contiki	Basé-évènements *& Protothreading*	Priorité donnée aux événements	Non	C	Cooja	Oui

5.2. La pile protocolaire des capteurs

Cette architecture est mise en place afin de structurer les protocoles de communication dans les RCSFs. Ce modèle comprend 8 couches, cinq d'entre eux ont les mêmes tâches que celles du modèle OSI (*Open System Interconnection*), et trois autres couches pour assurer la gestion d'énergie (*Power Management Plane*), la gestion de la mobilité (*Mobility Management Plane*), et la gestion des tâches (*Task Management Plane*).

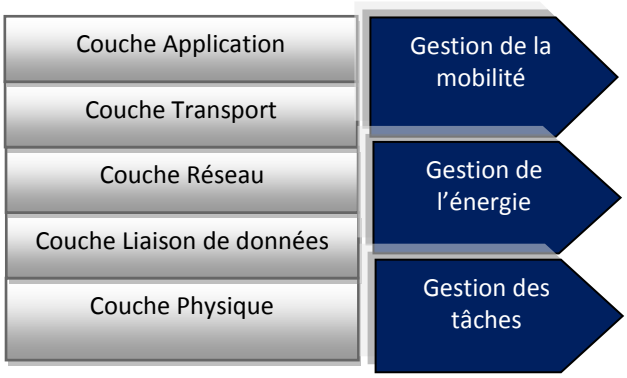


Figure 1.5. Modèle en couches pour la communication dans les RCSFs.

5.2.1. Les rôles des couches

Le rôle de chacune des cinq couches ainsi que les protocoles en vedette sont :

- **La couche physique :** Cette couche se charge de tout ce qui est spécifications des caractéristiques matérielles, la génération des ondes porteuses, la modulation de données et leur injection sur le support de transmission toute en sélectionnant les bonnes fréquences.

- **La couche liaison de données :** Spécifie comment les données sont expédiées entre deux nœuds dans une distance d'un saut. Elle est responsable de l'accès au media physique, du

multiplexage des données, du contrôle d'erreurs. Elle assure la liaison point à point et multipoint dans un réseau de communication. Parmi les protocoles qui opèrent au niveau de cette couche on cite : SMAC (*Self-organizing Medium Access Control for Sensor networks)* et EAR (*Eavesdrop And Register*).

- **La couche réseau :** La couche réseau a pour but principal de baliser une route optimale en vue d'acheminer efficacement les données captées depuis leur source jusqu'au puits, tout en minimisant la dissipation énergétique des nœuds capteurs inclus dans le chemin. La tâche de routage au sein d'un réseau de capteurs est spécifique du fait que :

    - L'écoulement de données récoltées à partir de multiples sources vers une seule destination (la station de base).
    - La forte redondance de données et l'exigence de l'agrégation.
    - La nécessité d'une gestion soigneuse des ressources (énergie, mémoire, bande passante).

  En effet, plusieurs protocoles ont été proposés pour répondre au besoin d'un routage efficace dans les RCSFs comme : LEACH (*Low-Energy Adaptive Clustering Hierarchy)* [3], HEEP (*Hybrid Energy Efficient Protocol*) [4], le protocole *O(1)-Reception* [131] et autres.

- **La couche transport :** Cette couche est chargée du transport de données, de la vérification de la qualité de la transmission et de la gestion des éventuelles erreurs. Dans le cas des réseaux de capteurs sans fil, la bonne qualité de transmission est souvent négligée car d'une part les pertes sont très probables avec un support de transmission sans fil et d'autre part, les mécanismes de gestion de la fiabilité sont trop lourd (tout comme le protocole TCP : *Transmission Control Protocol*). Ainsi, les pertes et les erreurs de transmission sont tolérables et peuvent même être camouflé par la redondance de données et l'agrégation.

  Le protocole UDP (*User Datagram Protocol*) qui fournit un service de transport en mode datagramme (sans connexion, sans gestion de congestion et sans fiabilité) est jugé d'être le protocole de transport le mieux adapté aux environnements capteurs en raison de sa faible empreinte mémoire et simplicité.

- **La couche application :** La couche application présente le niveau le plus proche des utilisateurs. de nombreux profiles d'applications peuvent être configurées et utilisées dans la couche application des réseaux de capteurs sans fil.

5.2.2. Les plans de gestion

Les plans de gestion d'énergie, de mobilité et des tâches permettent au nœud capteur de contrôler respectivement la dissipation d'énergie, le mouvement et la distribution de tâches. En effet, ces plans aident également les nœuds capteurs à coordonner la tâche de captage tout en

rationalisant la consommation énergétique. Ils sont donc nécessaires pour que les nœuds capteurs puissent collaborer ensemble pour acheminer les données dans un réseau mobile et partager les ressources entre eux avec une consommant efficace de l'énergie.

- **Plan de gestion de mobilité :** Offre des mécanismes de détection et enregistrement des mouvements du nœud capteur. Ainsi, le nœud capteur peut garder trace de ses voisins.

- **Plan de gestion d'énergie :** Permet le contrôle de l'utilisation de la batterie, par exemple : après la réception d'un message, le capteur éteint son récepteur afin d'éviter la duplication du message déjà reçu. En outre, si le niveau d'énergie résiduelle devient bas, le nœud capteur diffuse à ses voisins une alerte les informant qu'il ne peut pas participer au routage et il préserve l'énergie restante pour le captage.

- **Plan de gestion des tâches :** Responsable de l'ordonnancement des tâches de captage de données dans une région spécifique. Il n'est pas nécessaire que tous les nœuds capteurs de cette région effectuent la tâche de captage au même temps ; certains nœuds capteurs la font plus que d'autres suivant que l'énergie résiduelle leur soit suffisante ou non.

5.3. Le modèle cross-layer dans les réseaux de capteurs

La plupart des protocoles proposés pour les RCSFs et qui sont basés sur le modèle en couches (qu'on a traité dans la section précédente) ont réussis à apporter une certaine amélioration aux techniques de réduction de la consommation énergétique, par l'exploitation de la nature collaborative des nœuds dans les réseaux de capteurs. Bien que ces protocoles puissent atteindre un niveau de performance élevée en termes des métriques relatives à chacune des couches individuellement, ils ne sont pas optimisés conjointement pour maximiser les performances globales du réseau.

A cet effet, plusieurs travaux de recherche récents se sont concentrés sur la conception et le développement des solutions *cross-layer*. Le principe de base de telles solutions tourne autour de deux concepts :

- le premier consiste à préserver le modèle en couches traditionnel avec la considération des interactions inter couches ; chaque couche est informée des conditions des autres couches tout en gardant ses fonctionnalités de base.

- Dans le deuxième axe, les chercheurs ont constaté que les performances seront beaucoup plus optimisées si l'ancien modèle en couches est totalement modifié. L'idée principale est de fournir un seul module de communication dans lequel, les tâches communes des différentes couches sont incorporées.

Cette nouvelle tendance prometteuse qui n'est pas encore arrivée à sa maturité a pu montrer une efficacité significative tant en termes de conservation d'énergie qu'en termes d'amélioration des performances du réseau [5].

6. LES TYPES DES RÉSEAUX DE CAPTEURS SANS FIL

Selon des critères bien spécifiques, comme la mobilité, l'homogénéité des nœuds du réseau, la nature de l'application et le type des données captées, les RCSFs peuvent être classés en plusieurs classes [6].

6.1. Selon le critère de mobilité

Les nœuds capteurs, ainsi que la station de base dans un réseau de capteurs sans fil peuvent être stationnaires ou bien mobiles. On parle alors des réseaux de capteurs statiques ou mobiles respectivement.

6.1.1. Les réseaux de capteurs statiques

Dans les réseaux de capteurs statiques, et les nœuds capteurs et la station de base sont stationnaires ; ils gardent leurs positions initiales tout au long de leur durée de vie. Ce type de réseaux de capteurs est bénéfique dans certains types d'applications qui exigent que les capteurs soient placés dans des endroits stratégiques pour les contrôler. En effet, tel type de RCSFs est caractérisé par une topologie statique, une localisation facile des nœuds dans le réseau et des techniques de routages assez simples.

6.1.2. Les réseaux de capteurs mobiles

Contrairement aux RCSFs statiques, dans les réseaux de capteurs mobiles, les capteurs et/ou la station de base ont la capacité de se mobiliser. La mobilité du capteur se produit soit quand le capteur est collé sur un objet mobile, soit quand le capteur s'auto-déplace (cas d'un capteur muni d'un moteur).

La mobilité est indispensable dans les réseaux de capteurs destinés aux applications de suivi, par exemple, quand les capteurs sont embarqués sur des véhicules, ou sur des animaux. Elle est (la mobilité) également avantageuse du point de vue coût d'investissement ; au lieu de déployer plusieurs nœuds statiques, un nombre minime de dispositifs mobiles est suffisant. Cependant, lorsque la mobilité est trop fréquente, elle ne peut être considérée comme un problème secondaire. Ainsi, le changement fréquent de la topologie complique les mécanismes de routage et de localisation.

6.2. Selon le critère d'homogénéité

Suivant ce critère, on observe deux types des réseaux de capteurs sans fil : les réseaux de capteurs homogènes et les réseaux de capteurs hétérogènes.

6.2.1. Les réseaux de capteurs homogènes

Un réseau de capteurs est dit homogène si tous les nœuds capteurs sont équivalents sur le plan capacités et contraintes (faibles ressources et durée de vie courte). C'est le type qu'on trouve souvent dans la majorité des applications des réseaux de capteurs, car ils répondent au besoin d'autonomie.

6.2.2. Les réseaux de capteurs hétérogènes

A l'encontre des réseaux de capteurs homogènes, les réseaux de capteurs hétérogènes comportent deux types de nœuds capteurs : les nœuds capteurs contraints (*battery-powered*) et les nœuds capteurs puissants non limités en ressources (particulièrement les ressources énergétiques comme ils sont directement liés à un secteur d'alimentation électrique).

Dans ce type de RCSFs, les nœuds contraints doivent préserver autant que possible leur réserve énergétique en minimisant les tâches les plus coûteuses en énergie tout comme la communication radio. Ainsi, les calculs et les traitements compliqués sont délégués aux nœuds puissants pour équilibrer la charge et maximiser la durée de vie du réseau.

Bien que les RCSFs hétérogènes soient plus avantageux que les RCSFs ordinaires (homogènes), leur adoption est limitée à un nombre réduit d'applications. Cela est dû à la difficulté du déploiement des RCSFs hétérogènes dans des milieux hostiles, isolés ou inaccessibles.

6.3. Selon le type de l'application

Le déclenchement du processus de captage de données dans un réseau de capteurs sans fil dépend des exigences applicatives et de l'importance de la donnée captée en elle-même. Donc, on distingue deux types de RCSFS : temporels (*time-driven*) ou évènementiels (*event-driven*).

6.3.1. Les réseaux de capteurs temporels

Un réseau de capteurs temporel est approprié pour des applications qui nécessitent un prélèvement périodique des données. Tel est le cas par exemple dans les applications de monitoring (feu ou météo). Un écoulement en rafale, périodique, du trafic est très susceptible dans ce type d'applications. Par conséquent, des mécanismes de gestion raisonnable des ressources sont primordiaux.

6.3.2. Les réseaux de capteurs évènementiels

Dans certaines applications, les capteurs doivent réagir rapidement à des changements brusques des valeurs captées et donner des réponses immédiates à l'occurrence des évènements. Un prélèvement périodique des données est inadapté pour ce type de scénario.

6.4. Selon les données captées

Les données que récoltent les nœuds dans un réseau de capteurs peuvent être de type simple, comme ils peuvent être de type multimédia. De plus, un nœud capteur peut capter un seul type de données (exemple : que la température) ou plusieurs types à la fois (exemple : image, température et humidité).

6.4.1. Les réseaux de capteurs standards

Il s'agit des RCSFs ordinaires où les données récoltées sont de types scalaires, comme par exemple : la température, l'humidité, la pression, etc. les RCSFs de tel type partagent les caractéristiques déjà mentionnées.

6.4.2. Les réseaux de capteurs multimédia

Certaines applications des réseaux de capteurs, exigent que les données à capter soient de type multimédia (son, image ou vidéo) comme c'est le cas par exemple dans les applications médicales et les applications militaires. Néanmoins, les données multimédia sont reconnues pour être volumineuses et occupent donc, un espace mémoire important. Ainsi, des techniques de représentation différente que celles des données ordinaires sont nécessaires pour les données multimédia.

Les réseaux de capteurs multimédia (ou *Wireless Multimedia Sensor Networks:* WMSN) requièrent des protocoles performants ainsi que des considérations particulières pour répondre à leurs défis en matière de qualité de service et de capacités de traitement exigées. D'autres spécificités liées aux WMSNs sont données ci-dessous :

- **le déploiement :** les nœuds dans les réseaux de capteurs standards sont souvent déployés aléatoirement. En revanche, dans les réseaux de capteurs multimédia, le déploiement des nœuds est généralement précis et étudié d'avance, particulièrement quand il s'agit du captage des images.

- **La puissance de traitement :** les traitements à effectuer sur les données scalaires sont faibles. Néanmoins, pour le cas des données multimédia, les nœuds capteurs effectuent des traitements intensifs ce qui demande plus de performance matérielle.

- **Qualité de service** : les réseaux de capteurs multimédia revendiquent suffisamment de bande passante ainsi qu'une faible latence pour qu'ils soient opérationnels, ce qui n'est pas le cas dans les réseaux de capteurs standards où la qualité de service est relâchée pour un besoin en un moindre coût et une faible dissipation des ressources.

- **Consommation d'énergie** : puisque la qualité de service et les traitements intensifs sont pratiquement gourmands en énergie, les mécanismes de gestion de la consommation énergétique dans les réseaux de capteurs multimédia doivent être très efficaces. A cet effet, on note que dans ce cas, le remplacement des batteries des nœuds capteurs est souvent possible (tout dépend de la nature du champ de captage).

6.4.3. Les réseaux de capteurs multimodaux

Un nœud capteur dans un RCSF multimodal peut récolter plusieurs informations de types différents où les types peuvent être scalaires ou multimédia. Par exemple, un nœud capteur peut capturer et la température et l'image. Ainsi, un seul nœud capteur multimodal peut remplacer tout un groupe de capteurs ordinaires. Ceci est particulièrement avantageux dans le cas où l'on veut avoir plus d'une information environnementale sur le même endroit d'intérêt.

7. LES TOPOLOGIES DES RÉSEAUX DE CAPTEURS

La connectivité entre les nœuds capteurs et la (les) station(s) de base peut être principalement organisée en deux topologies [2].

7.1. La topologie plate

Les nœuds capteurs sont tous dans le même niveau de privilège. Ainsi, chaque nœud dans le réseau a la possibilité de communiquer avec n'importe quel autre nœud dans sa portée radio. La communication suivant cette architecture est dite communication en multi-sauts, c'est-à-dire que si un nœud veut envoyer un message vers la station de base et que celle-ci est en dehors de sa portée radio, il envoie son message à un nœud intermédiaire (dans sa portée radio) pour passer le message et la même procédure se répète récursivement, jusqu'à ce que le message arrive à la station de base. Donc, dans la topologie maillée, les nœuds capteurs ne se chargent pas que de la tâche de captage de données, mais ils se préoccupent aussi du routage.

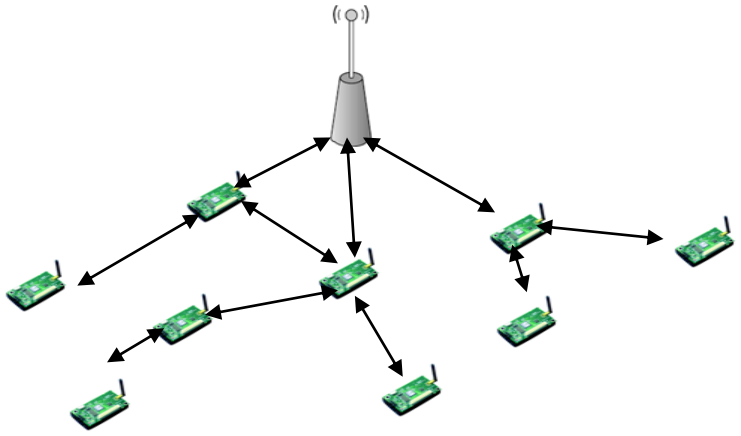


Figure 1.6. Modèle de la topologie plate.

Cette topologie a l'avantage d'être adaptée à l'évolutivité du réseau avec un certain niveau de redondance. La tolérance aux pannes est en fait un autre avantage de la topologie plate; si un nœud intermédiaire tombe en panne, la communication entre deux nœuds éloignés dans le réseau demeure possible s'il existe un chemin alternatif.

Avec un nombre considérable de nœuds capteurs, la consommation d'énergie sera de plus en plus importante car la majorité des nœuds capteurs sont sélectionnés dans, éventuellement, multiples routes menant vers la station de base ce risque d'épuiser rapidement leurs réserves énergétiques. Par conséquent, des protocoles de routage optimisés doivent être employés pour que la charge du réseau soit équilibrée et ne soit plus concentrée sur une zone particulière.

7.2. La topologie hiérarchique

Le principe est de partitionner le réseau en plusieurs groupes (ou *clusters*) dont chacun est vu comme un sous réseau ayant la topologie en étoile. Chaque groupe possède un chef qui relie les membres de son groupe à la station de base. La communication entre les nœuds capteurs et le chef du cluster peut être directe ou indirecte (en multi-sauts) pour les nœuds distants. Ainsi, il peut y avoir plusieurs niveaux dans la hiérarchie, où les chefs des clusters forment entre eux des chaînes menant vers la station de base. Cette topologie est particulièrement avantageuse car elle est flexible et permet une durée de vie du réseau, du fait que les nœuds capteurs sont dans la majorité du temps endormis et ils ne deviennent actifs que s'ils veulent communiquer des informations au chef du cluster, ou pendant la mise à jour de la topologie. Un autre avantage marquant de la topologie hiérarchique réside dans le fait qu'elle réponde mieux au besoin d'extensibilité et d'évolutivité du réseau. La figure suivante présente un simple modèle de cette topologie.

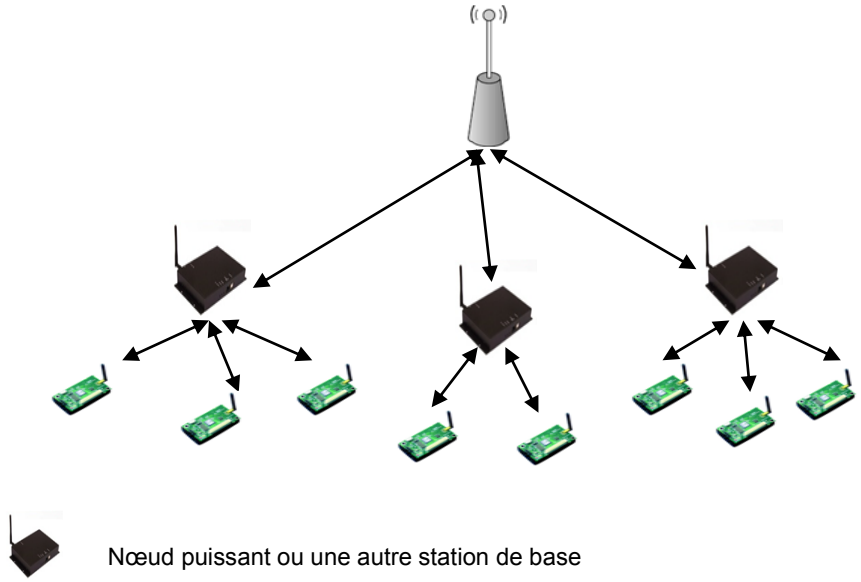


Figure 1.7. Modélisation de la topologie multi-étoile.

8. Les technologies de transmission dans les RCSFs

La communication sans fil dans les réseaux de capteurs est extrêmement importante et critique. Les RCSFs peuvent en supporter plusieurs types dont l'efficacité des communications et la conformité aux particularités du réseau et/ou l'application figée sont des critères clés pour le choix de telle ou telle technologie de transmission sans fil. Dans cette section, on cite les exemples de technologies les plus utilisées par les RCSFs.

8.1. Wi-Fi (IEEE 802.11a/b/g/n)

La technologie Wi-Fi (IEEE 802.11) permet la connexion d'un réseau local sans fil. Elle est disponible en plusieurs types : A, B, G et N. et récemment les normes AC et AD. La différence entre ces types tourne essentiellement autour du débit maximal qu'un dispositif connecté puisse atteindre et la portée (la distance maximale possible entre un dispositif connecté et le point d'accès). Wi-Fi utilise la bande de fréquence ISM (Industrial, Scientific and Medical) 2.4 GHz (à licence gratuite) ou la bande 5GHz. Cette technologie est caractérisée par un débit théorique nettement élevé allant de 11Mb/s (pour IEEE 802.11b) jusqu'à 54Mb/s (pour IEEE 802.11a,g). L'avantage du Wi-Fi est qu'il est couramment utilisé et donc, les nœuds capteurs peuvent être facilement connectés aux réseaux WLAN (Wireless Local Networks) existants.

A côté de ces avantages, cette technologie est inappropriée pour les réseaux de capteurs standards, en raison de la forte consommation d'énergie induite, ainsi que la complexité de sa pile protocolaire. Cependant, certains types de réseaux de capteurs, comme les réseaux de capteurs multimédia, exigent un débit effectif relativement important. Pour satisfaire cette exigence, la technologie Wi-Fi est utilisée.

8.2. Bluetooth (IEEE 802.15.1)

La technologie Bluetooth a été initiée en 1994 et actuellement gérée par le groupe SIG (*Special Interest Group*), elle a été standardisée sous la norme IEEE 802.15.1. Bluetooth est conçu pour fonctionner sur des appareils à faible puissance et faible consommation d'énergie, il a comme but la mise en œuvre des réseaux à portée personnelle où le transfert de données se fait par un débit moyen.

8.3. ZigBee (IEEE 802.15.4)

ZigBee est une association de plusieurs groupes de recherche qui visent le développement d'un standard global, complet et ouvert pour les communications sans fils avec un coût réduit et une basse consommation d'énergie. Il est fondé sur le standard IEEE 802.15.4 qui définit les couches basses (la sous-couche MAC et la couche physique) pour les réseaux WPAN à très faible débits LRWPAN (*Low Rate Wireless Personal Area Networks*). ZigBee offre des caractéristiques qui répondent mieux aux besoins des réseaux de capteurs sans fil parmi :

- Installation automatique/semi-automatique.
- Possibilité de rajouter/retirer des dispositifs avec souplesse.
- Coût avantageux.
- Débit : 10 kbps-115.2 kbps.
- Portée radio: 10-75 m.
- Jusqu'à 65536 nœuds par réseau.
- Jusqu'à 100 réseaux co-localisés.
- Jusqu'à 2 ans d'autonomie énergétique (durée de vie de batterie).

En raison de ses caractéristiques attractives, ZigBee est devenue la technologie la plus adoptée dans la plupart des applications des réseaux de capteurs où le besoin en débit est lâché pour une efficacité énergétique maximale, comme dans le cas des applications industrielles, et agricoles.

Il existe également d'autres technologies de transmission qui peuvent être supportées par les RCSFs et qui sont tout autant adaptées, comme la technologie UWB (IEEE 802.15.3) et Wibree.

Le tableau suivant récapitule les principales différences entre les trois technologies citées sur le plan des réseaux de capteurs sans fil.

Table 1.3. Comparaison entre les trois technologies.

Critère	Technologie		
	ZigBee	**Bluetooth**	**Wifi**
IEEE	802.15.4	802.15.1	802.11a /b/g
Besoin en mémoire	4-32 Kb	250 Kb +	1 Mb +
Autonomie batterie	Années	Jours	Heures
Nombre de nœuds	65 000 +	7	32
Débit	250 Kb/s	1 Mb/s	11-54-108 Mb/s
Portée	10 m	10-100 m	300 m

9. Les contraintes des réseaux de capteurs sans fil

Les principales contraintes imposées sur les réseaux de capteurs sont résumées dans les points suivants [1] :

- **La consommation d'énergie :** Le nœud capteur est limité en énergie (< 1.2V). Dans la plupart des cas, le remplacement de la batterie est quasi impossible ce qui fait que la durée de vie du réseau dépende grandement de la durée de vie des batteries des nœuds capteurs. D'autre part, la nature de réseau peut parfois entraîner une dissipation supplémentaire de l'énergie. Ceci pourrait être le cas par exemple lors de dysfonctionnement de quelques nœuds capteurs, ce qui nécessite un changement de la topologie du réseau et un re-routage des messages. Toutes ces opérations sont bien évidemment gourmandes en énergie. C'est pour cette raison que les recherches dans le domaine des RCSFs se concentrent principalement sur l'économie d'énergie.

- **La tolérance aux fautes :** Certains nœuds capteurs peuvent générer des erreurs ou ne plus fonctionner à cause d'un manque d'énergie, un problème physique. Ces problèmes ne doivent pas affecter le reste du réseau, suivant le principe de tolérance aux fautes, qui est la capacité de maintenir les fonctionnalités du réseau sans interruptions causées par les incidents matériels ou logiciels.

- **L'évolutivité du réseau :** Le nombre de nœuds capteurs déployés peut atteindre le million. Un nombre aussi important de nœuds capteurs engendre des flux intensifs dirigés vers la station de base. Cette dernière doit absolument être équipée de suffisamment d'espace mémoire pour stocker les informations reçues. Des techniques efficaces d'agrégation de données doivent être mises en place pour atténuer les réplications inutiles des messages.

- **L'environnement :** Les capteurs sont souvent déployés en masse dans des endroits difficiles d'accès, tels que des champs de batailles, à l'intérieur de grandes machines, au fond d'un océan, dans des champs biologiquement ou chimiquement souillés, etc. Par conséquent, il devient primordial d'assurer le bon fonctionnement du réseau sans le surveiller.

- **La topologie du réseau :** Le déploiement d'un grand nombre de nœuds capteurs nécessite une maintenance de la topologie. Cette maintenance consiste en trois phases: le déploiement, le post-déploiement (les capteurs peuvent bouger, ne plus fonctionner, etc.), le redéploiement de nœuds capteurs additionnels.

- **Les contraintes matérielles :** La principale contrainte matérielle est la taille du capteur qui doit être assez réduite, ainsi que la résistance du capteur aux susceptibles cassures et accidents.

- **La sécurité :** La sécurité physique des nœuds capteurs ainsi que la sécurité des communications inter-capteurs sont des contraintes très intéressantes. On étudie la sécurité des RCSFs, en détails, dans le chapitre suivant.

10. Conclusion

Les réseaux de capteurs sans fil représentent bel et bien une technologie prometteuse qui à travers ses caractéristiques particulières a pu attirer les chercheurs et les développeurs dans des domaines multiples. Cependant, certaines contraintes qui découlent d'un nombre de facteurs relatifs à la phase de conception des réseaux de capteurs doivent être prises en considération par les chercheurs, essayant ainsi d'éliminer ces derniers (ou alors les atténuer) et de maintenir le bon fonctionnement du réseau de capteurs.

Dans ce chapitre, nous avons décrit les principaux concepts liés aux réseaux de capteurs sans fil tels que : l'architecture, les applications en vedette, les principales caractéristiques et limitations. Dans

le prochain chapitre, nous allons nous intéresser au problème de sécurité dans le contexte des réseaux de capteurs sans fil.

CHAPITRE 2:

La sécurité dans les réseaux de capteurs sans fil

1. INTRODUCTION

La sécurité dans les réseaux de communication est par définition tout mécanisme permettant de couvrir les vulnérabilités du réseau et de protéger ses entités, ses utilisateurs ainsi que les informations échangées. En effet, différents types d'attaques existent déjà et plein de nouvelles catégories de menaces ne cesse d'émerger. De ce fait, la sécurité des réseaux informatiques constitue une discipline indépendante et un domaine très intéressant de la recherche scientifique attirant de nombreux chercheurs à travers le monde.

Assurer le service de sécurité pour des réseaux soumis à de fortes contraintes, tout comme les réseaux de capteurs, est une tâche pénible. Le défi majeur dans ce cas, consiste à réaliser le compromis entre un bon niveau de protection, un moindre coût et une meilleure considération des particularités du réseau. Puisque la mission d'un réseau de capteurs est souvent critique, toute altération ou subversion qui touche aux données captées et/ou les nœuds capteurs, peut causer des dégâts immenses. Les diverses vulnérabilités ainsi que la non surveillance des réseaux de capteurs déployés dans des régions éloignées et hostiles rendent les réseaux de capteurs exposés à de nombreuses menaces, ce qui nécessite l'élaboration des mécanismes sécuritaires robustes et bien adaptés.

Dans ce chapitre, nous présentons les aspects fondamentaux de la sécurité des réseaux de capteurs sans fil. Nous présentons également les solutions de base, qui sont proposées dans la littérature pour répondre aux besoins en termes de sécurité.

2. LES OBJECTIFS DE LA SÉCURITÉ DANS LES RÉSEAUX DE CAPTEURS

Les solutions de sécurité destinées aux réseaux de capteurs doivent remplir un ou plusieurs services de sécurité, parmi [8]:

2.1. La confidentialité

Seules les entités autorisées peuvent accéder les données échangées entre les entités communicantes. Les données doivent donc être chiffrées à l'aide des algorithmes de cryptage suffisamment robustes. D'autre part, la confidentialité des programmes des nœuds capteurs doit être garantie ; le nœud capteur ne doit en aucun cas se permettre la lecture de son contenu par des parties non autorisées. Donc, mêmes les données propriétaires au capteur, comme les clés cryptographiques, le programme du capteur et son identificateur, doivent être protégées.

2.2. L'intégrité

Le mécanisme de sécurité doit garantir que les données ne seront pas altérées le long de leur passage vers la station de base. Les deux entités doivent implémenter des techniques de détection de toute modification de données.

2.3. L'authentification

Il arrive qu'un attaquant ne cause pas que la modification des paquets qu'il intercepte mais aussi, il peut forger et injecter des paquets falsifiés dans le réseau. Dans tel cas, le nœud capteur doit pouvoir vérifier la validité des identificateurs des nœuds sources de données qui lui parviennent.

2.4. La fraîcheur de données

Dans la majorité des applications des réseaux de capteurs, la récence de données est vivement suggérée. Un attaquant peut violer cette propriété, en rejouant plusieurs fois des anciens messages. De ce fait, et les nœuds capteurs et la station de base doivent mettre en place des mécanismes appropriés pour s'assurer de la fraîcheur des données communiquées.

2.5. La disponibilité

Les RCSFs sont des réseaux orientés-services, ce qui veut dire que le réseau est spécialement mis en place pour rendre un service bien déterminé et souvent assez critique. Donc, même dans le cas où le réseau de capteurs est ciblé par des attaques, il doit résister tant que possible et préserver la disponibilité de ses ressources et services.

2.6. La sécurité de la localisation

Le réseau de capteurs a souvent besoin des informations précises de localisation concernant des objets contrôlés par les capteurs et/ou les capteurs eux-mêmes. Telles informations doivent nécessairement être protégées contre toute interception illégale ou manipulation mal intentionnée.

3. Analyse des vulnérabilités des réseaux de capteurs

Etant un réseau sans fil, un réseau de capteurs sans fil hérite les vulnérabilités communes dans les réseaux sans fil. De plus, les RCSFs ont également leurs propres vulnérabilités qui se découlent de leurs caractéristiques distinguées (et parfois de leur milieu de déploiement) qui les rendent prédisposés à différents types d'attaques. Les principales vulnérabilités des RCSFs sont résumées dans les points suivants [8]:

3.1. La communication sans fil et multi-sauts

Le support de transmission sans fil ouvre une porte d'insécurité pour les données qui peuvent être facilement interceptées et analysées par un attaquant qui est à proximité du réseau. La faible portée de la communication radio des nœuds capteurs et la nécessité d'une communication multi-sauts pour la diffusion des données introduit de nombreuses failles de sécurité comme la modification, la suppression ou la falsification des messages.

3.2. L'absence d'infrastructure

Les RCSFs ont une nature décentralisée; pas de centre de confiance entre les nœuds capteurs qui pourrait gérer, de manière sûre, la communication dans le réseau ainsi que, la sécurité (exemple : la génération et la distribution des clés cryptographiques sur l'ensemble des nœuds capteurs).

3.3. La limitation des ressources

L'énergie est sans doute le facteur limitatif le plus fort aux capacités d'un nœud capteur, la réserve énergétique limitée de chaque nœud capteur doit être gérée raisonnablement pour prolonger autant que possible sa durée de vie et ainsi que celle du réseau. Un attaquant peut exploiter cette contrainte pour effectuer des attaques qui épuisent les batteries des nœuds, surchargent les ressources de calcul (processeurs) et provoque le débordement de la mémoire.

3.4. L'autonomie et l'insécurité physique des nœuds capteurs

La plupart des applications des réseaux de capteurs exigent un déploiement étroit des nœuds à l'intérieur ou à proximité des phénomènes à surveiller. Cela peut conduire à de fréquentes captures et compromissions des nœuds capteurs. Comme le succès des applications des réseaux de capteurs sans fil dépend également de leur faible coût, les nœuds capteurs ne peuvent pas tous intégrer des solutions de protection physique inviolable. Par conséquent, un adversaire peut extraire des informations contenues dans le nœud capteur et peut même en empoisonner le programme pour que le capteur se comporte d'une façon malicieuse.

4. Les attaques visant les réseaux de capteurs

Comme tout autre type de réseaux sans fils, les réseaux de capteurs sont prédisposés à de diverses menaces qui apparaissent sous différents types et dans plusieurs niveaux de la pile protocolaire des capteurs.

4.1. les modèles d'attaques dans les réseaux de capteurs

Selon des critères bien spécifiques, comme l'ampleur de l'attaque, la puissance de l'attaquant, l'appartenance ou non de ce dernier au réseau, on distingue différentes classes d'attaques, à savoir [7]:

- **Les attaques accidentelles vs les attaques intentionnelles :** Les attaques accidentelles sont représentées par défaillances que subisse un nœud capteur depuis son entourage (par exemple, les cassures qui peuvent être causées par les animaux dans une application agricole). Cependant, les attaques intentionnelles et qui sont les plus fréquentes et les plus nuisibles aux RCSFs, sont gérées par des personnes malveillantes ayant un objectif malicieux.

- **Les attaques externes et les attaques internes :** Les attaques externes proviennent des nœuds qui n'appartiennent pas au réseau de capteurs (les nœuds intrus), et les attaques internes sont exercées par les nœuds de compromission qui font partie du réseau.

- **Les attaques impuissantes et les attaques puissantes :** Dans les attaques impuissantes (*mote-class*), l'attaquant utilise un certain nombre de nœuds ayant des capacités similaires à celles des nœuds capteurs du réseau pour l'attaquer, alors que dans les attaques puissantes (*laptop-class*) qui sont les plus dangereuses, l'attaquant fait appel à des dispositifs à fortes capacités (exemple : les ordinateurs portables) qui peuvent subvertir le réseau en entier.

- **Les attaques passives et les attaques actives :** Dans le cas où l'attaquant ne fait qu'écouter et analyser illicitement le trafic qui transite entre les nœuds capteurs, l'attaque est dite passive. Dans le cas échéant où l'attaquant se permet même de modifier, détourner, bloquer ou forger des données dans le réseau, l'attaque est dite active.

4.2. Les niveaux d'attaques dans les réseaux de capteurs

Les attaques qui ciblent les réseaux de capteurs peuvent opérer dans plusieurs niveaux de la pile protocolaire du capteur. A chaque fois les attaquants exploitent les failles de sécurité des protocoles ou des spécificités d'un niveau donné (physique, liaison de données, routage ou transport de données) [7-9].

4.2.1. Le niveau physique

La couche physique est très sensible aux attaques qui exploitent l'accessibilité du support de transmission pour intercepter les communications ou pour causer des problèmes plus grave comme, le brouillage qu'un l'attaquant puisse provoquer en envoyant des signaux parasites qui interfèrent avec les fréquences radio qu'utilisent les nœuds capteurs pour la communication. Si l'attaquant est assez puissant, ou encore s'il utilise plusieurs nœuds à faibles puissances, la perturbation de

communication peut s'étaler sur tout le réseau. La technique FHSS (*Frequency Hopping Spread Spectrum*) l'en est une contre mesure typique. FHSS est une technique d'étalement de spectre par saut de fréquences. Avant tout échange d'informations, l'émetteur et le récepteur se mettent d'accord sur une séquence pseudo-aléatoire correspondant aux sous canaux dans une bande fréquence sur lesquels le signal sera étalé pour une communication résistante aux interférences et un bon niveau de sécurité (un attaquant qui ne connait pas la combinaison aléatoire des canaux utilisés trouvera une grande difficulté pour intercepter toute la communication).

Une deuxième catégorie d'attaques possibles dans la couche physique des RCSFs, est la falsification des nœuds capteurs (*node tampering*). L'attaquant dans ce cas capture un nœud et extrait son contenu (ses programmes) à partir de la mémoire de ce même nœud capteur. Donc les clés cryptographiques et les autres informations sensibles seront dévoilées. Le nœud capturé pourrait même être corrompu (l'attaquant modifie le programme du nœud capteur en y insérant des codes malicieux) ou bien, remplacé par un nœud de compromission (qui est généralement plus riche en ressources) que l'attaquant puisse superviser. La détection de la falsification des nœuds (*tamper proofing*) est parmi les solutions qui peuvent faire face aux manipulations mal intentionnées des capteurs. Cependant, ces solutions sont jugées être relativement onéreuses comme elles nécessitent la conception des capteurs matériellement immunitaires contre les compromissions [10]. D'autres solutions purement logicielles (moins coûteuses) ont été proposées pour augmenter la résistance à la capture des nœuds. Dans [11] est définie une solution appelée PIV (*Program Integrity Verification*) qui exige qu'une entité centrale se charge de la vérification des programmes de tous les nœuds capteurs et de détecter les éventuelles altérations malicieuses. Toutefois, mêmes les solutions logicielles sont très compliquées pour être mises en place. Donc, il doit y avoir des mécanismes robustes de résistance aux attaques de compromission des nœuds, en attendant que des solutions préventives efficaces voient le jour. Ainsi, les protocoles de communication orientés RCSFs devraient être avertis de tel risque (la compromission des nœuds).

4.2.2. Le niveau liaison de données

Les attaques qui se concentrent dans ce niveau provoquent des collisions avec les communications inter-capteurs ou entre capteurs et station de base. Les collisions intensives causent des ruptures de communications (qui sont souvent urgentes dans un RCSFs) dans le réseau qui en résulte une consommation excessive d'énergie résultante des retransmissions répétées des trames corrompues.

Comme solutions possibles à ce problème : l'adoption des techniques préventives adaptées, comme les techniques d'évitement de collisions (CSMA/CA : *Carrier Sense Multiple Access with Collision Avoidance*) et la méthode d'accès au support à base de *beacon* (MAC IEEE 80.15.4) où une station dite coordinateur du réseau se fait consacrer pour la gestion des priorités entre les nœuds capteurs afin de synchroniser les communications entre elle et ces derniers (les capteurs). La fixation d'une borne supérieure pour le nombre maximal de retransmissions pour limiter l'impact de l'attaque sur la réserve énergétique des capteurs peut être également envisagée comme solution à ce niveau.

4.2.3. Le niveau routage de données

Le routage de données depuis leurs sources jusqu'à la station de base est une fonctionnalité qui est à la fois vitale et critique dans les RCSFs. Ce mécanisme est exposé à un large éventail d'attaques qui peuvent affecter la phase de construction des routes et même la phase d'acheminement des données. Dans cette section on cite les scénarios d'attaques les plus célèbres et les plus dangereux visant le mécanisme de routage dans les RCSFs [7].

A. L'attaque d'altération des tables de routage

Certains protocoles de routage exigent que les nœuds capteurs maintiennent localement des tables de routage contenant des informations sur les routes optimales menant vers la station de base. En vue de perturber ce processus, certains nœuds attaquants modifient, bloquent, retardent les messages de contrôle de routage ou génèrent de faux messages. Les tables de routage seront par conséquent empoisonnées, ce qui peut conduire à plusieurs problèmes, comme les boucles de routage et les déroutements des données, l'augmentation du délai de bout en bout, etc.

B. L'attaque *Sinkhole* :

Dans ce type d'attaque [12-13], l'intrus doit apparaitre très attractif aux autres nœuds et pour ce faire, il forge et diffuse des messages falsifiés annonçant qu'il est la meilleure prochaine destination des flux de données des autres nœuds. Selon les métriques adoptées par le protocole de routage suivant lesquelles se porte la décision du routage, les messages falsifiés peuvent annoncer un niveau d'énergie très élevé, un délai minime ou un nombre de sauts optimal vers la station de base.

Les nœuds recevant les messages falsifiés vont être facilement corrompus et vont tous orienter leurs paquets vers l'intrus. Dans le pire des cas, où le nœud attaquant est si puissant, il pourra couvrir le réseau en entier et par conséquent, il sera un aval de tous les flux de données dans le réseau. Le nœud attaquant peut se satisfaire d'analyser les données lui arrivant puis les faire passer à la station de base, ou bien il les supprime tous sans faire aucune distinction. Dans ce dernier cas, l'attaque se nomme Black hole. Le processus du routage et les performances du réseau risque d'être gravement affectés si le réseau cours une attaque Black hole. C'est pour cette raison que telle attaque est considérée comme une attaque très dangereuse qui a fait l'objet de plusieurs travaux de recherches récents.

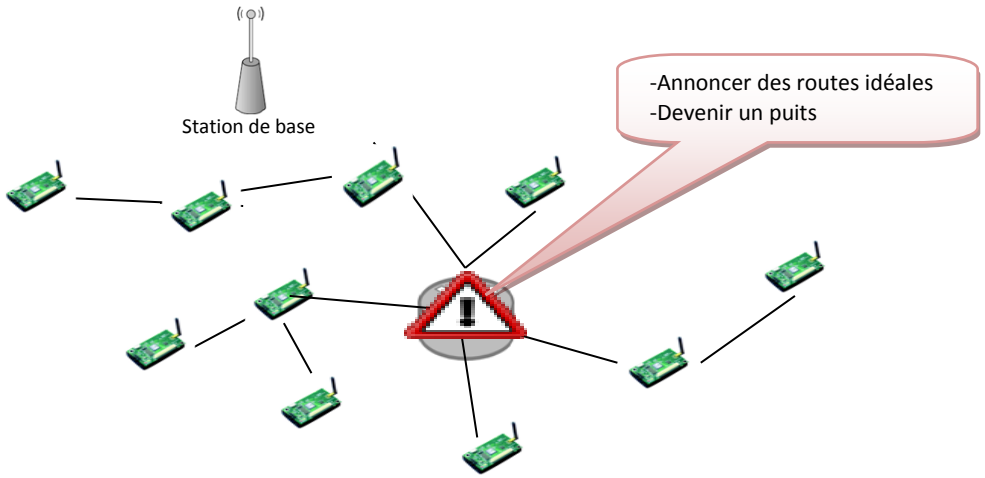


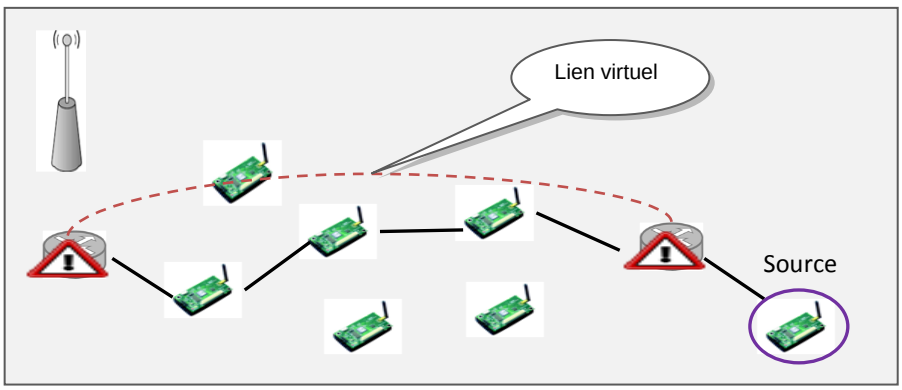
Figure 2.1. L'attaque sinkhole.

C. L'attaque trou de ver (*Wormhole)*:

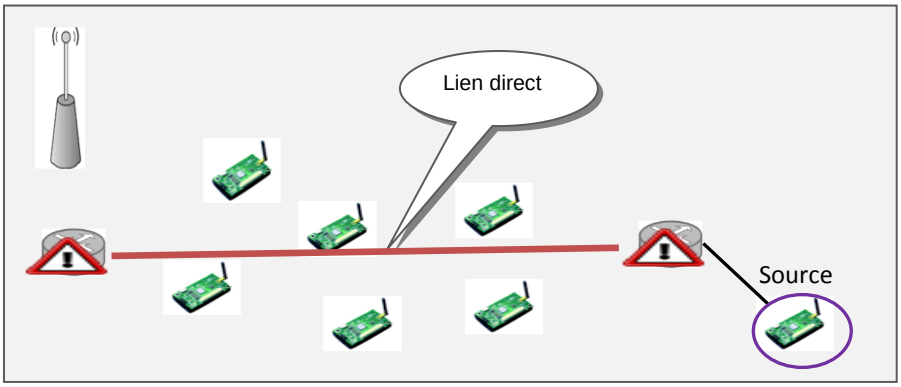
Le principe de cette attaque est que les nœuds malicieux dispersés dans le réseau collaborent ensemble via des tunnels virtuels pour entreprendre une attaque organisée. Ainsi, un attaquant collecte les données découlant des nœuds de son voisinage pour les réintroduire dans une autre zone du réseau, où il se trouve un autre attaquant. La communication entre ces deux attaquants peut se faire de deux manières :

- **Une communication multi-sauts :** dans ce cas les nœuds intermédiaires sont cachés par l'attaquant qui dans ce cas annonce une route à nombre de sauts minimale qui, en réalité, ne l'on est pas.
- **Une communication directe** : les deux attaquants sont liés directement par un lien à faible latence, ce qui conduit à ce que les protocoles qui se basent sur le délai comme métrique du routage soient affectés par cette attaque.

L'attaque *Wormhole* est très difficile à détecter [14] à cause de sa distributivité. Ainsi, une implantation précise des nœuds attaquant augmente l'ampleur de cette attaque, par exemple un attaquant implanté près de la station de base va complètement perturber le mécanisme du routage dans le réseau.



(a)



(b)

Figure 2.2. L'attaque *wormhole* : (a) communication multi-sauts. (b) communication directe.

D. L'attaque d'acheminement sélectif (*Selective* F*orwarding*) :

Dans cette attaque, l'adversaire essaye de tout faire pour s'instaurer dans un ou plusieurs chemins de routage que le protocole vient de créer. Pour cela, le nœud de compromission a souvent recours à l'attaque *sinkhole* qui lui facilite la tâche. Une fois que l'objectif de l'attaquant soit atteint, ce dernier commence à analyser le trafic, et il supprime aléatoirement les paquets.

E. L'attaque *Sybil* :

Dans cette attaque l'adversaire forge et diffuse plusieurs identités ou des positions géographiques différentes pour maximiser sa chance d'être un point d'intersection entre plusieurs chemins du routage. Notons que les identités annoncées peuvent correspondre à des nœuds réels qui existent dans le réseau, comme elles peuvent être des fausses identités. Cette attaque affecte sensiblement les protocoles de routage multi-chemins. Multiples routes construites peuvent représenter en réalité une seule route. Si le nombre d'attaquants est considérable, le trafic du réseau risquerait d'être détourné en totalité vers ces derniers.

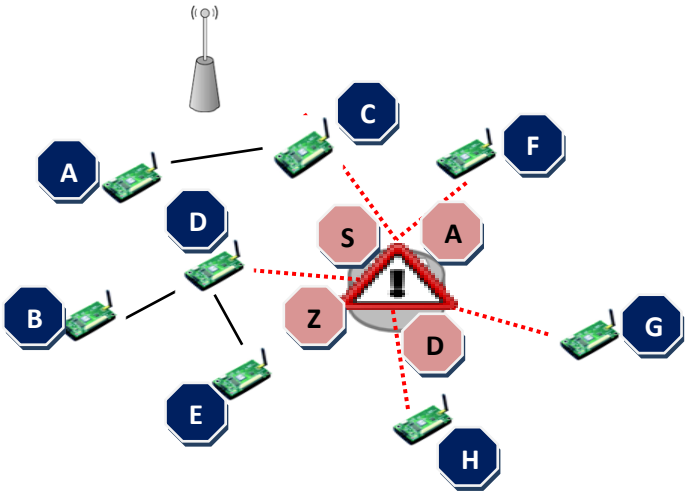


Figure 2.3. L'attaque Sybil.

F. L'attaque *Hello flooding* :

Les nœuds capteurs utilisent le message '*Hello*' pour découvrir les nœuds de leur voisinage immédiat ou pour annoncer leur état actuel. Un nœud adversaire utilisant un *laptop* peut exploiter le

fait que les nœuds capteurs ont des faibles portées radio, pour envoyer via un signal très puissant des messages annonçant une route optimale, à tous les nœuds du réseau. Ces derniers, vont par conséquent, mettre à jour leurs tables de routage avec des informations erronées. Puisque la liaison entre le nœud capteur et l'attaquant de est le plus souvent unidirectionnelle, les nœuds capteurs victimes ne pourront pas utiliser les routes annoncées par l'attaquant car il est en dehors de leurs portées de communication.

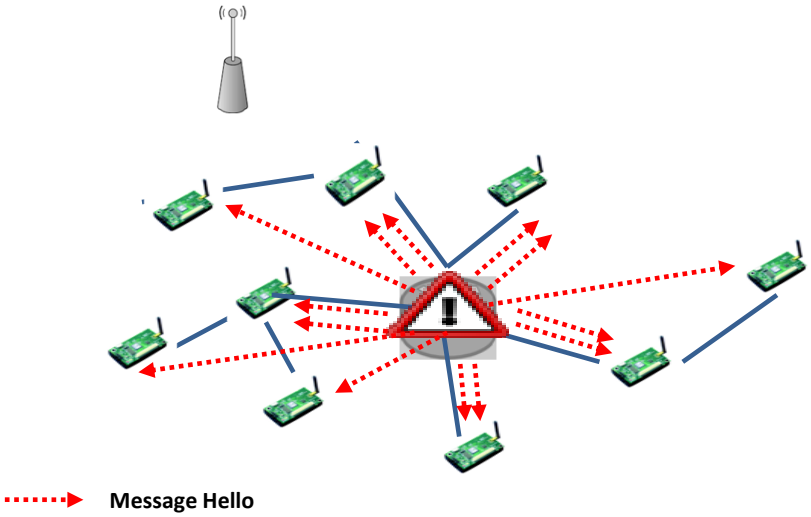


Figure 2.4. L'attaque Hello Flooding.

G. L'attaque par rejeu de données :

Cette fois-ci, le principe est très simple. Il suffit juste qu'un attaquant rejoue un ancien message plusieurs fois dans le réseau. Cependant, l'impact pourrait être assez négatif, comme cet acte peut entraîner des fausses alertes ou alors, empêcher le signalement d'une urgence. La situation risque d'être encore plus grave si l'attaque est effectuée par un attaquant interne (un nœud de compromission) car elle sera difficile à détecter [133].

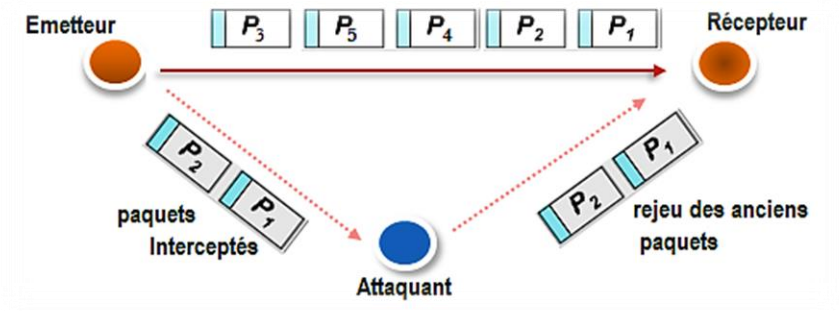


Figure 2.5. L'attaque par rejeu.

H. L'attaque par déni de service (DoS: *Dinial of Service*) :

L'attaque par déni de service peut apparaitre sous diverses formes, dont l'objectif est toujours de geler les services fournis par le nœud capteur et/ou le réseau à travers le surmenage des ressources des capteurs. L'épuisement des batteries des nœuds capteurs par le biais de l'envoi massif des messages, ou bien en confiant une masse importante de traitements, le débordement des tables de

routage causé par l'envoi intensif des fausses informations de routage, sont parmi les scénarios possibles pour déclencher une attaque DoS.

Le tableau ci-dessous résume les attaques citées au niveau routage, ainsi que les principaux remèdes.

Table 2.1. les attaques et les contremesures au niveau routage de données.

Attaques	Contremesures
Altération des tables de routage	Diffusion authentifiée des informations de routage
Sinkhole	L'authentification, la redondance.
Wormhole	Authentification, routage géographique, utiliser les techniques Leach.
Selective forwarding	Routage multi-chemins.
Sybil	Authentification, chiffrement symétrique (où chaque nœud partage une clé secrète avec le puits).
Hello flooding	Authentification, vérifier la bidirectionnalité de la communication.
Rejeu	Authentification, s'assurer du bon séquençage des messages
DoS	Mis en sommeil périodique pour les nœuds, routage tolérant.

4.2.4. Le niveau transport de données :

L'envoi fréquent des requêtes d'ouverture de connexion TCP (si ce dernier est supporté) sur des services bien définis dans les nœuds capteurs ciblés par l'attaquant, accélère l'épuisement des ressources requises (mémoire et énergie), ce qui conduit à un déni de service. Une autre attaque consiste à violer les connexions existantes par la désynchronisation des nœuds communicants.

Les solutions les plus évidentes consistent en la fermeture des services qui s'avèrent momentanément non nécessaires et la limitation du nombre maximal de connexions autorisées à la fois. L'adoption des mécanismes robustes pour l'authentification est également nécessaire.

5. Les exigences sécuritaires dans les réseaux de capteurs

Les solutions de sécurité proposées pour les réseaux de capteurs doivent satisfaire les exigences suivantes :

- **La sûreté:** toute technique de sécurité doit assurer au moins l'un des services cités dans la section 2.

- **La gestion soigneuse des ressources :** l'utilisation raisonnable des ressources, en particulier l'énergie, est primordiale dans tout mécanisme sécuritaire destiné aux RCSFs.
- **La résistance :** la technique de sécurité définie doit résister tant que possible aux attaques. plus la résistance est bonne, plus la solution est bonne.
- **La flexibilité :** la souplesse de configuration et de réalisation de la solution sont vivement recommandées.
- **Le support pour l'évolutivité du réseau :** même avec l'ajout des nouveaux nœuds capteurs, la solution doit maintenir ses performances et son bon fonctionnement.

6. Les fondements de la sécurité dans les réseaux de capteurs

Les paramètres qui sont à la base de la plus part des solutions de sécurité dans les réseaux de capteurs sont [9]:

6.1. La cryptographie

La cryptographie est une discipline fondamentale de la sécurité qui vise à protéger des messages en garantissant la confidentialité, l'authenticité et l'intégrité. Pour ce faire, il est nécessite d'employer un algorithme de chiffrement et de déchiffrement qui à leur tour, utilisent des secrets ou clés pour le cryptage/décryptage des messages.

En effet, il est connu que l'on a deux grandes classes de cryptographie. On parle alors de la cryptographie asymétrique et la cryptographie symétrique.

6.1.1. La cryptographie asymétrique

La cryptographie asymétrique, ou cryptographie à clé publique, est une méthode de chiffrement qui se base sur l'utilisation de deux types de clés pour chaque entité : une clé publique et une clé privée (secrète) pour le chiffrement et le déchiffrement. Chaque entité possède sa propre clé privée et l'ensemble des clés publiques des autres entités. Pour le chiffrement, l'expéditeur du message peut utiliser la clé publique du destinataire. Dans tel cas, seul le destinataire peut déchiffrer le contenu du message en utilisant sa clé privée, ce qui garantit la confidentialité de la communication. Dans le cas échéant, l'expéditeur utilise sa clé privée pour le chiffrement. Le message sera par la suite décrypté par le destinataire en utilisant la clé publique de ce dernier. Dans ce cas, l'expéditeur chiffre le message tout en affirmant son authenticité (l'expéditeur est la seule entité qui pourrait chiffrer le message). RSA est parmi les algorithmes de chiffrement les plus connus dans le contexte de la cryptographie asymétrique. Les algorithmes de signatures numériques DSA (*Digital Signature Algorithm*) s'appuient également sur la cryptographie asymétrique pour garantir l'intégrité et l'authentification des messages échangés.

Bien que la cryptographie asymétrique soit très robuste, elle est jugée pour être inadaptée aux RCSFs en raison de :

- La taille volumineuse des clés (allant de 1024 bits)
- Occupation de mémoire
- Calculs lourds et coûteux en énergie

La cryptographie par courbes elliptiques (ECC : *Elliptic Curve Cryptography*) [15] est une bonne alternative aux algorithmes RSA. Il s'agit de la cryptographie asymétrique économique qui assure un bon niveau de sécurité, comparable à celui de RSA, avec l'utilisation des clés de tailles réduites (il est par exemple possible d'utiliser des clés de 224 bits).

6.1.2. La cryptographie symétrique

Contrairement à la cryptographie asymétrique, la cryptographie symétrique, également dite à clé secrète permet le chiffrement et de déchiffrement des messages à l'aide d'une même clé secrète partagée uniquement entre deux entités communicantes. Ce qui est plus avantageux, en particulier du point de vue empreinte mémoire, c'est pour cela qu'elle est la plus adoptée dans les RCSFs.

Toutefois, l'échange sûr et sécurisé des clés secrètes présente un véritable problème dans la cryptographie symétrique. A partir de là, un pré-chargement de clés dans les nœuds capteurs avant le déploiement du réseau est la méthode prépondérante qui est la plus sûre dans la plupart des applications des RCSFs.

En effet, on distingue deux types d'algorithmes de chiffrement/déchiffrement symétrique : les algorithmes de chiffrement à flot (ou par flux) et les algorithmes de chiffrement par bloc. Le principe de chiffrement symétrique par flot est de traiter le message bit par bit en appliquant une opération quelconque (le plus souvent XOR). RC4 en est un exemple des algorithmes les plus répandus qui est d'ailleurs utilisé dans le protocole WEP (*Wired Equivalent Privacy*) du Wi-Fi [16].

Quant à lui, le chiffrement symétrique par bloc procède en découpant le message en plusieurs parties (blocs). Chaque bloc subit un traitement répétitif pour le chiffrement avant de passer au bloc suivant. Le nombre de tournées de chiffrement appliquées sur chaque bloc diffère d'un algorithme à un autre et plus le nombre d'itérations est important, plus la sécurité est bonne. Cette technique de chiffrement symétrique est la plus utilisée. Les algorithmes DES (*Data Encryption Standard*) et AES (*Advanced Encryption Standard*) sont des exemples bien connus.

6.2. La gestion de clés

La gestion de clés dans les réseaux de capteurs [17-18] englobe tout mécanisme qui s'attache à la méthode suivant laquelle les clés cryptographiques sont générées, distribuées et rafraichies. Les

nœuds capteurs ont besoin d'établir des liens de sécurité entre eux et/ou avec la station de base pour protéger ensuite les communications dans le RCSFs.

6.2.1. La gestion centralisée des clés

Dans les schémas de gestion de clés centralisés, les tâches de génération et de distribution des clés se réalisent par une station unique qui représente entre autres, un centre de confiance pour toutes les autres entités dans le réseau. Concernant les RCSFs, c'est plutôt la station de base qui devrait jouer tel rôle. Les nœuds capteurs lui confient cette mission qui est en fait assez coûteuses et qui exige que la station ait suffisamment de ressources, particulièrement un large espace mémoire.

L'inconvénient de tels schémas réside dans le fait qu'ils perdent une grande part de leur efficacité quand le réseau est étendu, là où la charge concentrée sur la station gérante devient sensiblement importante, ce qui risquerait d'avoir des blocages de service.

6.2.2. La gestion distribuée des clés

Dans ce cas, tous les nœuds sont au même niveau de privilège. Les nœuds communicants partagent la responsabilité de générer les clés nécessaires pour une communication sécurisée. Il existe plusieurs variantes des schémas distribués, dont on trouve par exemple les schémas de gestion de clés par groupes (appelés aussi schémas clusterisés) où les nœuds capteurs appartenant à un groupe, construit soit par le mécanisme de gestion de clé lui-même, soit par un autre mécanisme préalable (peut être un protocole de routage spécifique), coopèrent entre eux pour générer et partager une clé du groupe. Celle-ci (la clé), va être employée par la suite pour sécuriser les communications internes au groupe ou même externes avec la station de base.

Etant flexible et bien adaptés à l'évolutivité du réseau, les mécanismes de gestion de clés distribués sont beaucoup plus compliqués que les schémas centralisés, et par conséquent plus coûteux en énergie.

6.2.3. La gestion de clés avec des schémas aléatoires et probabilistes

L'idée de base des protocoles de gestion de clés probabilistes est qu'un grand ensemble *S* de clés soit généré, et pour chaque nœud capteur, *m* clés (qui constituent le trousseau de clés du nœud) seront aléatoirement choisis et chargés dans la mémoire. Le nombre de clés que doit contenir l'ensemble initial ainsi que les sous-ensembles m est choisi de telle façon que deux sous-ensembles aléatoires de *S* de taille m auront une certaine probabilité *p* d'avoir au moins une clé en commun.

Une fois déployés, les nœuds capteurs commencent à découvrir leurs voisins et plus particulièrement ceux avec lesquels ils partagent des clés communes pour établir des communications sécurisées.

6.3. La détection d'intrusion

Les solutions cryptographiques et les systèmes de gestion de clés sont très efficaces pour faire face aux attaques externes. Néanmoins, en cas d'existence des attaquants internes (nœuds de compromission) ces solutions deviennent insuffisantes. Cela est dû au fait que les adversaires internes disposent tout comme les nœuds légitimes, des clés cryptographies ainsi que de tout matériel de sécurité possible. Pour cette raison, il est nécessaire de renforcer la sécurité des RCSFs par des systèmes de détection d'intrusion (SDI) qui offrent une deuxième ligne de défense, en plus des solutions cryptographiques.

Un SDI [19] est un système chargé de la détection de l'existence des intrus dans le réseau, à travers un certain nombre d'agents (des nœuds capteurs ordinaires) de surveillance et d'évaluation des comportements et des réputations des nœuds capteurs. Ces agents collectent et diffusent des informations de contrôle visant la détection et l'isolation des nœuds malicieux.

6.3.1. Les exigences imposées sur la conception des SDIs dans les RCSFs

Comme toute autre solution destinée aux RCSFs, les systèmes de détection d'intrusion doivent faire preuve d'adaptabilité avec la nature exceptionnelle de tels réseaux. Dans les points suivants sont résumées les exigences et les contraintes principales qui doivent être remplies lors du développement d'un SDI pour les réseaux de capteurs :

- **Une moindre consommation d'énergie :** le SDI doit consommer un minimum d'énergie pour la détection et l'isolation des intrus.
- **Une moindre surcharge** : le volume des messages de contrôle à être échangés entre nœuds détecteurs ne doit pas être si important.
- **L'efficacité** : le SDI doit remplir ses objectifs avec robustesse, même si le réseau comportera un nombre important d'intrus. En d'autres mots, le taux de succès des détections devrait nécessairement être élevé et les intrus détectés doivent effectivement être détachées du réseau.
- **Le support de l'évolutivité du réseau**: le SDI doit préserver son efficacité si le réseau est élargi.

6.3.2. Classification des SDIs dans les RCSFs

Les systèmes de détection d'intrusion peuvent être classés dans plusieurs classes où les critères de classification sont basés sur l'architecture de communication adoptée par le SDI et sur la technique de détection utilisée. La figure ci-dessous présente les classes des systèmes de détection d'intrusion.

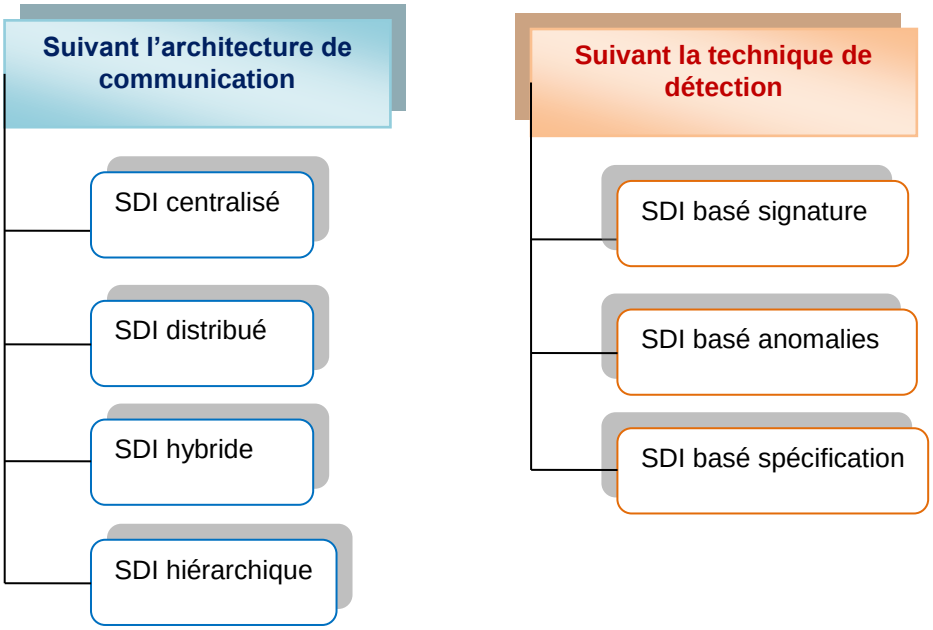


Figure 2.6. Les classes des systèmes de détection d'intrusion dans les réseaux de capteurs sans fil.

A. Les SDIs centralisés : dans cette classe de systèmes de détection d'intrusion, tous les agents détecteurs envoient leurs rapports à la station de base qui est dans ce cas la seule entité responsable de la prise de décisions finales sur la détection et l'isolation des intrusions. Etant donné que la station de base soit la plus puissante en termes de mémoire, d'énergie et de puissance de calcul, cela permettra l'adoption des techniques de détection beaucoup plus compliquées et robustes. Toutefois, la communication périodique des informations de détection vers un centre unique pourrait causer un excès de consommation d'énergie au niveau des nœuds capteurs, avec le risque de congestion des liens menant vers la station de base. Un autre problème de cette architecture réside dans la non-tolérance aux susceptibles pannes logicielles et/ou matérielles. La station de base prend également le risque d'être attaquée.

B. Les SDIs distribués : à l'encontre des SDIs centralisés, dans les systèmes de détection d'intrusion distribués c'est plutôt les nœuds capteurs qui se chargent de la détection d'intrusions. Cependant, pour plus d'efficacité des SDIs distribués, les nœuds capteurs détecteurs doivent collaborer ensemble pour détecter tous les comportements malveillants qui pourraient exister dans le réseau. Ainsi, les stratégies de détections doivent réaliser un compromis entre une détection crédible et un moindre coût (faible consommation d'énergétique, faible empreinte mémoire et un nombre réduit de messages de contrôle et d'alertes). Le problème marquant dans cette classe de SDIs est que les nœuds capteurs détecteurs puissent subir des actes de compromission ce qui va dégrader énormément l'efficacité du système. Comme remède à tel problème, il est exigé de changer

périodiquement les détecteurs et de mettre en place des techniques résistantes pour la sélection de ces détecteurs (exemple : des méthodes de sélection aléatoire).

C. **Les SDIs hybrides :** l'hybride entre l'architecture centralisée et distribuée consiste à réunir les deux modèles en un seul système de détection d'intrusion. Cela va permettre d'allier les avantages des deux architectures pour en avoir des SDIs à performances optimales.

D. **Les SDI hiérarchiques :** appelés également HIDS (*Hierarchical Intrusion Detection Systems*). Les SDIs hiérarchiques sont destinés aux réseaux de capteurs hiérarchiques où le réseau est organisé en clusters. Dans ce cas, les chefs des clusters et les membres se contrôlent mutuellement ; dans chaque cluster, il existe parmi les nœuds membres des agents qui surveillent le chef de leur cluster, et le chef du cluster lui-même peut être un agent qui surveille ses membres.

De plus, les systèmes de détection d'intrusion proposés pour les RCSFs peuvent soit adresser les attaques concentrées sur une seule couche dans la pile protocolaire des capteurs [12-13-14], ou bien être cross layer exploitant les interactions inter-couches pour détecter diverses attaques dans différents niveaux [20].

7. QUELQUES PROTOCOLES DE SÉCURITÉ DESTINÉS AUX RCSFs

Dans cette section nous présentons quelques protocoles de sécurité qui ont été proposés pour sécuriser les communications dans les RCSFs et qui ont connu plus de succès. Ces protocoles comportent un ou plusieurs aspects de sécurité traités dans la section 5.

7.1. Le protocole TinySec (Tiny Security)

Proposé dans [21], TinySec est un protocole de sécurité au niveau liaison de données ayant pour objectif l'assurance de l'authenticité, la confidentialité et l'intégrité des données dans un RCSF, tout en présentant un minimum d'exigences en termes de ressources de calculs, d'espace mémoire, d'énergie et de bande passante. TinySec définit deux sortes de protocoles : TinySec-Auth et TinySec-AE. TinySec-Auth assure uniquement l'authentification il est bon à utiliser dans les cas où les données Des capteurs ne requièrent pas d'être confidentielles. TinySec-AE garantit et la confidentialité et l'authentification des messages. Le protocole TinySec est repose sur le mécanisme d'authentification par code *(MAC : Message Authentication code)* pour l'authentification et le contrôle de l'intégrité, et l'algorithme CBC (*Cipher block chaining*) pour le chiffrement par blocs des données. Les résultats d'évaluation portée sur le protocole montrent que le débit et la latence sont acceptables avec les services sécuritaires au niveau liaison de données. La consommation énergétique induite est également admissible.

7.2. SPINS: Security Protocols for Sensor Networks

SPINS [22] est optimisé pour les réseaux sans fil dont les nœuds sont limités en ressources, tout comme les réseaux de capteurs. SPINS se base sur SNEP (*Secure Network Encryption Protocol*) et µTESLA (la version de *Timed, Efficient, Streaming, Streaming, Loss-tolerant Authentication Protocol*).

- **SNEP :** SNEP fournit la confidentialité des données, et l'authentification des données dans les deux côtés de communication, l'intégrité, et la fraîcheur des données. Avant de procéder au cryptage du message, l'expéditeur précède le message avec une chaîne binaire aléatoire. Cela empêche l'attaquant de déduire le texte en clair original du message crypté. Pour éviter d'ajouter des charges supplémentaires de transmission de ces bits supplémentaires, SNEP utilise un compteur partagé entre l'expéditeur et le récepteur pour le chiffrement par bloc en mode compteur (CTR). Les parties communicantes partagent le compteur et l'incrémentent après chaque bloc.

- **µTESLA :** µTESLA garantit l'authentification des entités de la communications. Il exige que la station de base et les nœuds soient lâchement synchronisés dans le temps, et que chaque nœud connaisse une limite maximale sur l'erreur de synchronisation. Pour envoyer un paquet authentifié, la station de base calcule tout simplement un MAC sur le paquet avec une clé qui est secrète. Quand un nœud reçoit un message, il peut vérifier que la clé correspondante au MAC n'a pas encore été divulguée par la station de base. Le nœud stocke le paquet reçu dans un tampon et au moment de la divulgation de la clé, la station de base diffuse la clé de vérification à tous les nœuds. Quand un nœud reçoit la clé divulguée, il peut facilement vérifier l'exactitude de cette dernière. Si la clé est correcte, le nœud pourra l'utiliser pour authentifier le paquet stocké dans son tampon.

SPINS est un protocole très avantageux pour les environnements capteurs. Il présente une bonne résilience aux attaques de rejeu et de fabrication de données.

7.3. RLEACH

Le protocole RLEACH [23] est une version sécurisée du protocole du routage LEACH, il traite le problème de sécurisation des communications inter et intra-clusters. Pour ce faire, il utilise un schéma de gestion de clé probabiliste appelé RPK (*Random Pair-wise Keys*) qui n'était pas directement applicable sur le protocole LEACH malgré ses avantages[1], car il ne parvient pas à garantir que chaque pair de nœuds aura une clé partagée. C'est pourquoi, il fallait l'améliorer pour l'adapter au protocole LEACH, par l'emploi d'une fonction de hachage à sens unique.

[1] Il assure et l'authentification nœud à nœud et l'économie d'énergie.

L'opération du protocole RLEACH est divisée en quatre phases, la phase de pré-distribution, la phase de découverte de clés partagées, la phase de construction des clusters et la phase de relais des données.

- **La phase de pré-distribution :** Tout d'abord un grand ensemble de clés est généré et sauvegardé dans le puits, l'ensemble des nœuds capteurs est divisé par RPK en plusieurs groupes $G_i$selon une certaine probabilité. A chaque groupe correspond une information publique dite *seed* qui est sauvegardée dans la mémoire de tous les nœuds du groupe et qui sert comme un identifient de ceci. Chaque nœud est pré-distribué avec un identificateur *IDx* et une clé originale K_i relative à son identificateur, la valeur du seed correspondante à son groupe et la fonction *H ()*. Ainsi, Il doit choisir aléatoirement *m* clés, qui vont être par la suite partagés avec d'autres nœuds. En plus de la plage de tous les clés, le puits doit sauvegarder aussi les identités de tous les nœuds et la fonction *H ()*.

- **La phase de découverte de clés partagées :** Dans cette phase, chaque nœud diffuse son identificateur à l'ensemble de ses voisins, par exemples : un nœud *x* diffuse son identificateurID_x, le nœud *y* qui est l'un des voisins, reçoit le message et vérifie tout d'abord si le nœud x appartient à son groupe. Si c'est le cas, il calcule la clé de conversation *K* = H ($seed_i$, ID_x, ID_y) et la renvoie avec son identificateur (ID_y) au nœud x. Le nœud x invoque la même fonction et s'il trouve le même résultat, la clé *K* sera validée comme clé partagée entre les nœuds x et y. En revanche, si les deux nœuds sont dans deux groupes différents, les nœuds vont vérifier dans leurs trousseaux de clés s'ils détiennent d'une clé commune.

- **La phase de configuration (la construction de clusters):** Le principe de construction des clusters est le même que celui du protocole LEACH, sauf que la sélection du cluster n'est pas basée uniquement sur la puissance de signal d'annonce envoyé par le chef du cluster, mais aussi sur l'existence ou non d'une clé partagé entre eux.

- **La phase de relais des données :** Dans cette phase, la communication des données ce fait en deux étapes : dans chaque cluster, les nœuds membres envoient leurs messages au chef du cluster suivant l'ordonnancement TDMA (*Time division Multiple Access*), le chef agrège les données reçues et communique le résultat obtenu directement au puits. La sécurité des communications intra-clusters est assurées par les clés que partagent chacun des membres avec le chef de son cluster (authentification nœud à nœud offerte par RPK), tandis que la sécurisation des communications entre les chefs de clusters et le puits est garantie clés originales pré-chargées.

8. Conclusion

A travers ce chapitre, nous avons présenté des généralités sur la sécurité dans les réseaux de capteurs sans fil. Nous avons accentué les défis et les principaux aspects liés à la sécurisation des communications dans tels réseaux contraints, ainsi que la protection des nœuds capteurs eux-mêmes

contre toute manipulation mal intentionnée. Finalement, nous avons présenté les blocs fonctionnels de la sécurité dans les RCSFs avec quelques exemples illustratifs de mécanismes et de protocoles les plus connus dans ce contexte.

CHAPITRE 3 :

Généralités sur l'Internet des Objets

1. Introduction

L'Internet des objets ou IoT (*Internet of Things)* [24], est un paradigme émergeant dans le monde des réseaux informatiques. Il peut être défini qui comme une évolution et extension de l'Internet de nos jours pour l'inclusion de tous les objets et les endroits dans notre entourage (réfrigérateurs, thermostat, maisons, véhicules, routes, etc.). Le concept prometteur de l'IoT va nous simplifier la vie, nous faire gagner du temps, décharger notre cerveau de la mémorisation de données logistiques (itinéraires, temps de prise des médicaments, etc.). Ainsi, l'accès ubiquitaire à différents types d'informations permettrait la sophistication du mode de vie et une amélioration significative de la qualité des services dans différents domaines.

L'IoT qui est une nouvelle vague de l'Internet, est en réalité une partie naissante de l'Internet du futur, appelé l'Internet de tous les objets ou IoE (*Internet of Everything*), qui vise à interconnecter les gens, les données et tous les objets, de telle sorte qu'il y ait une fusion entre le monde réel (physique) et le monde numérique (virtuel) ; les objets du monde physique vont être incorporés dans le monde virtuel de l'Internet. Cela fait appel à de nouvelles tendances et innovations que ce soit sur le plan architectures de communications ou sur le plan présentation et exploitation des services.

Ce chapitre est consacré à la présentation du domaine de l'Internet des objets et les aspects qui s'y rapportent.

2. Historique de l'Internet des objets

L'émergence de l'Internet des objets ce n'est qu'un résultat de convergence entre multiples technologies, à savoir l'Internet, la communication sans fil, les systèmes embarqués, systèmes micro-électroniques et la nanotechnologie. Dans cette section, nous citons les évènements les plus marquants sur le chemin de la concrétisation de l'IoT.

Le concept d'un réseau de dispositifs intelligents a été évoqué pour la première fois en 1982, avec le premier appareil connecté à Internet à l'Université Carnegie Melon capable de signaler à son inventaire si les boissons nouvellement chargées sont bien froides. Ainsi, en 1991, Mark Weiser a introduit l'informatique omniprésente à travers son papier intitulé : «L'ordinateur du 21ème siècle » et a présenté d'avance la vision contemporaine de l'Internet des objets. Un peu plus tard, en 1994, Steve Mann avait créé le *WearCam* qui était parmi les premières caméras à apparaitre sur le web. WearCam comporte les parties suivantes : (1) un groupe de caméras (ou uniquement une) qui sont fixées au corps, d'une manière quelconque, à deux mains libres (2) des moyens d'enregistrement, de traitement et de transmission des images capturées par les caméras (3) un moyen d'affichage qui a la capacité de présenter une image ou un flux d'images de l'appareil photo. Les images capturées seront communiquées vers une entité (une station de base) à la disposition de l'utilisateur. Ensuite, en 1998, l'informatique ubiquitaire a commencé d'attiré l'attention par le fait qu'elle permettrait l'incorporation flexible et efficace de l'informatique dans la vie quotidienne. Mark Weiser disait : « là où la réalité

virtuelle met l'humain en dedans du monde des ordinateurs, l'informatique ubiquitaire force plutôt l'ordinateur à s'instaurer dans le monde réel».

En 1999, la désignation Internet des objets a été prononcée pour la toute première fois par Kevin Ashton. Après, en 2000 la société LG annonce son premier réfrigérateur intelligent connecté à Internet. De plus, la technologie RFID (*Radio Frequency IDentification*) qui est l'une des technologies constitutionnelles de l'IoT, a commencé à être massivement déployée vers les années 2003 et 2004. D'autre part, une initiative très intéressante a été prise en 2008 ; un groupe de recherche appelé IPSo Alliance s'est consacré à promouvoir l'utilisation du protocole IP (*Internet Protocol*) pour les réseaux d'objets miniatures intelligents.

De nombreux travaux de recherches ont été succédés et se sont tous concentrés autour de la réalisation, dans les meilleures conditions, de la vision de l'Internet des objets et la mener à sa maturité en dépit de tous les défis soulevés. Cela avec la considération des progrès technologiques continus dans le marché des dispositifs intelligents et dans le domaine de technologies de télécommunication (comme : le *cloud computing*, le concept du SDN (*Software-Defined Networking*) [25], etc.).

3. Typologie des objets

Avec l'avènement de l'Internet des objets, la connexion Internet acquiert une troisième dimension; en plus de la possibilité de se connecter n'importe quand et n'importe où, il est désormais possible d'être connecté avec n'importe quel objet. De plus, les objets connectés sont identifiés de façon unique et sont capable de récolter des informations environnementales (liées aux changements des paramètres de l'environnement, comme la température) ou comportementales (issues des variations d'état de l'objet lui-même ou des objets contextuels), de les traiter et de les communiquer sur Internet. D'où vient leur appellation par objets intelligents.

CisCo prévoit que d'ici quelques années, spécifiquement en 2020, l'Internet des objets sera une réalité et le nombre d'objets connectés dépassera les 50 milliards [26]. A ce stade, il est nécessaire de noter que les données massives générées par un nombre immense d'objets intelligents connectés présente, partiellement, une source de la charge globale de données qualifiées de *BigData* sur Internet [27].

On distingue différents types de dispositifs connectés à l'IoT, ou qui font connecter d'autres objets à Internet, dont on cite principalement:

3.1. Les objets d'identification

Codes barre, marqueurs RFID et autres dispositifs miniaturisés qui servent à l'identification et la traçabilité des objets sur lesquels ils sont collé pouvant être collés sur les objets d'usage courant (ex. vêtements, marchandises, livres, véhicules, etc.). Ce type de dispositifs nécessite qu'il y ait un lecteur

pour récupérer leurs données qui seront par la suite téléchargées sur un serveur et deviennent alors accessibles via le système d'information d'une organisation ou directement sur Internet.

3.2. Les capteurs

Les capteurs dans l'IoT permettent de récolter des informations contextuelles concernant les objets dans lesquels ils sont intégrés, ou les environnements sur lesquels ils sont déployés. Les capteurs communiquent les informations collectées sur Internet d'une manière directe ou indirecte, tout dépend du modèle adopté pour l'intégration des réseaux de capteurs à l'internet.

3.3. Les drones

Un drone désigne un aéronef miniature sans pilote, pouvant porter des charges utiles, communiquer et exécuter des commandes en toute flexibilité. Les drones sont utilisés dans des applications civiles aussi bien que dans des applications militaires pour accomplir des missions bien déterminées. On entend parler de l'efficacité de l'utilisation des drones dans le domaine commerciale pour par exemple, les livraisons à domicile des commandes faites sur Internet. Aussi, des opérations de sauvetage, d'exploration et de surveillance sont réalisables par les drones dans le contexte des applications militaires. Bien que la technologie (ou bien son prototype) des drones en elle-même existait depuis bien longtemps, son exploitation idéale dans différentes applications demeure modeste. Récemment, les drones sont élus pour faire une importante part de l'Internet du futur, soit en tant que objets intelligents terminaux rapportant des données de contrôle, soit en tant que routeurs particuliers (mobiles et volants) de données entre les parties connectées à Internet. Comparés aux capteurs qui sont le plus souvent stationnaires ou dans certains cas mobiles mais dans tous les cas, manquent de l'aspect aérien, un drone parvient très efficacement à donner une vision aérienne sur l'état de la zone à contrôler même dans les zones isolée et/ou inaccessibles (là où il est difficile d'installer une infrastructure terrestre avec des points d'accès et des stations de base).

3.4. Smartphones et tablettes électroniques

Les smartphones et les tablettes qui sont déjà connectés à Internet par le biais de diverses technologies (Wi-Fi, 3G, 4G) permettent aux utilisateurs de communiquer à distances avec les autres types d'objets connectés dans l'IoT. Les objets intelligents peuvent rapporter en temps réel l'état actuel aux utilisateurs via Internet. Dans ce cas, les utilisateurs reçoivent des e-mails ou simplement des messages d'alertes sur leurs Smartphones ou tablettes, tout dépend de l'application. Il est meme possible que les utilisateurs supervisent ou ordonnent leurs objets connectés, à distance, via leurs smartphones ou tablettes.

La figure ci-dessous présente les principaux types d'objets dans l'IoT.

Figure 3.1. Typologie des objets dans l'IoT.

4. CYCLE DE VIE D'UN OBJET CONNECTÉ DANS L'IOT

Dans l'IoT, les objets intelligents passent par trois étapes : la phase préparatoire (*bootstrapping*, la phase opérationnelle et la phase de maintenance [24].

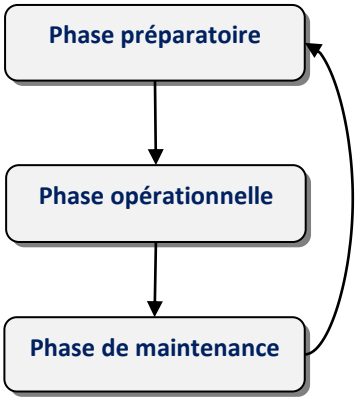


Figure 3.2. Cycle de vie de l'objet.

- **La phase préparatoire (*bootstrapping)* :** déploiement des objets (capteurs, tags), leur configuration avec les informations nécessaires, par exemple les identificateurs, les clés de sécurité, etc.
- **La phase opérationnelle :** dans la phase opérationnelle, l'objet connecté se met à réaliser sa mission qui diffère d'une application à une autre.
- **La phase de maintenance :** effectuer des mises à jours, régler les problèmes en faisant d'éventuelles réparations des objets en cas de défaillances par exemple. Il est même possible de remplacer carrément des objets et redémarrer à nouveau à partir de la phase préparatoire.

5. TECHNOLOGIES FONDATRICES DE L'INTERNET DES OBJETS

Bien que l'Internet des objets soit une notion relativement nouvelle, les technologies qui la rendent possible existaient depuis quelques années déjà. On parle alors des réseaux de capteurs sans fil et

de la technologie d'identification par radio fréquence. Les évolutions observées par les technologies sans fil et le domaine des réseaux de télécommunication d'une part, et l'Internet de l'autre part, ont permis d'ouvrir de nouvelles perspectives pour ces technologies, qui ont pu s'instaurer efficacement dans notre vie quotidienne et qui sont devenues de plus en plus omniprésentes. Ainsi, de nouvelles facilités et de nouveaux modes d'exploitation des services peuvent être envisagés si les capteurs et les marqueurs d'identification intègrent l'Internet. Dans cette section nous présentons les technologies basiques et nous accentuons leurs rôles dans le contexte de l'Internet des objets.

5.1. L'identification par radio fréquence (RFID)

Un système RFID [28] est composé d'un ou plusieurs lecteurs et d'un ensemble d'étiquettes (appelée aussi tags, marqueurs, identifiants ou transpondeurs) à micro-puissances. Les étiquettes sont des dispositifs minuscules équipées d'une puce contenant des informations et une antenne pour la communication radio. Elles sont placées sur les éléments que l'on veut identifier d'une manière unique ou tracer. Les étiquettes peuvent avoir différentes formes (figure 3.3) et peuvent être passives ou actives. Les étiquettes actives sont équipées d'une batterie, elles diffusent des signaux automatiquement et d'une façon autonome, tandis que les étiquettes passives ne disposent d'aucune source d'énergie et attendent à ce qu'un signal électromagnétique leur arrive et munit de l'énergie pour pouvoir envoyer leurs propres signaux. Les étiquettes passives sont plus déployées que celles qui sont actives car leur usage est beaucoup plus flexible avec un coût nettement réduit (comparé au coût relatif aux étiquettes actives qui est nettement élevé). Une autre spécificité pas moins importante dans les étiquettes passives qui est la durée de vie. Par le fait d'être passive, la durée de vie de l'étiquette est importante (elle reste valable tant qu'elle garde son bon état), ce qui n'est pas le cas pour une étiquette active où la durée de vie est restreinte (s'achève avec l'épuisement de la batterie).

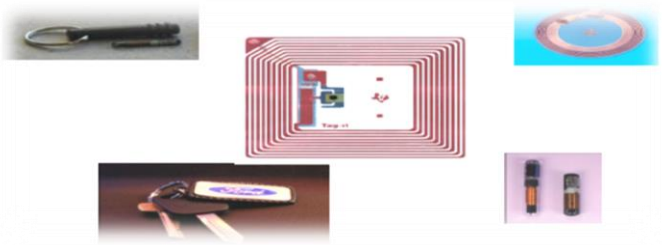


Figure 3.3. Formes des étiquettes RFID.

Etiquette passive **Etiquette active**

Figure 3.4. Types des étiquettes RFID.

Le processus d'identification se réalise à travers un scénario bien déterminé. En effet, le lecteur active les étiquettes qui passent devant lui en leur envoyant un signal électromagnétique puissant. Les étiquettent s'active et réagissent en répondant par un signal transportant les identités. Contrairement aux systèmes d'identification par codes barre qui exigent que le lecteur et le code barre soient exactement opposés et très proches l'un de l'autre, dans un système RFID, il suffit juste que le lecteur et l'étiquette soient l'un dans la portée de communication de l'autre pour que l'interaction puisse avoir lieu. La portée de communication radio (appelée aussi la distance de lecture) dans un système RFID dépend du type de tag (passif ou actif) et de la gamme de fréquences utilisée. Par exemple, la portée avec les étiquettes actives est plus importante qu'avec celles qui sont passives.

Dans le contexte de l'Internet des objets, les objets intelligents ont besoin d'être identifiés de façon unique. A partir de là, l'adoption de la technologie RFID s'est avérée nécessaire.

5.2. Les réseaux de capteurs sans fil

Nous avons déjà introduit les réseaux de capteurs sans fil dans le chapitre 1, où nous avons présenté profondément les aspects liés à cette technologie et comment elle permet la surveillance efficace de notre environnement et de nos activités. Outre les RCSFs classiques qui étaient déployés pour des applications privées où les données de captage étaient récupérables à partir des seules stations impliquées (station de base ou l'ordinateur du gestionnaire de taches), la nouvelle génération des RCSFs est désormais invitée à intégrer l'Internet en un bond audacieux, compte tenu de la nature ultra particulière de ce type de réseaux contraints. Dans ce dernier cas, les rapports des capteurs intégrés à Internet sont accessibles de n'importe où et n'importe quand, à partir d'un autre bout connecté également à Internet. Donc, l'accès et la récupération des données de captage deviennent ubiquitaires.

Les RCSFs jouent un rôle très intéressant dans l'Internet des objets. En effet, les capteurs permettent la représentation des caractéristiques dynamiques (température, humidité, pression, mouvements, …) des objets et des endroits du monde réel dans le monde virtuel représenté par le réseau Internet global. Ainsi, avec l'incorporation des réseaux de capteurs dans l'Internet, Les capteurs deviennent des serveurs (fournisseurs de services) dans ce que l'on désigne par le web des objets (dit WoT pour *Web of Things*) [29]. Ainsi, les services (applications) des RCSFs se rajoutent à l'ensemble des services et applications de l'Internet de futur qui réunira une variété de réseaux fortement hétérogènes (que ça soit sur le plan matériel ou logiciel), soumis à des contraintes différentes et qui sont déployés pour diverses applications, afin d'en avoir un monde réel très sophistiqué.

En plus de ces deux technologies principales (RFID et RCSFs), on trouve également d'autres technologies qui contribuent à la concrétisation du principe de l'Internet des objets. On parle alors des systèmes embarqués et la nanotechnologie (rétrécissement et incorporation des capteurs et autres

dispositifs miniatures dans les objets à faire connecter à Internet), comme montré dans la figure suivante.

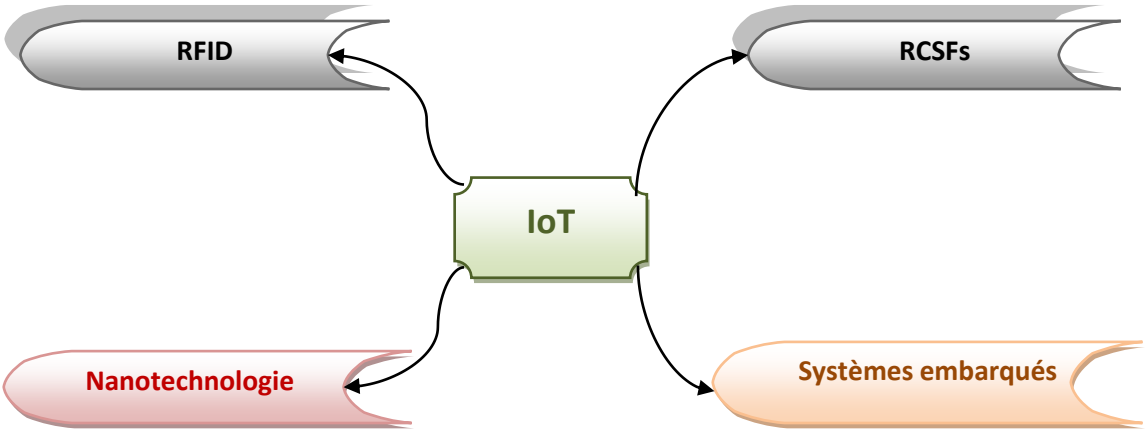


Figure 3.5. Technologies fondatrices de l'Internet des objets.

6. Architecture de l'Internet des objets

De point de vue architectural, on peut dire que l'Internet des objets est organisée en trois couches principales [26] : la couche de perception de donnée, la couche réseau et troisièmement la couche application. La figure ci-dessous illustre telle organisation.

6.1. La couche perception

La couche perception, au niveau bas dans la hiérarchie, est responsable de la capture de données, ainsi que leur identification dans leur environnement. Cette couche comprend ainsi le matériel nécessaire pour parvenir à la collection de données contextuelles des objets connectés, à savoir les capteurs, les étiquettes RFID, caméras, GPS (*Global Positioning System*), etc.

6.2. La couche réseau

Cette couche se charge de la transmission fiable des données générées dans la couche perception ainsi que l'assurance de la connectivité inter-objets connectés et entre objets intelligents et les autres hôtes de l'Internet. D'autre part, il est prévu que les données issues de la couche perception soient énormes car le nombre d'objets connectés à Internet ne cesse d'augmenter à grands pas. De ce fait, il s'est avéré nécessaire de mettre en place des mécanismes et des équipements de stockage et de traitement massif de ces données sur Internet, à faible coût. Cela est bel et bien garanti par les services *cloud* [30] qui assurent une gestion élastique des ressources de mémorisation et de traitement sur les géants centres de données résidant sur Internet et qui sont en mesure d'absorber efficacement la charge de données générée du côté de l'Internet des objets. à ce stade, il est important de noter que le *cloud* utilise un concept récent dénommé SDN (*Software Defined Networking*) qui vise une méthode de gestion abstraite basée sur le découplage des

fonctionnalités décisionnelles et opérationnelles des équipements réseau, en vue de pouvoir déployer les tâches de contrôle sur des plateformes beaucoup plus performantes que les commutateurs classiques. Cela va réduire davantage la latence réseau et rendre possible l'automatisation de la gestion du large ensemble de serveurs sur le *cloud* et leur auto-configuration.

6.3. La couche application

Quant à elle, la couche application définit les profiles des services intelligents et les mécanismes de gestion de données de différents types, provenant de différentes sources (différents types d'objets). Dans la section suivante nous abordons cet aspect applicatif et ce que représentent réellement les services intelligents dans chaque champ d'application.

L'architecture peut être étendue à une quatrième couche dite la couche middleware [31] entre la couche application et les deux autres couches. Cette couche sert pour une interface entre la couche matérielle et les applications. Elle comprend des fonctionnalités assez compliquées permettant la gestion des dispositifs, et traite aussi l'agrégation, l'analyse et le filtrage de données et le contrôle d'accès aux services. La couche middleware permet également la dissimulation de la complexité des mécanismes de fonctionnement du réseau et rend plus facile le développement des applications par les concepteurs.

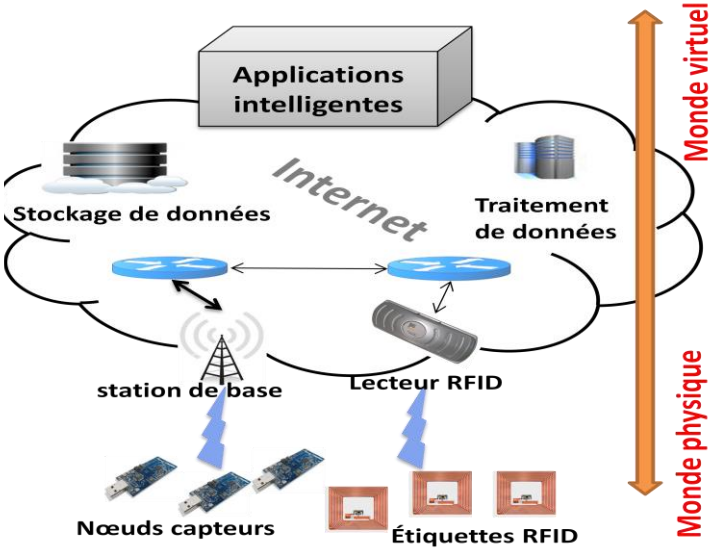


Figure 3.6. Architecture de l'internet des objets.

7. Paradigmes de communication

En plus des communications humain à humain qui ont régné sur l'Internet classique, de nouveaux styles d'interactions émergent avec l'apparition de l'Internet des objets comme le montre la figure ci-dessous qui illustre ces interactions inter objets connectés et entre l'humain et le(s) objet(s) dans l'IoT.

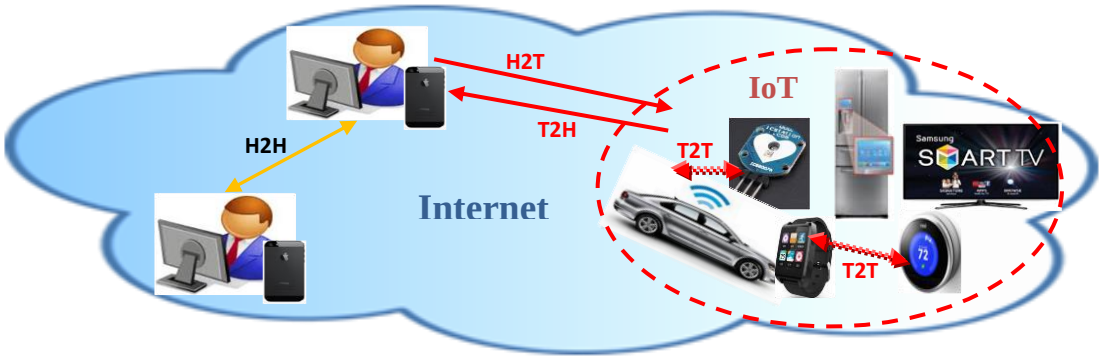


Figure 3.7. L'émergence de nouveaux paradigmes de communication dans l'Internet du futur.

7.1. Les communications humain-à-objet

L'utilisateur peut interroger des objets connectés à Internet à tout moment via son smartphone (ou autre dispositif connecté). Les communications humain-à-objet (dite aussi H2T pour *Human-to-Thing*) [32] sont très fréquentes dans certaines applications de l'Internet des objets (voir section 8) comme est le cas d'une application médicale ou de l'automatisation des maisons. Tel type d'interactions est caractérisé par une forte hétérogénéité matérielle et technologique car du côté de l'utilisateur on utilise généralement des équipements beaucoup plus puissants (ordinateur portable, Smartphone ou tablette) que les capteurs contraints du côté de l'objet sollicité dans l'IoT. Cependant, l'hétérogénéité dans toutes ses formes doit être traitée efficacement.

7.2. Les communications objet-à-objet

Les communications objet-à-objet (ou T2T pour *Thing-to-Thing*) sont appelées également machine-à-machine ou M2M (*Machine-to-Machine*) [33]. Cela désigne des communications automatiques et autonomes inter-machines sans l'intervention humaine. Rappelons que les communications M2M forment la base de l'informatique pervasive qui fait partie de l'ensemble des principes et concepts de l'Internet du futur. En fait, les interactions inter-objets intelligents dans l'IoT sont souvent homogènes, du moins au niveau des contraintes où on trouve des capteurs qui peuvent utiliser différentes technologies de transmission mais qui observent les mêmes limitations en termes de ressources et qui ont les mêmes vulnérabilités.

Rappelons à ce stade que les communications allant des objets connectés dans l'Internet des objets vers les hôtes ordinaires de l'Internet (les communications objet-à-humain ou T2H*: Thing-to-Human* en anglais) sont aussi considérées comme une variante des communications M2M. Par exemple, un capteur associé à une porte d'une salle à accès restreint dans une banque, est configuré de telle sorte qu'il avise par MMS (ou e-mail) le responsable de la sécurité dans la banque (via son smartphone) en lui transmettant le temps d'entrée de la personne ainsi que sa photographie. Cette opération se fait même si la personne était déjà authentifiée avant d'accéder la salle.

8. Les applications de l'Internet des objets

L'Internet des objets ce n'est pas qu'un immense ensemble d'objets intelligents interconnectés et connectés à Internet mais c'est également et plus considérablement, les applications qui sont en fait la raison d'être de cette nouvelle vague de connectivité sur Internet. L'existence des objets intelligents avec de nouvelles possibilités de communications automatiques et intelligentes vont sensiblement améliorer le mode de vie des gens ainsi que la qualité de services dans divers domaines à travers des degrés élevés d'autonomie et d'intelligence.

Les potentialités de l'Internet des objets ont mené à ce que des modèles de nouvelles applications soient développées sur Internet. Dans cette section, nous citons les applications en vedette de l'IoT [26].

8.1. Les applications médicales

L'IoT aura de nombreuses applications dans le secteur de la santé où l'objectif est d'arriver à prévenir des situations graves et de suivre à distance des patients atteints des maladies chroniques et agir rapidement si cela s'est avéré nécessaire. Des capteurs corporels implantés dans le corps du patient récoltent des informations relatives aux paramètres médicaux, telles que la température, la glycémie, le rythme des battements du cœur ou encore même la tension artérielle. Ces informations seront stockées et traitées sur Internet (plus précisément sur un *cloud*) et mises à la disposition du médecin qui pourra les consulter n'importe quand et depuis n'importe quel dispositif connecté à Internet (ex : son Smartphone ou sa tablette). Le médecin est alerté en temps réel (en lui envoyant un mail ou un SMS) de tout changement brusque concernant l'état de son patient. Suivant le degré de gravité de la situation, le médecin réagit soit en se déplaçant chez le patient ou juste en le contactant et lui indiquant ce qu'il faut faire pour revenir à l'état normal. Imaginons par exemple un patient avec un rythme cardiaque irrégulier. Le capteur détectant tel évènement déclenche une alerte au cardiologue s'occupant du patient. Le médecin peut également consulter à tout moment les rapports médicaux de ses patients ou bien interroger les capteurs pour avoir les valeurs actuelles.

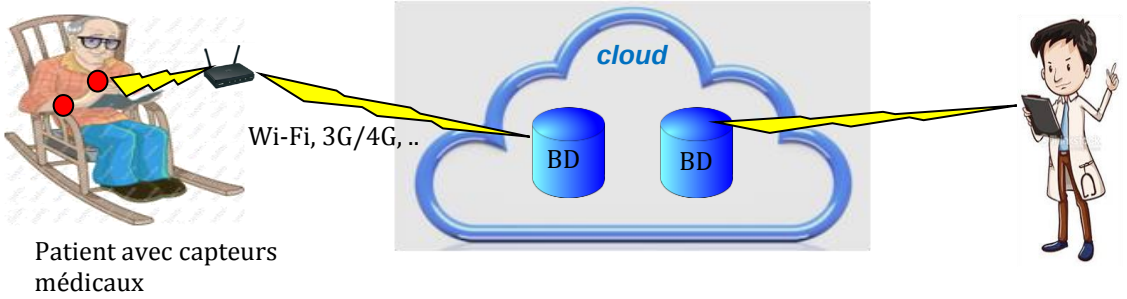


Figure 3.8. L'internet des objets dans le domaine médical.

8.2. Les applications militaires

L'Internet des objets est un domaine fertile tant pour les applications civiles que pour les applications militaires. Dans le domaine de défense les capteurs et les nano-drones connectés à Internet permettent d'envisager des applications sophistiquées pour l'exploration, la surveillance des champs de batailles et des frontières, ainsi que la poursuite et la localisation géographique des objets connectés. Les forces militaires ont la tendance d'utiliser des infrastructures propriétaires pour la connectivité et les communications. En transitant vers l'Internet, il sera plutôt possible d'utiliser des infrastructures *cloud,* qui offrent une flexibilité opérationnelle très intéressante. Le soldat en mission peut lui-même être connecté à Internet à travers les capteurs connectés, intégrés dans sa tenue. Ces capteurs peuvent être par exemple des capteurs médicaux qui rapportent l'état de santé du soldat, ou des capteurs multimédia qui captent des images, une vidéo ou du son depuis la zone où il se trouve (le soldat).

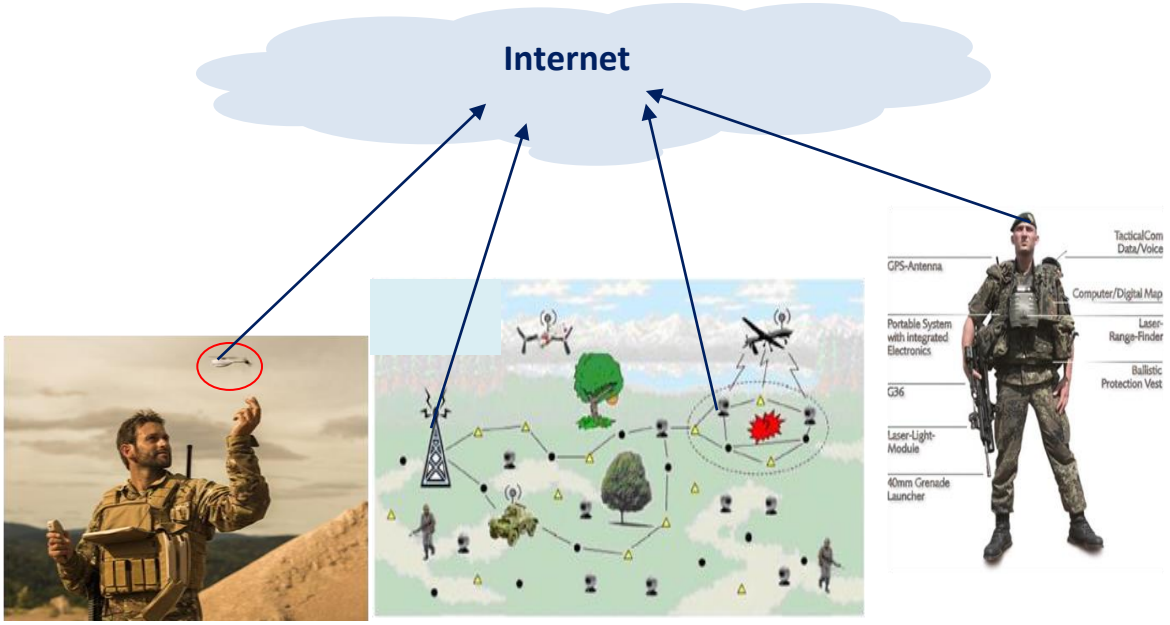


Figure 3.9. Le domaine militaire et l'Internet des objets.

8.3. Les applications industrielles

Le secteur industriel est un autre domaine qui va être bouleversé par l'Internet des objets. Une quantité considérable de capteurs et d'étiquettes RFID et les contrôleurs embarqués, s'accroît sensiblement dans les systèmes de production industrielle, sur la chaîne logistique et même dans les produits. Ce qui aide les entreprises à améliorer la qualité de leurs processus de fabrication et à fournir un service après-vente plus concurrentiel. Ainsi, les usines connectées à Internet sont plus productifs, efficaces et intelligents que ceux qui ne le sont pas. En effet, le producteur peut également avoir une idée sur la commercialisation de ses produits à travers le monde, à l'aide des informations collectées auprès des différents points de ventes. D'autre part, les produits connectés seront capables de transmettre les avis (*feedback*) des clients aux producteurs pour faire un sondage sur le taux de satisfaction de la clientèle. Ils acquièrent et communiquent aussi des données pertinentes concernant

les susceptibles pannes, les préférences des utilisateurs, ou autre. Il est important de noter que les communications M2M jouent un rôle prépondérant dans l'automatisation des processus industriels et des interactions inter composants opérationnels.

8.4. Les maisons intelligentes

La maison du futur sera un objet connecté à Internet accessible à distance par ses propriétaires via des Smartphones, tablette ou ordinateurs connectés. La porte, la télévision, le thermostat, le réfrigérateur, les parapluies, les montres, etc. de telle sorte qu'une porte connectée informe les parents par Internet de la rentrée de leurs enfants. La télévision qui était seulement un terminal récepteur. Connectée à Internet, elle (la télévision) devient plutôt un dispositif émetteur/récepteur qui fournit à ses téléspectateurs la possibilité d'envoyer et recevoir des e-mails, faire des appels téléphoniques sur Internet, ou autre. Un thermostat intelligent connecté au réseau Wi-Fi de la maison permet de contrôler facilement la température de celle-ci à partir de n'importe où, pour une amélioration du confort et une optimisation des économies énergétiques. Le réfrigérateur intelligent connecté à Internet et muni d'un système RFID traque les produits élémentaires qui y sont stockés et enregistre des informations pertinentes leur concernant (comme la durée du stockage et la date d'expiration). L'utilisateur peut l'interrogé à distance pour savoir ce qui reste et ramener les produits manquant avant de rentrer à la maison. Ou alternativement, le réfrigérateur peut être programmé pour commander automatiquement les produits qui manquent.

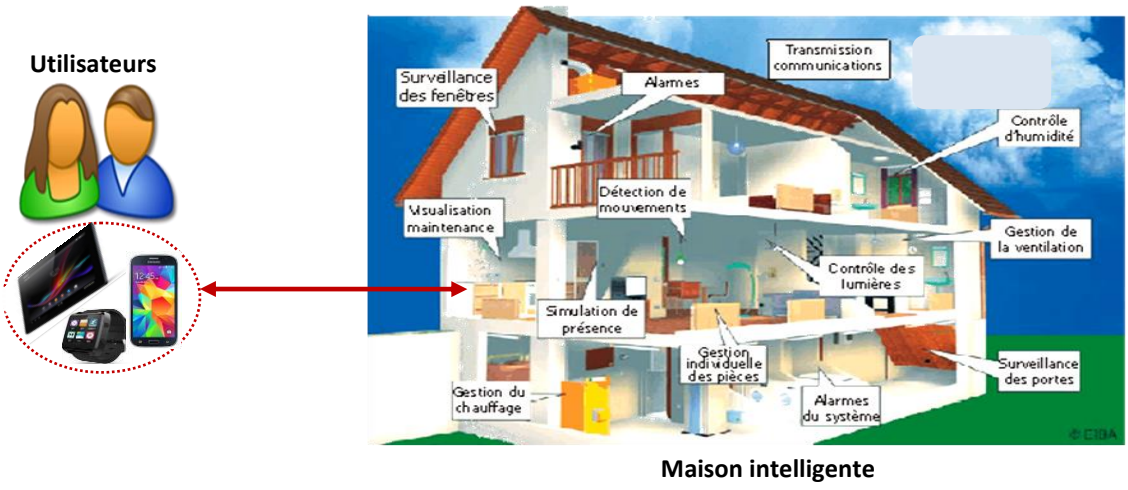


Figure 3.10. L'Internet des objets et la domotique.

8.5. Les villes intelligentes

Pas que les maisons, les routes, les bâtiments, les véhicules, les magasins, les parkings, etc. seront tous connectés à Internet et annoncent leurs présence les uns aux autres objets connectés pour contrôler le trafic routier, aider les citoyens (surtout les automobilistes) à gagner le temps en leur fournissant des informations pertinentes, en temps réel, sur l'endroit où il se trouve (par exemple le

plus proche parking, hôtel, restaurant, hôpital et autres) et des informations d'ordre général sur la ville, comme la température, le taux d'humidité les niveaux de radiation, … de même, les autorités de la ville intelligente trouveront une facilité de réalisation des tâches de contrôle de la pollution, l'éclairage urbain, etc. notons qu'une coexistence massive de multiples technologies est nécessaire pour la réalisation du projet de la ville intelligente, principalement les réseaux de capteurs.

Des applications avantageuses pas moins intéressantes peuvent être envisagées dans d'autres domaines à savoir l'agriculture de précision, où le principe est le même dans tous les cas: permettre un accès ubiquitaire aux informations relatives aux différents types d'objets intelligents existants dans notre environnement afin de parvenir à automatiser le contrôle et optimiser les rendements.

9. Les avantages de l'Internet des objets

Nous avons cité dispersement dans différentes parties du présent chapitre quelques avantages de l'Internet des objets. Dans cette section, nous résumons les principaux avantages de l'IoT.

- Accès ubiquitaire à l'information pour un monde plus intelligent et un mode vie sophistiqué et confortable.
- Amélioration de la qualité de service et de la télésurveillance dans différents domaines d'applications, à savoir le domaine industriel, médical, etc.
- Améliorer la productivité et l'expérience-client : les objets connectés envoient des rapports à leurs constructeurs indiquant les préférences et les habitudes des clients aidant davantage les entreprises à agir de manière proactive et adaptée qui satisfait la demande et les exigences de la clientèle.
- Le gain du temps est un autre avantage de l'IoT. Les déplacements inutiles sont dès lors remplacés par une simple navigation sur le web pour commander des produits, contrôler l'état des objets et/ou endroits connectés.
- Dans certaines applications, l'IoT nous permet même de rationaliser nos dépenses et faire des économies car on ne consomme qu'en cas de besoin, que ça soit pour les achats ou la consommation énergétique (nécessaire pour l'éclairage ou la climatisation) ou autre.
- Possibilité d'exploitation des ressources géantes de l'Internet pour le stockage et le traitement des données écoulées de l'IoT.

10. Les enjeux de l'Internet des objets

Bien que l'Internet des objets soit un concept qui est à la fois avantageux et prometteur, et qui pourra apporter des solutions efficaces des problèmes du suivi et de télésurveillance dans différents domaines. En contrepartie, l'IoT soulève certaines questions décisives, étroitement liées à sa maturité et son acceptabilité. On cite ci-dessous les enjeux les plus marquants.

- **La sécurité :** la sécurité des personnes, des communications, des données, des services, des réseaux et des équipements était et continue à être un problème sévère observé par l'internet courant. aujourd'hui avec la naissance de l'IoT, l'amplitude du problème va prendre un autre ordre de gravité. Des milliers d'objets contraints connectés en permanence à internet et intégrés dans toute sorte d'objets dans notre vie quotidienne, vont porter le risque d'être ciblés par les menaces classique de l'Internet. Il est même possible que de nouvelles générations d'attaques apparaissent. Donc, les objets intelligents dans l'IoT, la transmission et le stockage de leurs données sur Internet devraient être sécurisés.
  D'autre part, l'IoT peut lui-même menacer la sécurité des individus ou des institutions. L'armée chinoise proscrit les officiers et les soldats de porter des objets connectés (comme les montres et les lunettes connectées à Internet) et considère leur utilisation comme une violation de la réglementation sur le secret dans les casernes [129].

- **La protection de la vie privée des utilisateurs :** un grand nombre de capteurs connectés à Internet et intégrés dans des objets d'usage quotidien révèlent nos habitudes notre état de santé notre localisation géographique et autres types d'informations qui nous sont privées. Il devra absolument y avoir des mécanismes robustes qui peuvent assurer la confidentialité des données que l'utilisateur qualifie être sensibles. Les utilisateurs devraient également pouvoir savoir qui accède quelles données (concernant les utilisateurs) sur Internet et pour quelle raison.

- **Les limitations de ressources :** les capteurs et les tags RFID sont très limités en ressources de calculs, de stockage mémoire et d'énergie. A cet effet, les solutions (protocoles de communications ou de sécurité, technologies de transmission, etc.) destinées à l'Internet des objets doivent prendre en considération telles contraintes et limitations.

- **L'hétérogénéité :** des dispositifs de divers types ayant des capacités variées et appartenant à des réseaux de différentes natures, vont intégrer l'Internet en utilisant différentes technologies de communication (filaire, sans fil, satellitaire, …). Avec toutes ces formes d'hétérogénéités matérielles et technologiques, il serait primordial de mettre en place des mécanismes bien avertis qui soient capables d'en cacher et gérer.

- **L'interopérabilité :** c'est parmi les plus grands défis de la réalisation de l'Internet des objets. L'interopérabilité c'est, en réalité, la cohabitation des dispositifs, des systèmes et des mécanismes disjoints et la possibilité de les faire coopérer et interagir en toute flexibilité. Une tendance récente tend vers la standardisation et l'unification des systèmes et protocoles opérationnels dans l'IoT et de les présenter en *open source* (à accès libre). Ceci afin de faciliter la collaboration entre objets connectés, ainsi que le couplage avec les entités externes se trouvant sur Internet.

- **La virtualisation :** plusieurs capteurs connectés peuvent représenter un seul capteur virtuel qui rapporte une mesure virtuelle résultant de l'agrégation de plusieurs états secondaires. Par

exemple un capteur virtuel qui nous dit si l'état de santé du patient est bon ou non. Cette information n'est qu'une combinaison de plusieurs informations fournies par plusieurs capteurs médicaux réels incorporés dans le corps du patient. Ainsi, un model générique de virtualisation des objets connectés à l'IoT, nommé VoT (*Virtualization of Things*) [34] permet une représentation abstraite des objets et l'accumulation des données qui en proviennent, depuis différents endroits, pour faciliter leur contrôle.

- **La transparence :** l'objectif de l'informatique transparente est de rendre les systèmes informatiques des boites noires transparentes à travers des communications sans fil, automatiques et invisibles ne nécessitant pas l'interaction avec les utilisateurs. La transparence est la base de l'informatique pervasive qui est à son tour un facteur essentiel dans l'Internet des objets.

- **Le nombre croissant d'objets connectés :** il est prévu que le nombre d'objets intelligents qui vont peupler l'Internet du futur franchira les millions, voir les milliards. Avec cela, l'adoption de nouveaux mécanismes qui supportent efficacement l'évolutivité continue dans le nombre d'objets connectés, est vivement recommandée.

- **La mobilité :** un nombre immense d'objets connectés à Internet en tant que partie de l'Internet des objets, seront le plus souvent mobiles. De ce fait, des solutions flexibles de gestion de la mobilité doivent être mises en place pour permettre à tels objets d'accomplir leurs missions efficacement indépendamment de la fréquence et la vitesse de la mobilité.

- **La qualité de service des communications :** suivant que l'application est critique ou non, les communications inter objets connectés dan l'IoT et entre ces derniers et les hôtes ordinaires de l'internet, peuvent exiger ou non un minimum de qualité de service en termes de délais, débits, fiabilité, etc.

11. COMMERCIALISATION ET PROJETS DE RECHERCHE

De nombreuses entreprises trouvent du marché de l'Internet des objets un champ fertile pour y investir. Intel, IBM et Google sont les trois entreprises principales dans le domaine. Chacune d'entre elles se contente de s'adapter rapidement à l'évolution de l'IoT en développant des solutions innovantes, basées sur les facilités *cloud*, pour la connectivité des objets à Internet, tout en assurant une bonne sécurité dans les différents niveaux de la connectivité allant des objets connectés jusqu'au *cloud*.

Sur la voie de la réalisation de la vision de l'internet des objets, des groupes de recherche réunissant des chercheurs académiques ou institutionnels se sont créés. Ce dans le but de développer des produits et des solutions avancées qui répondent aux besoins de l'IoT en tant que projet global qui vient de devenir une réalité, dont les avantages et les rendements seraient tout comme attendu. Dans ce contexte, des projets prometteurs se sont déjà lancés. ERCIT (*European*

Research Cluster on the Internet of Things) [35] représente un large éventail de projets de recherches concernant l'application de l'Internet des objets avec des dimensions européennes. Garantir la collaboration et la communication entre ces projets est un prérequis essentiel pour une industrie compétitive et un déploiement sécurisé et sûr de l'IoT en Europe.

Butler [36] est un projet européen, son but est le développement des applications sécurisées et intelligentes basées sur des systèmes d'information omniprésents et contextuels. Bulter s'intéresse aux scénarios du genre villes intelligentes, maisons intelligentes, applications de santé assistées par l'informatique ubiquitaire et des applications commerciales intelligentes. En ce qui concerne les exigences de sécurité, le projet vise à permettre aux utilisateurs de gérer leurs profils distribués, ce qui implique le contrôle de duplication de données et des identités utilisées par les applications distribuées. L'objectif final étant de mettre en œuvre un système capable d'intégrer des données dynamiques de l'utilisateur (par exemple, la localisation, le comportement) dans les protocoles de sécurité.

Le projet Hydra [37], cofinancé par la commission européenne, sert à développer une couche middleware pour la connexion des réseaux d'objets intelligents à Internet, en se basant sur une architecture orientée services (SOA : *Service Oriented Architecture*). Ce projet envisage les questions de sécurité distribuée et de confiance sociale. Une telle middleware permet aux développeurs d'incorporer des dispositifs matériellement hétérogènes dans leurs applications en offrant des interfaces de service Web facile à utiliser pour contrôler tout type de périphérique sans se soucier aux différentes technologies de transmission adoptées dans le réseau, telles que Bluetooth, ZigBee et Wifi. Hydra incorpore des mécanismes pour la découverte des dispositifs et des services, une architecture orientée modèle sémantique et même les communications P2P (*Peer to Peer*). Un projet de recherche prometteur dénommé NITRD (*Networking and Information Technology Research and Development*) [38], commencé en 2012. Ce projet regroupe une dizaine d'agences fédérales, telles que NASA (*National Aeronautics and Space Administration*) et DARPA (*Defense Advanced Research Projects Agency*). L'objectif est de développer des infrastructures intelligentes pour la concrétisation efficace des différents scénarios applicatifs de l'IoT.

12. Conclusion

L'Internet des objets en tant qu'une évolution de l'Internet actuel permet une amélioration considérable de notre mode de vie et la façon dont les objets intelligents dans notre entourage interagissent entre eux et avec leurs utilisateurs de telle sorte que nos activités, nos biens, notre état de santé, nos dépenses,…puissent être contrôlés efficacement et d'une manière ubiquitaire. Dans ce chapitre, nous avons discuté principalement les technologies de base ainsi que les applications en vedette de l'IoT. Nous avons aussi mis en évidence les contraintes liées au déploiement de l'IoT et qui devraient être soigneusement traitées pour atteindre les objectifs prédéfinis et parvenir à optimiser les rendements.

CHAPITRE 4:

L'intégration des réseaux de capteurs sans fil à l'Internet des objets

1. Introduction

Au cours des dernières décennies, les réseaux de capteurs sans fil ont connu un très grand succès dans différents domaines. Dans leurs applications classiques, où les capteurs étaient totalement isolés de l'Internet, la connexion Internet était utilisée juste comme un support de transmission des rapports de captage vers le gestionnaire de tâches. Dans le contexte de l'Internet des objets, les réseaux de capteurs sont plutôt intégrés à Internet en tant qu'une partie prépondérante de l'Internet des objets. Par conséquent, les capteurs deviennent des hôtes (particuliers) de l'Internet pouvant interagir directement ou indirectement (tout dépend de la stratégie de l'intégration des RCSFs à Internet) avec n'importe quels autres hôtes sur Internet. Ainsi, les services fournis par les RCSFs vont être considérés comme des services web, où les capteurs jouent le rôle des clients ou serveurs web et les services demandés/rendus sont du genre inhabituel : température, pression, etc.

L'incorporation des RCSFs dans l'Internet est assez bénéfique. D'une part, ça permet à l'internet de s'étaler aux objets physiques (autres que les ordinateurs et les téléphones mobiles) intégrant des capteurs connectés à Internet. De l'autre part, l'accès ubiquitaire aux informations et services fournis par tels capteurs crée de nouvelles perspectives avantageuses aidant à maximiser le confort, optimiser les rendements et minimiser le gaspillage et les dépenses.

Dans ce chapitre nous allons nous intéresser à l'aspect technique de l'intégration des RCSFs à l'Internet des objets. Nous allons étudier en détail les techniques et les modèles proposés pour réaliser telle intégration.

2. Les prérequis

Avant de procéder à l'étude des différents modèles d'intégration des RCSFs à l'internet, il est tout d'abord nécessaire de souligner que lors de la conception d'un modèle qui va se server de l'intégration des réseaux de capteurs à l'Internet, certains facteurs doivent être sérieusement considérés. Il est donc important de définir une stratégie d'intégration qui répond efficacement aux exigences suivantes :

- **La considération des contraintes des RCSF :** même si certaines plateformes capteurs deviennent de moins en moins contraignantes, les ressources de calculs, d'énergie et de stockage détenues par un tel capteur demeurent quand même très limitées, comparées à celles des hôtes ordinaires de l'Internet (par exemple les ordinateurs classiques). Par conséquent, toute démarche d'intégration des RCSFs à l'IoT doit prendre en considération ka nature contraignante des capteurs dans un RCSF.

- **La flexibilité :** l'approche d'intégration devrait être souple donnant ainsi l'impression que le réseau de capteurs est une partie intégrante de l'internet. Les communications des capteurs avec le reste des hôtes doivent se dérouler en toute flexibilité sans que l'une des deux parties s'aperçoive que l'autre lui est différente.

- **La sécurité :** le modèle d'intégration doit assurer un bon niveau de sécurité des capteurs, des réseaux de capteurs connectés à Internet pendant les communications depuis et vers le RCSF. L'approche d'intégration doit également s'ouvrir à des mécanismes de protection de la vie privée des utilisateurs de tels capteurs. Faute de quoi, le modèle perd sa crédibilité.

- **La tolérance aux pannes :** la tolérance aux pannes est une caractéristique très importante qui implique que le disfonctionnement d'un ou plusieurs entités dans le réseau d'affecte pas l'accessibilité et la serviabilité du reste des nœuds dans le réseau de capteurs.

- **La qualité de services :** les données fournies par les capteurs sont généralement assez critiques. Pour cette raison, on exige dans les applications de l'Internet des objets, des communications à faibles délais. A ce niveau, le modèle d'intégration adopté joue un rôle important dans la qualité des communications avec les capteurs connectés, spécialement en termes de fraîcheur de données.

- **Le coût :** la stratégie d'intégration devrait présenter un coût modeste pour l'investissement et/ou la mise au point.

3. Les modèles d'intégration existants

On distingue deux grandes familles de modèles d'intégration des réseaux de capteurs à l'Internet des objets. La première et la plus intuitive est basée sur un proxy qui se serve de la connexion de tout le réseau de capteurs à Internet. Quant à elle, la deuxième catégorie des modèles d'intégration repose sur l'adoption des standards de communication sur Internet (le modèle TCP/IP). Dans cette section, on présente d'une façon assez détaillée les deux approches.

3.1. Modèle d'intégration basé proxy

Le principe de ce modèle est simple. Le réseau de capteurs reste isolé de l'Internet et la station de base joue le rôle d'un proxy qui est la seule entité connectée à Internet dans le RCSF. En effet, le proxy représente une interface entre l'Internet et les nœuds capteurs qui lui sont associés. Aucune communication directe de bout-en-bout n'est permise entre les hôtes externes de l'Internet et les nœuds capteurs. Ces derniers (les nœuds capteurs) communiquent entre eux et avec le proxy en utilisant n'importe quelle technologie de transmission (Wi-Fi, ZigBee, Bluetooth, ou autre) conjointement avec les protocoles de communications qui étaient initialement proposés pour les RCSFs isolés de l'Internet, et qui ne portent aucune considération au scénario de l'Internet des objets (exemple : le protocole LEACH). Cependant, les services du réseau de capteurs peuvent être fournis à l'extérieur (Internet), indirectement, au niveau applicatif à travers des interfaces web implémentées sur le proxy. De plus, et à l'arrivée d'une requête depuis un client externe de l'Internet, le proxy obtient les données de captage à partir de(s) nœud(s) capteur(s) concerné(s) suivant l'une des deux techniques suivantes :

- **La méthode réactive :** ou à la demande, quand le proxy reçoit une requête, il demande au capteur d'intérêt de lui remettre sa lecture récente concernant une grandeur bien déterminée, puis il retourne la réponse en un message bien adapté au client externe. Si plusieurs capteurs sont concernés par la requête, le proxy agrège d'abord les valeurs obtenues avant de transmettre la réponse. Notons que dans ce cas, il est possible au proxy de mettre en cache les réponses des capteurs afin de réduire le temps de réponse avec les prochaines requêtes, car dans tel cas, il répondra à partir des données stockées localement sans avoir à consulter les capteurs à nouveau. .

- **La méthode proactive** : les nœuds capteurs envoient leurs données au proxy dès qu'il y ait un changement dans les valeurs, même si aucune requête n'est reçue par le proxy. Cette alternative est suffisamment bonne si le taux de requêtes externes est important. Si par contre les requêtes parviennent avec une très faible fréquence, la solution devient inutilement onéreuse (du point de vue surcharge réseau).

Pour plus de performances, le proxy peut disposer d'une couche middleware regroupant tous les protocoles possibles dans les deux côtés : RCSF et Internet, pour assurer les translations protocolaires nécessaires pour l'intégration. De plus, le réseau de capteurs peut être organisé en groupes et à chaque groupe sa propre passerelle (ou proxy) connectant les nœuds capteurs du groupe à Internet. La figure ci-dessous illustre l'approche d'intégration par proxy.

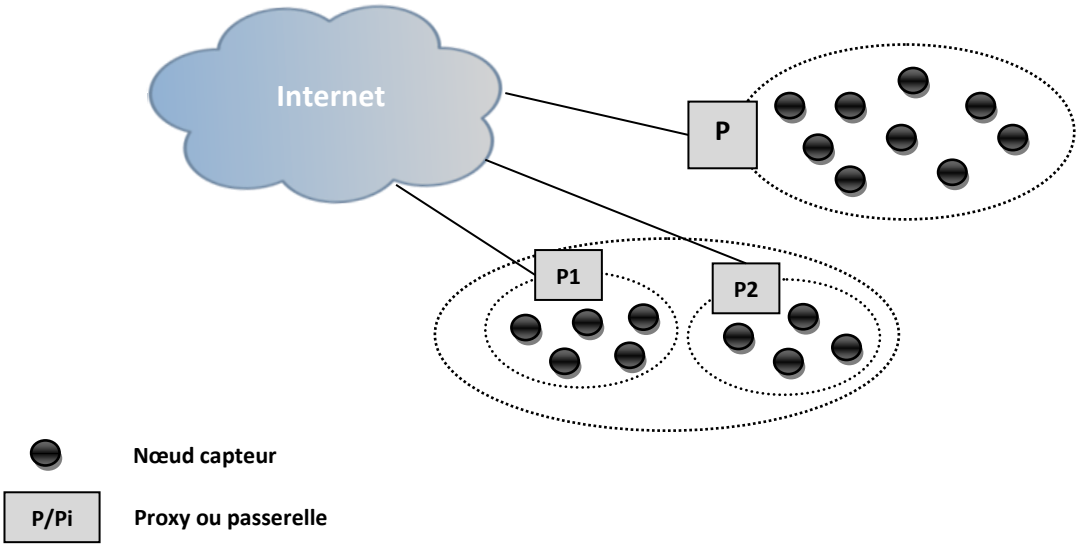


Figure 4.1. Modèle d'intégration des RCSFs à l'Internet basé proxy (avec un seul ou multiples proxy).

Les premières solutions d'intégration inspirées du modèle basé proxy ont été discutées dans [39-40]. Cette solution présente une architecture où les nœuds capteurs supportent les services web et comprennent le protocole HTTP (*HyperText Transfer Protocol*) [41] sans pour autant supporter le standard de communication dans Internet (TCP/IP). Le fait d'utiliser le même protocole applicatif présente un grand avantage pour ce qui concerne l'interopérabilité. Néanmoins, le coût relatif à

l'adoption de cette solution semble élevé car le protocole HTTP a un code volumineux et il est assez lourd pour être supporté par des dispositifs assez contraints comme les capteurs.

3.1.1. Quelques solutions middleware proposées pour l'intégration des RCSFs à l'IoT

Dans la littérature, on trouve une variété de solutions orientées middleware, pour l'Internet des objets. Dans cette section nous citons et présentons brièvement quelques exemples.

A. UBIWARE

UBIWARE [42] est un middleware sémantique intelligent pour l'IoT. La conception de UBIWARE est fondée sur le web sémantique. Son modèle définit une approche à base de règles pour le contrôle d'accès aux objets, celles-ci (les règles) peuvent être stockées et gérées par des parties externes. Cependant, il n'y a pas d'indications claires sur la façon de garantir que les politiques et les règles qui sont stockés sur des serveurs externes sont protégés et maintiennent leur intégrité. Le modèle ne traite pas ni les approches de représentation des nœuds extrêmes dans le réseau ni les techniques de filtrage/agrégation de données de l'IoT au niveau du proxy.

B. GSN

La solution GSN (*Global Sensor Networks*) [43] définit une couche middleware moins compliquée pour l'IoT. La solution n'est pas suffisamment détaillée, elle spécifie juste les règles de contrôle d'accès aux capteurs depuis l'extérieur et les mécanismes de vérification de l'intégrité des programmes du middleware.

C. VIRTUS

Le middleware VIRTUS [44] comporte une collection de mécanismes sécuritaires flexibles pour assurer l'authentification et le control d'accès. De plus, la solution offre la possibilité d'attribuer une instance personnelle du middleware pour les utilisateurs ou les dispositifs. Egalement, des interactions peuvent se produire avec les capteurs du domaine terminal (par exemple une maison connectée) uniquement si l'application ne demande pas un niveau élevé de sécurité. Néanmoins, la sécurité de telle application devient nécessaire lors du partage de données dans l'extérieur. A ce niveau, la couche middleware proposée intervient pour entreprendre des tunnels de sécurité sur Internet.

D. LinkSmart

L'avantage marquant de LinkSmart [45] par rapport aux autres solutions c'est qu'elle est un projet libre (*open source*) accompagné d'une documentation assez riche permettant aux chercheurs de mieux comprendre le modèle et même de le développer de la manière qui répond le mieux aux exigences de leurs applications. L'architecture du middleware LinkSmart est basée sur la spécification

des services web et sur le langage XML sécurisé. Le problème de l'utilisation de la sécurité avec XML réside dans le coût important en temps de calcul et en espace mémoire occupé au niveau des capteurs. Ce qui présente un véritable obstacle devant un large déploiement de la solution.

Ainsi, LinkSmart fournit un service de gestion de la confiance et des identités des objets dans l'IoT, qui fonctionnent en utilisant une infrastructure à clés publiques (PKI : *Public Key Infrastructure*) et les certificats. La solution ne spécifie pas comment ces certificats sont gérés et distribués dans le réseau d'objets intelligents. En plus, la solution ne définit aucun mécanisme de control d'accès aux données de l'IoT, et c'est pourtant une mesure très intéressante dans l'IoT, comme nous allons le voir dans le chapitre suivant.

Le tableau ci-dessous compare les solutions middleware citées.

Table 4.1. Comparaison entre les solutions middleware citées.

	Disponibilité du code	Filtrage	Control d'accès	Sécurité
UBIWARE	–	–	+	–
GSN	–	+	+	+
Virtus	–	+	+	+
LinkSmart	+	+	–	+

Où, les signes - et + signifient supporté et non supporté respectivement.

3.1.2. La solution SensorMAP

Une application populaire du modèle est présentée dans *SensorMap* [46] de Microsoft. Etant fondé sur un modèle à proxy multiples, *SensorMap* fournit une interface cartographique des capteurs déployés dans une zone géographique, comme présenté dans la figure ci-dessous. Il offre des outils pour la sélection et l'interaction avec les capteurs et de visualisation de leurs données qui peuvent être de types variés (température dans le lieu où se trouve le capteur, une image, …). Dans les requêtes des clients, les capteurs sont indexés par leurs types ou positions géographiques. Bien évidemment, l'accès aux capteurs définis dans la cartographie est soumis à des règles d'authentification et de control d'accès.

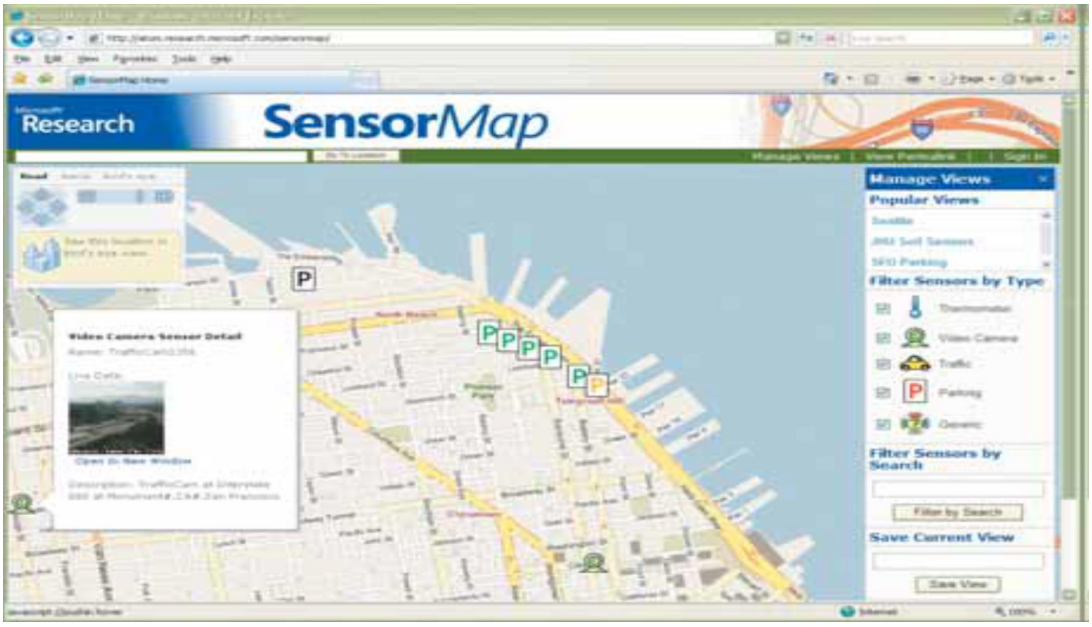


Figure 4.2. *SensorMap* : Interface utilisateur, fournit la facilité de zoom, des cartes routières, des images satellitaires et un mode de vue 3D [46].

L'intégration des réseaux de capteurs à l'IoT par l'intermédiaire d'un proxy est une approche plus ou moins intuitive. Elle permettant aux nœuds capteurs de joindre l'Internet tout en maintenant les protocoles de communication propriétaires et de communiquer indirectement avec des hôtes externes qui adoptent des standards de communication basés IP. Etant isolés du monde de l'Internet, les nœuds capteurs intégrés à l'IoT suivant cette solution se retrouvent implicitement protégés contre les menaces transportées sur la connexion Internet. Cependant, l'intégration par *proxy* est sensible aux pannes du fait que le proxy (ou l'ensemble des proxys) est la seule entité connectée à Internet dans le RCSF. De plus, cette approche est généralement jugée inappropriée compte tenu des aspirations et des exigences applicatives de l'Internet du futur qui demandent à ce que les capteurs soient des hôtes réels de l'Internet et que les solutions d'intégration répondent aux recommandations en termes de standardisation, d'interopérabilité et surtout d'ubiquité de la connexion Internet.

3.2. Modèle d'intégration par adoption du standard TCP/IP

Au lieu d'utiliser des solutions propriétaires développées en ad hoc pour les réseaux de capteurs, empêchant la communication directe entre capteurs et hôtes réguliers et même entre capteurs dans différents RCSFs connectés à Internet. L'alternative consiste plutôt en la standardisation et l'unification de la manière suivant laquelle on intègre les réseaux de capteurs à l'Internet. Dans ce contexte, la tendance actuelle se dirige vers l'extension de l'architecture de communication basée sur le standard TCP/IP aux RCSFs. L'adoption du modèle TCP/IP par les réseaux de capteurs est une approche à la fois audacieuse et prometteuse car d'une part elle va rendre les RCSFs des réseaux tout à fait IP avec tout ce que cela porte comme avantages. De l'autre part, elle invite les capteurs (reconnus par leurs limitations en termes de ressources) à accepter des protocoles et des mécanismes qui avaient été initialement développés pour fonctionner sur des réseaux IP beaucoup

moins contraints. Tels protocoles sont pratiquement onéreux en termes d'espace occupé, des traitements induits et d'énergie consommée. Donc, la projection du standard TCP/IP tel qu'il est sur les réseaux de capteurs est quasiment impossible. Pour cette raison, l'IETF (*Internet Engineering Task Force*) et certaines alliances (comme IPSO) ont déjà pris l'initiative d'adapter les standards de communication fondés sur IP, et même de développer de nouveaux mécanismes qui en sont inspirés, et qui seraient alertés des contraintes des réseaux de capteurs dans la nouvelle génération de l'Internet (l'Internet des objets).

Notons à ce propos que dans cette architecture d'intégration, le réseau de capteurs est entièrement ouvert sur Internet et les nœuds capteurs deviennent des hôtes réels de l'Internet, adressables, et ayant les mêmes concessions qu'un hôte ordinaire. Aussi, la station de base devient un routeur IP dirigeant le trafic depuis et vers les réseaux de capteurs. Ainsi, les communications entre les capteurs et les autres hôtes sur Internet (que ce soit d'autres capteurs connectés de la même façon, ou encore des hôtes ordinaires) se réalisent d'une manière directe et de bout-en-bout. L'interopérabilité est davantage assurée et les recommandations de l'Internet des objets en matière d'ubiquité de l'information, l'accessibilité et l'interactivité des objets seront bien remplies.

Trois groupes de recherches de l'IETF se sont concentrés sur le développement des mécanismes allégés et compatibles avec le standard TCP/IP qui sont destinés aux réseaux de capteurs basés sur la technologie IEEE 802.15.4, dans l'IoT. IL s'agit alors de 6LoWPAN (*IPv6 over Low power Wireless Personal Area Networks*) [47], RoLL (*Routing over Low power and Lossy networks*) [48] et CoRE (*Constrained RESTful Environments*) [49]. 6LoWPAN et RoLL se focalisent sur la couche réseau, tandis que le groupe CoRE s'occupe de la couche application. Comme le montre la figure suivante.

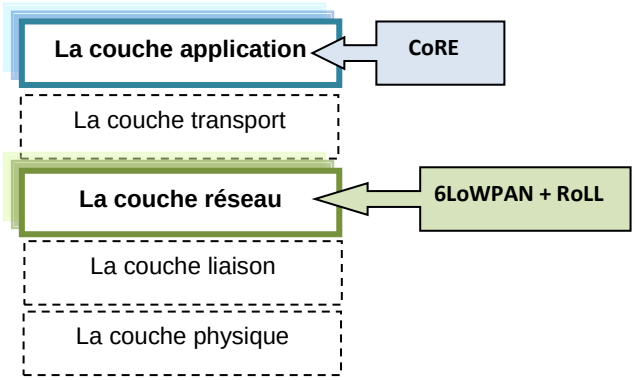


Figure 4.3. Orientation des groupes de recherche de l'IETF chargés de l'intégration basée IP des RCSFs.

3.2.1. Aperçu sur le standard IEEE 802.15.4

La technologie IEEE 802.15.4 [50], déjà introduite dans le chapitre 1, spécifie la couche physique et la sous couche MAC pour les réseaux LR-WPAN (*Low-Rate Wireless Personal Area Networks*) du fait qu'elle (la technologie) réponde mieux à leurs exigences en termes de faible consommation d'énergie, faible débit des dispositifs. Etant reconnu par son faible cout et sa moindre consommation

d'énergie, le standard IEEE 802.15.4 est devenu la technologie de transmission la plus utilisées par les réseaux de capteurs.

La technologie supporte un nombre important de nœuds (jusqu'à 65000 nœuds) pouvant être organisés en différentes topologies (en étoile). De plus, on peut trouver deux types de dispositifs :

- **Les nœuds FFD (*Full Function Device*):** des nœuds puissants destinés à effectuer les tâches les plus compliquées telles que la coordination de la communication et le routage.
- **Les nœuds RDF (*Reduced Function Device*) :** sont des nœuds limités en ressources, et destinés à la réalisation des tâches plus simples comme le captage de données.

Pour la communication, trios bandes de fréquences sont sollicitées : la bande 868 MHz en Europe, 915 Mhz en Amérique et la bande ISM (industriel, scientifique et médical) 2.4 GHZ disponible partout. Aussi, le débit des communications ne dépasse pas 250 Kbits/s avec une taille maximale des trames, que le réseau puisse écouler (MTU : *Maximum Transmission Unit*) fixée à 127 octets.

Au niveau MAC, les dispositifs sont adressés par des adresses IEEE 802.15.4 longues de 64 bits qui sont universelles et uniques par dispositif et assignées par IEEE (EUI-64 : IEEE *Unique Identifier*). Ces adresses peuvent être localement (dans le réseau) remplacés par des adresses courtes codées sur 16 bits, pour une éventuelle minimisation de la surcharge de communication des entêtes longs dans les : trames. Telles adresses sont attribuées localement par une entité centrale qui garde une table contenant toutes les adresses courtes des nœuds du réseau et leurs adresses longues correspondantes. Cette table sera nécessaire pour la translation des adresses courtes de 16 bits en des adresses globales de 64 bits en cas des communications externes.

Ainsi, le standard définit deux politiques pour l'accès au média : avec *beacon* et sans *beacon*. Suivant la première approche (avec *beacon*), le coordinateur du réseau qui est un nœud FFD central, se charge de la supervision et la synchronisation les communications au sein du réseau (par échange de la trame spéciale dite *beacon*), et il tient même à gérer les priorités entre les nœuds. Par ailleurs, dans le second mode de fonctionnement (sans beacon), la concurrence sur l'accès au média est plutôt organisée par l'algorithme CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*) où tous les nœuds sont sur la même ligne de priorité. La sécurité est optionnellement supportée au niveau de la couche MAC en utilisant l'algorithme de chiffrement symétrique par bloc AES-128. La figure ci-dessous illustre la trame 802.15.4 au niveau MAC et physique.

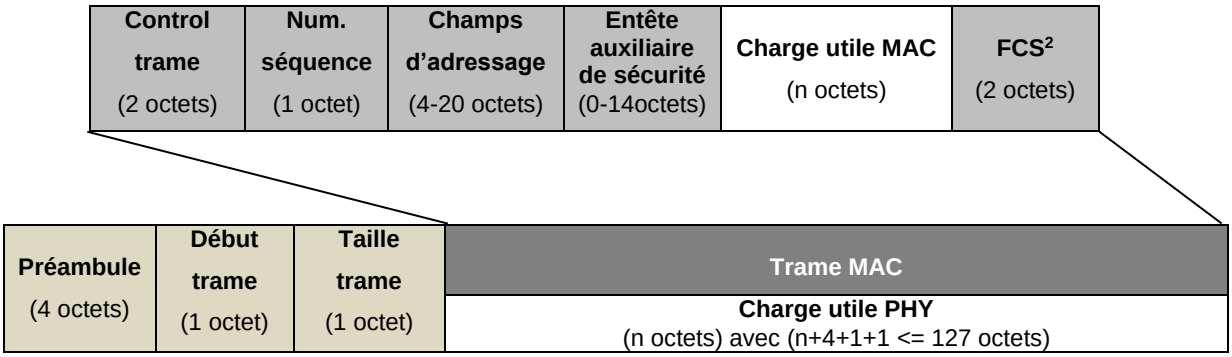


Figure 4.4. Structure de la trame IEEE 802.15.4.

3.2.2. La couche d'adaptation 6LoWPAN

Vu le nombre important des capteurs qui sont déjà connectés à Internet et ceux qui le seront dans les années à venir et qui seront beaucoup plus nombreux, nécessite d'avoir des adresses unique dans l'Internet du futur. Cependant, le plan d'adressage du protocole IPv4 qui est codé uniquement sur 32 bits, et qui est déjà saturé, ne peut pas satisfaire telle exigence. Pour remédier à cette problématique, l'utilisation du protocole IPv6 [51] qui est caractérisé par un espace d'adressage super large (adresses codée sur 128 bits et la possibilité d'adresser 340 sextillions, soit $340x10^{36}$, d'objets) est avérée incontournable. Néanmoins, le protocole IPv6 est assez coûteux en espace mémoire et en énergie nécessaire pour la communication des datagrammes IPv6 de tailles importantes. Le standard 6LoWPAN définit une couche d'adaptation des datagrammes IPv6 pour les réseaux de capteurs connectés. La figure suivante indique l'emplacement de la couche 6LoWPAN [52] dans la pile protocolaire des capteurs et de la station de base qui est appelée 6BR pour *IPv6 Border Router.*

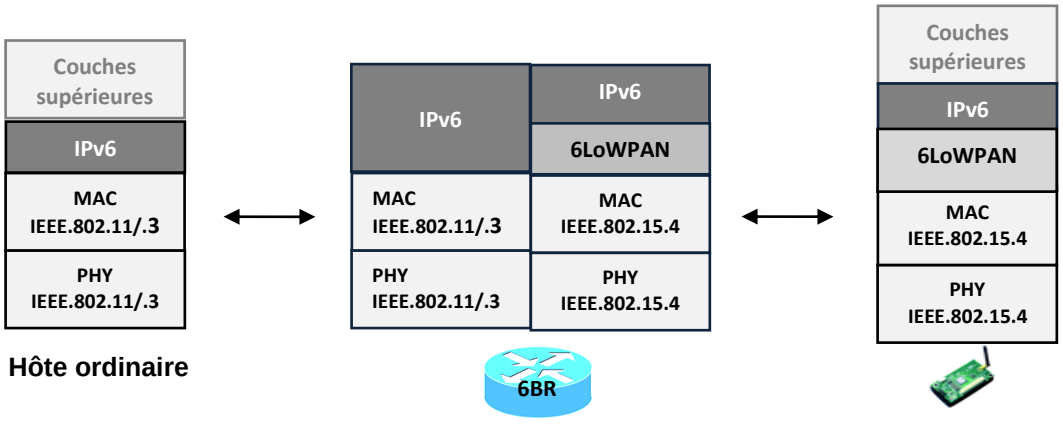


Figure 4.5. La position de la couche 6LoWPAN dans la pile protocolaire du capteur et du routeur de bord.

Pour l'adaptation des datagrammes IPv6 aux contraintes des capteurs, le standard 6LoWPAN présente deux techniques de base : la compression de l'entête IPv6 et la fragmentation des datagrammes IPv6.

[2] *Frame Check Sequence*, utilisé pour le control d'erreurs lors de la communication de la trame.

A. La compression de l'entête IPv6

L'entête IPv6 (présenté dans la figure ci-dessous) de 40 octets, comporte pas mal d'informations ne nécessitant pas d'être communiquées sur les réseaux de capteurs connectés au segment de l'IoT dans l'Internet.

<table>
<tr><td>Version
(4 bits)</td><td>Classe de trafic
(8 bits)</td><td colspan="2">Identificateur de flux
(20 bits)</td></tr>
<tr><td colspan="2">Longueur de données
(16 bits)</td><td>Entête suivant
(8 bits)</td><td>TTL
(8 bits)</td></tr>
<tr><td colspan="4">Adresse source
(128 bits)</td></tr>
<tr><td colspan="4">Adresse destination
(128 bits)</td></tr>
</table>

Figure 4.6. Format de l'entête IPv6.

Dans ce qui suit, on montre comment le standard 6LoWPAN compresse chaque champ dans l'entête IPv6 :

- **Version** : cette information est carrément éliminée par 6LoWPAN en supposant que toutes les communications avec les capteurs vont utiliser le protocole IPv6.
- **Classe de trafic** et **identificateur de flux** : ces deux champs qui sont utiles pour la gestion de la qualité de service sont maintenus dans l'entête si la gestion de la qualité de service est activée. Sinon, les deux champs vont être mis à zéro. Dans ce cas, ils sont à révoquer.
- **Longueur de données :** cette information est redondante car la taille des données peut être déduite à partir de la longueur de la trame au niveau MAC, ou à partir de l'entête de fragmentation du 6LoWPAN où est indiquée la taille totale du datagramme.
- **Entête suivant :** peut être soit gardé, soit révoqué si l'entête suivant est compressé par 6LoWPAN.
- **TTL (Time To Live) ou limite de sauts :** ce champ est compressé uniquement dans le cas des communications locales et directes. Dans le cas échéant, le champ est gardé.
- **L'adresse source et l'adresse destination :** les adresses source et destination IPv6 sont les champs les plus volumineux dans l'entête IPv6 (128 bits). Une adresse IPv6 est composée de deux parties : la première partie (64 bits de poids fort de l'adresse) représente l'identificateur du réseau (adresse réseau) tandis que la deuxième (les 64 bits de poids faible de l'adresse) définit l'identificateur de l'hôte. Pour des raisons de simplicité, l'adresse MAC IEEE 802.15.4 de l'hôte (EUI-64) est elle-même l'identificateur hôte dans son adresse IPv6 comme indiqué dans la figure ci-dessous.

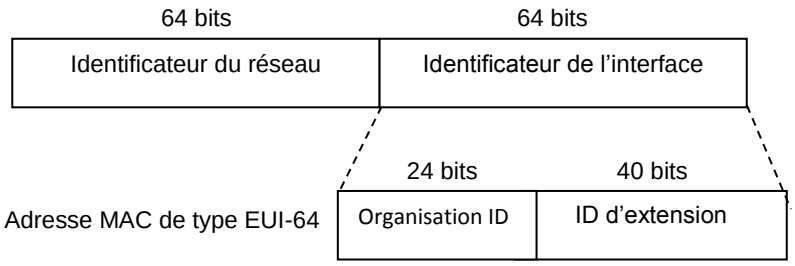


Figure 4.7. La structure générale d'une adresse IPv6.

Si la source et la destination sont dans le même réseau de capteurs et ils sont en plus directement liés, les deux adresses IP de la source et la destination sont compressées et les dispositifs s'adressent seulement au niveau MAC par leurs adresses MAC.

Si la source et la destination sont toujours dans le même réseau mais que l'une soit distante de l'autre, les adresses IPv6 seront réduites à 64 bits uniquement ou à 16 bits si les adresses courtes IEEE 802.15.4 sont envisagées.

Si la source (ou la destination) est en dehors du réseau alors, son adresse est maintenue avec compression de l'adresse destination (ou source) à 64 bits ou 16 bits.

Notons que la compression 6LoWPAN exige que l'entête compressé soit précédé par un octet *dispatch* qui identifie l'état de l'entête IPv6 (par exemple, si l'octet de *dispatch* prend la valeur 01000001alors l'entête qui vient juste après est un entête IPv6 non compressé et s'il vaut 01000010, l'entête suivant est un entête IPv6 compressé) et un octet (ou deux) d'encodage. Dans cet octet, on identifie l'entête compressé, et on indique les champs compressés. Le format général du datagramme 6LoWPAN (datagramme IPv6 compressé) est comme suit :

Octet dispatch	**Octets d'encodage**	**Entête IPv6 compressé**	Charge utile

Figure 4.8. Format général du datagramme 6LoWPAN compressé.

L'entête IPv6 compressé résultant comprendrait une quantité moindre d'information ; seules les informations nécessaires sont communiquées au sein du RCSF. Par conséquent, la taille de l'entête, et des datagrammes en général, deviennent réduites. Comme montré dans la figure ci-dessous, avec la compression 6LoWPAN, la taille l'entête IPv6 peut être réduite jusqu'à 2 octets (de l'encodage) dans le cas des communications directes locales au réseau. La taille peut atteindre 7 octets (2 octets d'encodage, 1 octet du champ TTL, 2 octets de l'adresse source, 2 octets de l'adresse destination) dans le cas d'une communication locale en multi-sauts. Dans le cas où l'une des entités communicantes est située en dehors du réseau de capteurs connecté, la taille de l'entête IPv6 compressé devient 21 octets (2 octets d'encodage, 1 octet du champ TTL, 2 octets de l'adresse source/destination, 16 octets de l'adresse destination/source).

Dispatch (1 octet)	**Octet(s) d'encodage**

(a)

Dispatch (1octet)	**Octet(s) d'encodage**	**TTL** (1 octet)	**Adr source** (2 octets)	**Adr destination** (2 octets)

(b)

Dispatch (1 octet)	**Octet(s) d'encodage**	**TTL** (1 octet)	**Adr source** (2 octets)	**Adr destination** (16 octets)

(c)

Dispatch (1 octet)	**Octet(s) d'encodage**	**TTL** (1 octet)	**Adr source** (16 octets)	**Adr destination** (2 bytes)

(d)

Figure 4.9. L'entête IPv6 compressé à l'aide du standard 6LoWPAN avec (a) communication locale directe, (b) communication locale multi-sauts, (c) la source appartient au réseau de capteurs connecté et la destination est à l'extérieur du réseau, (d) la source est en dehors du RCSFs et la destination y appartient.

Le mécanisme de compression 6LoWPAN ne concerne pas que l'entête du protocole IPv6, mais il peut aussi concerner ses protocoles d'extension (comme nous allons voir dans le chapitre suivant) et les entêtes suivants, comme l'entête du protocole UDP qui peut être compressé de 8 octets à 5 octets. Et comme l'entête UDP n'inclut pas un champ pour identifier l'entête du protocole applicatif qui le suit directement, il a été nécessaire de compresser les entêtes des ces protocoles comme une partie de la charge utile du paquet UDP. Cette variante de compression 6LoWPAN est dénommée 6LoWPAN-GHC [53].

Il est nécessaire de noter que les réseaux de capteurs intégrés à Internet par cette technique sont désormais appelés les réseaux 6LoWPANs.

B. La fragmentation des datagrammes IPv6

Après la compression des datagrammes IPv6, la deuxième fonctionnalité de la couche d'adaptation 6LoWPAN consiste, bel et bien, en la fragmentation de tels datagrammes. Etant donné que le MTU minimal dans les réseaux IPv6 vaut 1280 octets, opposé à uniquement 127 octets du coté réseau de capteurs, la transmission d'un datagramme IPv6 (compressé) dans le réseau de capteurs connecté ne peut se faire sans la division de la trame initiale en plusieurs trames unitaires (fragments) de tailles inférieures ou égales à 127 octets, au niveau du routeur de bordure 6BR (*6LoWPAN Border Router*). Une fois arrivées à leur destination finale (un capteur), celle-ci réassemble les fragments et constitue le datagramme 6LoWPAN initial, comme illustré dans la figure suivante.

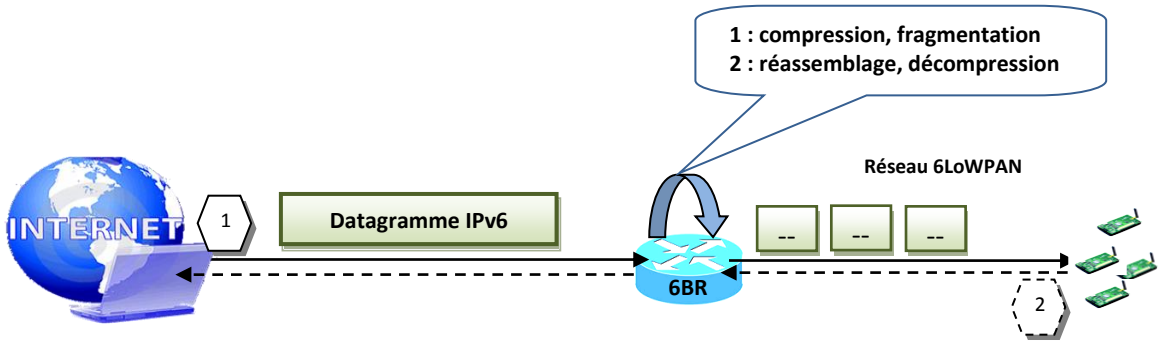


Figure 4.10. Le processus de fragmentation.

Chaque fragment 6LoWPAN doit contenir un entête spécifique au processus de fragmentation et aidant les entités à correctement fragmenter et réassembler les datagrammes 6LoWPAN. Le format de l'entête du premier fragment 6LoWPAN est comme suit :

1 1 0 0 0	Taille du datagramme (11 bits)	Identifiant du datagramme (16 bits)

Figure 4.11. Entête fragmentation du premier fragment 6LoWPAN.

Où les cinq premiers bits identifient l'entête du premier fragment. Le champ taille du datagramme donne la taille du datagramme 6LoWPAN initial avant son fragmentation. L'identification du datagramme est une information qui aide la station réceptrice à repérer les fragments correspondants au même datagramme.

Dans l'entête fragmentation du reste des fragments 6LoWPAN, il est nécessaire d'inclure une information supplémentaire relative à l'emplacement du fragment dans le datagramme initial (champ décalage). Comme indiqué dans la figure ci-dessous.

1 1 1 0 0	Taille du datagramme (11 bits)	Identifiant du datagramme (16 bits)	Décalage (8 bits)

Figure 4.12. Format de l'entête fragmentation pour la suite des fragments 6LoWPAN.

3.2.3. Le routage dans les réseaux 6LoWPANs

Le routage dans les réseaux 6LoWPANs est la fonctionnalité vitale qui assure le bon acheminement des datagrammes 6LoWPAN entre les capteurs appartenant au même réseau ou entre le routeur de bordure 6BR et les nœuds capteurs extrêmes. Deux mécanismes sont définis par l'IETF pour prendre en charge le routage dans ce type de réseaux : le routage maillé au niveau de la couche 6LoWPAN et le routage par le protocole RPL (*Routing Protocol for Low power and lossy networks*) du groupe RoLL.

a. Le routage maillé

Ce mécanisme exploite les informations de la couche MAC, plus précisément les adresses MAC, pour réaliser le routage des datagrammes IPv6 compressés (et fragmentés) au niveau de la couche d'adaptation 6LoWPAN. La communication entre la source et la destination est considérée comme un seul saut IP dont les nœuds intermédiaires (les routeurs) prennent la décision de routage en se basant sur l'analyse de l'adresse MAC de la destination. Si celle-ci ne correspond pas à l'adresse MAC d'un nœud de relai, ce dernier se rend compte que le fragment (et le datagramme) ne lui est pas destiné et consulte donc sa table de routage au niveau liaison pour trouver le nœud prochain. Dans ce cas, les fragments sont acheminés indépendamment les uns des autres. Cela veut dire que les fragments du même datagramme peuvent emprunter différents chemin pour arriver à leur destination finale. Le problème avec cette solution est qu'en cas de perte d'au moins un fragment, la perte ne peut être détectée qu'au niveau de la destination finale et dans tel cas, tous les fragments (y compris le manquant) doivent être retransmis à nouveau pour la récupération. La figure suivante illustre le format de l'entête du mécanisme de routage maillé dans les réseaux 6LoWPAN[3].

1 0	O	F	Nb sauts (4 bits)	Adresse source (16 ou 64 bits)	Adresse destination (16 ou 64 bits)

Figure 4.13. Entête du mécanisme de routage maillé définit dans la couche 6LoWPAN.

Avec deux premiers bits servant à l'identification de l'entête. Et comme le standard IEEE 802.15.4 supporte deux sortes d'adresses (les adresses courtes et les adresses longues). Les indicateurs O (*originator*) et F (*Final*) valent un si les adresses MAC des entités source et destination, respectivement, sont courtes (codées sur 16 bits) et zéro si les adresses sont longues (codées sur 64 bits). La valeur du champ nombre de sauts est limitée à 15 et elle est décrémentée à transit du datagramme. Quand la valeur du champ devient zéro, l'entité qui l'a reçu l'écarte si elle n'est pas sa destination finale.

b. Le protocole RPL

Le protocole RPL [54] est un protocole de routage IPv6 destiné aux réseaux 6LoWPAN dans l'Internet des objets. il forme une topologie dynamique et optimisée avec l'évitement des boucles et la considération des paramètres de qualité de service pour l'acheminement des datagrammes IPv6 depuis et vers les nœuds capteurs.

Chaque nœud intermédiaire se comporte comme un routeur IP, il réassemble d'abord tous les fragments pour reconstruire le datagramme IPv6 initial ensuite, il analyse l'adresse IPv6 de destination pour décider si le paquet va passer à la couche transport ou bien s'il doit être communiqué vers un autre nœud capteurs, jusqu'à ce qu'il arrive à la bonne destination finale.

[3] L'ordre des entêtes dans un datagramme 6LoWPAN de gauche à droite est : entête du routage maillé, entête de fragmentation et finalement l'entête 6LoWPAN.

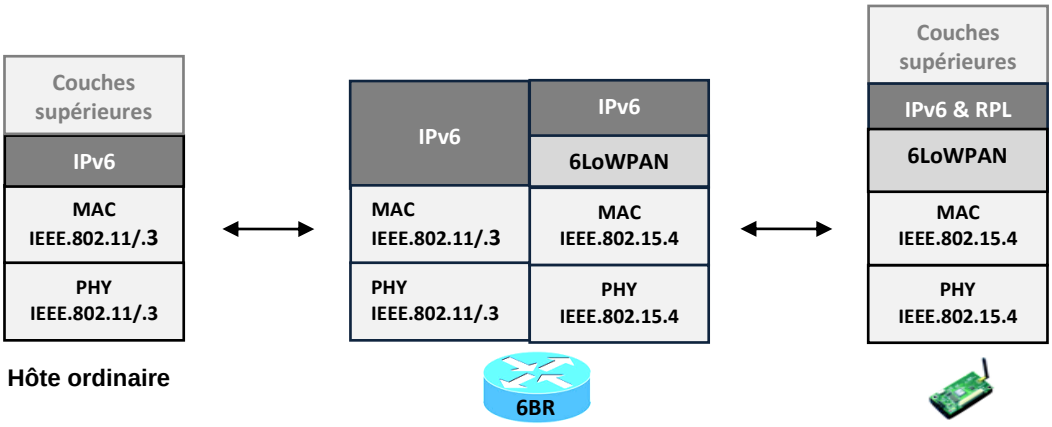


Figure 4.14. Position du protocole RPL dans la pile protocolaire du capteur.

RPL construit un graphe acyclique dit DODAG (*Destination Oriented Directed Acyclic Graph*) qui route les informations vers ou depuis une seule destination appelée racine DODAG. Dans certains cas, le même réseau physique 6LoWPAN devrait être optimisé pour supporter plusieurs applications ayant chacune son propre graphe (communément appelée instance RPL), construite selon une métrique de routage bien déterminée. La métrique peut être le niveau d'énergie résiduelle des nœuds capteurs dans le réseau 6LoWPAN, le nombre de transmissions nécessaire pour atteindre la racine (EXT), le délai moyen des communications, le taux de pertes, etc. Lors de la construction du graphe, les nœuds utilisent la fonction objective qui définit la méthode de calcul de la métrique du routage, et s'échangent quatre types de messages : DIO (*DODAG Information Object*), DIS (*DODAG Information Solicitation*), DAO (*DODAG Destination Advertisement Object*) et DAO-ACK (*DAO Acknowledgement*).

- Le message DIO est diffusé en premier lieu par le 6BR (la racine) pour déclencher le processus de construction du graphe. Les nœuds capteurs voisins de la racine reçoivent le message et décident s'ils peuvent joindre le graphe ou non (la décision dépend de plusieurs facteurs tels que la fonction objective et le coût du chemin annoncé). Une fois le nœud a rejoint le graphe, il a automatiquement une route vers la racine. Si le nœud est configuré pour être un routeur, il diffuse à son tour, sa connaissance locale sur le graphe (ses liaisons) à ses voisins.
- Le message DIS est utilisé par les nœuds pour demander des informations concernant le graphe à partir des nœuds voisins qui vont répondre en envoyant un message DIO.
- Les messages de type DAO sont utiles pour annoncer la présence du nœud à son parent dans le graphe. Ce dernier met à jour sa table de routage en y rajoutant une entrée correspondante au nœud fils. Le processus se reproduit récursivement et d'une manière ascendante jusqu'à ce que l'on arrive à la racine (6BR).
- Le message DAO-ACK est envoyé par le nœud parent au nœud fils, en réponse à son message DAO (pour en accuser réception).

De plus, RPL supporte deux techniques de routage : le routage par source de données (sans état) et le routage avec décision local du chemin (avec état). Dans le routage par source de données, la totalité du chemin à emprunter est mentionnée dans le paquet, et les nœuds intermédiaires le passe jusqu'à sa destination finale en se basant sur ces informations. Par revanche, dans la deuxième technique, le paquet porte uniquement l'adresse de la destination finale, et le routage est décidé au niveau de chaque nœud intermédiaire suivant les informations contenues dans une table de routage locale. La table de routage comporte des informations pour la distinction des flux ascendant (orientés vers la racine 6BR) des flux descendants (orientés vers les nœuds capteurs). Le 6BR maintient donc, une liste complète de tous les nœuds de l'arborescence. Notons que pour l'évitement des boucles de routage, chaque nœud doit calculer sa position (ou rang) dans la hiérarchie par rapport à la racine. La valeur du rang devient importante plus la distance entre la racine et le nœud est importante. Des considérations relatives à la métrique du routage peuvent affecter la procédure de calcul de la position.

Ainsi, les communications locales entre les nœuds capteurs ayant un parent en commun, ne nécessitent pas de passer par la racine. Cependant, les nœuds n'ayant pas une racine secondaire commune doivent passer par la racine principale (le routeur de bord 6BR), comme indiquer dans la figure ci-dessous, où les flèches rouges représentent les communications les communications locales, et les numéros 1 jusqu'à 4 représentent les rangs des nœuds dans le graphe.

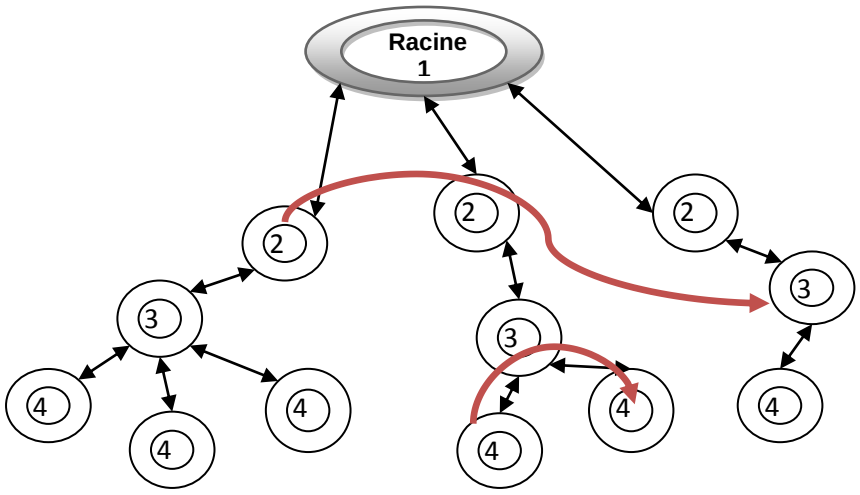


Figure 4.15. Exemple d'un graphe DODAG construit par le protocole RPL.

3.2.4. Les protocoles applicatifs pour le web d'objets intelligents

Dans cette section, on étudie les principaux protocoles applicatifs qui sont destinés à offrir les services web aux capteurs connectés à l'IoT.

A. Le protocole CoAP

Le groupe de travail CoRE de l'IETF a défini le protocole CoAP (*Constrained Application Protocol*) [55] de transfert web, destiné à fonctionner sur des équipements très limités en ressources (de

traitement, de stockage mémoire et d'énergie) et appartenant à des réseaux sans fil peu fiable, tout comme les réseaux de capteurs, dans l'Internet des objets. Pour cela, CoAP devrait être un protocole web léger et devrait surtout supporter les communications de type M2M (Machine à Machine) entre capteurs connectés dans l'IoT.

Ce protocole est spécialement conçu pour consommer un minimum de ressources et pour cela, il définit des messages de taille assez réduite, avec une moindre surcharge liée aux informations de control dans l'entête fixe codé sur 4 octets seulement. L'entête est suivi par un champ de longer variable entre 0 et 8 octets appelé jeton[4] ensuite vient une séquence de zéro ou plusieurs options CoAP, et optionnellement un champ de charge utile.

Entête CoAP	**Jeton**	**Options**	**Charge utile**
(4 octets)	(0-8 octets)	(Variable)	Variable

Figure 4.16. Format du message CoAP.

Notons que la taille réduite des messages CoAP est tant bénéfique pour l'atténuation de la surcharge dans le réseau que pour l'affaiblissement de la fréquence de fragmentation des messages au niveau des couches inférieures. D'autre part, le protocole CoAP opère sur le protocole de transport UDP qui fournit un service de transport assez simple et dont l'adoption pour les réseaux de capteurs est bien approuvée. Par conséquent, l'échange des messages CoAP se déroule d'une façon asynchrone, contrairement au cas synchrone qui caractérise les communications du protocole HTTP (*Hyper Text Transfer Protocol*), qui est plutôt fondé sur TCP.

L'intégrabilité avec les standards du web existants a été aussi tenue en compte comme paramètre essentiel, lors de la conception du protocole CoAP. Ce dernier est très semblable au fameux protocole de transfert web, le protocole HTTP; CoAP et HTTP sont tout deux basés sur le modèle REST (*Representational State Transfer*) qui fournit une architecture d'interaction basée requête/réponse. Bien que CoAP soit très semblable à HTTP, les deux protocoles ne sont pas tout à fait compatibles et la communication entre un hôte HTTP et un hôte CoAP nécessite l'interposition d'un proxy assurant la translation entre HTTP/TCP et CoAP/UDP.

[4] Le jeton est une valeur permettant de lié le message de la requête au message de la réponse correspondant.

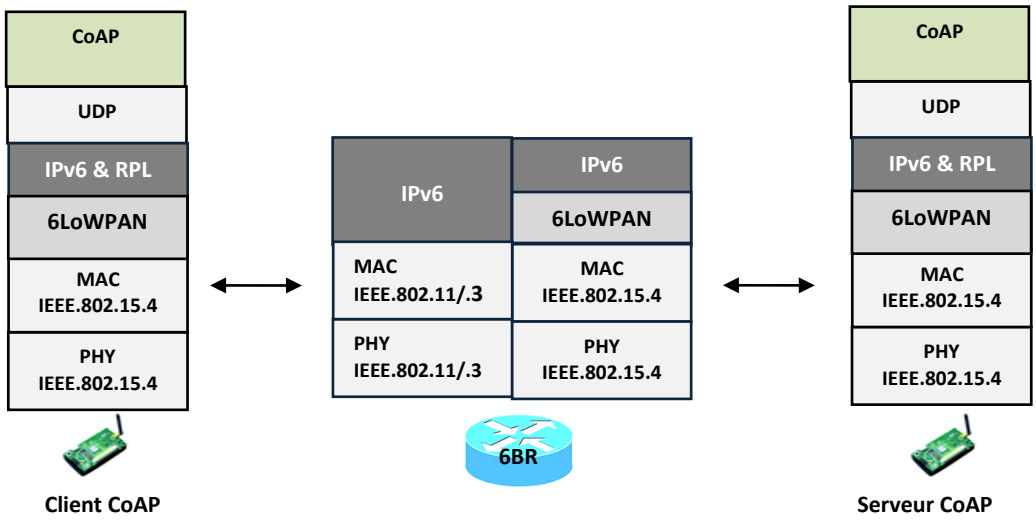


Figure 4.17. Les piles protocolaires des dispositifs impliqués dans la communication objet-à-objet entre un client CoAP et un serveur CoAP.

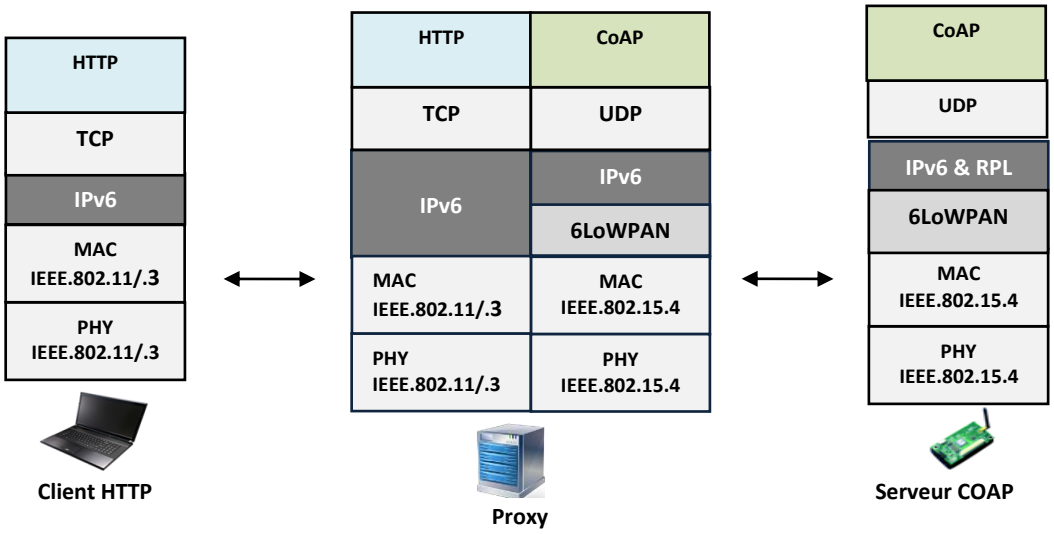


Figure 4.18. Les piles protocolaires des dispositifs impliqués dans la communication humain-à-objet entre un client HTTP et un serveur CoAP.

Comme CoAP se focalise sur le modèle REST, le serveur CoAP (un capteur) rend les ressources accessibles aux clients à travers des URI (*Uniform Resource Identifier*) identifiant les ressources, et des méthodes bien déterminées que les clients spécifient dans leurs requêtes. On distingue donc quatre types de méthodes :

- La méthode *GET :* Le client CoAP utilise cette méthode pour demander une ressource. elle a le code 0.01.
- La méthode *PUT* : Ayant le code 0.02, la méthode *PUT* est utilisée pour mettre à jour ou créer une ressource dans le serveur CoAP.
- La méthode *POST :* Cette méthode requière que la représentation indiquée dans le message soit appliquée sur la ressource. Son code est 0.03.

- La méthode *DELETE* : La ressource identifiée dans la requête contenant la méthode *DELETE* doit être éliminée au niveau du serveur. le code correspondant à la méthode est 0.04.

Le serveur répond à une requête du client par un message de réponse contenant la ressource demandée, si elle est disponible, avec un code de réponse 2.05 (contenu). Sinon, le serveur répondrait par un message d'erreur exprimant le problème rencontré, le code de la réponse pourrait par exemple être 4.04 (non trouvé) ou 5.03 (service indisponible).

La découverte de ressources est un besoin clé dans une architecture web et notamment pour les applications M2M. Les serveurs CoAP doivent être capables de fournir une URI pour que leurs services soient découverts par les clients CoAP. Par exemple un client envoie la requête : `GET/.well-known/core`. Le serveur CoAP retourne un message dans lequel il exprime les ressources qu'il a. la réponse pourrait être : `2.05 Content </light> </temperature>`.

Et comme déjà mentionné, les messages CoAP (requêtes ou réponses) peuvent inclure, dans l'entête, des options qui ont des formats et des valeurs spécifiques et qui donnent les informations nécessaires concernant la façon suivant laquelle le message devrait être traité par la station qui le recevra. Par exemple: l'option *Max-Age* prend la valeur 60 par défaut et révèle que le délai de fraicheur du message de la mémoire cache du proxy CoAP (la station de base) est exactement 60 secondes.

En effet, CoAP peut gérer optionnellement la fiabilité des communications (au niveau de la couche application) entre les clients et les serveurs. Ainsi, un client CoAP peut envoyer une requête dans un message *CON* (confirmable) ou un message *NON* (non confirmable). A la réception de la requête repérée par *CON*, et si la ressource est disponible, le serveur envoi immédiatement la réponse dans un message *ACK* (acquittement). Si le client ayant envoyé un message *CON* au serveur ne reçoit aucun message d'acquittement de la part de celui-ci, il retransmet la requête. Et dans le cas où le récepteur d'un message *CON* ne peut plus le traiter ni même d'y répondre par un message d'erreur, il retourne un message de type *RST* (pour la réinitialisation).

Et concernant les ports utilisés par le protocole, les ports UDP 61616 et 61631 sont réservés respectivement pour le serveur et le client si le standard 6LoWPAN est employé conjointement avec CoAP dans le réseau de capteurs connectés. Sinon, si 6LoWPAN n'est pas employé, le numéro de port est utilisé 5683 pour le serveur.

La figure suivante illustre un exemple de communication entre un client et un serveur CoAP dans le cadre des interactions objet-à-objet (M2M) et entre un client HTTP et un serveur CoAP (une communication humain-à-objet). Dans les deux cas, le client demande la lecture récente de la température à partir d'un serveur CoAP. L'exemple montre comment le processus de mise en cache de la ressource peut se produire.

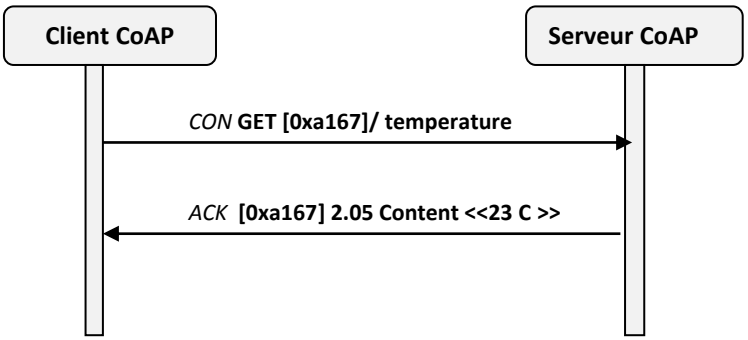


Figure 4.19. Exemple de communication entre client-serveur CoAP

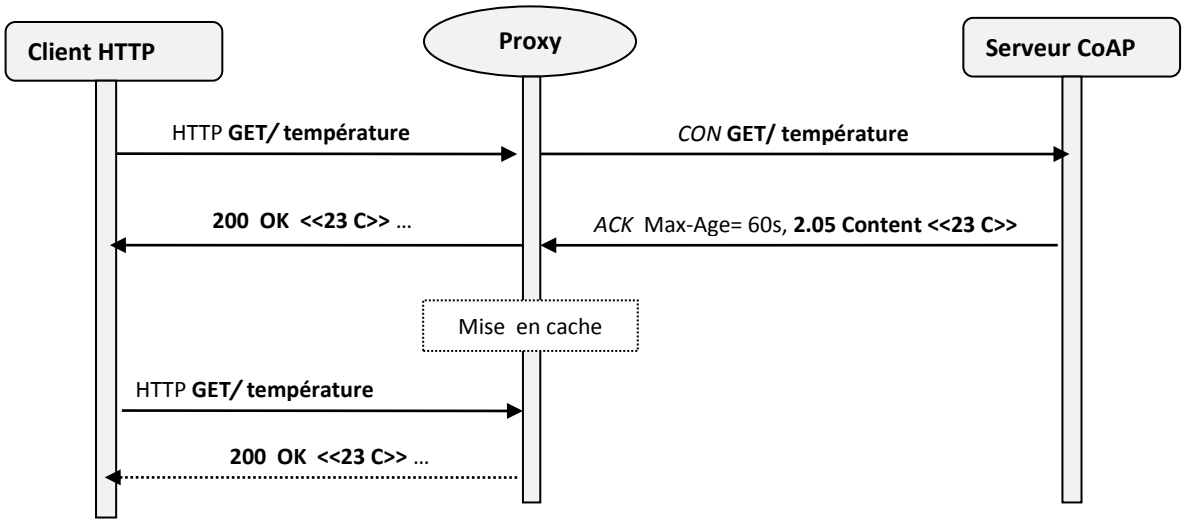


Figure 4.20. Exemple de communication entre un client HTTP et un serveur CoAP.

Un avantage très intéressant du protocole CoAP est qu'il supporte les communications multicast. Dans certaines applications, il est nécessaire d'interroger plusieurs capteurs en même temps. On peut imaginer un exemple d'une application de maison intelligente où l'on ordonne les capteurs dans une chambre pour allumer les lumières en même temps.

B. Le protocole MQTT

MQTT (*Message Queuing Telemetry Transport*) [56] est un autre exemple de protocole applicatif de messagerie sur le web, dont son efficacité est de plus en plus approuvée dans de célèbres applications comme la messagerie sur le réseau social *Facebook*. Maintenant, son application pour les applications fondées sur les communications M2M dans l'IoT est largement investiguée. Le principe du fonctionnement du protocole MQTT est généralement concentré autour du modèle *publish/subscribe*. Le protocole fonctionne au-dessus du protocole TCP, avec un mécanisme de communication suffisamment simple pour mieux répondre aux fortes contraintes des réseaux de capteurs connectés à Internet.

Du point de vue architectural, les nœuds capteurs sont des publieurs qui se connectent tous à une entité centrale dite *broker*. Chaque message est publié au niveau du *broker* dans une rubrique qui

convient au type de la donnée contenue dans le message. Les abonnés peuvent souscrire à plusieurs rubriques. A chaque fois qu'un nouveau message est publié dans l'une des rubriques d'intérêts, il sera immédiatement diffusé aux abonnés intéressés. Dans la figure ci-dessous, on illustre un exemple de l'architecture de communication M2M dans MQTT. Le modèle est composé de deux abonnés qui élaborent des connexions TCP avec le *broker*. L'abonné numéro 1 adhère à la rubrique température ensuite, l'abonné 2 publie un nouveau message dans cette rubrique. Le *broker* se met, immédiatement, à transmettre une copie du message au client 1.

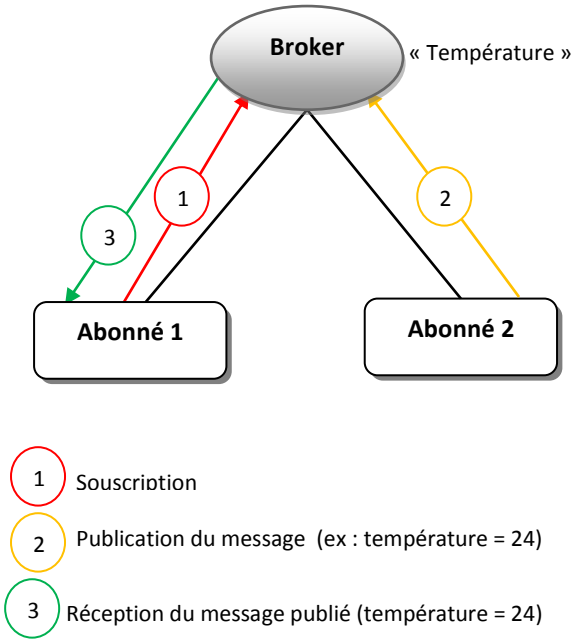


Figure 4.21. Exemple de communication machine-à-machine suivant le protocole MQTT.

L'efficacité du protocole MQTT a été approuvée pour certaines applications comme : les applications médicales et industrielles. Il s'est avéré même qu'il représente une bonne solution pour les applications mobiles dans des environnements contraints car il présente une faible empreinte mémoire, une faible consommation d'énergie avec une meilleure distribution de l'information aux destinataires.

4. Comparaison entre les modèles d'intégration

Dans cette section, on compare les modèles d'intégration des réseaux de capteurs à l'Internet., en observant plusieurs critères.

Table 4.2. Comparaison générale entre les modèles d'intégration des RCSFs à l'IoT.

Critère de comparaison	Intégration basée Proxy	Intégration par adoption des standards IP
Considération des contraintes des RCSFs.	Très bonne	Tout dépend de la stratégie d'adaptation des standards IP pour les RCSFs
Flexibilité et Interopérabilité	Mauvaise	Bonne

Modèle de communication	Solutions propriétaires développées en ad-hoc.	Modèle TCP/IP adapté pour les RCSFs
Qualité de l'intégration	Mauvaise (nœuds capteurs connectés indirectement).	Bonne (intégration unifiée avec accessibilité des capteurs connectés depuis l'extérieur).
Sécurité	RCSFs Mieux sécurisés	RCSFs ouverts à diverses menaces.
Tolérance aux pannes	Sensibilité aux pannes de la station de base.	Sensibilité aux pannes des nœuds capteurs ciblés par les attaques externes.
Délais des communications	Elevés sans la mise en cache	Courts
Fraîcheur de données	Peut être mauvaise en cas de mise en cache	Bonne
Coût de l'intégration	Réduit	Relativement élevé

D'après le tableau, nous constatons que l'avantage marquant de l'approche d'intégration par projection du standard TCP/IP aux réseaux de capteurs, à laquelle nous nous intéressons dans cette thèse, réside dans sa flexibilité et bonne qualité qui convient mieux aux perspectives des applications de l'Internet du futur. Cependant, les coûts relatifs à l'adoption des standards IP (adaptés) et les enjeux liés à la sécurité demeurent les principaux facteurs limitatifs sur lesquels les travaux de recherches doivent se concentrer le plus.

5. Conclusion

Nous avons étudié, à travers le présent chapitre, les approches proposées dans la littérature pour l'incorporation des réseaux de capteurs à l'Internet des objets. En fait, nous avons présenté deux stratégies principales : celle basée proxy et celle qui repose sur l'extension soigneuse des standards basés IP. Malgré le fait que la deuxième approche soit la plus prometteuse sur plusieurs plans, le choix entre les deux modèles d'intégration revient à réaliser un compromis entre les exigences de l'application et les coûts induits. Si par exemple l'application exige que le réseau de capteurs soit sécurisé dans une grande mesure, la restriction de l'ouverture du RCSF à Internet devient une obligation donc, l'utilisation du modèle basé proxy est nécessaire dans telle situation. Et si par contre, l'application requière que le RCSF soit suffisamment ouvert à Internet avec un niveau modéré de sécurité, l'adoption du modèle d'intégration par TCP/IP saurait le bon choix.

Une nouvelle approche, alternative, pour l'incorporation des RCSFs dans l'internet des objets est appelée *SIG-Weightless* [57]. L'idée de base consiste à de réutiliser l'infrastructure GSM pour les capteurs assurant ainsi une connectivité longue portée, à faible consommation d'énergie. La solution est en phase expérimentale et son efficacité n'a pas encore été approuvée.

CHAPITRE 5:

La sécurité de l'intégration des réseaux de capteurs sans fil à l'Internet des objets : état de l'art

1. Introduction

La sécurité sur Internet, qui représente un milieu non fiable et non sécurisé, était et continue à être un grand problème menaçant les gens, leurs données, leurs équipements connectés et toute autre fonctionnalité critique se trouvant sur Internet. De nombreuses attaques de types variés ayant une ampleur de gravité croissante, ne cessent d'apparaitre et maintenant avec l'avènement de l'Internet des objets, la situation risque de s'empirer encore. Cela parce que des capteurs en nombres explosifs, incorporés dans notre environnement et qui révèlent nos habitudes et des informations étroitement liées à notre vie privée, vont être connectés à Internet. Donc, les risques potentiels relatifs à la sécurité vont être amplifiés principalement à travers l'ubiquité de la connexion Internet. Encore plus, l'alliance entre les deux mondes physique et virtuel créée par l'IoT présente un nouveau champ, fertile, pour l'émergence de nouvelles classes d'attaques beaucoup plus dangereuses que les attaques classiques bien connues dans l'Internet d'aujourd'hui.

La plus grande inquiétude est donc que le basculement vers l'Internet des objets exposera ses utilisateurs, ainsi que les réseaux et les dispositifs qui y sont impliqués à de véritables problèmes de sécurité. Cette imagination peut devenir une réalité, à moins que des contremesures sécuritaires robustes soient mises en place.

Comme nous avons déjà vu dans les chapitres précédents, les réseaux de capteurs jouent un rôle très important dans la concrétisation de l'Internet des objets. Ce type de réseaux est souvent déployé pour accomplir des missions critiques ou pour surveiller nos activités quotidiennes. Donc leur sécurité dans le contexte de l'IoT est plus que nécessaire.

Dans ce chapitre nous allons d'abord étudier les vulnérabilités de l'intégration des RCSFs à l'IoT, ensuite nous présentons les attaques ciblant les RCSFs connectés à Internet. Après, nous présentons et analysons les solutions proposées dans la littérature pour la sécurité de l'intégration.

2. Les vulnérabilités de l'intégration

Les réseaux de capteurs sans fil sont intrinsèquement vulnérables à différents types d'attaques (comme montré dans le chapitre 2). La communication sans fil, les limitations de ressources et la prédisposition aux actes de compromission, étaient les principales vulnérabilités dans les RCSFs classiques (isolés de l'Internet). Dans le contexte de l'Internet des objets, l'ensemble des vulnérabilités s'élargit aux points suivants [58] :

2.1. L'hétérogénéité des communications

La communication entre les hôtes ordinaires de l'Internet et les capteurs connectés à l'IoT est caractérisée par de fortes hétérogénéités matérielles et technologiques. Les hôtes réguliers sont plus performants, caractérisés par une abondance énergétique, appartiennent à des réseaux plus ou moins fiables fournissant des débits de communications très élevés. Contrairement aux nœuds

capteurs minuscules, qui sont munis de ressources limitées, et qui font partie des réseaux peu fiables autorisant des communications avec des débits pratiquement faibles. Ces hétérogénéités peuvent être facilement exploitées par des attaquants pour nuire aux nœuds capteurs et/ou au RCSF en entier.

2.2. Les vulnérabilités liées à la fragmentation des paquets

La grande différence entre les tailles maximales des unités de données qui peuvent être communiquées dans un réseau de capteurs et dans l'Internet (le MTU minimal est de 68 octets dans les réseaux IPv4 et 1280 octets pour les réseaux IPv6, comparés à par exemple, 127 octets de MTU maximal dans un RCSF utilisant la technologie IEEE 802.15.4) ce qui entraîne des fragmentations fréquentes au niveau du routeur de bordure du coté réseau de capteurs. Cette fonctionnalité (la fragmentation) est une source majeure de vulnérabilités, elle risque d'être malicieusement exploitée pour causer des problèmes graves au RCSF connecté à Internet.

2.3. L'ubiquité de la connexion Internet

Dans les applications classiques des RCSFs, les attaques étaient lancées à proximité du réseau de capteurs, nécessitant la présence physique de l'attaquant pour qu'il puisse subvertir le réseau. Dans le scénario de l'Internet des objets, les attaques sont beaucoup plus aisées ; un attaquant connecté à Internet peut atteindre le RCSF à distance, depuis n'importe quel endroit et à n'importe quand.

2.4. L'héritage de la vulnérabilité aux menaces classiques

Etant des réseaux IP (particuliers), les réseaux de capteurs intégrés à Internet sont susceptibles d'hériter les vulnérabilités existantes déjà sur le réseau Internet. Ces vulnérabilités sont diverses et elles sont généralement issues des lacunes de conception du standard de communication TCP/IP et peuvent facilement être exploitées pour lancer des attaques à différents niveaux. A titre d'exemple, et au niveau de la couche IP, un attaquant a la possibilité d'utiliser d'une façon illégale des adresses IP des stations légitimes pour attaquer une station victime sans que celle-ci découvre l'identité réelle de l'attaquant original.

3. Les attaques menaçant les RCSFs dans l'IoT

L'Intégration des réseaux de capteurs à l'IoT a permis de créer un autre espace sur Internet qui va être ciblé par les attaques classiques[5] [62] aussi bien que par de nouvelles catégories de menaces ayant des perspectives plus évoluées. Dans cette partie, nous citons les typologies d'attaques ciblant les réseaux de capteurs dans l'IoT :

[5] Les menaces portées sur la connexion Internet, comme les *Virus*, *Spam, Trojan, Worm, ransomware*, etc

3.1. Les attaques de type déni de service (DoS)

Les attaques de cette classe constituent un énorme danger sur les réseaux de capteurs ouverts à Internet. Elles ont différentes formes et surviennent généralement quand un adversaire (souvent externe) exploite les contraintes des nœuds capteurs (spécialement les limitations de ressources) et de l'hétérogénéité des communications entre hôtes puissants et nœuds capteurs, pour épuiser les capacités de ces derniers et du RCSF en entier. Cela se fait à travers la concentration des flux en rafale sur les capteurs connectés, provoquant ainsi un débordement de la mémoire, une surcharge des ressources de traitements et un excès de la consommation de l'énergie. Les services des capteurs affectés seront alourdis ou totalement bloqués. Une autre variante des attaques par déni de service consiste à exploiter certaines vulnérabilités pour causer des problèmes au niveau du capteur extrême conduisant à son blocage ou à une déstabilisation dans son fonctionnement. La figure ci-dessous illustre le risque de l'attaque DoS.

Figure 5.1. Modèle de l'attaque par déni de service visant les RCSFs dans l'IoT.

Un autre type des attaques DoS, dont l'impact sur les réseaux de capteurs est encore plus sévère, est les attaques par déni de service distribuées (DDoS). Une attaque DDoS est exercée par un réseau d'attaquants, appelé *Botnet*, qui adressent tous la même cible. Les attaquants (dits aussi zombies) dans un *Botnet* sont le plus souvent eux-mêmes des nœuds victimes que l'attaquant principal a employées, à leur insu, pour attaquer d'autres victimes connectées à Internet. La distributivité de l'attaque est bien avantageuse pour l'attaquant car il serait pratiquement difficile de découvrir son identité et sa localisation du fait qu'il se cache derrière plusieurs machines de compromission.

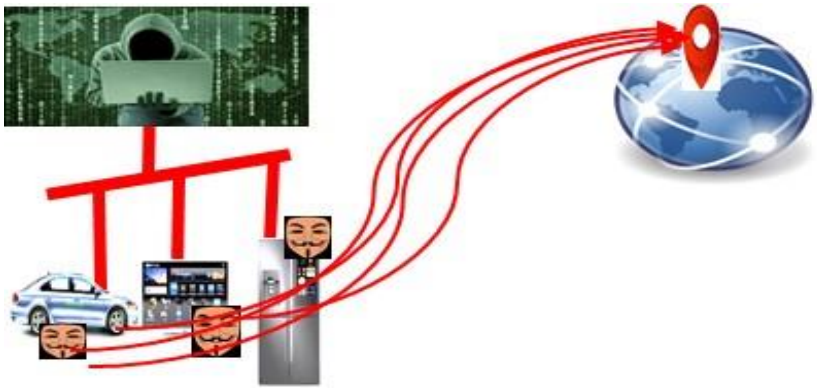


Figure 5.2. Manipulation des capteurs connectés à l'IoT dans le contexte d'un *Thingbot*.

En effet, l'attaquant externe qui est supposé être suffisamment puissant n'a pas besoin de recourir à d'autres machines pour l'assister à attaquer les nœuds capteurs connectés. Donc, il utilise plutôt les

milliers de capteurs de l'IoT pour constituer le *Botnet* appelé dans ce cas *Thingbot* (*Botnet* d'objets intelligents) [59] (figure 5.2). Les capteurs corrompus sont configurés pour générer une très grande masse de *spams* sur Internet. Donc on imagine un réfrigérateur et une télévision connectés qui deviennent des attaquants sans que leur propriétaire s'en aperçoive.

3.2. Attaques sur la fragmentation

Le processus de fragmentation dans le coté des RCSFs intégrés à l'IoT est sensible à une variété d'attaques [60], comme les attaques par amplification des messages. Le principe de l'attaque consiste à générer des messages de très grandes tailles destinés au réseau de capteurs connecté à l'Internet des objets. A leur arrivée, les messages seront fractionnés au niveau du routeur de bordure en plusieurs messages de tailles inférieures qui seront après injectés et communiqués dans le réseau de capteurs. Ce dernier va se retrouver inondé par l'abondance de messages qui lui parviennent.

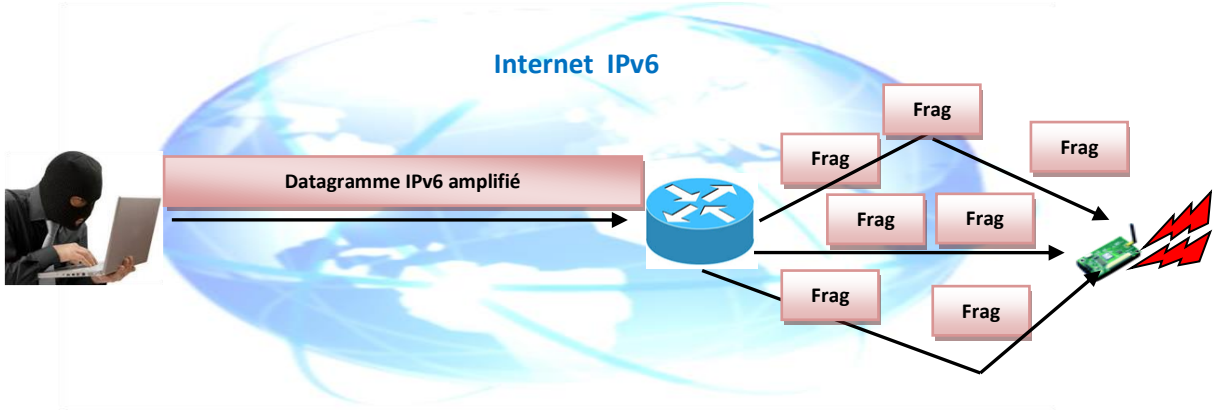


Figure 5.3. Attaque par amplification des messages.

Un attaquant externe envoyant des messages amplifiés vers le réseau de capteurs peut collaborer avec un attaquant interne (un capteur) qui lui-même vise à perturber la procédure de fragmentation/réassemblage des messages communiqués au sein du réseau de capteurs. Par exemple : un adversaire peut bloquer un fragment d'un message et après la réception de tous les autres fragments par le capteur final, il détecte le manque du fragment et ignore le message incomplet qui vient d'être reconstruit et signale l'erreur. Un adversaire interne a la possibilité de forger un fragment et de l'envoyer à un capteur destinataire (la victime) en lui faisant croire qu'un nouveau message lui est destiné. Ce dernier réserve le tampon pour ce nouveau message et reste en attente des fragments suivants que l'attaquant n'enverra jamais, juste pour empêcher la victime de recevoir le bon message. Cette attaque est appelée attaque par réservation de tampon.

3.3. Attaques sur les données de captage au niveau *cloud*

Certaines applications (en particulier les applications médicales et de villes intelligentes) des réseaux de capteurs dans l'IoT engendrent des données massives qui nécessitent d'être stockées et traitées dans les centres de données du *cloud.* Le risque que les données de captage, qui sont

souvent critiques, soient altérées, analysées et/ou divulguées illégalement par une tierce partie ou par des services *cloud* corrompus, est très probable [58].

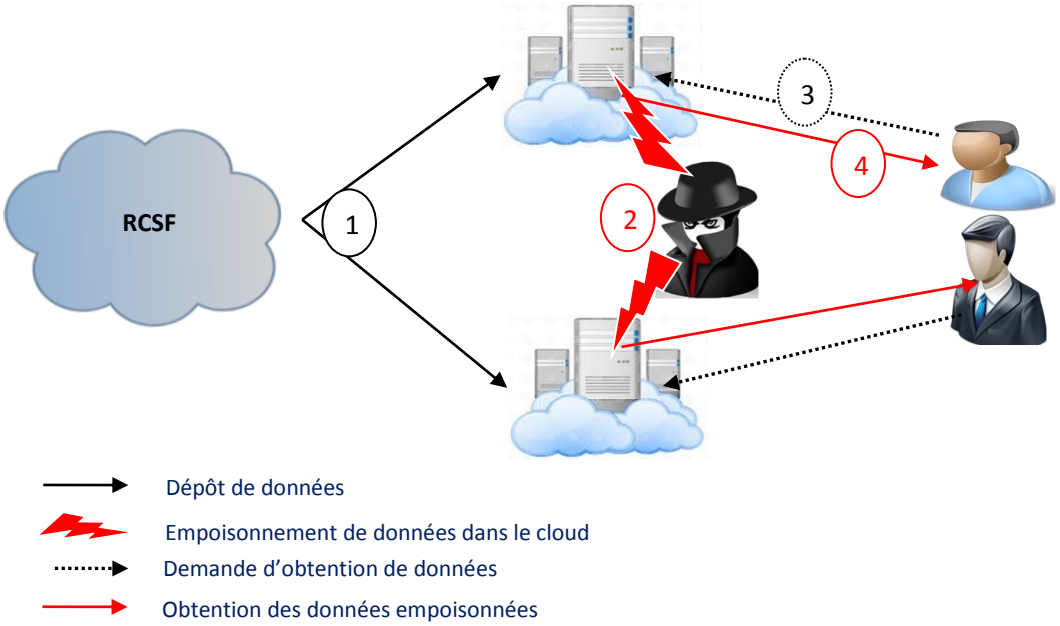


Figure 5.4. Attaques menaçant les données des RCSFs stockés sur les centres de données dans le cloud.

3.4. Menaces liées à la vie privée des utilisateurs

Les capteurs connectés à Internet et intégrés dans notre entourage (nos corps, nos maisons, nos biens, etc.) récoltent des informations qui nous sont privées, à titre d'exemple : l'état de santé, la localisation géographique, le contenu du réfrigérateur, … etc. Ces capteurs apprennent avec le temps sur le comportement, les préférences et les habitudes de leurs utilisateurs et cela demande à ce que ces utilisateurs aient le droit de se faire protéger la vie privée contre toute fuite d'informations qu'ils jugent critiques, sur Internet. La confidentialité de données ainsi que l'identification des parties qui les manipulent sur Interner. En d'autres mots, les utilisateurs doivent savoir qui utilise quoi comme données les concernant et pour quelle raison. D'autre part, il est même nécessaire de permettre aux utilisateurs de l'IoT de d'autoriser la récupération de leurs données par des tierces parties (pour par exemple faire des études statistiques) ou simplement la refuser [61].

La figure ci-dessous illustre le risque que les données de l'IoT collectées dans le cadre d'une application de maison intelligente et une application médicale soient divulguées sur les réseaux sociaux sur l'Internet.

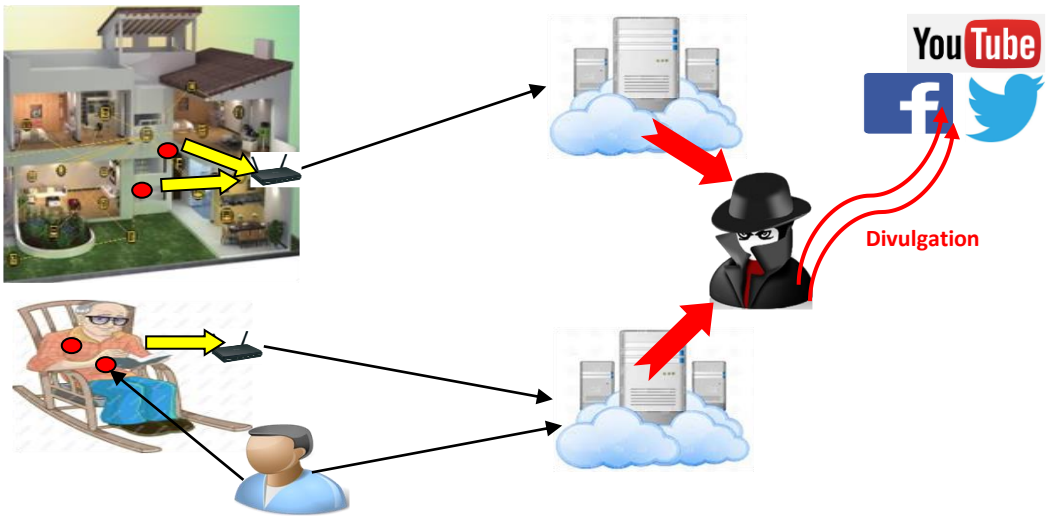


Figure 5.5. La vie privée des utilisateurs et les RCSFs connectés à l'IoT.

4. Les blocs fonctionnels dans la sécurité de l'intégration des RCSFs à l'IoT

Pour la sécurité de l'intégration des RCSFs à l'Internet des objets, les services sécuritaires suivants doivent être efficacement traités.

4.1. La sécurité des différents types de communications avec les capteurs

La sécurité devrait couvrir les communications inter-capteurs connectés à l'IoT et/ou entre les capteurs et les hôtes réguliers de l'Internet. En d'autres mots, que ce soit la communication avec les capteurs connectés à Internet est de type objet-à-objet (M2M ou T2T) ou humain-à-objet (H2T), la protection des capteurs et de leurs données est dans tous les cas nécessaire. En effet, le besoin en la sécurité est une question relative car elle dépend tant de la criticité de l'application que des configurations et des préférences des utilisateurs qui peuvent par exemple privilégier la sécurité d'un type de communication sur un autre (généralement c'est les communications de type humain-à-capteur qui demandent plus de sécurité) et/ou un segment sur un autre (la communication entre le routeur de bordure et l'Internet peut être la seule partie concernée par la sécurité si l'on se fie de la sécurité interne du réseau de capteurs.

4.2. La détection d'intrusion

La détection d'intrusion est primordiale dans la sécurité des réseaux de capteurs. Dans le contexte de l'Internet des objets, les intrusions logiques ayant différentes formes et différents objectifs, transportées via la connexion Internet présentent un risque immanent. Donc la mise en place des mécanismes robustes pour la prévention de l'intrusion, et le filtrage des flux destinés à engendrer des trous d'intrusion au niveau du RCSF, est vivement recommandée.

4.3. La gestion des clés

La mise en place des mécanismes pour l'échange de clés entre les stations connectées à Internet, y compris les capteurs dans l'IoT, sur une plateforme reconnue par son insécurité (le réseau Internet) en est un défi majeur pour l'Internet du futur. A cet effet, des solutions de sécurité doivent permettre aux capteurs connectés d'échanger, en toute sécurité, des clés cryptographiques avec les autres dispositifs connectés à Internet. Cependant, un échange de clés avec une faible dissipation énergétiques est vivement recommandé dans l'IoT.

4.4. La protection de la vie privée des utilisateurs et La gestion de confiance

Les utilisateurs des capteurs connectés à l'IoT doivent savoir à quel point leur vie privée est protégées et ils doivent avoir le droit d'autoriser ou de proscrire les manipulations des données reportées par les capteurs qui les entourent. Ainsi, une évaluation préalable de la confiance est plus que nécessaire entre les réseaux de capteurs et les fournisseurs de services de stockage de données de captage sur le *cloud* et mêmes les entités (les clients) qui demandent de les avoir (les données des capteurs). A ce stade, il est nécessaire de noter que dans le contexte de l'IoT, la gestion de confiance ne tourne pas uniquement entre objets intelligents mais elle se déroule également entre ces objets intelligents et leurs utilisateurs [61].

4.5. Le contrôle d'accès aux services du RCSF

Comme les services des réseaux de capteurs sont souvent très critiques, il est vivement recommandé de définir des règles pour le contrôle d'accès à ces services depuis l'extérieur. Cela signifie que seules les entités autorisées ont le droit d'accéder les ressources des réseaux de capteurs (un capteur, une donnée de captage depuis les capteurs ou le *Cloud*, etc.) dans l'IoT. Il y a plusieurs techniques qui peuvent être utilisées pour garantir le contrôle d'accès, en tenant en compte certaines considérations liées aux objectifs derrière l'accès. En limitant l'accès aux donnés et services des capteurs, certaines problématiques de la protection de la vie privée seront résolues.

Des solutions de contrôle d'accès ont été développées bien avant pour les services de l'Internet classique, comme les techniques de contrôle d'accès à base de rôles ou à base de politiques, mais telles solutions sont très exigeantes en ressources de calcul et en espace mémoire nécessaire pour le stockage des règles de contrôle d'accès. Cela remet en question l'applicabilité de ces solutions pour les réseaux de capteurs intégrés à l'IoT.

4.6. Assurance des services classiques de sécurité

En plus des services sécuritaires déjà cités, il est également nécessaire d'assurer les services de sécurité classiques comme l'assurance de l'authentification et la gestion de confiance entre les

entités, l'intégrité et la confidentialité de données, la disponibilité des services, la sécurité de la localisation, la non-répudiation.

Dans ce qui suit, nous allons voir comment les solutions proposées dans la littérature adressent ces aspects pour répondre à la question de la sécurité de l'intégration des RCSFs à l'IoT.

5. Taxonomie des solutions proposées pour la sécurité de l'intégration des RCSFs à l'IoT

La sécurité et la protection de la vie privée des utilisateurs sont les plus grandes clés de la maturité et le succès de l'Internet des objets. Pour cette raison, de nombreux travaux de recherche récents, ayant pour objet la protection des RCSFs dans l'IoT, sont proposés dans la littérature. Dans cette section nous allons classer, étudier et comparer les solutions de sécurité proposées dans ce contexte.

En réalité, l'ampleur des risques de sécurité diffère selon le modèle d'intégration des réseaux de capteurs à l'IoT. Elle est modeste avec le modèle d'intégration basé proxy et plus importante avec l'intégration par adoption des standards IP. A cet effet, on organise les solutions en deux grandes classes : la classe de solutions de sécurité pour les RCSFs intégrés par proxy et la classe des solutions destinées aux réseaux de capteurs utilisant la technologie IP pour se connecter à l'IoT (les réseaux 6LoWPAN).

5.1. Sécurité des RCSFs intégrés à l'IoT par proxy

Avec ce modèle d'intégration qui isole complètement les capteurs du monde extérieur (l'Internet), les RCSFs se retrouvent déjà bien protégés contre les menaces externes provenant de l'Internet. Cependant, des risques d'infraction à la sécurité par des attaques internes lancées à proximité des capteurs), demeurent susceptibles et donc les solutions de sécurité propriétaires (voir chapitre 2) peuvent être utilisées. Le proxy est dans ce cas, la seule station exposée aux attaques externes. Dans ce cadre, les efforts de sécurité se concentrent tous sur le proxy. La sécurité de ce dernier devrait comprendre sa protection matérielle et logicielle contre les virus et les autres types de codes malicieux, la sécurité de ses couches middleware, sa capacité d'authentifier les hôtes externes qui tentent de communiquer avec lui. Néanmoins, les solutions proposées dans la littérature se concentrent beaucoup plus autour de la sécurité des couches middlewares.

La couche middleware UBIWARE [42] définit le modèle SURPAS (*Smart Ubiquitous Resource Privacy and Security*) [63] qui présente des architectures de sécurité fondées sur les principes du web sémantique. Peu de détails sont donnés sur les méthodes de sécurité utilisées dans le modèle. Par exemple : bien que le modèle présente une approche orientée politiques pour le control d'accès aux services du réseau de capteurs connecté, il n'y a pas un modèle clair pour la gestion des identités, ni pour la protection et la vérification de l'intégrité des politiques stockées au niveau du proxy.

SMEPP (*Secure Middleware for Peer-to-Peer*) [64] est une solution middleware assurant la sécurité de l'architecture pair-à-pair dans l'IoT. La sécurité de SMEPP repose sur la cryptographie par courbes elliptiques et le concept de groupe. Quand un nœud tente de joindre un groupe, une technique de challenge-réponse est employée pour l'authentification mutuelle entre ce nœud et tous les membres du groupe. A ce point, le nouveau nœud obtient une clé partagée avec le reste des membres du groupe avec lesquels il peut entreprendre des communications sécurisées. La solution ne définit aucun mécanisme de contrôle d'accès, pourtant c'est nécessaire dans les scénarios de l'Internet des objets.

La solution middleware VIRTUS [44] est riche en mécanismes de sécurité. Elle garantit la tunnelistaion sécurisée des communications entre le proxy et les hôtes externes à l'aide du protocole TLS (*Transport Layer Security*) [65], l'authentification à travers un mécanisme flexible qui supporte différents techniques comme identifiant/mot de passe, approches basées jeton comme OAuth2 et les certificats X.509. La solution comprend également un mécanisme pour le control d'accès via le standard XMPP (*Extensible Messaging and Presence Protocol*) très utilisé dans l'Internet classique.

5.2. Sécurité des RCSFs intégrés à l'IoT par adoption du standard TCP/IP

Bien que l'approche d'intégration par proxy soit mieux positionnée compte tenu de la sécurité des RCSFs, l'intégration des réseaux de capteurs sans fil à l'Internet des objets en tant qu'une partie de l'infrastructure IP (IPv6) est l'approche la plus avantageuse et la plus attractive car elle répond mieux aux exigences applicatives de l'IoT. En effet, le défi réel c'est de permettre l'ouverture des RCSFs à l'Internet tout en les protégeant contre les risques potentiels qui menacent et la station de base (jouant le rôle d'un routeur de bordure ou un proxy HTTP-CoAP) et les capteurs dans un réseau 6LoWPAN. Dans cette section, nous présentons les solutions de sécurité proposées pour protéger les réseaux 6LoWPAN dans l'Internet des objets.

Les solutions de sécurité des réseaux de capteurs adoptant telle politique d'intégration peuvent être classées en deux grandes classes : la classe des solutions de sécurité interne au réseau 6LoWPAN et une classe de solutions de sécurité de l'intégration, comme le montre la figure suivante.

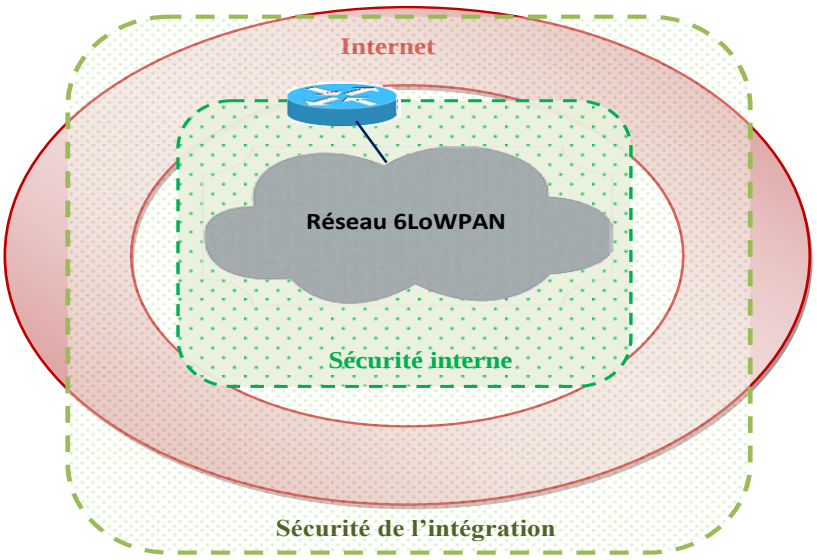


Figure 5.6. Deux classes pour la sécurité des réseaux 6LoWPANs dans l'IoT.

5.2.1. La sécurité interne des réseaux 6LoWPANs

La sécurité locale (ou interne) des réseaux 6LoWPANs englobe essentiellement les mécanismes de sécurité du déploiement du réseau, la sécurité des communications au niveau liaison, la sécurité de la fragmentation, la sécurité du routage (par le protocole RPL) et la détection d'intrusion physique.

A. La sécurité du déploiement

La phase de mise en route du réseau de capteurs (*bootstrapping)* est très importante dans le cycle de vie des objets de l'IoT, elle affecte même le bon fonctionnement des nœuds capteurs dans la phase opérationnelle. La sécurité des réseaux de capteurs dans cette phase consiste essentiellement en la configuration sécurisée des capteurs avec le matériel sécuritaire initial (pouvant être des clés cryptographiques, des algorithmes de chiffrement/déchiffrement, des programmes de détection d'intrusion ou autre), nécessaire pour entreprendre des communications sécurisées sur Internet. La meilleure façon de pré-charger telles données dans les nœuds capteurs est de le faire manuellement (par l'administrateur) et à travers une liaison filaire avec chaque capteur juste avant le déploiement. La sécurité dans cette phase consiste aussi en l'établissement des liens de sécurité locaux entre les nœuds capteurs voisins du même réseau.

Une fois le réseau de capteurs est déployé, la sécurité physique des capteurs et la prévention contre toute forme d'attaque d'extraction de leurs informations, l'insertion des objets malicieux dans le réseau et même la compromission des nœuds capteurs légitimes. Dans le chapitre 2, nous avons cité un projet européen [10] destiné au développement des nœuds capteurs physiquement résistants aux opérations de compromission, tout en maintenant leur faible coût. Dans [66], un standard de l'IETF traite la sécurité de *boostrapping* pour les réseaux 6LoWPAN, par les protocoles de gestion de clés et d'authentification dans l'Internet.

B. La sécurité au niveau liaison de données

Le standard IEEE 802.15.4 supporte optionnellement la sécurité au niveau de la sous-couche MAC. Le modèle est basé sur le partage d'une clé entre tous les nœuds capteurs y compris le routeur de bordure. Cette clé est dite la clé du réseau, elle est souvent pré-chargée (ou transporté aux nœuds en toute sécurité) et elle est utilisée pour la sécurité locale des communications en utilisant l'algorithme AES-128 [67].

Bien que cette méthode soit simple et facile à mettre en œuvre, le partage au sens large de la clé de sécurité présente une véritable vulnérabilité, car si la clé est découverte par un attaquant, il pourra espionner sur toutes les communications internes dans le réseau.

C. La sécurité de la fragmentation

René Hummen et.al. ont présenté dans l'article [60] une étude approfondie des menaces liées à la fragmentation des datagrammes IPv6 au sein des réseaux 6LoWPANs. Par exemple pour faire face à l'attaque par amplification des datagrammes IPv6, le routeur de bordure peut fixer une taille maximale que tout datagramme ayant une taille excédentaire sera automatiquement rejeté.

Concernant l'attaque par injection de fragments, les auteurs proposent un mécanisme qui garantit le bon chaînage entre les fragments correspondant au même datagramme IPv6. Le principe de la solution est d'ajouter une valeur de hachage à chaque fragment, tel que la valeur de hachage du fragment i est obtenue à partir de la partie donnée et la valeur de hachage du fragment (i-1), comme indiqué dans la formule (5.1). Il est préférable que les nœuds capteurs adoptent un système de gestion de clés (utilisées pour générer les valeurs de hachage), qui soient à la fois économiques en énergie et immunitaires aux attaquants internes (les nœuds de compromission), par exemple, les schémas de pré-chargement probabiliste des clés.

$$hi = H(Données + h(i-1)) \qquad (5.1)$$

Et pour l'attaque par réservation du tampon ou l'attaquent forge un fragment et l'envoie au nœud capteur cible juste pour réserver le tampon et prévenir la réception du bon message. Comme solution à cette attaque, les auteurs proposent une solution basée sur les scores des émetteurs de fragments. Le nœud capteur terminal calcul le temps moyen entre deux fragments consécutifs a et le temps l depuis le dernier fragment reçu. Le récepteur pénalise l'émetteur en lui réduisant le score si l est important par rapport à a. les sources avec les scores les plus faibles auront également des faibles priorités pour occuper le buffer de la destination. A chaque réception d'un fragment depuis une telle source, le nœud capteur récepteur met à jour les valeurs de l et a. le score suivant l'algorithme suivant :

Si ($l = a + x$, avec x suffisamment important) alors

    Minimiser le score.

Sinon ($l = a$ ou $l = a - x$)

    Augmenter le score.

Fin

Les résultats d'évaluation de performances effectuées par simulation ont montré l'efficacité des solutions proposées.

D. La sécurité du routage

Bien que le protocole RPL assure un routage efficace au sein du 6LoWPAN, il est vulnérable aux attaques classiques ciblant le processus de routage dans les RCSFs, du fait qu'il ne comporte aucune mesure de sécurité explicite par défaut. Dans [68], les auteurs ont étudié par simulation, la résistance du protocole RPL aux attaques communes dans la couche réseau. Les résultats ont démontré que RPL résiste implicitement, par son principe de fonctionnement, à l'attaque *Hello flooding* (les liens entre les nœuds en RPL sont bidirectionnels). Mais il est affecté par les attaques *sinkhole*, *selective forwarding*, *wormhole* et *sybil*. Les auteurs proposent une contremesure adaptée pour sécuriser RPL de l'attaque *selective forwarding.* Il s'agit d'utiliser la technique de battement de cœur (*heartbeat*) réalisée par l'envoi périodique des messages *echo-request/echo-reply* du protocole ICMPv6 (*Internet Control Message Protocole version 6*) protégés dans le protocole IPsec. Le 6BR envoie un message *echo-request* à chaque nœud, et reste en attente des réponses. Ceci permet de détecter si le trafic est filtré envers et/ou depuis le nœud.
Des mécanismes additionnels sont nécessaires pour sécuriser le protocole RPL contre les attaques *sinkhole*, *wormhole* et *sybil* [127].

E. La détection d'intrusion physique

La détection d'intrusion est nécessaire pour l'optimisation de la sécurité des réseaux. Elle présente en fait, une deuxième ligne de défense après les solutions cryptographiques. L'intrusion physique est concrétisée par les nœuds capteurs ayant subi une opération de compromission. Ces nœuds disposent du matériel sécuritaire (les clés cryptographiques, les algorithmes de chiffrement / déchiffrement) et accèdent ainsi, les services du réseau tout à fait comme les nœuds légitimes. Le problème de détection d'intrusion dans les RCFSs isolés de l'internet, était largement étudié et plusieurs systèmes de détection d'intrusion (SDI) ont été proposés.

Pour le cas de la détection d'intrusion physique dans le 6LoWPAN, les SDIs proposés se concentrent généralement sur la détection des attaques ciblant la couche réseau. Dans [69], le premier SDI pour les réseaux de capteurs basés IP est proposé. Il considère trois modèles d'attaques, parmi eux, le modèle des attaques internes au 6LoWPAN. Deux modules de détection d'intrusion sont proposés : un module pour analyser les paquets issus de l'internet, un deuxième module d'analyse

des paquets du 6LoWPAN. Les nœuds du réseau sont de deux types. Les nœuds du premier type sont dits nœuds esclaves, chargés de la surveillance et de la génération des messages d'alertes. Le second type de nœuds c'est les nœuds maitres, qui prennent la décision et applique leurs politiques à leurs nœuds esclaves. Ces derniers sont prévus pour détecter les anomalies du trafic et le brouillage au niveau MAC.

Dans [70], un SDI pour les 6LoWPAN est proposé. Le module de détection est fondé sur deux composants. Le premier composant se préoccupe de la détection des attaques du routage. Les techniques utilisées dans ce composant sont basées sur la spécification du protocole RPL. Le deuxième composant est chargé de la détection des anomalies qui ne dépendent pas nécessairement du RPL, comme les attaques de dégradation de la qualité de service (QoS) dans le réseau. Les deux composants coopèrent ensemble pour plus d'efficacité du SDI.

Dans [71] est proposée une solution dénommée SVELTE : un SDI pour protéger les réseaux 6LoWPAN dans l'IoT. Pour la sécurité physique du 6LoWPAN, le système proposé définie deux modules implantés au niveau du 6BR. Le 6Mapper (6LoWPAN Mapper) est un module vital utilisé pour la reconstruction périodique, du graphe DODAG de RPL donnant une cartographie complète du réseau. Les données de ce module sont nécessaires pour le module de détection. Ce dernier utilise l'hybride entre plusieurs techniques, comme la détection par signature et la détection d'anomalie, pour détecter diverses attaques principalement les attaques d'inconsistance du graphe, *sinkhole*, *selective forwarding* et *sybil* avec un bon taux de précision (*true positive rate*).

5.2.2. La sécurité de l'intégration

La sécurité de l'intégration qui est la plus importante, comporte des procédures d'établissement de la sécurité avec les hôtes externes et des techniques de protection des communications avec eux. Elle inclut la sécurité de bout-en-bout des communications, la gestion de clés, les techniques d'authentification et de contrôle d'accès, la sécurité des données des réseaux 6LoWPANs dans le *cloud* et la détection d'intrusion logique.

A. La gestion de clés

La gestion de clés est un aspect très important dans la sécurité de l'IoT. Cet aspect définit comment se déroule la négociation entre un hôte de l'Internet et un nœud capteur pour l'établissement des liens de sécurité pour sécuriser les communications. Cette fonctionnalité est à la fois et difficile car d'une part les nœuds capteurs dans l'IoT sont soumis à de fortes contraintes qui leur empêche de supporter les mécanismes de gestion de clés classiques de l'Internet. D'autre part, l'échange de clés entre les hôtes communicants sur une plateforme non sécurisée comme l'Internet est un véritable défi.

- **Les pré-requis :**

En fait, la conception des systèmes de gestion de clés pour l'IoT doit remplir les critères suivants :

- **La distribution à la demande des clés :** tandis que dans les réseaux de capteurs il est beaucoup plus approprié de faire la distribution hors ligne (proactive, effectuée avant le déploiement) pour des raisons de simplicité. Dans le contexte de l'IoT, la distribution dynamique des clés est jugée nécessaire, afin de répondre mieux aux exigences en termes de flexibilité et d'évolutivité. Dans tel cas, les clés sont distribuées dans la phase de négociation entre les entités voulant communiquer. Donc, un nœud peut communiquer avec n'importe quel autre nœud sans avoir à partager des clés pré-chargées.

- **Authentification mutuelle des entités communicantes :** les systèmes de gestion de clés doivent permettre aux clients et aux serveurs de s'authentifier mutuellement, pour rassurer les clients/serveurs que les données sont communiquées depuis/vers les bons serveurs/clients.

- **Moindre surcharge** : le coût des calculs à effectuer ainsi que, la quantité des messages de signalisation nécessaires dans un système de gestion de clés doivent être aussi réduits que possible.

- **La résistance :** les systèmes de gestion de clés sont appelés à être suffisamment robustes pour pouvoir résister aux attaques d'extraction des clés.

- **Le support de l'évolutivité du réseau et l'extensibilité :** l'efficacité du système de gestion de clés doit être indépendante du nombre des entités communicantes aussi bien que du taux d'informations à stocker dans chacune d'elles.

❖ L'algorithme *Diffie-Hellman*

L'échange de clés *Diffie-Hellman* [72] inventé en 1976, est un algorithme célèbre d'échange de clés sur Internet. Il permet à deux entités nommés conventionnellement Alice et Bob de se mettre d'accord sur un secret partagé qui va être employé par la suite pour protéger leurs conversations. Il est fondé sur la cryptographie à clé publique (PKC : *Public Key Cryptography*) et il tire sa robustesse de la difficulté du problème du logarithme discret. Le principe de l'algorithme est comme suit :

- Alice et Bob se mettent d'accord sur un grand nombre *p* tel que : (*p*-1)/2 soit premier et sur un générateur *g* primitif par rapport à *p*.
- Alice choisit un nombre secret au hasard *a,* et calcule sa clé publique $A = g^a$ *modulo p.* Bob procède de même ; il choisit un secret initial *b,* et calcule sa clé publique *Diffie-Hellman* $B = g^a$ *modulo p.*
- Ensuite Alice et Bob s'échangent leurs clés publiques en clair, sans aucun risque qu'une troisième partie découvre les secrets initiaux à partir des informations publiques (*g, p, A* et *B*).
- Finalement Alice et Bob calculent la clé secrète de la session *Diffie-Hellman* :

$K = B^a \text{ modulo } p = A^b \text{ modulo } p$.

Bien que l'algorithme soit suffisamment robuste, il est cependant vulnérable à l'attaque homme au milieu (*Man in the middle*). Donc, une entité malveillante Eve a la possibilité de s'interposer entre Alice et Bob en faisant croire chacun d'entre eux qu'il est en train d'élaborer la session de sécurité avec la bonne partie, alors que ils sont tous deux en train d'établir une session avec l'attaquant.

L'algorithme *Diffie-Hellman* est largement utilisé sur Internet et il est également disponible en une deuxième version qui est basée sur la cryptographie à base de courbes elliptiques ECDH (*Elliptic Curve Diffie-Hellman*) [130] dont la complexité de calculs est moindre. Mais par rapport aux réseaux de capteurs les primitives sécuritaires impliquées dans l'algorithme demeurent onéreuses. Dans ce contexte, quelques travaux de recherches ont adressé l'adaptation de l'algorithme afin qu'il puisse être utilisé par tels réseaux dans l'Internet des objets. Dans ce cas, l'adaptation est matérialisée soit par la délégation des calculs à des stations bien puissantes dans le RCSF, ou plutôt par la distribution de ces calculs sur un ensemble d'agents assistants, comme nous allons voir dans la section suivante.

- Dans [73], les auteurs proposent un nouveau protocole d'échange de clés pour les réseaux de capteurs intégrés à l'IoT. Le protocole considère la sécurité des communications multicast où un hôte de l'Internet initie une requête destinée à multiples nœuds capteurs en même temps. Le protocole proposé suppose une structure hiérarchique du réseau de capteurs formée par plusieurs groupes de nœuds. Un hôte de confiance distribue, en toute sécurité, les clés sécuritaires aux nœuds capteurs membres de chaque groupe. La distribution des clés se fait en deux étapes. la première étape correspond à la communication sécurisée de la clé du groupe depuis l'hôte vers le nœud capteur chef du groupe, passant par la station de base (routeur). Ensuite, le chef du cluster diffuse la clé aux nœuds de son groupe par une communication sans fil multicast. L'hôte de l'Internet a le pouvoir de rajouter un nouveau nœud capteurs à un groupe, et même d'en révoquer un autre déjà existant. Les résultats d'évaluation du protocole proposé montrent qu'il a un coût de calcul et de communication acceptable. L'empreinte mémoire est également faible.

- Les auteurs proposent dans [74-75], des schémas de gestion de clés robustes adaptés pour assurer la sécurité des applications médicales dans l'IoT, et des réseaux de capteurs mobiles connectés à l'IoT.

B. La sécurité de bout-en-bout

La sécurité de bout-en-bout est un facteur indispensable dans la sécurité de l'Internet. Ce mécanisme repose sur les mesures de protection implémentées sur les hôtes extrêmes (les terminaux) qui permettes d'entreprendre des communications sécurisées d'un bout de communication à un autre sans que les nœuds intermédiaires puissent accéder le contenu des messages échangés. La sécurité de bout-en-bout peut être assurée à différents niveau du modèle TCP/IP (réseau, transport ou application). Dans les RCSFs isolés de l'Internet, la sécurité de bout-en-bout entre les nœuds capteurs et la station de base était un peu difficile à garantir car les nœuds capteurs intermédiaires

doivent dans la majorité des cas accéder les données pour les agréger. Une fois que le réseau de capteurs est intégré à Internet, la sécurité de bout-en-bout devient nécessaire, comme l'information communiquée entre les nœuds capteurs et les hôtes de l'Internet est dans la plupart des cas critique et liée à la vie privée des utilisateurs. Néanmoins, les contraintes sévères imposées sur les réseaux 6LoWPANs, spécialement les limitations de ressources, forment un obstacle pour la projection directe des standards de sécurité initialement proposés pour l'Internet (protocoles basés sur le modèle TCP/IP) qui sont reconnus par leur robustesse et efficacité. A cet effet, l'adaptation de ces standards pour les réseaux 6LoWPANs est fortement encouragée.

Récemment, les chercheurs ont intensivement investigué cette problématique en vue de trouver des techniques adéquates qui permettent l'extension des protocoles classiques de sécurité de bout-en-bout pour les réseaux de capteurs intégrés à Internet avec un minimum de surcoût. Les solutions proposées dans ce cadre, se focalisent davantage sur la compression des messages pour affaiblir la consommation d'énergie nécessaire pour la communication des messages au sein du RCSF et ainsi, réduire leur besoin en mémoire des capteurs. Dans très peu de cas les tentatives d'adaptation se basent sur la distribution de la charge de calcul apportée par les primitives sécuritaires définies dans tels standards.

Suivant le niveau d'application de la sécurité de bout-en-bout dans le modèle TCP/IP, on distingue principalement trois grandes classes de solutions : les solutions proposées au niveau application, transport ou réseau.

B.1. les solutions de sécurité de bout-en-bout au niveau de la couche réseau

Dans cette classe, la sécurité de bout-en-bout est assurée au niveau de la couche réseau par le fameux protocole IPsec (IP sécurité) [76]. IPsec représente une encapsulation sécurisée pour le standard IP. Il comporte deux sous protocoles :

- **AH *(Authentication Header)***: AH [77] est conçu pour assurer l'intégrité de données, et l'authentification de l'origine de ces données sans chiffrement (la confidentialité n'est pas assurée par le mécanisme AH), et la protection contre le rejeu. Son principe est d'adjoindre au datagramme un bloc de données supplémentaire, appelés " valeur de vérification d'intégrité, (ICV : *Integrity Check Value*)", permettant au récepteur de vérifier l'authenticité des données incluses dans le datagramme. La protection contre le rejeu est réalisée grâce à un numéro de séquence. La figure ci-dessous décrit le format des messages AH.

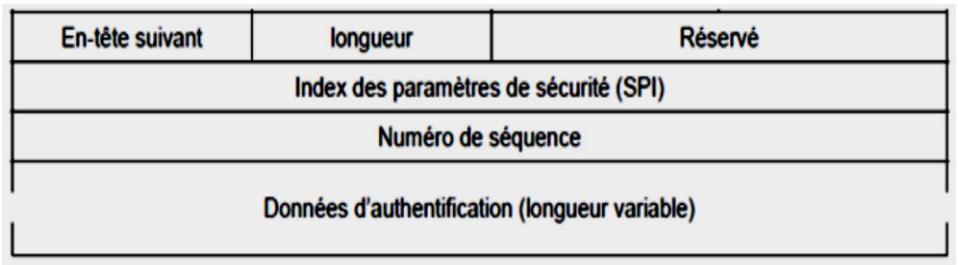


Figure 5.7. Format de l'entête du protocole AH.

Tel que :

- ✓ En-tête suivant : identifie le protocole de niveau supérieur (TCP : 6, UDP : 17, ...).
- ✓ Réservé : emplacement réservé pour une future utilisation.
- ✓ SPI : identifiant unique de l'association de sécurité entre les deux nœuds communicants.
- ✓ Données d'authentification : résultat du hachage effectué sur la totalité de paquet IP.

- **ESP *(Encapsulating Security Payload)*** : ESP [78] son rôle principal est d'assurer la confidentialité mais il peut aussi assurer l'authentification et l'intégrité des données. Contrairement à AH, où on ajoute un en-tête supplémentaire au paquet IP, ESP fonctionne selon le principe d'encapsulation; les données sont chiffrées puis encapsulées entre un en-tête et un en-queue (*trailer*). La figure suivante illustre le format d'un message ESP.

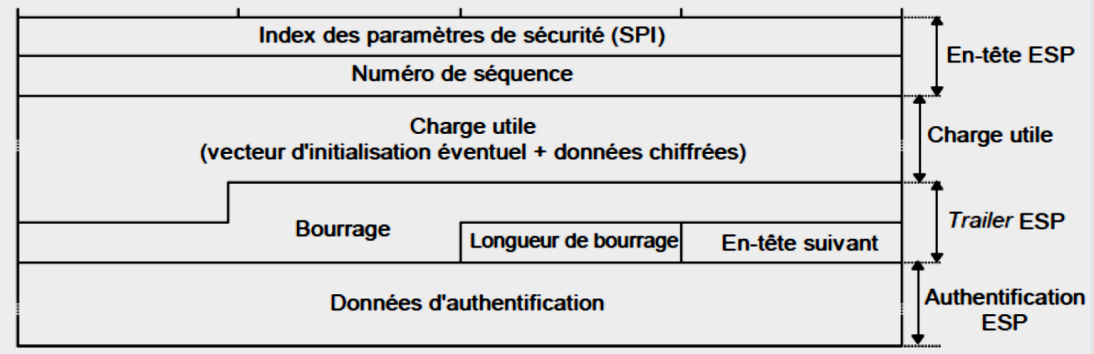


Figure 5.8. Format du paquet IP sécurisé par ESP.

Le champ données d'authentification ESP concerne uniquement l'entête, la charge utile et l'en-queue ESP, sans prendre en considération l'en-tête IP).

De plus, le protocole IPSec peut opérer en mode transport ou tunnel comme illustré dans la figure ci-dessous :

- ✓ **Le mode transport** : protège uniquement le contenu du paquet IP sans toucher à l'en-tête. Ce mode n'est utilisable que sur les équipements terminaux (clients et serveurs).
- ✓ **Le mode tunnel** : permet la création d'un tunnel de sécurité pour la protection des datagrammes IP entre les passerelles. Chaque datagramme IP est encapsulé dans un nouveau datagramme IP. Ainsi, la protection porte sur toutes les parties d'un datagramme IP arrivant à l'entrée d'un tunnel, y compris les champs de l'en-tête (adresse source et destination des hôtes terminaux, par exemple).

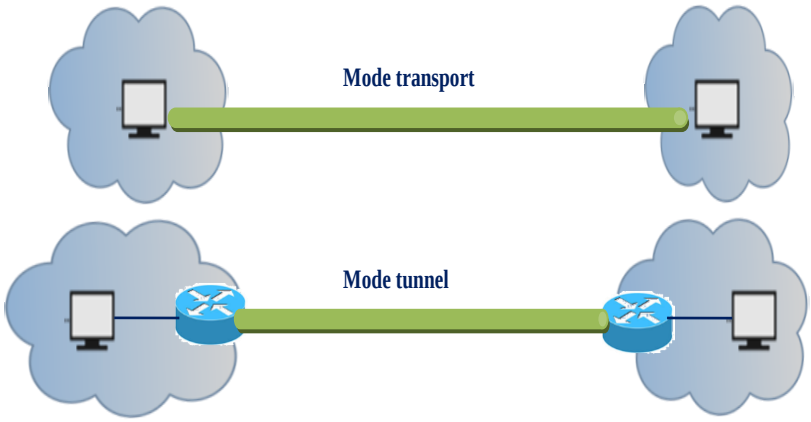


Figure 5.9. Les deux modes de fonctionnement du protocole IPsec.

- Dans [79], une solution propose d'utiliser le protocole IPsec en mode transport pour la sécurité des communications avec les capteurs connectés à l'IoT. La solution présente une extension de la compression 6LoWPAN pour la compression des en-têtes de sous-protocoles AH et ESP. Les résultats d'évaluation ont montré que la solution permet un gain énergétique considérable lors de la communication des messages IPsec.

Le protocole IPsec nécessite qu'il soit accompagné d'un autre protocole qui gère la négociation des paramètres de sécurité entre les entités communicantes. L'ensemble de ces paramètres est appelé association de sécurité (SA : *Security Association*) et englobe les clés de sécurité, les algorithmes de chiffrement, les algorithmes d'authentification, etc. Systématiquement, IPsec peut fonctionner conjointement avec le protocole IKEv2 ou le protocole HIP.

❖ **IPsec/IKEv2 :**

IKEv2 (Internet Key Exchange version 2) [76] est un protocole d'échange de clés et de configuration de la sécurité sur Internet. Il fonctionne au niveau de la couche application pour préparer le matériel sécuritaire que va utiliser par la suite le protocole IPsec pour la protection des datagrammes IPv6. D'autre part, la procédure d'échange de clés dans le protocole IKE peut se réaliser statiquement (configuration des hôtes avec clés pré-partagées) ou automatique en utilisant l'algorithme *Diffie-Hellman* (la sécurité est établie à travers l'échange de messages et les calculs cryptographiques compliqués pour la génération des secrets).

- Les auteurs du travail [79] ont considéré dans leur solution le mode statique, où les clés partagées sont manuellement pré-chargées dans les entités communicantes, afin d'éviter la lourdeur du mode automatique. Cette solution est à la fois simple et économique mai, le mode automatique demeure plus pratique pour les applications de l'IoT. Tenant cela en compte, les mêmes auteurs dans [80] proposent une extension de compression 6LoWPAN pour les en-têtes du protocole IKE, afin d'atténuer les coûts énergétiques de la communication dans la phase d'établissement automatique (et dynamique) de la sécurité. Toutefois, les coûts de calcul des opérations cryptographiques asymétriques restent très importants pour être supportés par des nœuds capteurs.

- Dans ce même contexte, une solution dans [81] propose de déléguer ces opérations à la passerelle qui est une entité suffisamment puissante. Une autre solution proposée dans [82] définit une méthode pour la distribution de la charge sécuritaire introduite par telles opérations sur un ensemble de proxy. Les coûts de calculs avec ces deux dernières solutions citées sont très faibles dans le coté des capteurs mais comme les messages IKE ne sont pas compressés, le coût des communications de ces messages est important.

❖ **IPsec/HIP :**

Le protocole HIP (*Host Identity Protocol*) [83-84] est une solution alternative à IKE. HIP fonctionne juste au-dessus de la couche IP et introduit un système d'identification qui découple les informations d'identification des informations de localisation. En conséquence, les applications référencent les hôtes par des identifiants générés d'une manière cryptographique au lieu des adresses IP correspondantes qui informent sur l'emplacement physique des hôtes. Ainsi, HIP facilite la mobilité dans les réseaux IP et assure un bon niveau de sécurité de la localisation des hôtes ce qui est fortement recommandé dans la majorité des applications de l'Internet des objets.

HIP définit un mécanisme de négociation de sécurité authentifié, appelé *Base Exchange* (HIP-BEX). Comme dans IKE, HIP-BEX est basé sur *Diffie-Hellman* pour établir dynamiquement les associations de sécurité entre les pairs HIP communicants sur Internet. Uniquement quatre messages sont nécessaires pour la l'établissement d'une clé secrète de la session HIP qui sera utilisée par la suite par le protocole ESP de IPsec. En outre, chaque pair HIP doit avoir une clé publique servant comme identificateur de l'hôte (HI), dont la contrepartie privée (une clé) n'est connue et utilisée que par son propriétaire légitime. Ces deux clés sont utiles pour des objectifs d'authentification et de vérification de la validité des identités.

La sémantique et le contenu de ces messages, ainsi que, le scénario de HIP-BEX sont illustrés dans la figure ci-dessous.

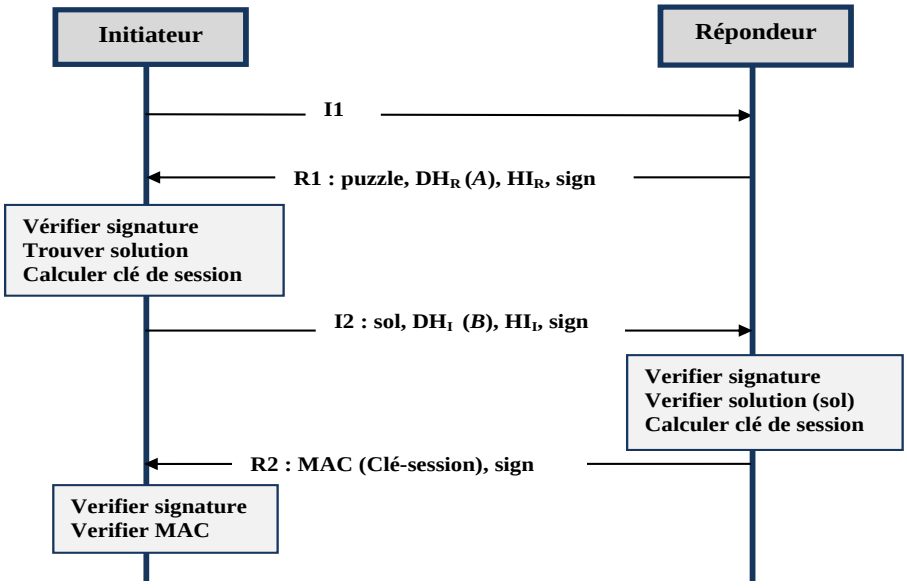


Figure 5.10. Le mécanisme HIP Base-Exchange.

Tout d'abord, le nœud initiateur initie l'échange en envoyant le message vide I1 (contenant uniquement l'en-tête). À la réception de I1, le répondeur HIP répond à l'initiateur par un message R1 qui transporte sa clé publique *Diffie-Hellman* (*DH*), son identifiant d'hôte (HI) et un puzzle utilisés pour prévenir les attaques DoS et dont la complexité peut prendre différents degrés de difficulté tout dépond du nœud initiateur ; s'il est un nœud de confiance le puzzle sera plus ou moins facile à résoudre. Pour l'authentification des pairs, le message est signé avec la clé privée correspondante à la clé publique HI de l'émetteur. L'initiateur calcule la clé de session *Diffie-Hellman* puis, il envoie un message signé I2, où il donne la solution du puzzle, en plus de ses informations cryptographiques nécessaires (*HI* et *DH*). Après les vérifications de validité et de correspondance de la solution avec le puzzle, le répondeur calcule à son tour la clé secrète de session *Diffie-Hellman*. Enfin, le répondeur prouve la propriété de la clé secrète auprès de l'initiateur par l'intermédiaire d'un MAC (*Message Authentication Code*) dérivée de la clé, dans un message final R2. A la réception du message R2, l'initiateur vérifie la signature du message et la validité de la clé de session.

En fait, le mécanisme HIP-BEX implique des primitives cryptographiques asymétriques très lourdes. Pour cette raison, il est constaté qu'il n'est pas pratique de l'appliquer directement pour les nœuds capteurs qui sont équipés des microcontrôleurs très limités en puissances de calcul et d'énergie. Par conséquent, plusieurs solutions ont été proposées pour alléger le processus HIP-BEX et, le rendre plus adapté aux RCSFs intégrés à Internet.

- Dans [85], les auteurs proposent HIP *Diet Exchange* (HIP-DEX) pour réduire le coût des calculs de HIP-BEX sur les nœuds capteurs dans l'IoT, en introduisant cryptographie à courbes elliptiques dans le protocole *Diffie-Hellman :* ECDH (*Elliptic Curve Diffie-Hellman*). Une seule clé publique est nécessaire pour calculer la clé de session et pour identifier les pairs HIP et la détention de la clé de session est suffisante pour authentifier le nœud et prouver sa légitimité. Bien que la solution proposée semble assez simple et légère, l'altération des formats de message dans le standard HIP par élimination de certains champs dans les messages échangés, peut introduire des problèmes d'incompatibilité lors de l'établissement de la session avec les hôtes de l'Internet en utilisant le HIP originale. En outre, les principes de l'ECC (*Elliptic Curve Cryptography*) adoptés par Diet-HIP demeurent une solution onéreuse pour les capteurs extrêmement contraints.
- Dans [86], les auteurs proposent une variante intuitive du protocole HIP pour l'adapté au coté des RCSFs dans l'IoT. La solution est appelée LHIP (*Lightweight HIP* ou HIP Léger). LHIP conserve la même syntaxe pour les messages HIP et fournit un mécanisme de sécurité qui se concentre uniquement sur les fonctions de hachage pour vérifier l'intégrité des messages échangés. Par conséquent, les champs destinés à contenir des clés et d'autres informations de sécurité sont maintenus et communiqués dans les messages, mais, ils sont ignorés par les pairs. Malgré la solution est compatible avec le standard HIP, la sécurité fournies par LHIP est jugée très faible du fait que des mécanismes d'authentification mutuelle et d'échange de clés sont négligés.
- Toujours dans le but d'abaisser la complexité de calcul des primitives sécuritaires de HIP-BEX, les auteurs dans [87] proposent de distribuer les opérations les plus coûteuses en temps de calcul et en énergie, dans la procédure de génération de la clé secrète sur un groupe de nœuds situés aux

alentours et qui sont beaucoup moins contraints. L'autorisation et l'authentification du groupe des nœuds assistants sont effectuées en utilisant les fonctions de hachage à sens unique. Chaque nœud assistant calcule une partie Ki de la clé de session entière K, de façon parallèle, et la remet au nœud capteur terminal (le pair HIP) qui accumule les parties Ki reçues auprès des collaborateurs pour obtenir la clé de session. Notons que les collaborateurs vérifient même la signature des messages destinés au capteur.

Cette solution a l'avantage de réduire considérablement le coût relatif aux traitements. Cependant, les coûts énergétiques relatifs à la communication augmentent sensiblement dans le RCSF, car il y a une quantité importante de messages non compressés à échanger dans le réseau de capteurs. Un autre inconvénient de la solution tourne autour de la difficulté de gérer la fiabilité des communications avec les collaborateurs, et la complexité de la surveillance de leurs comportements. Nous soulignons également que le modèle de distribution proposé est explicite à l'hôte externe de l'Internet; ce dernier est conscient de l'interposition des nœuds assistants (il reçoit et envoie des informations vers et depuis eux), ce qui n'est pas pratique.

- Même si HIP (ou Diet-HIP) nécessite la communication d'un nombre réduit de messages de signalisation afin de mettre en place une session de sécurité entre deux hôtes de l'Internet, ces messages sont très longs et nous ne pouvons pas ignorer le coût élevé en termes de consommation d'énergie et de surcharge liée à la fragmentation/réassemblage résultant de la communication de ces messages dans un réseau de capteurs. Afin de surmonter ce problème, une couche de compression pour Diet-HIP nommé Slimfit a été proposée en [88]. Certains champs de l'en-tête et dans les paramètres de signalisation sont compressés, de telle sorte que le coût de la communication peut être aussi réduit que possible. Les résultats de l'évaluation montrent que le modèle de compression proposé permet d'affaiblir la fréquence de fragmentation des paquets Diet-HIP avec une consommation d'énergie nettement raisonnable. Toutefois, la solution souffre encore des problèmes de compatibilité avec le standard HIP, en plus de la complexité de calcul qui reste relativement importante, quand la solution est complètement prise en charge par les nœuds de capteur contraints. Un autre inconvénient réside dans le fait que la compression n'est pas normalisée selon les règles de compression du standard 6LoWPAN.
- Une autre solution de sécurité basée sur Diet-HIP est proposée dans [89]. En plus de HIP-DEX, la solution repose sur un autre processus d'échange de clés léger appelé AMIKEY nommé (*Adapted Multimedia Internet Keying*) [90] pour générer des clés de paires. Il est supposé dans la solution que chaque réseau de capteur (appelé dans la solution un domaine IoT) dispose d'un gestionnaire central pour gérer la sécurité à l'intérieur du réseau. Ainsi, chaque dispositif intelligent dans le domaine IoT établit une session sécurisée avec le gestionnaire de domaine en utilisant HIP-DEX. Le mécanisme AMIKEY permet aux capteurs connectés de mettre en place des liens de sécurité de bout-en-bout entre eux. En effet, le coût de la communication peut être important car le capteur connecté à Internet doit échanger plusieurs messages de signalisation de tailles importantes. En outre, l'adoption des deux mécanismes de sécurité (HIP-DEX et AMIKEY), peut être coûteuse en traitements pour les capteurs.

B.2. les solutions de sécurité de bout-en-bout au niveau de la couche transport

Suivant que le protocole de transport utilisé dans le coté des réseaux de capteurs soit UDP ou TCP, deux types de protocoles peuvent être envisagés pour assurer la sécurité de bout-en-bout sur Internet. On parle alors du protocole TLS (*Transport Layer Security*) et DTLS (*Datagram Transport Layer Security).*

- **Les solutions basées sur TLS :**

TLS [65] est le protocole de sécurité le plus utilisé sur Internet. Il est spécialement conçu pour fonctionner sur des services de transport fiables, comme TCP, et créer une connexion sécurisée entre un client et un serveur web. Pour ce faire, TLS spécifie deux couches : TLS Record et TLS Handshake. TLS Record assure la sécurité de la connexion entre le client et le serveur par des outils bien déterminés pour le cryptage de données, comme DES. TLS Record peut également opérer sans chiffrement. Lors de la phase TLS Handshake, le client et le serveur web négocient les algorithmes de sécurité, ainsi que l'ensemble de clés à utiliser. La négociation est réalisée par l'échange de messages *ClientHello*, *ServerHello* et *ClientKeyExchange.*

Même si TLS répond efficacement aux besoins de sécurité il comprend des opérations cryptographiques asymétriques couteuses tant en temps de calcul qu'en énergie, ce qui fait que son application pour les réseaux de capteurs qui utilise le plus souvent UDP dans la couche transport, soit un grand défi que certaines solutions ont essayé de surmonter.

- Les auteurs dans [91] étaient parmi les premiers à avoir proposé TLS pour sécuriser les communications entre les hôtes sur Internet et les nœuds capteurs dans le cadre des applications médicales. La solution est nommée Tiny 3-TLS. La solution met l'accent sur l'adaptation d'établissement de sécurité de TLS standard de sorte qu'il peut être supporté par les réseaux de capteurs connectés à Internet via un proxy. Le principe de l'adaptation proposée consiste en l'introduction d'un tiers (représenté par la station de base) entre le nœud capteur terminal et l'hôte distant. Cette passerelle sécurisée aide les nœuds capteurs contraints à établir des connexions TLS avec des entités externes sur Internet. La solution accepte deux scénarios possibles selon que la passerelle est considérée comme un nœud partiellement ou entièrement de confiance. Selon le premier cas, l'entité assistante (la passerelle) va seulement aider les deux parties communicantes à mettre en place un secret partagé qui lui est inconnu. Dans le deuxième scénario la passerelle participe à la procédure d'établissement de la connexion de sécurité et partage même le secret de la session avec le nœud capteur et le client externe. En outre, Tiny 3-TLS utilise la cryptographie par courbes elliptiques ECC pour des clés de tailles assez réduites et des coûts de traitements. Les auteurs affirment que leur solution offre une sécurité de bout-en-bout suffisamment efficace, dans ses deux versions, tout en déchargeant les capteurs contraints des tâches sécuritaires les plus compliquées dans TLS. Toutefois, le coût relatif à la communication de plusieurs messages volumineux de TLS dans le coté du RCSF serait considérable. Dans un autre côté, la nature centralisée de la solution (l'accréditation d'une

passerelle intermédiaire unique) conduit à une importante source de faiblesse et de problèmes de tolérance aux pannes.

- Dans [82] les auteurs proposent une solution pour permettre des communications sécurisées de bout-en-bout avec les nœuds capteurs dans l'IoT. La proposition est basée sur l'introduction d'un ensemble de nœuds puissants dans le coté du RCSF auxquels on affecte des taches de calculs cryptographiques dans le processus d'établissement de sécurité de TLS. Le schéma de distribution est très semblable aux schémas de distribution de la charge sécuritaire des protocoles IKE et HIP, ils se sont d'ailleurs proposé par les mêmes auteurs. distribution se compose d'un transport clé en collaboration et en accord la négociation TLS. Donc les solutions partagent les mêmes avantages (principalement l'abaissement du temps de calculs) et les inconvénients (essentiellement la surcharge du réseau et surcout des communications nécessaires).

A ce stade, il est nécessaire de mentionner que le problème commun des solutions basées sur TLS réside dans le fait que TLS et tout autre protocole qui gère la fiabilité des communications et de la livraison en séquence des paquets, ne convient pas pour les environnements peu fiables et très limitées en ressources, comme les réseaux de capteurs sans fil.

- **Les solutions basées sur DTLS**

DTLS (*Datagram Transport Layer Security*) [92] est une variante du protocole TLS basée sur le protocole UDP et destiné à fournir une protection de bout-en-bout pour les communications fondées sur le service de transport orienté datagramme. Comme UDP est plus approprié pour les environnements capteurs que TCP, et puisque le protocole CoAP lui-même repose sur UDP, de nombreuses solutions ont privilégié DTLS pour la sécurité de bout-en-bout dans l'Internet des objets. Néanmoins, DTLS a le même ordre de complexité comme TLS ; il exige l'échange d'une quantité importante de messages de signalisation, en plus de la cryptographie asymétrique qui est nécessaire durant le processus d'établissement de la liaison de sécurité entre les dispositifs communicants sur Internet. Donc, DTLS doit être d'abord adapté aux contraintes des réseaux de capteurs avant son application aux capteurs.

- Dans [93] les auteurs proposent une adaptation du protocole DTLS pour l'IoT où le principe est d'utiliser des capteurs qui sont équipés d'un module d'accélération matérielle des opérations cryptographiques RSA. La solution proposée permet donc d'appliquer le protocole DTLS tel qu'il est sur les nœuds capteurs, tout en minimisant la surcharge et le temps écoulé pour effectuer les calculs cryptographiques induits. Cependant, comme la solution est purement matérielle, la solution proposée augmente le coût des nœuds capteurs. Un autre inconvénient de la solution réside dans le fait qu'elle ne prend pas en considération la contrainte d'énergie des nœuds capteurs ; elle se concentre uniquement sur l'accélération des traitements et ignore complètement la consommation d'énergie nécessaire pour la communication et la réalisation de tels traitements, qui est pourtant un facteur limitatif très intéressant.
- Dans [94-95], les auteurs proposent un modèle de compression 6LoWPAN pour les en-têtes des messages DTLS. Les sous-protocoles concernés par la compression sont le protocole de base et

les messages ClientHello et ServerHello dans le processus d'établissement de session de sécurité. Le scénario de la communication suppose que même les hôtes de l'Internet qui communiquent avec les nœuds capteurs utilisent DTLS. La solution proposée permet de réduire considérablement les coûts de communication en termes de dissipation de l'énergie et de surcharge du réseau. Mais, le coût de calcul des primitives cryptographiques asymétriques reste très important sur les nœuds capteurs extrêmes.

- En outre, le scénario supposé se produit rarement, car les hôtes ordinaires de l'Internet utilisent généralement TLS. En suivant cette orientation, les auteurs en [96] proposent une solution qui utilise DTLS pour sécuriser les communications de bout en bout entre un nœud capteur et un hôte Internet, tout en protégeant le réseau 6LoWPAN contre les attaques par déni de service, qui pourraient être déclenchées depuis l'extérieur par des hôtes malveillants qui tentent de surcharger le nœud capteur en le forçant à ouvrir un nombre important de sessions DTLS. cela conduit à un débordement de mémoire et une consommation excessive des ressources énergétiques, provoquant un déni de service. Cette protection est matérialisée par la mise en place d'un mécanisme pour l'authentification de l'hôte externe qui est implémenté au niveau de la station de base (le routeur de bordure). Comme DTLS est moins déployés sur Internet, les auteurs supposent que TLS est utilisé par les hôtes externes non-capteurs sur Internet, et comme la solution proposée implique DTLS dans le côté du réseau 6LoWPAN, la station de base devrait effectuer la translation entre les deux protocoles (TLS et DTLS). La solution définit les règles à respecter par la station de base pour réaliser la traduction des messages TLS en des messages DTLS et vice-versa. Bien que la délégation de la tâche de traduction à la station de base qui est supposée être largement puissante, et qui est censé être un centre de confiance pour les nœuds capteurs, semble avantageuse car elle décharge les nœuds capteurs de la traduction entre les deux protocoles de sécurité. Les capteurs contraints doivent communiquer les messages DTLS longs et accepter sa complexité de calcul, ce qui est un inconvénient majeur de la solution.
- Dans [97] est proposée une collection de solutions d'adaptation de la sécurité du protocole DTLS. En plus de la compression de l'en-tête proposé dans [94], la solution supporte une technique de reprise de session, et les certificats basés ECC pour l'authentification efficace de la procédure d'établissement de la clé de l'association DTLS. La solution supporte également un outil d'accélération matérielle des traitements pour faire face à la complexité des calculs de la cryptographie à clé publique au niveau des nœuds capteurs. Donc, la famille de solutions proposée favorise l'adaptabilité de la sécurité DTLS pour l'IoT au moyen du regroupement des techniques logicielles et matérielles.

L'utilisation du protocole DTLS pour la sécurité de bout-en-bout des communications qui tournent entre les nœuds capteurs et le reste des hôtes (capteurs ou non capteurs) dans l'Internet du futur peut être considérée comme une mauvaise solution pour les raisons suivantes :

- le protocole DTLS ne supporte pas les communications multicast qui sont une partie clé dans le protocole CoAP et l'IoT en général.

- DTLS nécessite l'échange d'un nombre important (environ 15) de messages de signalisation afin qu'une session de sécurité puisse être établie.
- DTLS est moins utilisé sur Internet et son adoption pour la sécurité des communications humain-à-objet n'est pas pratique car elle nécessite l'intervention d'une troisième entité qui assure la traduction entre TLS et DTLS ce qui viole le principe de la sécurité de bout-en-bout.
- le protocole de négociation DTLS est vulnérable aux attaques d'épuisement de ressources énergétiques des nœuds capteurs, qui peuvent par exemple être lancées par envoi massif de plusieurs messages *Client Hello*.

B.3. les solutions de sécurité de bout-en-bout au niveau de la couche application

On ne peut parler de la sécurité de bout-en-bout au niveau de la couche application que si les deux nœuds communicants adoptent le même protocole applicatif (CoAP ou MQTT) et cela ne se réalise que dans le cas des communications inter-capteurs connectés à Internet ; les communications de machine-à-machine de type objet-à-objet. Notons que dans le cas d'une communication M2M de type humain-à-objet entre un capteur et un hôte ordinaire il sera pratiquement difficile d'appliquer la sécurité de bout-en-bout dans la couche application.

Puisque la sécurité du protocole MQTT se focalise essentiellement sur TLS, nous allons nous intéresser dans cette partie, à la sécurité du protocole CoAP. Dans ce contexte, les spécifications de CoAP proposent deux protocoles pour assurer les services de sécurité. Ces protocoles sont : le protocole DTLS au niveau de la couche transport ou le protocole IPsec au niveau IP. D'autres spécifications ont été proposées pour instaurer la sécurité au niveau du protocole CoAP pour une moindre surcharge.

La solution proposée dans [98]:

C'est une solution concentrée au niveau de la couche application et qui spécifie un ensemble d'options à employer pour fournir l'authentification, l'intégrité et le chiffrement des messages CoAP. La solution permet également de prévenir l'attaque par rejeu de messages. Le concept de base repose sur l'ajout de trois options sécuritaires supplémentaires au protocole CoAP:

L'option *CryptoInitiate*

CryptoInitiate est une option critique, utilisée pour la création d'un contexte cryptographique entre le client et le serveur CoAP. Dans le cas où le client CoAP veut mettre en place un contexte de sécurité utilisable avec l'option *CryptoEncap*, il émet au serveur un message confirmable contenant l'option *CryptoInitiate* avec les paramètres suivants :

- un Contexte ID spécifiant la valeur de l'identificateur attribué au contexte par le client.
- un identificateur de la clé secrète partagée qui sera utilisée dans le contexte.
- le paramètre *CryptoAlgos* dans le champ Options Valeur contenant une liste d'un ou plusieurs algorithmes cryptographiques proposés et supportés par le client.

Lorsque le serveur CoAP (un nœud capteur) reçoit le message, il vérifie d'abord les paramètres de l'option. Si l'ID du contexte n'est pas déjà utilisé pour un contexte précédemment initialisé par le serveur, et qu'au moins l'un des algorithmes cryptographiques proposés est supporté par le serveur pour assurer les services d'authentification, intégrité, confidentialité …, et l'identificateur de la clé est reconnu, alors le serveur doit renvoyer un message d'accusé de réception contenant l'option *CryptoInitiate* et une option *CryptoAlgos* indiquant les algorithmes de sécurité sélectionnés parmi la liste des algorithmes proposés.

- **L'option *CryptoEncap***

Il s'agit d'une option critique utilisée pour sécuriser les messages qui la comportent. Elle est utilisée avec un contexte cryptographique déjà initialisé par le client. L'option sert donc à repérer les messages concernés par le contexte de sécurité préalablement négocié à l'aide de l'option *CryptoInitiate*.

- **L'option *CryptoTerminate***

Cette option est utilisée pour supprimer un contexte cryptographique partagé entre le client et le serveur CoAP. Lorsqu'un client CoAP veut supprimer un tel contexte, il émet une demande confirmable contenant l'option *CryptoTerminate*.

La solution présente plusieurs inconvénients résumés dans les points suivants:

- Un attaquant peut facilement provoquer une attaque de type déni de service DoS par l'envoie successive des messages d'initialisation contenant l'option *cryptoInitiate*.
- Son principe est basé sur le choix entre plusieurs algorithmes de chiffrement et d'authentification, que l'espace mémoire limité des nœuds capteurs ne suffirait pas pour les contenir tous.
- Les capacités du serveur ne permettent pas le stockage des contextes d'un nombre important de clients.

La solution proposée dans [99]:

Une solution située au niveau de la couche application pour la sécurité des communications objet-à-objet entre les dispositifs CoAP. Comme les dispositifs ont les mêmes contraintes et sont équivalents en termes de disponibilité de ressources, il n'est pas nécessaire d'opter pour les solutions de sécurité des couches basses qui sont assez onéreuses.

La solution révoque la procédure de négociation réactive du matériel sécuritaire (choix des algorithmes cryptographiques à utiliser pour le cryptage et la signature, l'établissement des clés, etc.) entre les nœuds CoAP et la remplacer par une pré-négociation du contexte sécuritaire. Les nœuds CoAP appartenant au même réseau 6LoWPAN se mettent d'accord sur des algorithmes cryptographiques qui vérifie le compromis entre la robustesse sécuritaire et le coût en matière de consommation de ressources (exemple : algorithmes de chiffrement symétrique ultra légers, …). Ainsi,

les clés à utiliser sont dans ce cas pré-chargées dans les nœuds capteurs et elles sont périodiquement rafraîchies. Pour l'établissement du lien de sécurité entre les dispositifs CoAP communicants résidant dans différents réseaux 6LoWPANs, l'utilisation de l'algorithme *Diffie-Hellman* à base de courbes elliptiques (ECDH) est suggérée.

Le tableau ci-dessous compare les solutions de sécurité de bout-en-bout des communications avec les nœuds capteurs.

Table 5.1. Comparaison générale entre les solutions de sécurité de bout-en-bout.

Solution	Protocol de base	Couche opérationnelle	Technique d'adaptation principale	style d'adaptation	Coût de d'adaptation	Coût de communication	Coût de calculs	Translation dans le 6BR	Taux des msgs de signalisation	Protection contre les attaques DoS	Tolérance aux fautes
[91]	TLS	Transport	Délégation des calculs	Logicielle	_ _	+	_	x	+	Serveur	_ _
[82]	IKE & TLS	Application & transport	Distribution des calculs	Logicielle	_ _	++	_ _		++	Serveur	_
[93]	DTLS	Transport	Accélération de la sécurité	Matérielle	++	+	_	x	+	Serveur	
[94-95]	DTLS	Transport	Compression	Logicielle	_ _	_	++	x	+	Serveur	
[96]	DTLS	Transport	Translation des messages TLS/ DTLS	Logicielle	_ _	+	++	xx	+	Serveur & RCSF	_ _
[97]	DTLS	Transport	compression & accélération des calculs	Logicielle & matérielle	+	_	_	x	+	Serveur	
[79-80]	IPsec & IKE	Application & réseau	Compression	Logicielle	_ _	_	++		+	Serveur	
[81]	IKE	Application	Délégation des calculs	Logicielle	_ _	+	_ _		+	Serveur	_ _
[85]	HIP	Couche HIP	Adoption de l'ECC	Logicielle	_ _	+	+	x	_	Serveur	
[86]	HIP	Couche HIP	Délégation des calculs	Logicielle	_ _	+	_ _		_	Non supporté	
[87]	HIP	Couche HIP	Distribution des calculs	Logicielle	_ _	++	_ _		+	Serveur	_
[88]	HIP	Couche HIP	Compression	Logicielle	_ _	_	+	x	_	Serveur	_
[89]	HIP	Couche HIP	Légère gestion de clés	Logicielle	+	++	+		+	Serveur	_
[98]	CoAP	Application				_	+		_		
[99]	CoAP	Application	Deux niveaux de sécurité	Logicielle	_	++	++		++		_

Où +, ++, _, _ _, x, xx représentent respectivement: important, très important, faible, très faible, peut être nécessaire, nécessaire.

C. Solutions pour le contrôle d'accès aux services des RCSFs dans l'IoT

L'authentification est une technique sécuritaire indispensable, c'est même la première mesure à entreprendre par les entités avant d'engager les communications. Dans l'IoT, les données échangées sont parfois extrêmement critiques, l'authentification de données et des dispositifs devient plus que

nécessaire. L'objectif étant d'interdire toute malveillance, ayant rapport avec l'usurpation des identités, ainsi, de n'autoriser que les entités qui font preuve d'authenticité, à accéder les services de l'IoT. Le processus d'authentification mutuelle est souvent réalisé en compagnie avec la phase d'échange des clés et d'établissement de session de sécurité, définie dans les différents protocoles de sécurité de bout en bout (IPsec, HIP et DTLS).

- Les auteurs dans [100], proposent d'améliorer l'adaptabilité du mécanisme d'authentification du DTLS, en proposant comme idée la pré-validation des certificats par la station de base. Celle-ci n'autorise que les messages dont les certificats sont valides, aux nœuds capteurs. Ceci permet davantage de réduire la charge introduite par les messages relatifs aux tentatives d'initialisations de connexions indésirables. Cependant, une station de base malicieuse risque de corrompre tout le réseau d'objets par des certificats falsifiés.

- Dans [101], les auteurs définissent un nouveau schéma pour l'authentification lors de l'échange de clés, utilisant la cryptographie à base de courbes elliptiques. Et un mécanisme de control d'accès inspiré de la méthode RBAC (*Role Based Access Control*) de contrôle d'accès à base de rôles. L'accès d'un utilisateur appartenant à une organisation à des services ou ressources, est contrôlé par le rôle qu'il est autorisé à jouer.

D. Sécurité de données de l'IoT au niveau cloud

Les utilisateurs des capteurs connectés à l'IoT ont besoin de connaitre si leurs informations sont sécurisées dans les centres de données cloud et éventuellement les parties qui gèrent cette sécurité (le fournisseur de services cloud ou les utilisateurs eux-mêmes) et qui utilisent les données et pour quelle raison. Plusieurs considérations sécuritaires doivent être prises pour sécuriser les données provenant de capteurs connectés à Internet et protéger la vie privée des utilisateurs [102]. Les considérations les plus importantes sont résumées dans les points suivants :

- La sécurité des interactions entre les nœuds capteurs et les services cloud. TLS/DTLS sont utilisés.
- La sécurité des données sur le cloud (cryptage).
- Le contrôle d'accès aux données sur le cloud (contrôle contextuel ou à base de rôles).
- Classification des données selon leur degré de criticité et stocker les données les plus critiques des clouds privés et les données moins critiques dans des clouds publiques.
- Gestion des identités des capteurs et anonymats.
- La gestion de confiance et des certificats avec le fournisseur et les services cloud.
- ….

- Très peu de solutions adressent la sécurité dans le cloud, au profit des données de l'IoT. Dans [103] une solution nommée *SensorCloud* qui utilise TLS pour la sécurité des opérations de téléchargement via HTTPS. La plate-forme fournit également des mécanismes de gestion des autorisations d'accès aux données sensibles stockées sur le cloud, d'une application par exemple médicale. Toutes les données de captage sont privées par défaut, et les propriétaires de données

ont la possibilité d'envoyer des invitations à d'autres utilisateurs pour les faire impliquer dans l'application, par exemple pour aider à l'analyse et le traitement de données. SensaTrack [104] est une autre solution qui offre des mécanismes pour la mise en place des comptes d'utilisateurs et des privilèges d'accès sécurisé. La solution supporte quelques passerelles qui utilisent le service de réseaux privés virtuels (VPN)) du protocole IPSec pour protéger les interactions avec les serveurs cloud. La confidentialité, l'intégrité et l'authentification des communications de bout-en-bout avec ces serveurs sont donc bien assurées.

E. La détection d'intrusion logique

Par l'ouverture des RCSFs à l'internet, il devient plus facile d'attaquer les nœuds capteurs, comme l'attaquant du côté de l'Internet peut subvertir le réseau à distance, où qu'il soit, sans avoir besoin d'y accéder physiquement. Ce qui représente une véritable contrainte de l'intégration des RCSFs dans l'IoT.

Donc, il est nécessaire de protéger le 6LoWPAN contre les intrusions externes, déclenchées par les hôtes de l'internet. Ceci est concrétisé par l'insertion des mécanismes de filtrage de trafic entrant au 6LoWPAN, dans le routeur 6BR. Dans [69], un module qui réside dans le 6BR, responsable de l'analyse des paquets de l'internet. Le module est composé d'un détecteur d'anomalie et d'un classificateur des modèles d'attaques. Le détecteur d'anomalie tente de détecter le trafic malicieux destiné au réseau de capteurs, alors que le classificateur présente une classification des attaques par type comme : l'attaque *IPspoofing* et *Ping of death* et plein d'autres. Les interactions entre ce module et le module d'analyse des paquets en internes du 6LoWPAN sont vivement recommandées, pour pouvoir signaler et bloquer les nœuds malicieux externes qui réussissent, quand même, à pénétrer le mur de défense contre les attaques externes.

Dans [71], les auteurs proposent également, que le 6BR soit doté d'un mini pare-feu pour le filtrage du trafic. Et les hôtes de l'Internet et les nœuds capteurs, sont concernés par l'opération de filtrage, afin de prévenir un nœud capteur malicieux, de demander au 6BR de filtrer le trafic que génère un hôte légitime pour l'en interdire l'accès à tous les nœuds du 6LoWPAN.

6. RÉCAPITULATION

Le schéma ci-dessous récapitule les classes de solutions sécuritaires étudiées.

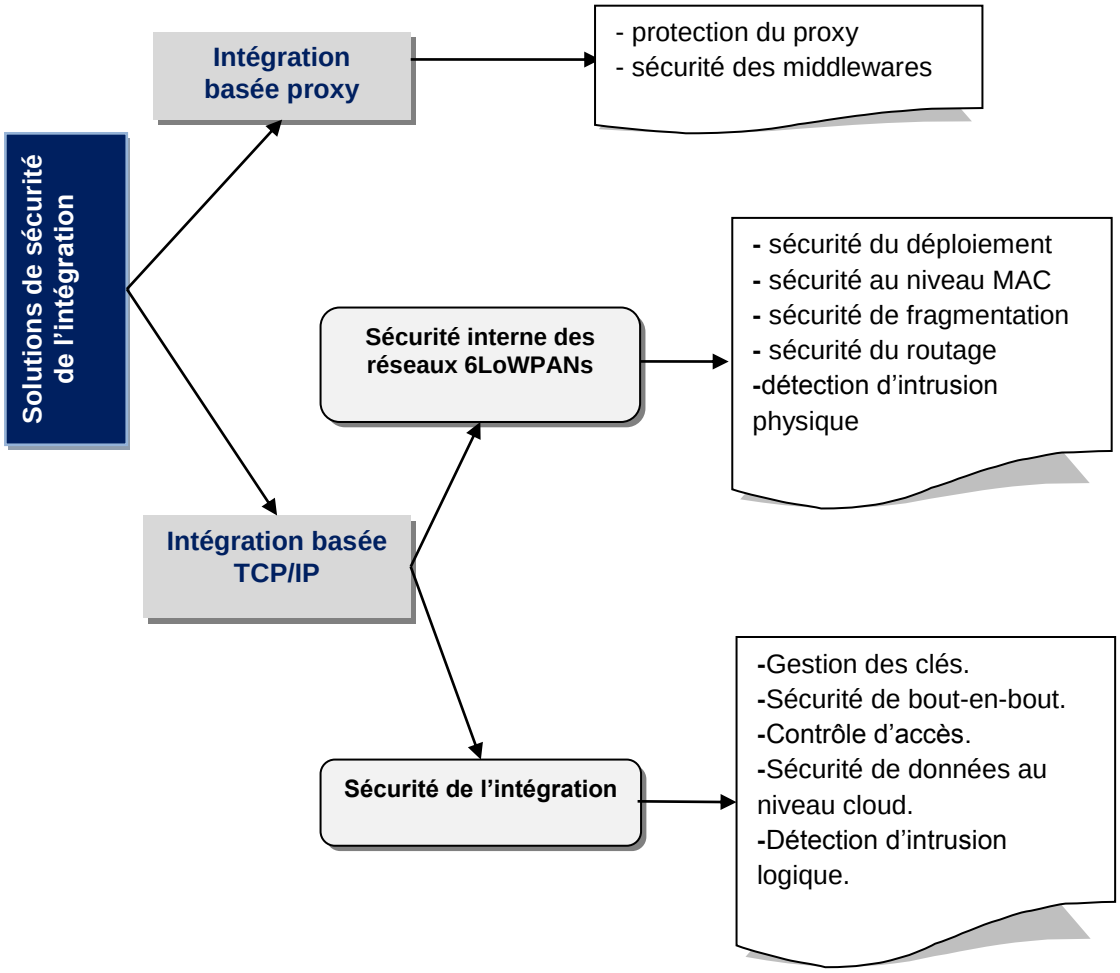


Figure 5.11. Schéma récapitulatif des classes de solutions de sécurité de l'intégration.

7. Conclusion

Dans ce chapitre, nous avons présenté un état de l'art sur les solutions récentes adressant la problématique de sécurité de l'intégration des réseaux de capteurs à l'Internet des objets. Nous avons commencé d'abord par une étude des vulnérabilités de l'intégration ainsi que les menaces susceptibles de cibler les réseaux de capteurs connectés à Internet. Finalement, nous avons récapitulé l'étude par une comparaison générale entre les différentes solutions.

Le chapitre suivant sera consacré à la présentation des solutions que nous proposons dans le cadre de la protection des réseaux de capteurs intégrés à l'Internet comme une partie de l'Internet des objets.

PARTIE 2 :

Contribution

CHAPITRE 6:

Les solutions proposées

1. Introduction

Dans cette thèse, nous nous focalisons sur la sécurité de l'intégration des réseaux de capteurs à l'Internet des objets. Dans le présent chapitre, nous présentons les solutions proposées pour répondre à cette question. En effet, nos propositions sont concentrées autour des réseaux de capteurs sans fil intégrés à l'IoT via l'adoption des standards basés-IP (les réseaux 6LoWPANs). Le motif principal est dérivé du fait que cette approche soit le meilleur choix pour répondre efficacement aux perspectives applicatives de l'IoT surtout en ce qui concerne la flexibilité de l'incorporation des nœuds capteurs à Internet et l'ubiquité de l'accès à leurs données.

2. Solution pour assurer la sécurité de bout-en-bout dans l'IoT

L'hétérogénéité technologique et matérielle dans l'Internet des objets empêche la généralisation des solutions de sécurité qui sont déjà approuvées dans l'Internet de nos jours, pour englober tous les réseaux qui composent l'IoT, tels que les réseaux de capteurs. Dans cette thèse, nous proposons deux solutions efficaces pour assurer la sécurité de bout-en-bout des communications entre les capteurs connectés à l'IoT et les hôtes ordinaires de l'Internet, tout en prenant en considération les différences en termes de capacités et de natures des réseaux auxquels appartiennent les dispositifs communicants.

Les solutions déjà proposées dans la littérature et qui sont destinées à assurer la sécurité de bout-en-bout des communications entre les nœuds capteurs et le reste des hôtes sur Internet ne sont pas bien adaptées aux contraintes des réseaux de capteurs car d'une part, elles se concentrent uniquement sur la compression des messages pour économiser et l'énergie de communication et l'espace mémoire occupé au niveau du nœud capteur terminal. Et dans très peu de solutions, l'adaptation est plutôt réalisée par la délégation ou la distribution de toute la charge sécuritaire liée aux opérations cryptographiques nécessaire pour l'établissement de la liaison de sécurité, en vue de décharger le nœud capteur des calculs lourds et coûteux en énergie. D'autre part telles solutions se focalisent dans certains cas sur des protocoles de sécurité qui reste quand même assez couteux pour être supportés par des hôtes contraints comme les nœuds capteurs dans l'Internet du futur et/ou qui ne peuvent être appliqués que pour un type particulier de communications (comme DTLS qui ne convient pas pour sécuriser les communications de type humain-à-objet).

La solution que nous proposons considère le protocole *Host Identity Protocol* (HIP) comme plateforme de sécurité de bout-en-bout et lui apporte deux techniques d'adaptation (la compression des messages avec la distribution de la charge cryptographique). Les deux techniques sont combinées dans HIP pour une adaptation optimale aux réseaux de capteurs.

2.1. Expression des motivations et de la problématique

Comme déjà révélé, la solution que nous proposons cible le protocole HIP qui prépare le matériel sécuritaire (principalement la clé secrète de session) que va utiliser par la suite le protocole IPsec (spécialement ESP) pour la protection de bout-en-bout des messages échangés. La sélection de HIP/IPsec pour la sécurité de bout-en-bout dans l'Internet des objets est motivée par :

- Avec le découplage des informations d'identification des informations d'adressage, HIP facilite d'une manière inhérente la mobilité des hôtes et ainsi, assure un bon niveau de sécurité à la localisation des utilisateurs, chose qui est fortement recommandée dans de nombreuses applications de l'IoT, à titre d'exemple : les applications médicales et militaires. Ces caractéristiques bénéfiques ne sont pas définies dans les autres protocoles de sécurité basés IP, comme DTLS et IKE qui reposent pour l'identification des pairs communicants sur les adresses IP correspondantes qui informent explicitement sur les emplacements physiques.
- HIP fournit un mécanisme flexible pour la gestion de clé flexible qui exige une légère quantité de messages à échanger entre les pairs (seulement quatre messages). Cela n'est pas le cas avec d'autres protocoles (tel que DTLS) qui peuvent nécessiter l'échange de plus de dix messages de signalisation dans la phase de négociation de sécurité.
- L'adoption du protocole HIP pour sécuriser les applications des réseaux de capteurs a attiré récemment plus d'attention [105-106].
- L'activation de la sécurité de bout-en-bout des données dans la couche réseau par le biais du protocole IPsec (que ça soit il est couplé avec HIP ou IKE) permet de sécuriser tous les types de trafic transporté. En d'autres termes, IPsec peut être utilisé pour sécuriser les applications qui sont fondées sur TCP ou UDP sans avoir besoin de traductions indésirables entre les protocoles de sécurité incompatibles au niveau de la passerelle.

Malgré ses nombreux avantages, le protocole HIP n'est pas adapté, tel qu'il est, pour fonctionner sur des environnements capteurs, car il a été initialement conçu pour les réseaux IP non soumis aux contraintes de disponibilité de ressources. En effet, la communication des messages longs et le calcul des opérations cryptographiques asymétriques onéreuses pour établir l'association de sécurité entre pairs HIP, présentent les principaux obstacles de l'extension du protocole HIP pour les réseaux de capteurs qui sont reconnus par leurs limitations en termes de ressources d'énergie, de stockage et de traitement. Par conséquent, la mise en place des mécanismes d'adaptation efficaces pour le protocole HIP s'est avérée nécessaire, pour pouvoir l'utiliser sans risques sur le coté des réseaux 6LoWPANs dans l'Internet.

En fait, toutes les solutions proposées pour adapter HIP (même les autres solutions non-HIP), étudiées dans le chapitre précédent, se basent idéalement soit sur la compression des messages ou sur la distribution et la délégation de la charge sécuritaire introduite par le mécanisme HIP-BEX afin d'atténuer le coût et la surcharge de la communication ou de la complexité de la sécurité, respectivement. Par conséquent, ces solutions sont insuffisantes car elles ne fournissent pas une

adaptabilité optimale et une considération complète de la faiblesse de ressources des nœuds capteurs.

Afin de remédier à ce problème, nous proposons deux techniques d'adaptation pour le protocole HIP: un modèle de compression 6LoWPAN pour l'entête HIP, ainsi qu'un système de distribution efficace et sécurisé de des tâches sécuritaires dans le processus HIP-BEX. Ces deux mécanismes sont en outre combinés ensemble pour une bonne adaptabilité du protocole HIP aux réseaux 6LoWPANs.

2.2. La solution proposée pour la sécurité de bout-en-bout à base du protocole HIP dans l'IoT (CD-HIP)

Dans cette section, nous détaillons nos propositions qui impliquent un modèle optimal de compression 6LoWPAN pour l'en-tête du protocole HIP ainsi qu'un système de distribution efficace des primitives sécuritaires nécessaires dans HIP-BEX. Pour une sécurité de bout-en-bout extrêmement adaptée aux RCSFs, nous proposons de combiner les deux mécanismes pour le protocole HIP quand il s'exécute dans le côté des réseaux de capteurs, dans l'Internet des objets. La variante résultante du protocole HIP est nommée CD-HIP pour HIP compressé et distribué [107-108].

En effet, lors de la conception des solutions de sécurité de bout-en-bout dans l'Internet des objets, une attention particulière doit être prise pour la considération des contraintes des nœuds capteurs avec très bon niveau de sécurité nœuds capteurs eux-mêmes, et leurs communications avec les hôtes externes. En d'autres termes, un compromis entre la sécurité et les coûts induits (taux de dissipation d'énergie, empreinte mémoire, surcharge du réseau, etc.) devrait être réalisé. En outre, des modifications qui pourraient être effectuées sur les standards de sécurité basés sur le modèle TCP/IP (HIP dans notre cas), afin de les rendre capables de fonctionner sur des environnements contraints comme les réseaux de capteurs, ne devrait pas affecter la compatibilité avec les protocoles originaux qui sont toujours opérationnels dans le coté de l'Internet. Cela permettra davantage l'évitement des traductions entre les différents protocoles par un proxy intermédiaire, ce qui présente une contradiction du principe de la sécurité de bout-en-bout. Donc, les trois principales exigences auxquelles doit répondre une solution de sécurité de bout-en-bout dans l'Internet des objets sont:

- La robustesse de la sécurité de bout-en-bout.
- La considération des contraintes des réseaux de capteurs.
- La compatibilité avec les standards de sécurité basés sur IP.

2.2.1. Le modèle de compression 6LoWPAN proposé pour l'entête HIP

Rappelons que la compression est une technique d'optimisation très importante. Actuellement dans l'Internet des objets, cette technique est indispensable pour surmonter les grandes différences dans les capacités de réseaux IP (IPv4 ou IPv6) et les réseaux de capteurs, notamment en termes de capacité d'admission des unités de données dans les deux réseaux (minimum MTU de 1280 octets

dans les réseaux IPv6 et seulement 127 octets dans les réseaux 6LoWPANs utilisant la technologie IEEE 802.15.4). La technique de compression des messages présente bien évidemment plusieurs avantages parmi:

- Coût minime de communication: les paquets compressés ont une longueur réduite et une plus faible quantité d'informations sera communiquée.
- Faible consommation d'énergie pour la communication des messages compressés.
- Courts délais de communication.
- Faible débit de fragmentation: les tailles réduites des paquets compressés aident à atténuer la fréquence de fragmentation.
- Minimisation du besoin en espace de stockage.
- Optimisation du débit effectif: par la révocation des informations de contrôle non nécessaires dans l'en-tête des paquets, ce qui améliore le débit car un espace important dans le message sera libre pour contenir plus de données applicatives.

La première partie de notre solution représente une proposition de la première extension de la compression 6LoWPAN pour l'entête HIP. Dans cette section, nous détaillons la solution, mais avant de le faire, nous discutons d'abord la compressibilité de l'entête des messages HIP.

A. Etude de la compressibilité de l'entête HIP

Le protocole HIP définit huit types de messages, quatre d'entre eux (I1, R1, I2, R2) sont nécessaires pour le processus HIP-BEX. Les messages HIP restants (UPDATE, NOTIFY, CLOSE, CLOSE_ACK) sont utilisés à des fins diverses. Le message UPDATE est utilisé pour mettre à jour des informations relatives à la session HIP (par exemple, on peut imaginer un message de mise à jour portant des informations que le pair HIP doit combiner avec la clé de session courante afin de la rafraichir). De plus, un hôte HIP envoie le message de notification (NOTIFY) qui est facultatif pour signaler les erreurs de fonctionnement du protocole ou l'échec de la négociation du secret entre les deux pairs HIP dans le processus HIP-BEX. Ce message porte des paramètres qui donnent plus de détails sur le problème signalé. Et pour annoncer la clôture de la session, un pair HIP envoie un message CLOSE. A la réception du message de fermeture, hôte HIP répond par un autre message qui en accuse réception, on parle bien du message CLOSE_ACK) et il ferme immédiatement la session en question.

Tous les paquets HIP ont un entête fixe de 40 octets. Le format de l'en-tête HIP est illustré dans la figure suivante.

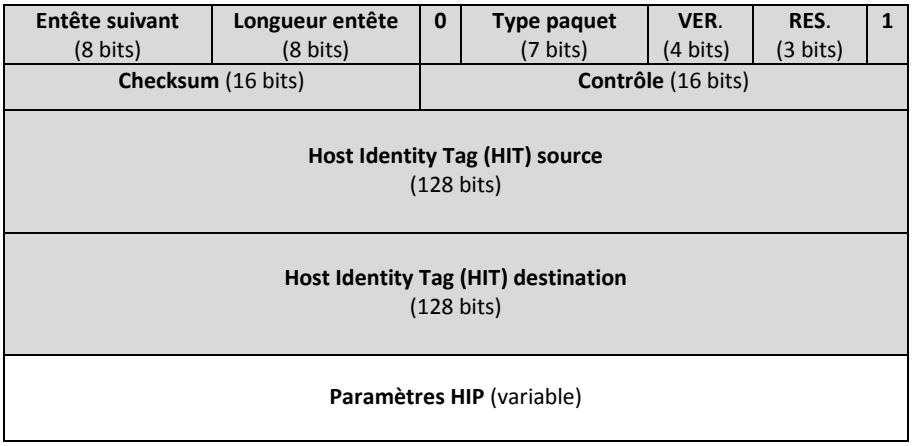


Figure 6.1. L'entête fixe (champs en gris) des messages HIP.

L'en-tête des messages HIP a une taille pratiquement importante et contient beaucoup d'informations inutiles et redondantes. En effet, le champ longueur de l'en-tête révèle la longueur de l'en-tête (à l'exception des 8 premiers octets) et les paramètres HIP. Cette information est statique pour tout type de message HIP et peut être déduite à partir des couches inférieures, par exemple au niveau de la sous couche MAC, en cumulant les longueurs des trames (précisément, les longueurs des unités de données des trames MAC correspondantes à un même paquet HIP, avec la soustraction des 8 octets exclus. Donc, l'octet de longueur d'en-tête peut être éliminé.

Le champ version du protocole HIP (VER.) peut être également révoqué parce que nous supposons que les pairs HIP utilisent la dernière version stable (actuellement la version 2). Dans le cas de versions différentes, nous pouvons encore supposer que tous les nœuds capteurs dans le réseau 6LoWPAN utilisent la même version du protocole HIP. Dans ce cas, le champ de version peut être compressé en toute sécurité dans tous les paquets sortants (générés par les nœuds de capteurs) et gardé inchangée dans les paquets entrants (arrivant de l'Internet). Dans notre solution, nous considérons la première supposition lorsque les pairs utilisent la même et dernière version stable. En plus, les trois bits réservés (RES.), qui sont réservés pour une future utilisation et valent zéro, peuvent être éliminés.

Ainsi, les datagrammes IPv6 entrants portant les paquets HIP seront compressés, puis fragmentés au niveau de la couche 6LoWPAN. Donc, il n'est plus nécessaire de maintenir la valeur du champ checksum originale qui porte sur le paquet HIP avant sa compression et fragmentation. Par conséquent, le champ *checksum* peut être éliminé de l'entête HIP dans le côté du réseau de capteurs, et son calcul et vérification peuvent être délégués au routeur de bordure 6BR. En plus, la vérification des erreurs de communication de chaque fragment 6LoWPAN est implicitement prise en charge dans le champ FCS (*Frame Check Sequence*) de la trame MAC qui l'encapsule.

Un seul bit (le bit A) est utilisé dans le tableau de bits de contrôle (16 bits). Ce bit indique si l'identificateur de l'hôte est anonyme dans le paquet HIP actuel et s'il doit être conservé ou non par les pairs HIP. Comme nous assistons à étendre HIP à un environnement contraint (les réseaux de capteurs), il est préférable de révoquer la gestion des entités anonymes, afin d'éviter une éventuelle surcharge. Par conséquent, le champ contrôle peut être complètement supprimé. Les deux bits fixes 0

et 1 dans l'en-tête servent à la gestion de la compatibilité et doivent uniquement être définis dans les implémentations adhérant à des spécifications particulières et, par conséquent, ils peuvent être éliminés.

Les champs compressibles restants dans l'en-tête HIP sont le HIT (*Host Identity Tag*) de la source et la destination qui sont tous deux de longueur 128 bits. HIT est une représentation concise de l'identifiant de l'hôte, il a la même longueur qu'une adresse IPv6 ce qui permet aux applications fonctionnant sur IPv6 de continuer à opérer de manière transparente au-dessus de HIP. HIT est composé de trois parties: 1) 28 bits fixes servant à la distinction de HIT d'une adresse IPv6 et, 2) l'indicateur de l'algorithme de génération de HIT (4 bits). 3) les 96 bits restants représentant une valeur de hachage de l'identificateur de l'hôte HIP en utilisant l'algorithme de génération correspondant. Les champs HIT peuvent être réduits à 96 bits, en omettant le préfixe du champ HIT (codé sur 32 bits : 4 bits pour identifier l'algorithme générateur et 28 bits fixes pour distinguer un HIT d'une adresse IPv6) et on garde seulement la valeur de hachage. Le HIT de l'expéditeur peut même être complètement éliminé dans les entêtes des messages R1 et I2 (voir figure 5.10) qui transportent l'identificateur de l'hôte source.

Finalement, le champ type du paquet est toujours gardé avec cinq bits de longueur au lieu de sept bits, comme on a huit types de messages HIP, où la valeur maximale que puisse prendre le champ est 19 (les valeurs de type du paquet dans les messages I1, R1, I2, et R2 sont respectivement 1, 2,3 et 4. Avec UPDATE, NOTIFY, CLOSE et CLOSE_ACK, le champ type de paquet prend les valeurs 16, 17, 18 et 19 respectivement).

B. Le modèle de compression 6LoWPAN proposé

Logiquement, l'entête HIP est considéré comme un entête d'extension IPv6. Par conséquent, nous proposons un modèle de compression 6LoWPAN d'entête d'extension IPv6 pour le protocole HIP. L'octet d'encodage proposé pour l'en-tête HIP compressé est défini dans la figure ci-dessous.

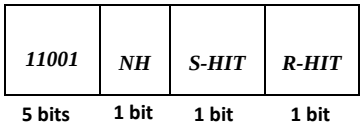


Figure 6.2. L'octet d'encodage de l'entete HIP compressé suivant le modèle 6LoWPAN.

- Les cinq premiers bits représentent l'identificateur de l'entête suivant compressé. Nous définissons sa valeur sur 11001. En fait, ces bites servent à l'identification unique de l'en-tête HIP compressé parmi tous les entêtes existants qui sont déjà compressés dans 6LoWPAN et qui correspondent aux différents protocoles (IPv6, UDP, IPsec, DTLS, IKE). Selon la norme 6LoWPAN [52], il n'y a aucune restriction sur la longueur ou la valeur du champ identificateur dans de l'entête compressé dans l'octet d'encodage. Cependant, nous devrions garantir l'unicité de la valeur pour éviter les possibles confusions d'identification. La séquence 11001 n'est actuellement pas attribuée dans la norme 6LoWPAN (les valeurs d'identité existants sont: 011 et 11110 pour les entêtes compressés IPv6 et UDP respectivement [52], 11011 pour la compression de la charge utile UDP [53]. 1110101 et 1110110 pour les entêtes ESP et AH.

[79]. 1000, 1001, 1010 et 1011 pour les entêtes de DTLS compressé [94]. 1101 pour l'entête compressé du protocole IKE [80]).

- NH (entête suivant): s'il est mis à 0: le champ entête suivant est retiré: le cas où l'entête HIP est l'en-tête final, comme spécifié dans [83-84] et c'est généralement le cas. Sinon, si la valeur du bit vaut 1, le champ entête suivant est maintenu ; il ya un autre entête d'extension ou de données après l'entête HIP, comme précisé dans [109].
- S-HIT (le HIT de l'expéditeur): prend 0 si le champ HIT de la source est réduit à 96 bits. Sinon, s'il prend 1, le champ est totalement éliminé (le cas du message R1 ou I2).
- R-HIT (le HIT de récepteur): toujours mis à 0, pour indiquer que le champ HIT de la destination est réduit à 96 bits.

Le format standard de paquets HIP avec le modèle de compression proposé est présenté dans la figure suivante.

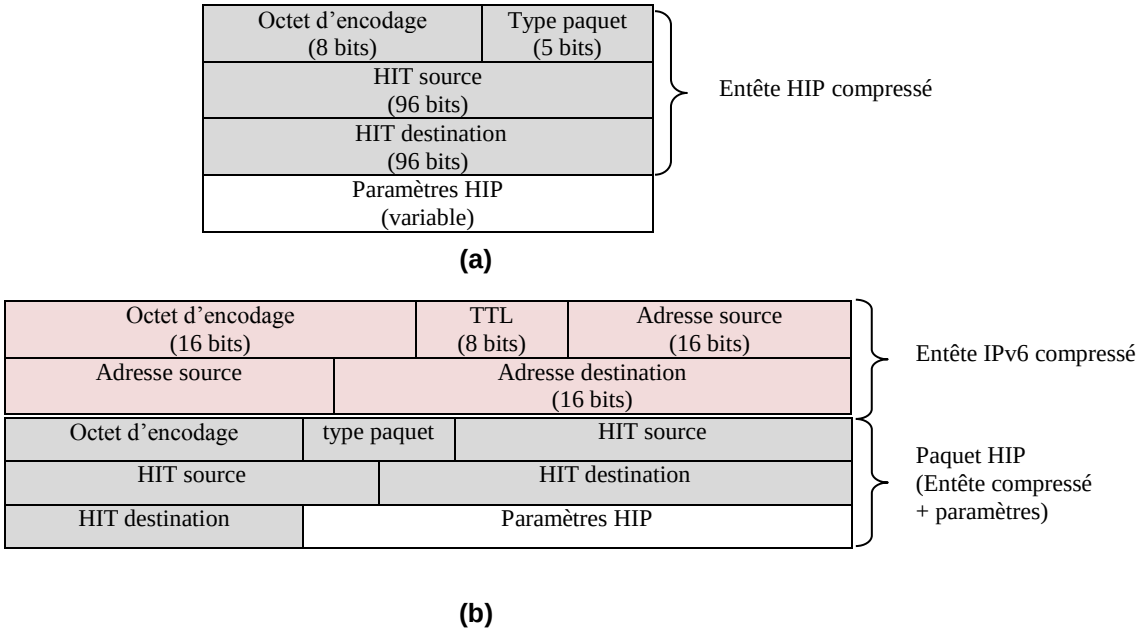


Figure 6.3. Le modèle de compression 6LoWPAN proposé pour l'entête HIP avec **(a)** le paquet HIP et **(b)** le datagramme HIP/IPv6 compressé. L'entête IPv6 est compressé en considérant l'exemple d'une communication locale multisauts entre les deux entités communicantes.

En effet, la compression et la décompression des paquets HIP sont pris en charge par le routeur de bordure. A la réception de chaque paquet HIP d'un hôte externe, le routeur de bordure contrôle dans un premier temps la validité du checksum. Si la somme de contrôle calculée ne correspond pas à celle reçue, le paquet est rejeté. Sinon, si le routeur de bordure constate que le checksum est intègre, il passe le paquet HIP à la couche 6LoWPAN où il supprime les champs et les bits concernés par la compression dans l'entête HIP, exactement comme dicté par le modèle de compression proposé. Après la compression de l'entête, le paquet HIP sera fractionné en plusieurs petits fragments en suivant les règles de fragmentation 6LoWPAN. Enfin, le 6BR communique les fragments du paquet HIP dans le réseau de capteurs vers leur destination finale (le pair HIP).

Inversement, lorsque le routeur 6BR reçoit un paquet HIP compressé typiquement divisé en plusieurs trames, à partir d'un nœud de capteur, il réassemble les fragments correspondant à ce même paquet HIP. Ensuite, il décompresse l'entête HIP, afin de construire l'entête HIP original, non compressé, contenant tous les champs. Systématiquement, le 6BR ajoute le champ longueur de l'entête, et élargit le champ de type de paquet à sa taille originale (7 bits). De plus, le routeur de bordure ajoute les informations de version du protocole (version 2 dans notre cas), les bits réservés et les deux bits fixes (0 et 1). A ce niveau, le 6BR calcule la somme de contrôle comme dicté par la norme [83-84], et met sa valeur dans le champ de checksum (16 bits). Il ajoute également le tableau contrôles avec le bit A mis à 0 parce que la gestion de l'anonymat des identificateurs dans le réseau 6LoWPAN est ignorée, comme précédemment clarifié dans la section précédente. Ainsi, le HIT de l'initiateur et le répondeur sont tous deux prolongés à 128 bits en ajoutant les informations de préfixe HIT. Enfin, le paquet HIP est prêt pour être communiqué sur Internet pour atteindre sa destination finale (l'hôte Internet distant agissant comme un pair de HIP).

Les mécanismes de compression et décompression illustrés dans la figure suivante.

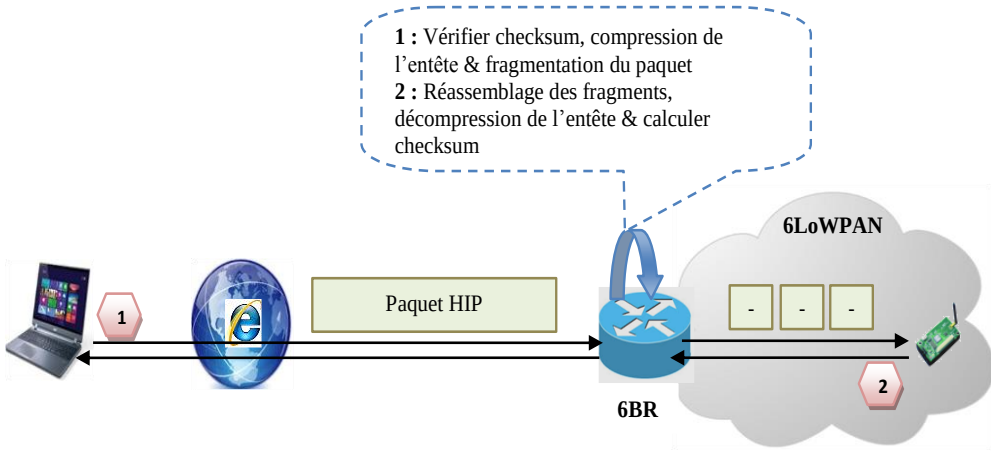


Figure 6.4. La schématisation de la communication de paquets HIP à (1) et depuis (2) le RCSF dans le l'IoT, avec le modèle de compression proposé.

Le tableau ci-dessous résume l'impact du modèle de compression proposé sur chaque champ de l'en-tête de du protocole HIP.

Table 6.1. Les champs de l'entête HIP avant et après l'application de la compression 6LoWPAN.

Champ	Longueur avant compression	Longueur après compression
Entête suivant	8 bits	0 or 8bits
Longueur entête	8 bits	0 bits
Type paquet	7 bits	5bits
VER.	4 bits	0 bits
RES.	3 bits	0 bits
0,1	2 bits	0 bits
Checksum	16 bits	0 bits
Contrôle	16 bits	0 bits

HIT source	128 bits	0 or 96 bits
HIT destination	128 bits	96 bits
Longueur de l'entête	**40 octets**	**Min = 13 octets, Max = 25 octets**

La compression 6LoWPAN proposée pour l'entête HIP présente une solution avantageuse pour réduire de façon significative et le temps de communication, et l'empreinte mémoire, et la dissipation d'énergie. Ceci résulte de la quantité minimale d'informations contenues dans les messages HIP compressés communiqués et stockés. En effet, à partir du tableau ci-dessus, nous nous rendons compte que l'entête HIP peut être réduit à 13 octets dans les messages R1 et I2 et peut atteindre jusqu'à 25 octets dans tous les autres messages de HIP. Le tableau ci-dessous compare le coût de la communication de l'entête HIP dans les deux cas: avec et sans le système de compression proposé.

Table 6.2. Surcharge totale liée à la communication de l'entête HIP avec et sans la compression 6LoWPAN proposée.

	Sans compression	**Avec compression**	**Gain proportionnel**
HIP-BEX (quatre paquets)	160 octets	76 octets	52%
Tous les paquets HIP (huit paquets)	320 octets	176 octets	45%

Le modèle de compression 6LoWPAN proposé présente un gain proportionnel important, de 52% en termes d'octets communiqués dans la phase HIP-BEX. Pour la communication de tous les messages HIP, le gain est d'environ 45%, ce qui est assez acceptable.

2.2.2. Le modèle de distribution proposé pour le mécanisme HIP *Base Exchange* (HIP-BEX)

La compression des messages, en tant qu'une technique d'adaptation, ne suffit pas pour une adaptation bien alertée des contraintes des RCSFs car les standards de sécurité basés classiques sont couteux tant en communication qu'en calculs. De même, le mécanisme de mise en place des sessions de sécurité HIP (HIP-BEX) fait appel aux opérations cryptographiques asymétriques assez lourdes et onéreuses en énergie, en particulier les exponentiations nécessaires pour le calcul de la clé publique et privée *Diffie-Hellman* [111]. A cet effet, nous proposons dans la deuxième partie de la solution de sécurité de bout-en-bout basée sur le protocole HIP, un système de distribution adapté pour alléger la charge sécuritaire du mécanisme HIP-BEX. Le but de la distribution est de rendre le cout de la sécurité beaucoup plus raisonnable, à travers la dispersion la charge cryptographique afin de décharger les nœuds capteurs terminaux qui sont sévèrement limités en ressources de calculs et d'énergie des opérations les plus compliquées dans le calcul du secret.

A. Modèle du réseau et suppositions

Nous considérons un réseau 6LoWPAN hétérogène et nous supposons l'existence de trois types d'entités opérationnelles:

- Un routeur de bordure 6LoWPAN (6BR) : entité de confiance jouant le rôle d'un point de relais entre les deux réseaux (6LoWPAN et Internet).
- Les nœuds RFD (*Reduced Function Devices*): nœuds capteurs contraints limités en ressources, qui ne peuvent pas supporter les primitives cryptographiques asymétriques coûteuses.
- Les nœuds FFD (*Full Function Devices*): nœuds de confiance assez puissants et et sans contrainte, qui sont situés dans le quartier de nœuds RFD. Ces nœuds sont capables d'effectuer des tâches de calcul complexes, pour aider à atténuer l'hétérogénéité dans les capacités de calcul entre les nœuds RFD et les hôtes réguliers d'Internet. Les nœuds FFD peuvent être similaires au 6BR en matière de disponibilité de ressources et peuvent même être branchés à un secteur d'alimentation électrique. En outre, ils devraient être couverts par la topologie que construit le protocole de routage RPL, et inclus dans les chemins de communication reliant les nœuds RFD et le 6BR. Nous soulignons ainsi que, si aucun nœud FFD n'est interposé entre un nœud RFD et la 6BR (si par exemple ils sont reliés directement ou si le nœud FFD est exclu du réseau en raison d'une panne), le 6BR, dans ce cas, se comporte comme un nœud FFD pour ce nœud de capteur RFD.

La figure ci-dessous illustre le modèle de réseau considéré.

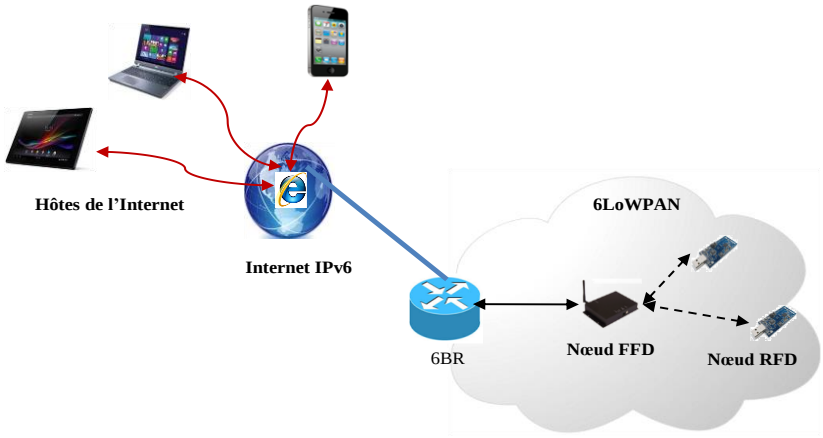


Figure 6.5. Le modèle du réseau considéré dans la solution proposée.

B. Le modèle de distribution proposé

L'idée centrale du mécanisme de distribution proposé pour la négociation de la sécurité dans le protocole HIP est d'introduire un tiers de confiance en toute sécurité et de manière transparente entre le nœud capteur terminal et le pair HIP distant. Cette troisième partie est représentée par un nœud collaborateur puissant (FFD dans le 6LoWPAN), qui collabore avec le nœud capteur terminal pour équilibrer la charge de calcul. Ceci est réalisé par la délégation des opérations cryptographiques les plus lentes et exigeantes en CPU dans le HIP-BEX au nœud collaborateur. Le fonctionnement du

modèle de distribution proposé est divisé en trois phases principales: la phase d'initialisation, la phase d'établissement de la sécurité et la phase de détection d'intrusion.

B.1. la phase d'initialisation

Cette phase initie et prépare les entités impliquées dans la procédure de distribution, ainsi que, le matériel de sécurité requis pour le modèle de distribution proposé. La phase d'initialisation comprend le déploiement du réseau, et le pré-chargement, en toute sécurité, de la clé du réseau partagée par tous les nœuds du réseau (opérations effectuées par un administrateur réseau), suivant le modèle de sécurité de la couche MAC défini par la norme IEEE 802.15.4 [110]. Cette clé sert pour l'authentification mutuelle entre les nœuds dans le réseau 6LoWPAN (les nœuds capteurs et les collaborateurs) et pour la protection des communications entre les capteurs et les nœuds assistants. En outre, la clé partagée du réseau peut être rafraîchie périodiquement, afin de résister aux menaces d'extraction de clé qui exposent le réseau aux risques d'intrusions.

A ce stade, il est important de noter que cette phase ne concerne que le réseau 6LoWPAN et les hôtes HIP externes ne sont pas conscients de ce qui se passe à l'intérieur du réseau de capteurs. Cela revient à dire que la distribution proposée est transparente à tout hôte externe dans l'Internet.

B.2. la phase d'établissement de la sécurité

Dans cette phase, les pairs HIP, qui sont dans notre cas un hôte ordinaire de l'Internet et un nœud capteur, établissent une association de sécurité HIP selon le modèle de distribution proposé pour le mécanisme HIP-BEX. En considérant que l'initiateur HIP est un hôte ordinaire et puissant sur Internet, et que le répondeur HIP est un nœud de capteur contraint, le schéma de distribution proposé est tel que décrit dans la figure 6.6.

Dans un premier temps, le nœud collaborateur (FFD) et le nœud capteur terminal (RFD) effectuent obligatoirement l'authentification mutuelle, car ils vont échanger des informations de sécurité assez critiques, en démontrant la possession de la clé du réseau pré-partagée. En outre, les données critiques communiquées entre les deux nœuds et qui sont utiles pour le calcul des clés *Diffie-Hellman* sont protégées et chiffrées en utilisant la même clé (du réseau).

À la réception du message I1 et, après une authentification réussie, le nœud capteur (le répondeur HIP) envoie en toute sécurité, les données nécessaires pour le calcul de sa clé publique *Diffie-Hellman*, au collaborateur. Ce dernier calcule la clé publique *A* (voir figure 6.6) et l'envoie au nœud capteur correspondant. Il est important de noter que le calcul de la clé *A* peut être effectué de manière proactive juste après l'authentification mutuelle. Dans ce cas, le répondeur HIP communique le secret initial (*a*) au collaborateur bien à l'avance. Là le risque d'extraction du secret par un tiers malveillant sera relativement plus élevé et une fois le secret (*a*) est récupéré, la clé de session sera très facile à découvrir (l'attaquant devrait juste intercepter le message I2 et calculer B^a *modulo p*). Pour cette raison, il est préférable d'opter pour un calcul réactif de la clé publique *A*, juste après la réception du message d'initialisation I1. Ainsi, le calcul à la demande n'affecte pas considérablement le délai

d'établissement de la session HIP car le collaborateur est suffisamment puissant et il effectue les calculs *Diffie-Hellman* qui lui sont délégués beaucoup plus rapidement que le nœud capteur terminal.

A ce stade, le répondeur HIP prépare et transmet le message signé R1 à l'initiateur. Pendant la traversée du message depuis l'initiateur vers le répondeur correspondant dans le réseau 6LoWPAN, le message I2 s'arrête au niveau du collaborateur qui vérifie sa signature en utilisant la clé publique correspondante à l'identifiant de l'hôte HIP (HI) de l'initiateur et calcule la clé de session secrète au nom du nœud capteur répondeur. Par la suite, le collaborateur crypte (à l'aide de la clé partagée du réseau) et communique le secret *Diffie-Hellman* calculé avec la solution du puzzle contenue dans le message I2, au nœud capteur afin qu'il vérifie si la solution correspond réellement au puzzle proposé. Enfin, le répondeur calcule la valeur MAC à partir de la clé de session et envoie le résultat dans le message signé R2 à l'initiateur pour lui confirmer le secret. Nous soulignons que les deux messages R1 et R2 sont signés par le nœud capteur répondeur en utilisant sa clé privée (la partie privée de son identifiant HIP : HI). Nous soulignons également que la communication entre le collaborateur et le répondeur est supposée fiable.

Comme les nœuds dans un réseau de capteurs sont prédisposés à des risques de compromission, le collaborateur porte donc ce risque qui pourrait lui entraîner un empoisonnement du comportement pour par exemple espionner ou altérer les communications entre l'initiateur et le répondeur HIP. Pour éviter que tel problème se produise, nous proposons une contre-mesure optionnelle à entreprendre par les pairs HIP en vue de cacher la clé de session finale au collaborateur.

Après avoir calculé la clé de session, les pairs HIP pourraient opter pour la combinaison de cette clé avec une autre clé (*seed key*) pour préserver l'image de la sécurité de bout-en-bout. Cette clé peut être par exemple, intelligemment extraite des identificateurs HIP (HI) de l'initiateur et le répondeur, juste pour éviter l'échange de messages de contrôle supplémentaires. Une méthode possible pour la génération de la clé de combinaison peut être inspirée de la technique FHSS (*Frequency Hopping Spread Spectrum*) où, l'émetteur et le récepteur se mettent d'accord, à l'avance, sur une séquence pseudo-aléatoire correspondante aux canaux de fréquence sur lesquels le signal sera propagé pour une communication à la fois sécurisée et résistante aux interférences.

Ainsi, les pairs HIP se mettent d'accord sur une séquence pseudo-aléatoire représentant les parties de la concaténation des deux identifiants HIP. Les parties correspondantes doivent être correctement reliées pour former la clé de combinaison. Notons qu'il est préférable de définir des tailles variables pour les diverses parties afin de renforcer la sécurité de la procédure de génération de la clé. Enfin, la combinaison de la clé *seed* avec la clé de session HIP initiale peut être faite avec une opération cryptographique simple (par exemple : XOR). De cette façon, il sera difficile au collaborateur de deviner la bonne clé de combinaison, même s'il connaît les identificateurs publics des pairs HIP. Ainsi, la clé finale de la session HIP serait:

$$HIPCléSession = DHCléSession \oplus cléSeed \qquad (6.1)$$

Dans ce qui suit, nous donnons un simple exemple de dérivation d'une clé de combinaison :

Supposons que HI1 et HI2 sont les identifiants HIP (codés en binaire) de l'initiateur et le répondeur, respectivement.

H1: (10100100101110010000101010 11) H2: (01100111110101011111000100101)

La clé initiale (obtenue par concaténation des deux identifiants):
(10100100101110010000101010110110011111010101111000100101)

La clé initiale est par exemple divisée en 11 parties, où les tailles conventionnelles de toutes les parties sont par exemple:
(Partie, Taille en bits): (1,4), (2,8), (3,2), (4,3), (5,3), (6,6), (7,6), (8,5), (9,7), (10,5), (11,7).

La séquence pseudo-aléatoire est par exemple : **1- 8- 5-11-2-7**

1010	**01001011**	10	010	**000**	101010	**110110**	**01111**	1010101	11100	**0100101**
1	**2**	3	4	**5**	6	**7**	**8**	9	10	**11**

La clé *seed* résultantes est **: 10100111100001001010100101111 0110**

En plus des avantages liés à la distribution de la charge cryptographique, qui soulage le nœud capteur des primitives de calcul les plus onéreuses en termes de temps de calculs et de dissipation d'énergie dans la phase d'établissement de sécurité dans HIP (principalement les exponentiations *Diffie-Hellman:* g^a et B^a). Le système proposé comprend un autre avantage qui réside dans sa transparence; la distribution est cachée à l'hôte Internet externe (dans notre cas, l'initiateur), afin de préserver l'image de la négociation de sécurité de bout-en-bout.

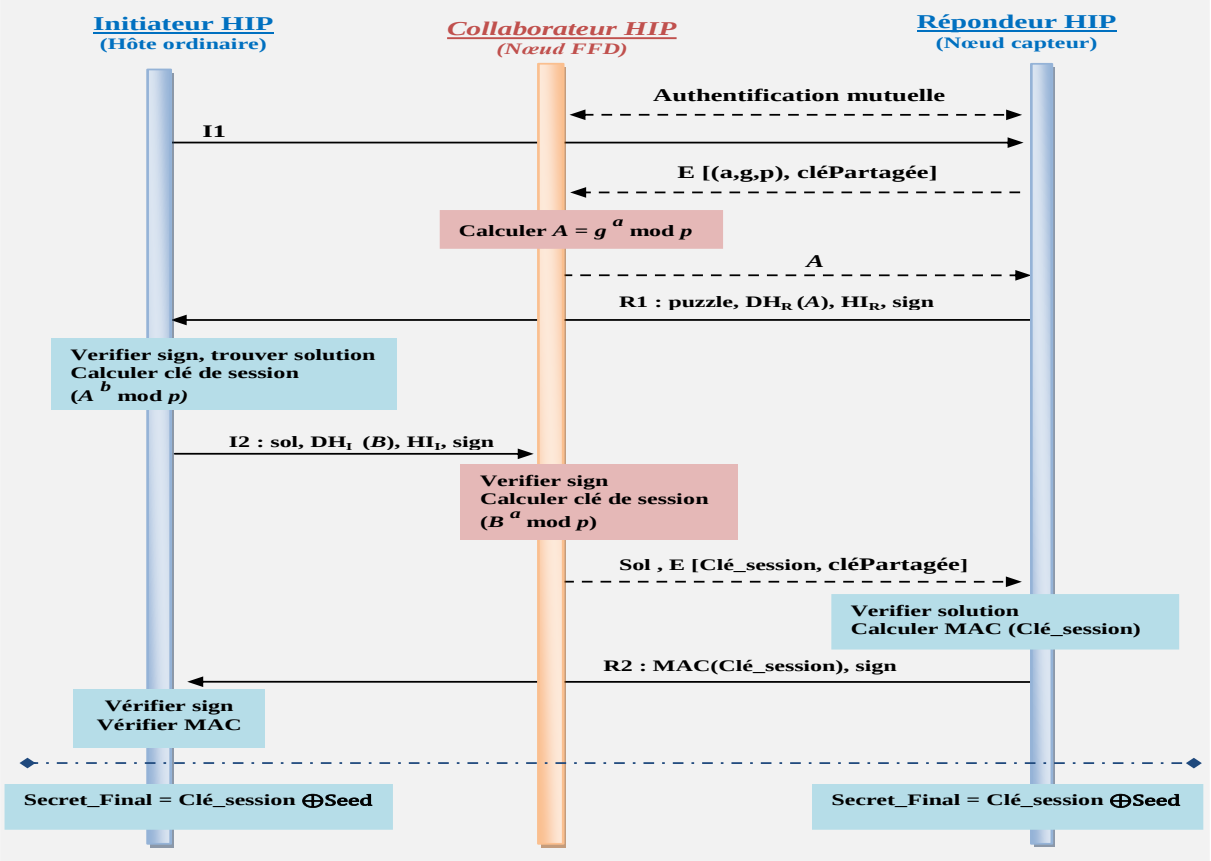


Figure 6.6. Le schéma proposé pour la distribution sécurisé de la charge computationnelle dans HIP Base Exchange.

B.3. la phase de détection d'intrusion

L'utilisation de la technique d'authentification mutuelle entre le collaborateur et le nœud capteur terminal, selon les règles cryptographiques, sert à l'exclusion de tout adversaire externe au réseau 6LoWPAN, qui peut tenter d'usurper l'identité d'un nœud légitime (collaborateur ou capteur ordinaire), de voler la clé de session, et exercer probablement une ou plusieurs attaques possibles, telles que: l'écoute clandestine des échanges, l'attaque de détournement de session. Cependant, l'authentification mutuelle seule ne fournit pas une sécurité optimale pour le modèle de distribution, du fait que les nœuds capteurs déployés dans une zone éloignée sans surveillance, et connectés à Internet, sont susceptibles d'être compromise soit physiquement ou par l'exécution de codes malveillants provenant de l'Internet. Les nœuds de compromission tentent d'exercer des attaques internes, qui sont nuisibles parce que les attaquants internes ont tout le matériel de sécurité requis, tels que les algorithmes de chiffrement/authentification, les clés cryptographiques, et bénéficient même de toutes les autres autorisations et les responsabilités dans le réseau (communication des données, la décision du routage, le relais de données, etc.) tout à fait comme les nœuds légitimes, tout en se comportant de manière malintentionnée.

Dans notre cas, le nœud de collaboration et le nœud de capteurs terminal (le répondeur HIP) portent ensemble le risque d'être compromis, mais le pire des scénarios se produit lorsque le nœud

collaborateur est affecté. Dans tel cas, le collaborateur malicieux peut empêcher les nœuds capteurs légitimes sous-jacents d'établir des sessions HIP avec des hôtes externes.

Dans notre solution, nous supposons que le réseau de capteurs est équipé localement d'un système robuste de détection d'intrusion (IDS), comme celui proposé dans [71], pour faire face aux attaques internes et aussi pour détecter et isoler les nœuds dont le comportement est malveillant (nœuds capteurs terminaux ou nœuds collaborateurs) dans le réseau 6LoWPAN.

2.3. CD-HIP et les autres solutions de sécurité de bout-en-bout basées sur HIP dans l'IoT

Contrairement aux solutions basées-HIP (et mêmes celles qui ne sont pas basées HIP) existantes, qui reposent dans le meilleur des cas soit sur la compression des messages avec des schémas de compression non conformes au standard 6LoWPAN, ou sur une politique de distribution compliquée pour le processus HIP-BEX. Notre solution (CD-HIP) qui combine un modèle de compression 6LoWPAN optimal pour l'entête HIP avec un modèle de distribution sécurisé, efficace et bien adaptée pour la procédure d'établissement de session de sécurité définie dans HIP-BEX. Cela permet une sécurité de bout-en-bout des communications avec les capteurs connectés à l'IoT, avec une réduction importantes des coûts de la communication et de l'établissement de la sécurité, tout en maintenant une bonne compatibilité avec le standard HIP. Le tableau ci-dessous compare CD-HIP, avec l'ensemble de solutions basées HIP dans l'IoT.

Table 6.3. Comparaison entre CD-HIP et le reste des solutions d'adaptation HIP pour la sécurité de communications de bout-en-bout dans l'IoT.

Protocole	Coût des communications	Coût des calculs	Compatibilité avec le standard HIP
R. Moskowitz, et al,. [85]	Important	Encore important	Mauvaise
T. Heer [86]	Important	Très faible	Bonne
Y. Ben-Saied et al,. [87]	Très important	Très faible	Bonne
R. Hummen, et al,. [88]	faible	Encore important	Mauvaise
F.V. Meca, et al,. [89]	Important	Moyen	Mauvaise
CD-HIP	**Faible**	**Faible**	**Bonne**

3. SOLUTION PROPOSÉE POUR LA SÉCURITÉ DES COMMUNICATIONS HUMAIN-À-OBJET DANS L'IOT

En plus des communications humain-à-humain qui règnent sur l'Internet classique, deux nouvelles classes de communications émergent avec l'émergence de l'IoT. On parle alors des communications humain-à-objet (*Human-to-Thing* : H2T) et les communications machine-à-machine ou M2M qui comprennent les communications objet-à-objet (*Thing-to-Thing* : T2T) et les communications objet-à-humain (*Thing-to-Human* : T2H). D'un point de vue sécuritaire, les communications H2T (qui sont toujours initiées par l'utilisateur) sont les plus importantes car elles représentent une source

importante de menaces sévères ciblant les réseaux de capteurs dans l'IoT, où les différences de capacités entre les nœuds capteurs et les hôtes ordinaires puissants sont malicieusement exploitées. A partir de là, la sécurité des communications H2T est considérée comme un véritable défi à surmonter.

3.1. Expression de la problématique

Les solutions proposées pour sécuriser les services des réseaux de capteurs et leurs communications, de bout-en-bout, dans le contexte de l'Internet des objets, ne considèrent généralement que le cas des communications inter-dispositifs (capteurs). Ce genre de communications (objet-à-objet) est caractérisé par un degré élevé d'homogénéité matérielle et protocolaire au niveau des parties communicantes, ce qui facilite la mise en œuvre de la sécurité.

Les interactions entre les hôtes ordinaires et les nœuds capteurs dans le cadre des communications humain-à-objet, dans l'Internet du futur sont assez fréquentes dans certaines applications comme les applications de maisons connectées [112], villes intelligentes [113], et parfois même dans les applications médicales [114], etc. où l'utilisateur a besoin d'interroger ou changer la configuration des objets connectés.

Si elle est considérée, la sécurité des communications humain-à-objet devrait toujours reposer sur l'intervention d'une troisième partie (un proxy) pour effectuer des traductions entre les protocoles applicatifs (CoAP et HTTP) et/ou entre différents protocoles de sécurité (TLS et DTLS), ce qui contredit expressément le principe de la sécurité de bout-en-bout. D'autre part, les solutions proposées ne prennent pas en compte les spécificités des réseaux 6LoWPANs (réseaux de capteurs IPv6) et ne considèrent pas sérieusement les restrictions en termes de disponibilité des ressources au niveau des nœuds capteurs.

Jusqu'à présent, la sécurité des interactions humain-à-objet dans l'IoT n'est pas clairement et efficacement adressée, et pourtant il s'agit d'une classe de communications qui est souvent à l'origine des attaques par déni de service (DoS) exploitant malicieusement l'hétérogénéité entre un capteur et un client puissant sur Internet pour subvertir tout un réseau de capteurs connectés à l'IoT.

Dans ce travail, nous proposons une adaptée pour la sécurité des communications humain-à-objet. Notre solution exploite les hétérogénéités inhérentes dans les dispositifs et les réseaux impliqués dans ce genre de communications, avec une bonne considération des particularités des nœuds capteurs et des réseaux 6LoWPANs. La solution considère des clients HTTP et des serveurs (nœuds capteurs) CoAP en interactions, et définit la première solution qui assure la sécurité de bout-en-bout des communications entre ces entités tout en limitant l'impact des attaques DoS sur les serveurs CoAP.

Nous avons déjà mentionné dans le chapitre précédent que les solutions de sécurité concentrées au niveau de la couche application ne sont pas tout à fait adaptées pour protéger les communications H2T entre des hôtes HTTP/CoAP ou alors HTTP/MQTT. Par conséquent, les solutions sous-jacentes apparaissent plus pratiques. Cependant, l'emploi de deux protocoles de sécurité différents au niveau de la couche transport (TSL du coté de client HTTP et DTLS du coté de serveur CoAP) nécessite

l'intervention d'un proxy pour la traduction entre les deux protocoles de sécurité, en plus de la translation obligatoire entre les protocoles HTTP et CoAP. Afin de remédier à ce problème, nous supportons l'adoption du protocole IPsec pour la sécurité des communications H2T au niveau de la couche réseau des deux entités communicantes. Cependant, le problème de rupture de la sécurité de bout-en-bout au niveau du proxy est toujours persistant.

Avec IPsec (et même dans les protocoles de sécurité de la couche transport) une fois que l'association de sécurité est établie entre les pairs communicants, tous les messages issus de la couche application (appelés PDUs : *Protocole Data Units*) seront encapsulé par IPsec, dans les deux extrémités de la communication, sans aucune distinction, jusqu'à la fermeture de la session. Cette approche n'a pas suscité des problèmes dans les réseaux IP classiques où les entités étaient suffisamment puissantes. Toutefois, quand telles solutions sont appliquées aux réseaux contraints, tels que les réseaux 6LoWPAN, elles doivent présenter une attention particulière à la rationalisation de la consommation de l'énergie des nœuds capteurs.

La solution proposée [115] consiste en un système de sécurité asymétrique et sélectif pour protéger les communications humain-à-objet de bout-en-bout, dans l'Internet des objets.

3.2. la sécurité asymétrique proposée

En plus des différentes formes d'asymétrie dans les capacités des dispositifs et des réseaux qui rendent même la nature des communications entre les hôtes ordinaires de l'Internet et les nœuds capteurs asymétrique (la taille des messages est plus importante du coté Internet que dans le réseau 6LoWPAN). Les communications requête/réponse avec les nœuds capteurs dans un réseau 6LoWPAN ont également un degré asymétrique d'importance, d'un point de vue sécuritaire. En effet, les messages de requêtes ne comportent pas vraiment des informations critiques, sauf le type de la ressource demandée qui peut être dans certains cas un indice d'une information privée. Par revanche, les messages de réponse sont les plus intéressants (toujours de point de vue sécuritaire) et les plus demandeurs en sécurité, parce qu'ils transportent les données de captages, effectives, qui sont généralement assez critiques et confidentielles. Ainsi, les messages échangés sont beaucoup plus critiques dans le sens de communication allant du serveur CoAP vers le client HTTP que dans le sens inverse (allant du client HTTP vers le serveur CoAP). Toutes les solutions existantes fournissent des modèles de sécurité symétriques où les requêtes et les réponses sont traitées de la même façon. Cela n'a pas posé des problèmes dans l'Internet classique car les entités communicantes étaient homogènes et avaient des capacités équivalentes. Cela n'est plus le cas dans les communications humain-à-objet dans l'IoT, ce qui empêche l'approche de sécurité symétrique d'être une solution pratique. Notons qu'on aurait préféré sécuriser tous les types de messages communiqués vers et depuis et réseau 6LoWPAN, qu'ils soient des réponses ou des requêtes. Mais dans ce cas, les coûts relatifs à la sécurité deviendraient nettement importants, en plus des coûts déjà consacrés à l'adoption des standards de communication basés-IP par tels réseaux contraints.

Donc pour plus d'efficacité de la sécurité des communications humain-à-objet dans l'IoT, nous proposons de concentrer la sécurité uniquement sur les messages de réponses générés par les serveurs CoAP et destinés aux clients HTTP. Ce principe est inspiré de la technique ADSL (*Asymmetric Digital Subscriber Line*) [116] qui offre un débit asymétrique car le flux de données est plus important quand il est descendant, que quand il est ascendant.

Afin de garantir une sécurité de bout-en-bout des communications H2T, la procédure de conversion des réponses CoAP en des réponses HTTP est décalée et confiée au client qui est dans notre cas un hôte HTTP puissant. Le modèle du réseau est illustré dans la figure ci-dessous.

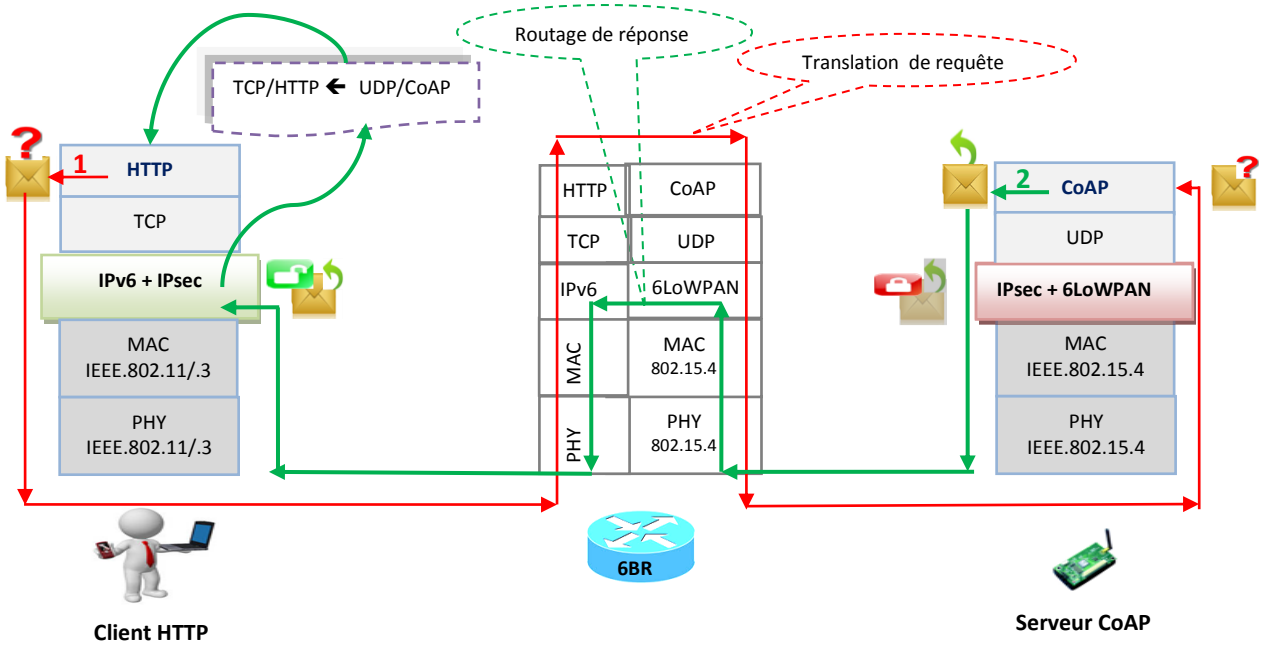


Figure 6.7. Illustration du modèle de la sécurité asymétrique proposé.

Après que la session de sécurité soit établie entre le client HTTP et le serveur CoAP (par négociation, à l'aide du protocole HIP ou IKE), le client initie et envoie une requête HTTP au serveur CoAP (le nœud capteur connecté) via son smartphone, son ordinateur ou autre. Lorsque le 6BR reçoit la requête HTTP, il la traite en tant que proxy et il la traduit en une requête CoAP. La requête CoAP résultante est ensuite communiquée vers le nœud capteur cible (le serveur). Notons que le proxy 6LoWPAN assure le service de translation de protocoles pour tous les messages de requêtes HTTP entrants. Cette tâche lourde ne peut pas être prise en charge par les nœuds capteurs qui n'ont pas suffisamment d'espace mémoire pour inclure les deux protocoles, et qui surtout se consacrent pour la tache de captage. Le chemin du message requête à partir de son générateur (client HTTP) jusqu'à la destination (serveur CoAP), est tracée en ligne rouge, dans la figure ci-dessus.

A la réception de la requête, le serveur CoAP répond par une réponse contenant la ressource demandée, si elle est disponible. Le message de réponse est encapsulé et crypté au niveau de la couche réseau du serveur au moyen du protocole IPsec, en utilisant le secret partagée avec le client HTTP, qui apparaît au serveur comme un client CoAP (grâce à la traduction entre les protocoles). Avant que le message soit envoyé au client correspondant, il traverse d'abord le réseau 6LoWPAN et

quand il arrive au 6BR, ce dernier achemine le message de réponse au réseau externe vers sa destination. Contrairement au cas de requêtes HTTP entrantes où le 6BR joue le rôle de proxy HTTP-CoAP, le 6BR se comporte seulement comme un routeur pour toutes les réponses CoAP sortantes. Cela parce que dans la solution proposée, il n'est pas permis au routeur de bordure de connaitre les clés secrètes partagées entre les nœuds capteurs et les hôtes externes. Par conséquent, le 6BR ne pourra plus convertir la réponse CoAP contenue dans le datagramme IPv6 chiffré, au format HTTP. Ainsi, on propose de décaler la procédure de traduction du CoAP vers HTTP au client HTTP qui est censé être suffisamment puissant et qui est la seule entité autorisée à décrypter le message de réponse. En fait, CoAP est standardisé pour être le premier protocole applicatif équivalent à HTTP et destiné au web d'objets intelligents (WoT: *Web of Things*). Par conséquent, les hôtes classiques de l'Internet devraient être conscients de ce protocole émergent.

À la réception de la réponse, le client HTTP décrypte le(s) datagrammes IPv6 correspondants dans la couche réseau. A ce niveau, la réponse ne peut pas être directement exploitée par la couche application (le client web), car elle est dans le format UDP/CoAP. Ainsi, le paquet de réponse doit d'abord passer par un module d'adaptation local qui garantit une traduction efficace des réponses CoAP en des réponses HTTP. En fait, certains travaux de recherches vont vers l'intégration du CoAP dans les navigateurs web (la solution *copper* [132] pour le navigateur Mozilla Firefox) mais la validité de cette approche n'est pas encore approuvée. Cela revient au fait que les applications web se basent sur le protocole HTTP. De plus, dans le côté IoT, des protocoles applicatifs autres que CoAP peuvent être adoptés (comme MQTT, XMPP [134], etc.). Donc, l'intégration de tous les protocoles applicatifs déjà existants au niveau du WoT (et éventuellement ceux qui vont voir la lumière dans le futur) dans les navigateurs web ne semble pas être une solution pratique. Pour pallier à ce problème, l'adoption des dispositifs ou modules qui prennent en charge la traduction et l'adaptation des protocoles du WoT avec le protocole HTTP est actuellement primordiale.

Le module de traduction locale défini dans notre solution doit implémenter les règles de conversion du format UDP/CoAP vers le format TCP/HTTP. Ces règles ont été précédemment suivies par le proxy après le décryptage de la réponse CoAP et l'interruption de la sécurité de bout-en-bout.

- **Protection de la vie privée des utilisateurs :**

Le message de requête, et plus particulièrement la ressource demandée, est dans certaines applications une information critique qui risque, en cas où elle reste non-protégée, d'affecter la vie privée des utilisateurs. Si nous prenons l'exemple d'une application médicale de surveillance de la glycémie : une requête de type `GET taux_glucose` indique que la personne (ou le patient) concernée est probablement atteinte de diabète. Cependant, le problème ne se pose pas si la requête serait plutôt abstraite (destinée à un capteur virtuel), comme `GET état_actuel` où l'état représente une agrégation de plus d'une grandeur (ex : glycémie, tension artérielle, température, …).

Donc afin de protéger la vie privée des utilisateurs envoyant des requêtes comportant des indices critiques, deux solutions peuvent être envisagées :

- La première consiste en l'établissement d'un lien de sécurité (tunnel) entre le client HTTP et le proxy, comme indiqué dans la figure 6.8.

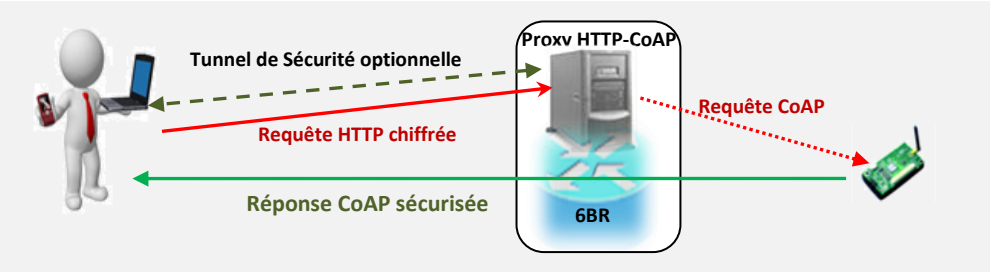


Figure 6.8. La sécurité asymétrique avec un tunnel de sécurité entre le client HTTP et le proxy.

- Pour plus d'efficacité, et pour prévenir le proxy de découvrir ce que le client demande exactement du serveur CoAP, il serait préférable de protéger, de bout-en-bout, les parties sensibles dans la requête. Il s'agirait alors de sécuriser l'URI identifiant la ressource CoAP en question. Ainsi, dans [117], il est spécifié que l'URI CoAP peut être directement inclus dans la l'URI de la requête HTTP, ce qui facilite son extraction par le proxy lors de la génération de la requête CoAP équivalente. La figure 6.9 illustre le modèle supposé.

  Étant donné que l'URI de base HTTP est : `http://p.example.com/hc` et l'URI CoAP: `coap://sensor1.contiki.com/light`. L'URI final à envoyer par le client HTTP est donc : `http://p.example.com/hc/coap://sensor1.contiki.com/light`.

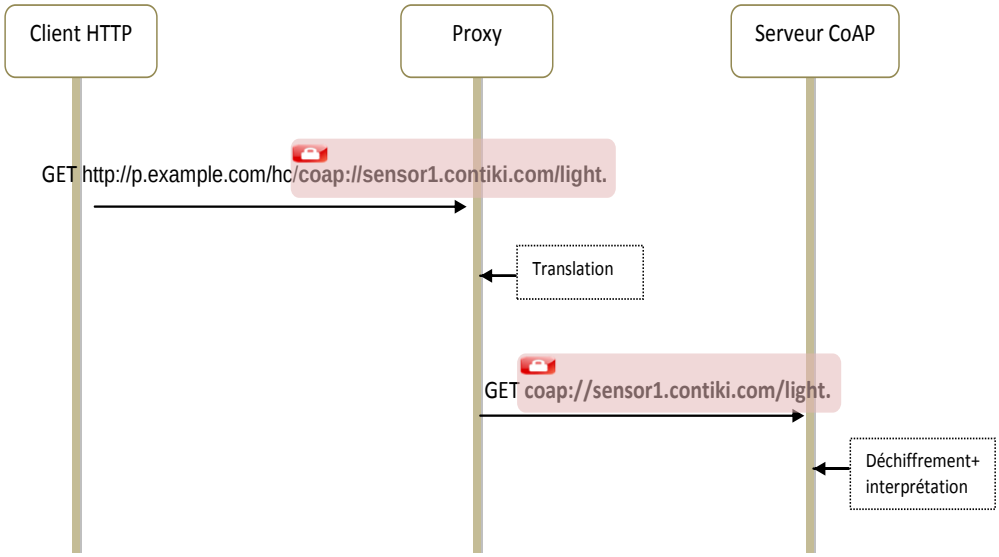


Figure 6.9. La protection de la vie privée des utilisateurs envoyant des requêtes HTTP critiques.

3.3. La sécurité sélective proposée

Les messages qui sont communiqués dans le réseau 6LoWPAN n'ont pas tous besoin d'être sécurisés. Nous avons déjà montré dans la partie sécurité asymétrique que les messages de requête peuvent ne pas être sécurisés, notamment dans les communications Humain-à-objet, car ils ne

comprennent pas vraiment des données sensibles. Nous avons aussi signalé que les messages de réponse ont un degré de criticité le plus élevé du fait qu'ils comportent les rapports de captage.

À leur tour, les messages de réponse peuvent ne pas la même sensibilité à la sécurité n'ont pas et ça dépend du degré de criticité de la ressource (la donnée captée), et/ou de la typologie de la ressource elle-même. En effet, on peut configurer le serveur CoAP (le nœud de capteur) au niveau applicatif pour spécifier les critères de sélection des messages de réponse qui vont être sécurisés au niveau de la couche réseau via le protocole IPsec. Par exemple, un serveur CoAP peut être configuré pour ne sécuriser que les messages dont les données de captage récoltées dépassent un seuil prédéfini de criticité. Cette politique peut être pratique dans plusieurs scénarios applicatifs de l'IoT, telles que les applications de suivi (*tracking*) et les applications médicales.

Un autre cas pratique de l'utilisation de la sécurité sélective apparaît clairement avec les réseaux de capteurs multimodaux connectés à Internet. Nous rappelons que dans un RCSF multimodal, un nœud capteur peut capte plus d'une grandeur, par exemple température et pression. Par conséquent, si une session de sécurité de bout-en-bout est établie entre un nœud capteur et un autre hôte externe sur Internet, ce dernier pourra demander d'obtenir divers types d'informations auprès du même capteur multimodal. Dans tel cas, il est possible de protéger uniquement les messages incluant les lectures correspondantes à un type particulier d'information. Par exemple, un capteur de température et d'image, configuré pour ne sécuriser que les images.

Pour concrétiser la sélection, nous proposons d'ajouter une nouvelle option CoAP pour les messages de réponses afin d'indiquer s'ils sont concernées par l'encapsulation de sécurité de IPsec dans la couche réseau, ou non. L'option est nommée *Securable* et elle a la définition suivante:

Table 6.4. Définition de l'option *Securable*.

Numéro	Classe	Nom	Format	Longueur	Valeur par défaut
n	Facultative	*Securable*	uint	8 bits	0

Le numéro de l'option *Securable* peut prendre sa valeur de l'intervalle 256..2047 qui est réservé pour les options couramment utilisées avec des spécifications publiques [55]. *Securable* est qualifiée d'une option facultative. Les options facultatives CoAP sont simplement ignorées si elles ne sont pas reconnues par les entités manipulant les messages qui les contiennent. Ainsi, l'option est codée sur 8 bits dont le type est entier non signé. Sa valeur par défaut est 0, et si la sécurité est recommandée pour un message donné, la valeur de l'option devient 1.

L'option *Securable* a une portée locale quand elle est ajoutée aux messages à sécuriser (la valeur de l'option est égal à 1). Si tel est le cas, l'option est manipulée et comprise par la couche application et la couche réseau du serveur CoAP. Ainsi, lors de la réception d'une requête CoAP (qui était dans notre cas une requête HTTP), le serveur CoAP génère le message de réponse correspondant à lequel il attache l'option *Securable* avec la valeur appropriée (0 ou 1) en tenant compte de la sensibilité de la ressource. La couche réseau du serveur consulte l'option et en fonction de sa valeur, elle décide de sécuriser ou non le message correspondant. Le processus de sélection est illustré dans la figure ci-dessous.

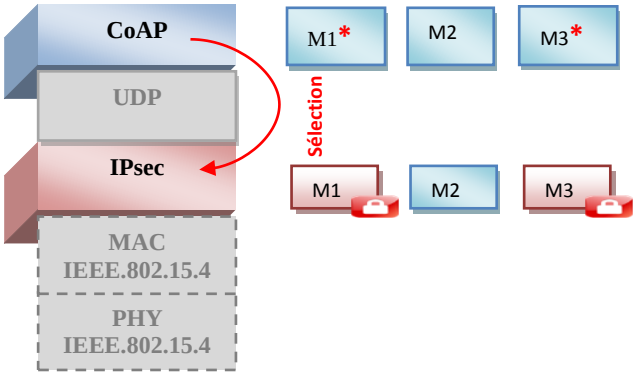


Figure 6.10. Le scénario proposé pour la sélectivité de la sécurité au niveau des serveurs CoAP

Dans l'exemple représenté à la figure 6.9, la couche application génère trois messages CoAP (réponses) M1, M2 et M3. Parmi eux, M1 et M3 sont sélectionnés pour être sécurisés dans la couche réseau. En d'autres termes, les messages M1 et M3 ont l'option *Securable* mise à 1 et ils seront protégés par IPsec, alors que le message M2 ne sera concerné que par les protocoles 6LoWPAN et IPv6 dans la couche réseau, comme la valeur de son option *Securable* vaut 0.

Lorsque la valeur de l'option *Securable* est fixée à 0 dans les messages CoAP moins critiques, la portée de l'option est étendue pour inclure le proxy qui se charge de traduire les messages de réponse CoAP en des réponses HTTP appropriées.

Le diagramme d'activité UML (*Unified Modeling Language*) suivant modélise le scénario de sécurité des communications humain-à-objet selon la solution proposée.

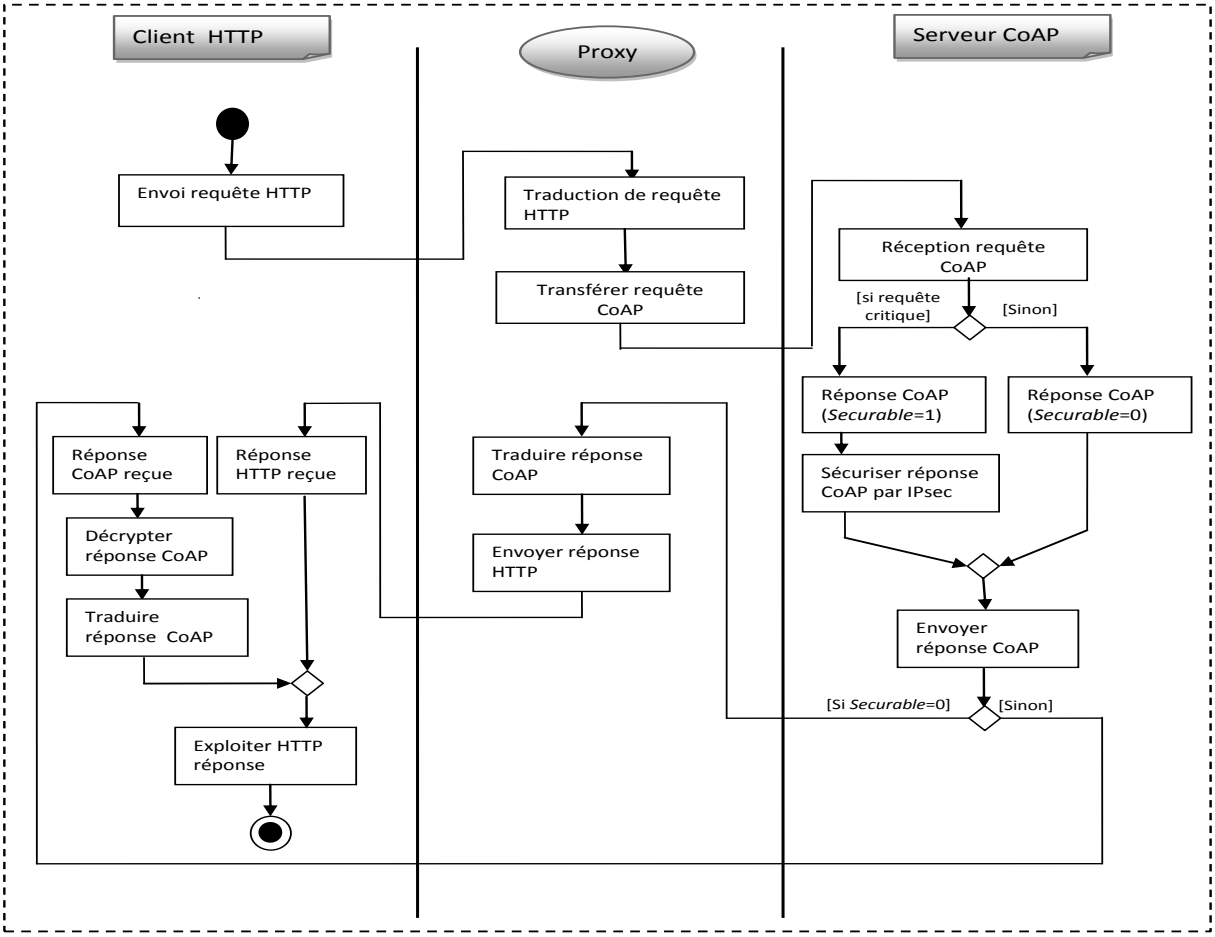


Figure 6.11. Diagramme d'activité UML modélisant le système de sécurité asymétrique et sélective pour les communications entre un client HTTP et un serveur CoAP contraint suivant la solution proposée.

3.4.Les règles supplémentaires pour la translation CoAP-HTTP

Les règles de traduction CoAP-HTTP qui sont clairement et profondément discutées dans [118] doivent être soigneusement appliquées par toute entité assurant la translation (le proxy 6LoWPAN ou le module local de traduction dans le côté du client HTTP) pour garantir la cohérence des communications et donner l'illusion à chaque partie (le client HTTP et le serveur CoAP) qu'elle est en train de communiquer avec l'entité qui lui est équivalente.

De plus, dans notre solution, nous définissons des règles supplémentaires concernant la traduction des paquets TCP/HTTP vers des paquets UDP/CoAP et vice-versa. Ces règles doivent être appliquées à la fois par le serveur proxy et le module de traduction de l'hôte HTTP.

Malgré le fait que les traductions entre les protocoles UDP et TCP soient primordiales dans toutes les communications HTTP-CoAP, cette partie n'a pas reçu l'attention qu'elle mérite dans les documents spécifiant les règles de conversion entre les protocoles CoAP et HTTP. Dans cette section, nous présentons en détails les règles de traduction supplémentaires que nous proposons.

Tout d'abord, le proxy 6LoWPAN établie, systématiquement, la connexion TCP en trois étapes avec les clients HTTP au nom de tous les serveurs CoAP contraints ciblés dans son domaine. A la

génération d'une requête HTTP du coté client, et la réception de la requête par le proxy, et le module de traduction local au client HTTP et le proxy extraient des informations pertinentes à partir du paquet TCP contenant la requête et qui les aideront plus tard à générer correctement des paquets TCP/HTTP à partir des réponses UDP/CoAP. Ces informations sont:

- L'adresse IP du client HTTP (ou son HIT si le protocole HIP est adopté).
- Le numéro du port client HTTP.
- Le numéro de séquence du paquet de la requête.
- La taille (en octets) des données du paquet TCP contenant la requête.
- La valeur du champ acquittement contenu dans l'en-tête TCP.

Après avoir sauvegardé les informations pertinentes concernant la requête HTTP qui vient juste d'être reçue, le proxy commence à l'adapter à une requête UDP/CoAP. La translation de l'entête TCP en un entête UDP est assez simple; les champs qui apparaissent dans l'entête TCP et ne figurent pas dans UDP sont tout simplement éliminés. Ensuite, le proxy aura juste à adapter les numéros de ports de la source et la destination aux numéros de ports UDP qui conviennent. Systématiquement, un paquet TCP correspondant à une requête HTTP contient un numéro de port client (port source) et le port du serveur HTTP 80 (port de destination). Le proxy les remplace par les ports UDP prédéfinis pour le protocole CoAP 61631 et 61616 pour le client et le serveur respectivement. Par la suite, le proxy transmet la requête CoAP résultante de la traduction de la requête HTTP initiale, vers le serveur CoAP ciblé.

Selon que la réponse CoAP est cryptée ou non, le proxy la traduit lui-même ou bien la fait passer vers la machine cliente où elle sera traduite localement par le module d'adaptation des réponses UDP/CoAP en des réponses TCP/HTTP. Ce dernier a déjà gardé trace des informations utiles qui l'aident à faire la traduction TCP vers UDP. Si la réponse CoAP n'est pas sécurisée (l'option *Securable* est mise à 0), le proxy effectue les traductions requises pour en obtenir la réponse HTTP correspondante. Pour ce faire, le proxy exploite l'état sauvegardé pour définir les bonnes valeurs des champs port source, port destination, le numéro de séquence ainsi que la valeur du numéro d'accusé de réception. Dans les ports de source et destination le proxy met 80 et le numéro de port du client HTTP qui a été déjà stocké. Le numéro de séquence du message de réponse prend la même valeur du nombre d'accusé de réception de la requête HTTP. Enfin, le nombre d'accusé de réception de la réponse est obtenu par l'accumulation des valeurs de numéro de séquence de la requête HTTP correspondante avec la taille des données de paquet TCP qui contenait la requête HTTP.

Sinon, si la réponse CoAP est protégée par IPsec (l'option *Securable* vaut 1), le module de traduction défini dans notre solution localement dans le client HTTP, effectue les mêmes actions entreprises par le proxy dans le cas des réponses CoAP non sécurisées.

3.5. Les avantages de la solution proposée

La solution proposée qui consiste en un mécanisme de sécurité asymétrique des interactions humain-à-objet inspirée de la technique ADSL et basée sur le protocole IPsec, et une politique de

sélectivité de la sécurité, supervisée par le protocole CoAP au niveau de la couche application, semble adaptée et beaucoup mieux avertie des contraintes et des spécificités des réseaux de capteurs connectés à l'Internet des objets. La solution proposée a trois principaux avantages:

- **La préservation de la sécurité de bout-en-bout:** le principe de la sécurité asymétrique proposée garantit une sécurité de bout-en-bout pour les communications entre les clients HTTP et les serveurs CoAP, contrairement aux approches classiques où la sécurité était toujours interrompue par une troisième partie (le proxy) pour faire les traductions protocolaires requises.
- **Moindre coût de la sécurité:** selon la stratégie proposée, seuls les messages critiques (dans notre cas: seules les réponses CoAP hautement sensibles) sont concernés par la sécurité dans la couche réseau. Cela représente une optimisation importante de la sécurité avec une bonne considération des contraintes du réseau 6LoWPAN.
- **La protection implicite contre les attaques par déni de service (DoS):** les réseaux de capteurs sont des réseaux orientés-services, cela veut dire qu'un RCSF est spécialement déployé pour fournir un service, qui est souvent assez critique. Par conséquent, les nœuds capteurs doivent préserver leur énergie et leurs ressources le plus longuement possible. Cependant, cette propriété risque d'être facilement violée par un adversaire externe exploitant la grande différence entre le MTU (*Maximum Transmission Unit*) minimal dans les réseaux IPv6 qui est fixé à 1280 octets et le MTU maximal dans les réseaux 6LoWPAN qui vaut juste 127 octets. Donc en envoyant des messages IPv6 en rafale, ou juste quelques paquets bien amplifiés, vers un réseau 6LoWPAN, ce dernier va être inondé par les fragments issus des paquets IPv6, ce qui entraînera un alourdissement des services du réseau, ainsi qu'un épuisement rapide de ses ressources, notamment si les messages sont chiffrés.

  La solution proposée contribue à réduire l'impact des attaques par déni de service (DoS) sur les réseaux 6LoWPANs de la manière suivante : en assurant la sécurité seulement pour les messages qui vraiment en nécessitent, la solution aide le nœud capteur terminal (serveur CoAP) à atténuer l'impact des attaques DoS, qui peuvent apparaître sous la forme de la réception d'une quantité abondante de requêtes. En fait, tous les messages de requêtes reçus sont en clair et le serveur CoAP ne va absolument pas avoir à les déchiffrer avant de renvoyer les réponses correspondantes qui, à leur tour, peuvent ne pas toutes être chiffrées, tout dépend de leur degré de criticité.
- **La minimisation des délais de communications :** l'approche de sécurité asymétrique proposée permet de réduire les opérations sécuritaires que doit réaliser chaque entité impliquée dans la communication humain-à-objet dans le contexte de l'IoT. D'une part les requêtes ne sont pas chiffrées/déchiffrés au niveau du proxy et éventuellement au niveau du client et du serveur. D'autre part, si les réponses doivent être protégées, leur sécurisation se fait toujours de bout-en-bout : le serveur (CoAP) crypte et le client (HTTP) décrypte les messages de réponses.

- **Moindre surcharge sur le proxy :** le système proposé distribue la tâche de translation des protocoles applicatifs entre le proxy HTTP-CoAP et les clients HTTP. Cela permet de réduire davantage la surcharge concentrée sur le proxy en cas des communications intensives introduites par un ou multiples clients HTTP.

4. Conclusion

Dans ce chapitre nous avons présenté les solutions que nous proposons dans le cadre de la protection des communications avec les réseaux de capteurs connectés à Internet. Nous avons ciblé les réseaux de capteurs de type 6LoWPAN où les nœuds capteurs sont traités comme étant des hôtes IPv6 particuliers. Les communications entre tels capteurs et le reste des hôtes (autres nœuds capteurs connectés à Internet ou hôtes classiques) se réalisent d'une manière directe. Dans ce contexte, la sécurité de bout-en-bout des communications devient très importante surtout dans le cas où les données rapportées par les nœuds capteurs sont hautement critiques. Cependant, la nature hétérogène de la communication entre les différents types d'entités impliquées, ainsi que les contraintes sévères des réseaux de capteurs, font que la sécurisation de bout-en-bout des communications entre les nœuds capteurs dans l'IoT et le reste des hôtes sur Internet soit un véritable défi.

Notre contribution est essentiellement matérialisée par deux solutions : la première sert à assurer un établissement efficace de la sécurité de bout-en-bout entre les nœuds capteurs et les hôtes externes dans un Internet IPv6. Quant à elle, la deuxième solution proposée consiste en un système adapté destiné à la sécurité des communications humain-à-objet (H2T: *Human-to-Thing*) dans l'IoT tout en affaiblissant l'effet des attaques par déni de service qui menacent la réussite de l'incorporation des réseaux de capteurs à l'IoT.

Le chapitre suivant sera consacré à l'évaluation des performances des solutions proposées à l'aide de la simulation.

CHAPITRE 7:

Evaluation des performances des solutions proposées

1. INTRODUCTION

Dans ce chapitre, nous évaluons les performances de nos contributions. Les évaluations sont effectuées à l'aide du simulateur réseau Cooja du système d'exploitation Contiki qui est dédié à la simulation des réseaux de capteurs dans le contexte de l'Internet des objets. Les évaluations sont divisées en deux volets : les évaluations de la solution de sécurité de bout-en-bout (le protocole CD-HIP) et l'évaluation du mécanisme de sécurité proposé pour les communications humain-à-objet (nœud capteur) dans l'IoT.

2. ENVIRONNEMENT DE SIMULATION

Dans cette section, nous donnons une présentation générale du simulateur Cooja de Contiki que nous utilisons pour l'évaluation de performances des solutions proposées. Nous présentons ainsi la plateforme Tmote Sky sélectionnée pour les capteurs terminaux.

2.1. Aperçu sur le simulateur

Pour effectuer nos tests d'évaluation, nous avons utilisé Cooja Contiki 2.5 [118]. Cooja (*Contiki os java simulator*) a été inventé en 2002 à l'institut de recherche suédois SICS ICT [119]. Cooja est reconnu pour être un simulateur bien développé, destiné à la simulation des réseaux de capteurs connectés à l'IoT (les réseaux 6LoWPANs). Il est open source, portable et s'installe sur machine virtuelle VMware. Cooja appartient à la famille des simulateurs à évènements discrets, son code (et même celui du système Contiki) est écrit en langages C et Java et repose sur le concept de *protothreading* [128].

Etant destiné à la simulation des réseaux 6LoWPANs, Cooja implémente une pile protocolaire TCP/IP complète et adaptée pour les environnements capteurs (IPv6, 6LoWPAN, RPL, ICMPv6, TCP, UDP, etc.). Ainsi, Cooja définit plusieurs émulateurs de capteurs, à savoir TMote Sky, Z1, Micaz, etc. avec un modèle de simulation détaillé et très proche de la réalité des capteurs. Cooja supporte même la simulation des réseaux de capteurs hétérogènes regroupant différents types de plateformes capteurs ce qui présente un avantage majeur.

Cooja comprend plusieurs plugins comme : `Log listener`, `Timeline`, `Radio logger`, `Collect view`, `Mobility`, etc. le plugin `Log listener` est utile pour l'affichage des messages d'état de chaque nœud ou les messages échangés lors des communications. Le plugin `Timeline`, sert plutôt à la visualisation des communications radio (transmission, réception, collision) ainsi que les états éveillé et endormis des nœuds capteurs. Le plugin `Radio logger` permet d'afficher les formats et le contenu des trames échangées au niveau MAC, et pour activer la mobilité des nœuds capteurs le plugin `Mobility` est utilisé. Cooja inclut d'autres plugins pas moins importants et peut même être étendu pour en avoir plus. La figure ci-dessous montre un exemple d'une instance graphique de simulation dans Cooja d'une communication entre un client UDP (nœud 1) et un serveur UDP (nœud 2). Trois plugins sont utilisés : le plugin Simulation Visualizer, Log listener et Timeline.

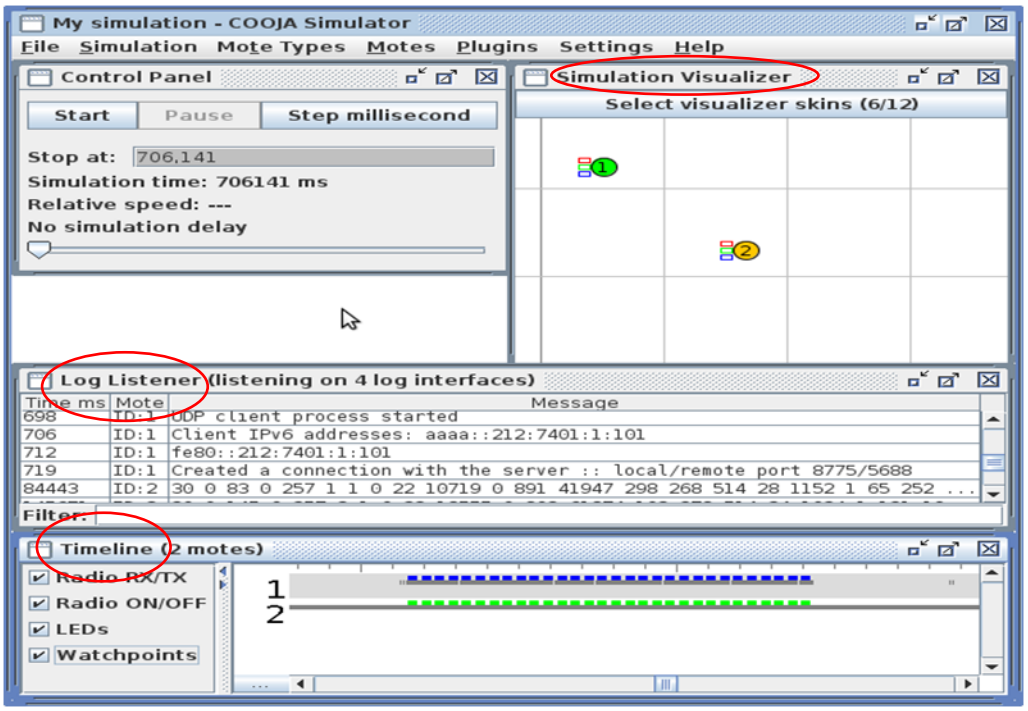


Figure 7.1. Interface graphique de la simulation dans Cooja.

Pour l'estimation de la consommation d'énergie au niveau des nœuds capteurs, le système Contiki définit un module appelé *Energest* (*Energy estimation*) [120] qui donne le temps dépensé pour les traitements CPU, pour la transmission, la réception avec écoute et le temps pendant lequel le capteur était en mode de faible consommation d'énergie LPM (*Low Power Mode*). A partir de ces informations, l'énergie consommée peut être calculée comme suit :

$$Energie(mJ) = \frac{Temps}{STicks} * Dissipation\_Courant * Tension \quad (7.1)$$

Pour mesurer l'énergie consommée par un nœud capteur, en milli-joules (mJ), lors de la réalisation d'une telle tâche (calculs ou communications radio), on doit multiplier le temps écoulé (en secondes) pour effectuer cette tâche par la dissipation du courant (en milliampère mA) et la tension d'alimentation (en volt). Notons que la valeur de la dissipation élémentaire du courant et la tension d'alimentation dépendent de la plateforme utilisée. Le module *Energest* exprime le temps (*Temps* dans l'équation 7.1) en *Ticks* (nombre d'impulsions d'horloge). Pour obtenir le temps en secondes, on divise le nombre de *Ticks* donné par *Energest* par le nombre d'impulsions que le *Timer* (horloge) génère par seconde. Dans Contiki 2.5, le *Timer* génère exactement 32768 *Ticks* par seconde.

2.2. Aperçu sur les dispositifs réseau sollicités

Dans les simulations effectuées nous avons utilisé la plateforme TMote Sky [126] pour les nœuds capteurs. Cette plateforme est assez célèbre et elle est utilisée très souvent dans les travaux de recherche ciblant les réseaux de capteurs [68-71]. Elle utilise un microcontrôleur MSP430 qui fonctionne à une fréquence CPU de 3.9 MHz. Concernant la mémoire, un capteur TMote Sky dispose

de 48 Ko de ROM et 10 Ko de RAM. Les autres caractéristiques de la plateforme sont résumées dans le tableau ci-dessous.

Table 7.1. Caractéristiques de la plateforme TMote Sky.

Fonctionnalité	**Valeur**
Le mode LPM (*Low Power Mode*)	0.0545 *mA*
Opération CPU	1.8 *mA*
Transmission	17.7 *mA*
Ecoute	20 *mA*
Tension d'alimentation	3 *V*

Du tableau 7.1, on constate que l'énergie élémentaire consommée pour les communications radio (*20 mA* pour l'écoute et *17.7 mA* pour la transmission) est beaucoup plus importante que l'énergie des calculs (uniquement *1.8 mA*) ce qui justifie le fait que les communications radio soient plus onéreuses en termes de consommation d'énergie que les traitements CPU.

Le routeur de bordure et même le nœud collaborateur sont exécutés sur la machine linux[6] sur laquelle le simulateur est installé et ils sont connectés à Cooja (aux nœuds capteurs simulés) via des sockets série, comme précisé dans [121]. Notons juste que le routeur de bordure est dans notre cas configuré pour agir aussi comme un pair HIP (initiateur) qui génère localement des messages HIP et comme un proxy HTTP/CoAP pour la deuxième solution.

3. Evaluation de la solution de sécurité de bout-en-bout proposée (CD-HIP)

Nous considérons un réseau 6LoWPAN composé de 100 nœuds capteurs Tmote Sky émulés (les nœuds RFDs) avec une interface radio IEEE 802.15.4. De plus, les nœuds du réseau sont fixes et déployés de manière aléatoire. La figure suivante illustre le modèle du réseau 6LoWPAN considéré.

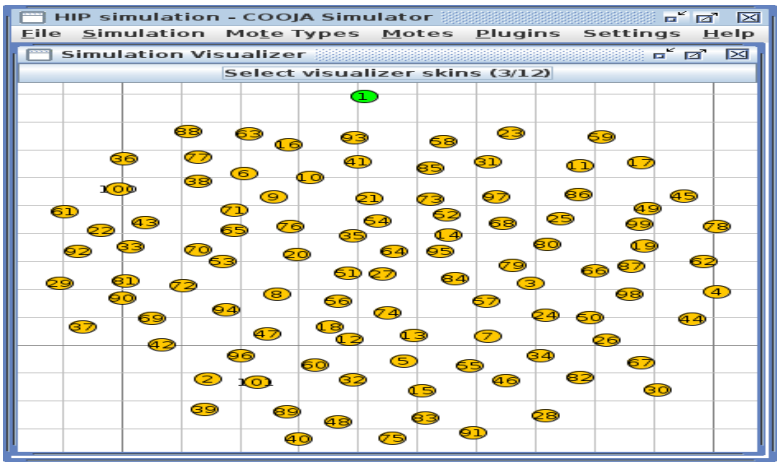


Figure 7.2. Modèle du réseau : le nœud vert portant le numéro 1représente le routeur de bordure et les 100 nœuds jaunes sont les nœuds capteurs.

[6] (Ordinateur *HP G62*, RAM de 4 Go, mémoire cache 3 Mo, processeur dual-core, CPU Intel Core i5 430M / 2.26 GHz)

Les simulations réalisées dans cette partie peuvent être divisées en deux parties:

- Nous évaluons d'abord le modèle de compression 6LoWPAN proposé pour l'entête HIP.
- Ensuite, nous menons une évaluation approfondie sur CD-HIP.

Comme la consommation d'énergie est un critère décisif de l'efficacité des solutions destinées aux réseaux de capteurs, dans nos évaluations, nous nous basons principalement sur la quantification de la consommation d'énergie par le nœud capteur pour réaliser à la fois la communication des paquets et le calcul de la clé secrète de la session HIP.

3.1. Evaluation du modèle de compression proposé

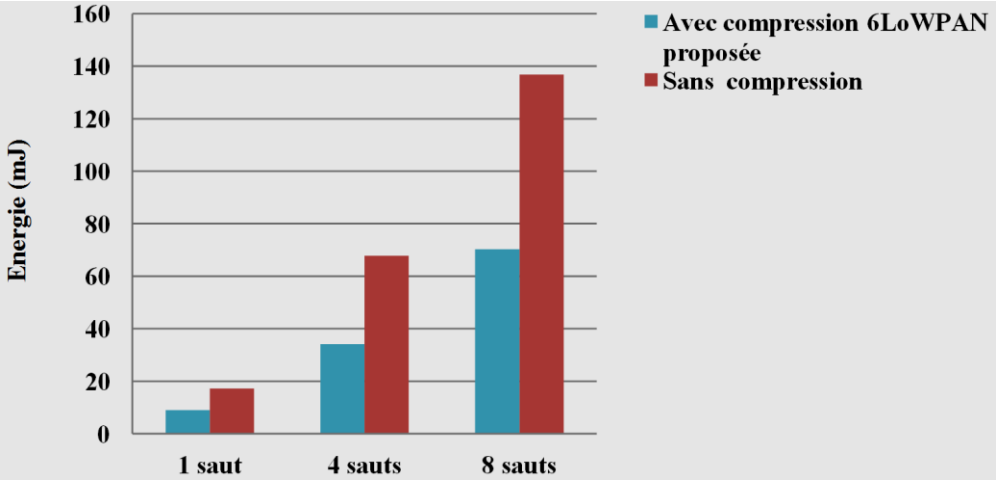
Nous évaluons d'abord le gain énergétique du modèle de compression 6LoWPAN proposé pour HIP. L'énergie de la communication est calculée avec l'équation suivante, où *Tx* et *Tr* sont respectivement les temps de transmission et du temps d'écoute.

$$EnergieComm(mJ) = \frac{[(Tx*17.7mA)+(Tr*20mA)]}{32768}*3V \qquad (7.2)$$

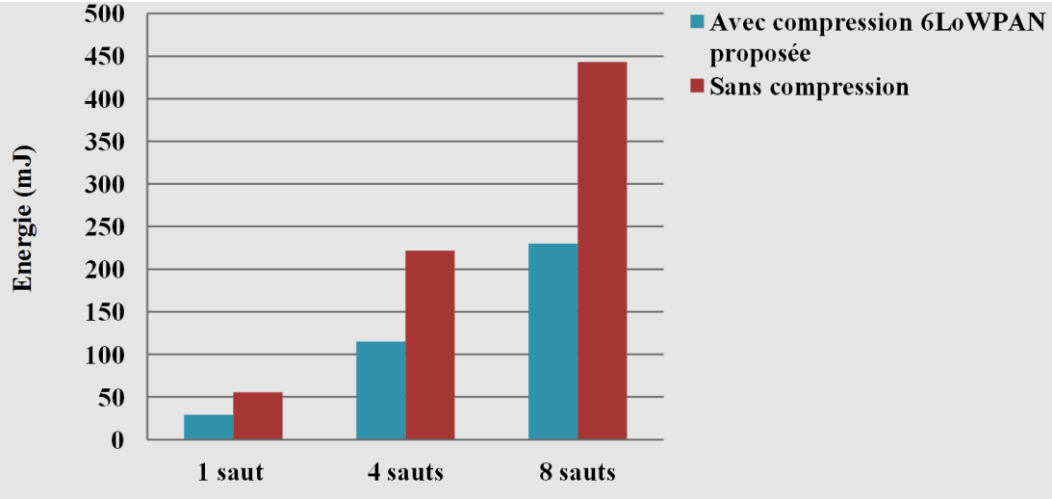
Le tableau ci-dessous résume les résultats d'évaluation de la consommation d'énergie pour la communication des entêtes HIP dans les deux cas: avec et sans le modèle de compression 6LoWPAN proposé. Cette évaluation porte sur les paquets échangés lors de la phase HIP-BEX, ainsi que la communication de tous les paquets HIP. La figure 7.3 représente les résultats obtenus.

Table 7.2. Résumé des coûts énergétiques de la communication de l'entête HIP.

	Parquets HIP-BEX (mJ)			**Tous les paquets HIP (mJ)**		
	1 saut	**4 sauts**	**8 sauts**	**1 saut**	**4 sauts**	**8 sauts**
Avec compression	9.032	34.128	70.234	29.237	114.971	230.031
Sans compression	17.225	67.8	136.803	55.368	221.47	442.946



(a)



(b)

Figure 7.3. La surcharge de communication de l'entete HIP avec (a) les paquets du processus HIP-BEX et (b) tous les paquets HIP.

Les résultats d'évaluation montrent qu'avec le modèle de compression proposé, le coût des communications devient extrêmement faible soit pour la communication des quatre messages du processus HIP-BEX soit pour la communication de tous les messages HIP. Ceci est particulièrement tangible lorsque la communication entre le nœud capteur terminal et le routeur de bordure 6BR se fait en multi-sauts.

Les résultats obtenus montrent également que sans compression, même avec une seule session HIP établie dans le réseau, les coûts énergétiques de communication, ainsi que la surcharge du réseau induite sont importants, spécialement dans le cas de communications multi-sauts entre le routeur de bordure et le répondeur HIP (le nœud capteur extrême). Ainsi, plus le taux de sessions HIP en cours d'établissement avec les nœuds capteurs augmente dans le réseau 6LoWPAN, plus le coût énergétique de la communication des paquets HIP non compressés n'est négligeable.

Suite à cela, nous avons évalué le total de dissipation d'énergie dans le réseau 6LoWPAN pour la communication de l'entête HIP pendant la phase HIP-BEX, et avec deux messages HIP supplémentaires qui sont nécessaires pour la fermeture de session (les messages CLOSE et CLOSE_ACK). Le nombre de sessions HIP établies est variable et incrémental, tandis que les répondeurs HIP sollicités (nœuds capteurs terminaux) sont liés au 6BR avec différentes distances (sauts) de communication. La figure ci-dessous présente les résultats obtenus.

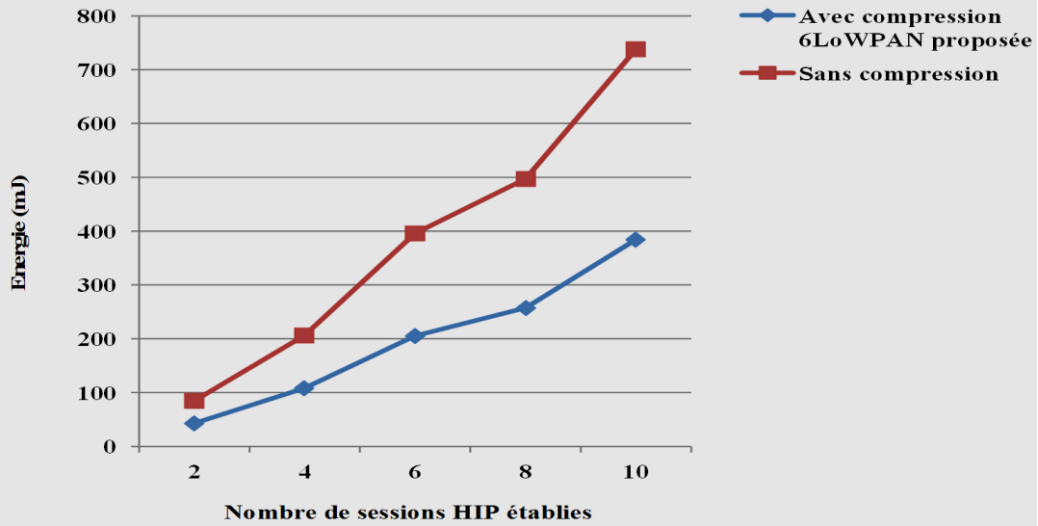


Figure 7.4. Surcharge générale de communication de l'entête HIP pour l'établissement des sessions HIP dans le réseau 6LoWPAN.

Nous remarquons à partir de la figure qu'à chaque fois le nombre de sessions HIP mises en œuvre dans le réseau 6LoWPAN est important, le total de la consommation d'énergie dans le réseau pour la communication augmente senssiblement sans la considération du modèle de compression proposé. Ceci n'est pas le cas avec la communication des messages HIP contenant des entêtes compressés (selon la solution proposée) où les taux de consommation d'énergie augmentent légèrement permettant des gains énergétiques très importants.

3.2. Evaluation de CD-HIP

Les performances du protocole CD-HIP sont évaluées avec les mêmes conditions énoncées dans le premier paragraphe de la section 3. Pour le matériel de sécurité initial *Diffie-Hellman,* nous avons considéré le groupe MODP 1536 bits avec des exposants de 180 bits, ce qui assure un niveau de sécurité acceptable [122]. Pour les signatures numériques qui doivent être effectuées par le nœud capteur terminal (répondeur HIP), l'algorithme ECDSA-163 bits [123] avec l'algorithme SHA-224 ont été utilisés. Les algorithmes de signature numérique à base de courbes elliptiques ECDSA (*Elliptic Curve Digital Signature Algorithm*) sont supportés dans les implémentations du protocole HIP et ils sont avantageux en termes de temps de calcul et de consommation d'énergie [84]. Nous avons utilisé l'algorithme SHA-224 (*Secure Hash Algorithm*) qui appartient à la famille des fonctions de hachage SHA-2 (SHA-224, SHA-256, SHA-384 et SHA-512) [124] qui sont plus efficaces et plus résistantes aux collisions, que leur prédécesseur SHA-1. Une autre raison de choisir l'algorithme SHA-224 est qu'il génère des valeurs de hachage pratiquement plus courtes (codées sur 224 bits).

De plus, nous employons l'algorithme de chiffrement AES-128 (*Advanced Encryption Standard* avec une clé de 128 bits) [67] qui est suffisamment robuste et qui offre une très bonne sécurité pour protéger uniquement les informations les plus critiques et les plus exigeantes en sécurité dans les messages échangés entre les nœuds capteurs terminaux et les collaborateurs. Rappelons qu'AES est optionnellement supporté pour la sécurité au niveau de la couche MAC dans les réseaux de capteurs

adoptant la technologie IEEE 802.15.4 [110]. Donc nous n'avons plus besoin d'intégrer un autre algorithme de sécurité et de surcharger la mémoire des nœuds capteurs.

Nous comparons CD-HIP avec le standard HIP et la solution de distribution de HIP-BEX proposée dans [87] (avec quatre nœuds assistants). Ainsi, nous comparons notre solution avec ses deux solutions partielles: HIP compressé (C-HIP) qui fait référence à HIP implémentant seulement le modèle de compression proposé (sans distribution), et HIP distribué (D-HIP) n'intégrant que le modèle de distribution proposé (sans le modèle de compression d'entête proposé). Les critères d'évaluation sont basés sur l'énergie consommée par le nœud capteur terminal pour la communication de messages HIP et pour effectuer les calculs nécessaires à la dérivation de la clé de session. Ainsi, l'énergie totale consommée par le protocole proposé (CD-HIP) est estimée. En outre, nous considérons dans notre évaluation l'impact de notre solution sur le délai moyen de l'établissement de la session de sécurité, et enfin, nous mesurons les exigences en matière d'espace mémoire nécessaire pour CD-HIP (ROM et RAM).

3.2.1. Évaluation de la consommation d'énergie

Le coût de communication dans CD-HIP et les autres solutions est estimé suivant la formule (7.2) tandis que le cout des calculs est évalué selon l'équation suivante, où *Tcpu* représente le temps écoulé dans les traitements CPU.

$$EnergieCompt(mJ) = \frac{Tcpu}{32768} * 1.8mA * 3V \qquad (7.3)$$

La figure suivante représente les résultats relatifs aux coûts énergétiques de communication des messages HIP et de calculs cryptographiques dans CD-HIP et les autres solutions.

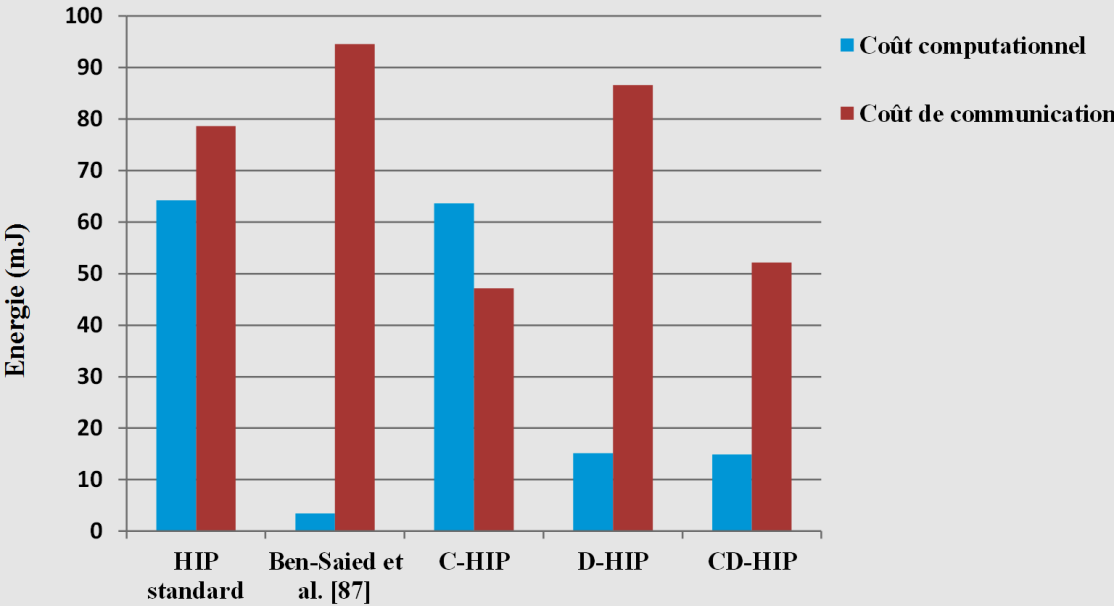


Figure 7.5. La consommation d'énergie pour les communications et les calculs.

De la figure ci-dessus, on constate que les solutions qui n'incluent aucun mécanisme de compression des messages (le HIP standard, la solution de Ben-Saied et.al. [87] et D-HIP) présentent

des coûts de communication importants. Néanmoins, la solution de Ben-Saied et.al. a le taux le plus élevé, en raison de la communication de plusieurs messages de signalisation entre le nœud capteur extrême et l'ensemble de collaborateurs durant le processus de distribution de la sécurité. Par conséquent, la surcharge globale de communication devrait être très importante surtout en cas d'établissements fréquents de sessions HIP avec les nœuds capteurs dans le réseau de capteurs connecté à Internet. Aussi, toute solution supplémentaire qui peut être définie pour assurer la fiabilité de communications des messages de distribution risque d'augmenter encore plus la surcharge de communication dans le RCSF.

Avec C-HIP, l'énergie consommée pour la communication des messages HIP (transmissions et réceptions avec écoute) est très faible, grâce au mécanisme de compression proposé pour l'entête HIP. Cependant, le coût de calcul est considérable, tout à fait comme dans le standard HIP, où toutes les opérations cryptographiques lourdes (essentiellement de l'algorithme *Diffie-Hellman*) sont réalisées par le nœud capteur terminal. En revanche, les coûts énergétiques relatifs aux traitements sont minimes dans la solution Ben-Saied et al. [87] à cause de l'implémentation du mécanisme de distribution de sécurité qui attribue le calcul des exponentiations *Diffie-Hellman* et aussi le calcul de signatures des messages HIP aux collaborateurs. D'autre part, comme la charge de calcul de la sécurité dans notre modèle de distribution est répartie entre le collaborateur et le nœud de capteur terminal, les coûts de calculs avec les protocoles D-HIP et CD-HIP qui adoptent le modèle de distribution proposé sont réduits, et sont liés aux opérations cryptographiques restantes (principalement le calcul des signatures qui doit être effectué par le nœud de capteur (HIP répondeur) comme dicté dans le standard HIP.

Avec CD-HIP, les résultats obtenus montrent une amélioration significative de la consommation d'énergie à la fois pour la communication et pour les calculs, avec un coût de communication légèrement supérieure à celui obtenu avec C-HIP. Cela est dû à la communication obligatoire des messages supplémentaires qui sont primordiaux pour le processus de distribution entre le répondeur HIP et le collaborateur.

La consommation totale d'énergie par le nœud capteur (répondeur HIP) pour l'établissement de la sécurité HIP est estimée avec l'équation suivante:

$$Energie(mJ) = EnergieComm + EnergieCompt \qquad (7.4)$$

Notons que nous n'avons pas examiné la dissipation d'énergie pendant le mode de mise en veille LPM, car il a un impact négligeable sur les résultats obtenus.

La figure suivante présente le total d'énergie consommée par le pair HIP (répondeur) capteur dans CD-HIP et dans le reste de solutions.

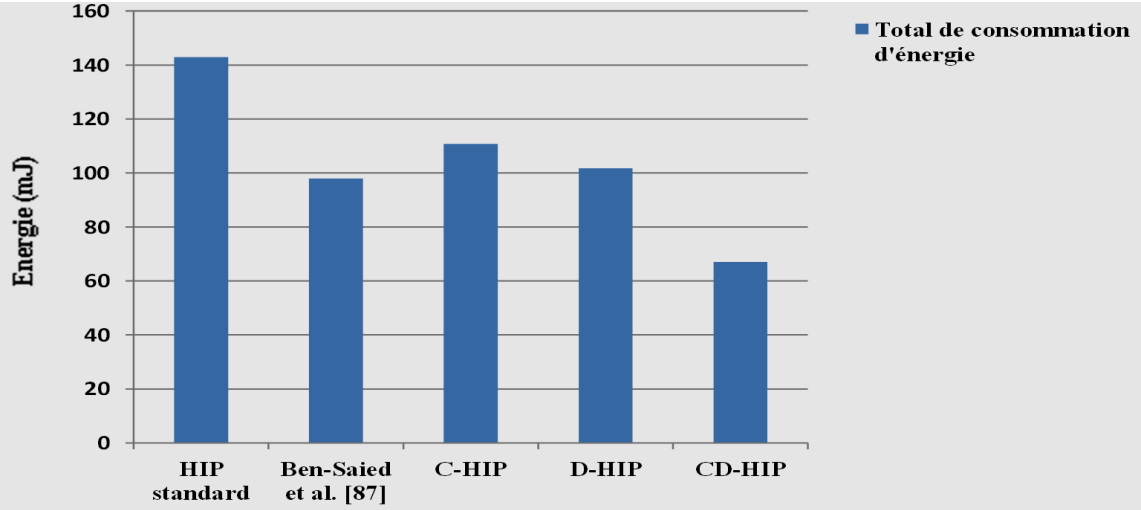


Figure 7.6. Le total de consommation d'énergie.

Par la combinaison de deux techniques d'adaptation proposées (la compression 6LoWPAN et la répartition de charge cryptographique) dans le standard HIP, les avantages des deux techniques sont également alliés. Par conséquent, la consommation globale d'énergie est minimale et beaucoup plus raisonnable, ce qui rend CD-HIP un protocole de sécurité économique en énergie dans l'Internet des objets.

3.2.2. Estimation du délai d'établissement de la session de sécurité

Nous évaluons les performances du protocole CD-HIP aussi selon le délai moyen d'établissement de session de sécurité, dans le réseau 6LoWPAN. Ce délai moyen est estimé par rapport aux délais de session mesurés en 1 saut, 4 et 8 sauts de communications entre les nœuds capteurs terminaux et le routeur de bordure 6LoWPAN. Comme le système de distribution proposé par Ben-Saied et al. dans [87] est concentré sur une structure de communication multi-chemins, le délai de l'établissement de la session HIP pour cette solution n'est pas mesuré en moyenne. Les résultats obtenus sont présentés dans la figure ci-dessous.

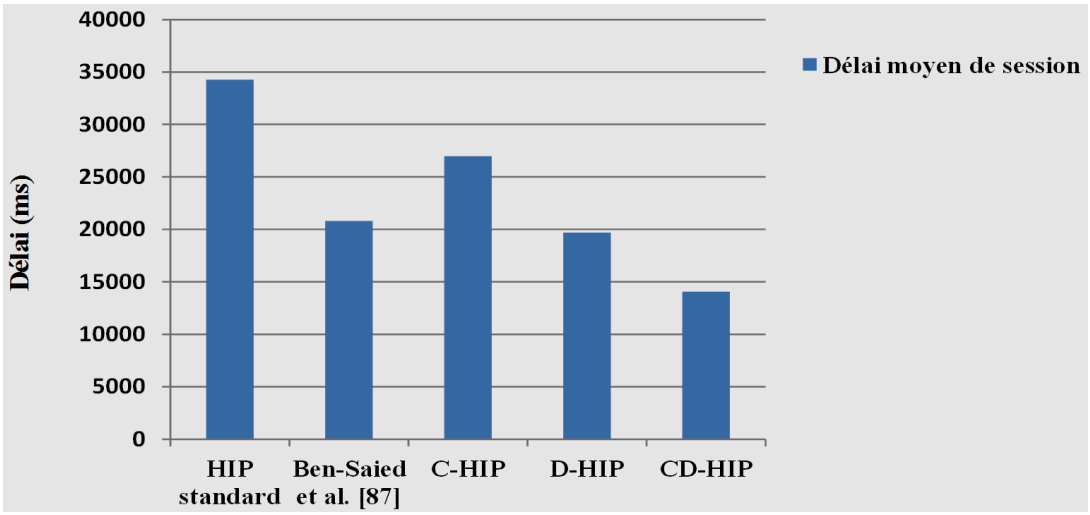


Figure 7.7. Le délai moyen de l'établissement de la session de sécurité dans CD-HIP.

Les courts délais de transit des paquets HIP compressés, ainsi que la génération accélérée du secret de session, conduisent à diminuer considérablement le délai de session dans CD-HIP. Ceci n'est pas le cas avec les autres solutions qui sont caractérisées par des délais de mise en place de session pratiquement importants, car ils nécessitent beaucoup de temps pour effectuer les primitives cryptographiques lentes dans mécanisme HIP-BEX, et / ou pour communiquer des messages HIP non compressés. Cependant, il est clairement remarquable que les délais moyens de session dans C-HIP et dans le standard HIP sont beaucoup plus importants que dans les solutions supportant la distribution, en raison de la lourdeur des opérations cryptographiques asymétriques induites. De ces résultats, on déduit que CD-HIP est approprié pour la sécurisation des applications sensible au temps dans l'IoT.

Le tableau 7.3 récapitule les résultats d'évaluation de CD-HIP.

Table 7.3. Résumé des résultats d'évaluation obtenus.

Protocole	Coût Computationnel (mJ)	Coût des Communications (mJ)	Total du coût énergétique (mJ)	Délai moyen de session (ms)
Standard HIP	64.231	78.67	142.901	34275
Ben-Saied et al. [87]	3.43	94.55	97.98	20817
C-HIP	63.622	47.186	110.808	26992
D-HIP	15.17	86.59	101.76	19700
CD-HIP	14.93	52.16	67.09	14056

3.2.3. Estimation de l'empreinte mémoire

Afin de quantifier le besoin en mémoire ROM et RAM par notre solution, nous avons utilisé l'outil *msp430-size*. D'après les résultats présentés dans le tableau ci-dessous, on trouve que l'occupation mémoire dans CD-HIP est modeste elle est environ 11,7 Ko de ROM et 2 Ko de RAM, grâce à l'adoption du modèle de compression 6LoWPAN proposé. En fait, ces résultats peuvent être prohibitifs pour les dispositifs capteurs extrêmement limités en mémoire, comme les plateformes *Tmote Sky* et *TeloseB* ayant tous deux 10 Ko de RAM, 48 Ko de ROM. Néanmoins, nous utilisons ces nœuds capteurs seulement pour faire des tests par simulation. Pour tester note solution réellement, nous projetons d'adopter des plateformes capteurs matérielles plus puissantes comme *Zolertia* Z1 (8 ko de RAM, 92 Ko de ROM) ou *WiSMote* (16 ko de RAM, 128 kB de ROM), sur lesquelles l'empreinte mémoire de CD-HIP apparaît beaucoup plus faible. Le tableau ci-dessous présente les besoins en mémoire pour CD-HIP.

Table 7.4. Besoins en mémoire dans la solution proposée.

Extension	ROM (octets)	RAM (octets)
Contiki OS	32145	4979
Contiki avec le standard HIP	46525 (+14380)	7242 (+2263)
Contiki avec CD-HIP	43917 (+11772)	7096 (+2117)

4. Evaluation du système de sécurité asymétrique et sélective pour la sécurité des communications Humain-à-objet

Cette partie est consacrée à l'évaluation du système de sécurité asymétrique et sélective proposé pour les communications Humain-à-objets dans l'Internet du futur. On présente d'abord le contexte général d'évaluation de la solution ensuite, on discute les résultats obtenus.

4.1. Contexte d'implémentation et d'évaluation

Nous avons évalué la solution en utilisant le même simulateur (Cooja de Contiki 2.5) et la même plateforme capteur (TMote Sky) utilisés pour l'évaluation de CD-HIP, sauf que cette fois-ci, nous considérons plutôt un réseau 6LoWPAN multimodal composés de nœuds capteurs qui jouent le rôle de serveurs CoAP fournissant chacun deux ressources : la lecture de la température et la lecture de la lumière. Les librairies `dev/light-sensor.c|.h` et `dev/sht11-sensor.c|.h` définies par Contiki, sont utilisée pour le captage de la lumière et la température respectivement.

De plus, les règles de conversion entre HTTP et CoAP dictées dans [118] sont ajoutées au programme du routeur de bordure pour qu'il puisse agir comme proxy. Nous avons déjà mentionné que le routeur de bordure s'exécute sur la machine virtuelle Linux (distribution Ubuntu 10.04) et connectée à Cooja par le biais d'un socket série. Les requêtes HTTP sont générées localement par le proxy va jouer le rôle d'un client HTTP. Donc, il traduit les requêtes avant de les communiquer vers le serveur CoAP ciblé et traduit les réponses CoAP reçues en des réponses HTTP, éventuellement après les déchiffrer.

Concernant le matériel sécuritaire, nous avons utilisé le protocole IPsec compressé proposé dans [79] avec pré-chargement de la clé de session entre serveur CoAP et le client HTTP, et avec adoption de CD-HIP pour une session de sécurité dynamiquement établie. Pour le chiffrement de données, l'algorithme AES-128 est employé.

Nous évaluons la surcharge de sécurité dans le coté du serveur CoAP (nœud capteur) avec la solution de sécurité de bout-en-bout proposée pour les communications humain-à-objet. Nous assumons un scénario de communication H2T dans lequel le client HTTP envoie les requêtes à une fréquence incrémentale. D'abord une seule requête est envoyée au serveur CoAP une fois toutes les cinq et deux secondes et puis, une requête est envoyée chaque seconde. Le client (ou un ensemble de clients) demande à chaque fois la température et la lumière alternativement. Ces ressources sont censées être volatiles; elles ne devraient pas être mises en mémoire cache du proxy.

Nous avons aussi estimé l'impact de l'attaque par déni de service (DoS) qui est une attaque très grave menaçant les réseaux de capteurs connectés à Internet [125]. L'attaque est particulièrement dangereuse lorsqu'elle est exercée par des hôtes externes puissants. Nous supposons alors un client HTTP malveillant qui transmet des requêtes en rafale et à une fréquence croissante vers un serveur CoAP contraint.

L'évaluation du délai des communications (le paramètre RTT: *Round Trip Time*) avec la solution proposée est également nécessaire car le rapport entre la sécurité et la qualité de service des communications est très important dans les réseaux orientés services, tout comme les réseaux 6LoWPANs dans l'IoT.

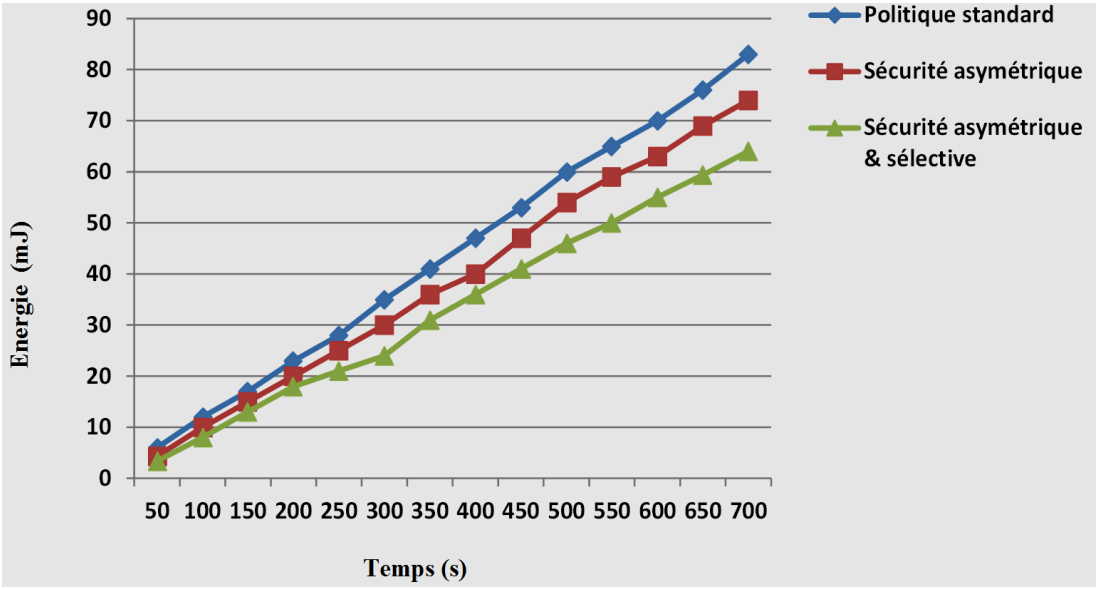
Il est à noter également que la température est configurée pour être la ressource la plus critique. C'est-à-dire que seules les réponses CoAP transportant les lectures de température sont concernées par la sécurité au niveau du serveur CoAP. Le temps de simulation est fixé à 700 secondes.

4.2. Résultats et discussion

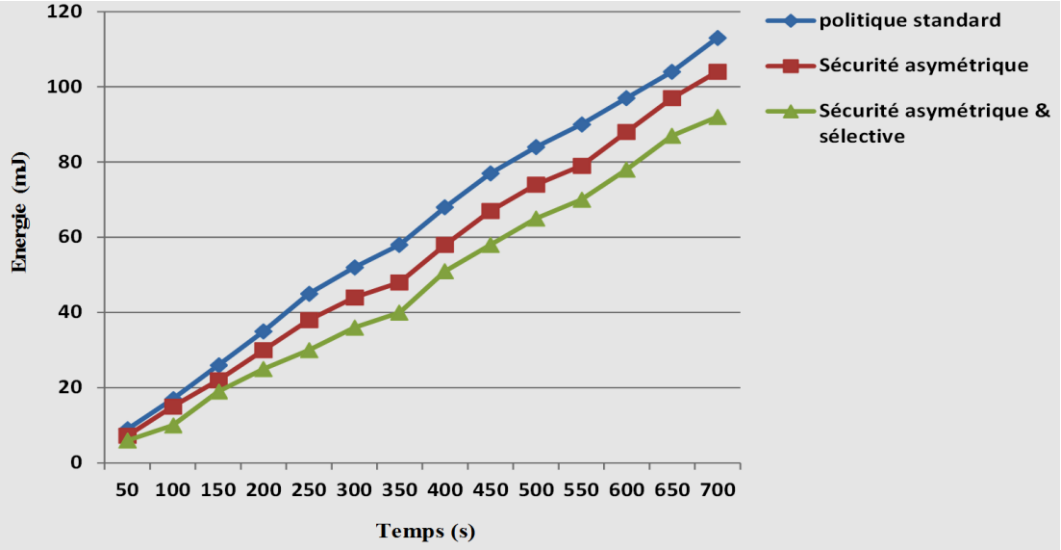
Dans cette section, nous présentons les résultats d'évaluation de la solution proposée pour une sécurité de bout-en-bout adaptée des communications humain-à-objet. Dans nos évaluations, nous nous sommes concentrés principalement sur l'estimation de la consommation d'énergie par le serveur CoAP adoptant le système de sécurité asymétrique et sélective proposé, avec et sans l'attaque DoS.

4.2.1. Résultats obtenus sans existence de l'attaque DoS

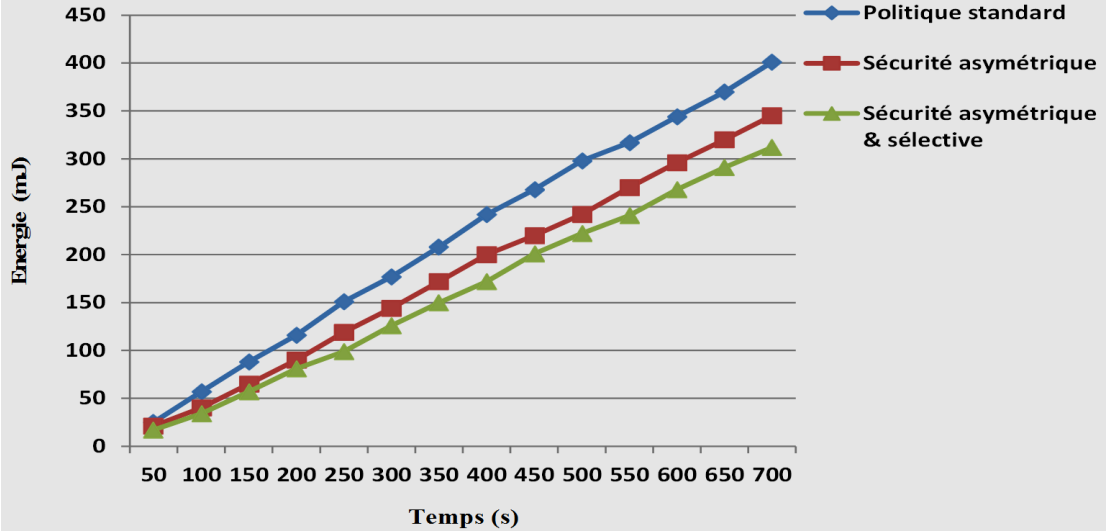
La figure 7.8 montre la surcharge computationnelle de la sécurité avec l'approche standard, où les requêtes HTTP et les réponses CoAP sont toutes sécurisées, et la sécurité de bout-en-bout est brisée dans le proxy pour effectuer les translations HTTP/CoAP. Le coût des calculs est également estimé et comparé avec la stratégie proposée (la sécurité asymétrique et sélective) où seules les réponses les plus sensibles sont sécurisées et le proxy se comporte dans ce cas juste comme un routeur 6BR.



(a)



(b)



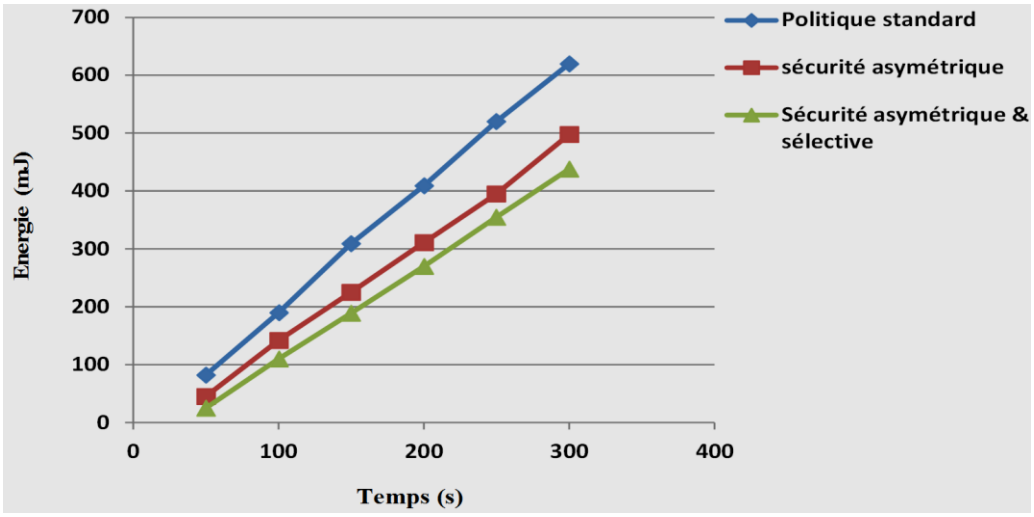
(c)

Figure 7.8. Le coût des calculs avec la solution proposée. **(a)** une requête est envoyée au serveur CoAP toutes les 5 secondes. **(b)** 1 requête est envoyée toutes les 2 secondes. **(c)** 1 requête est envoyée par seconde.

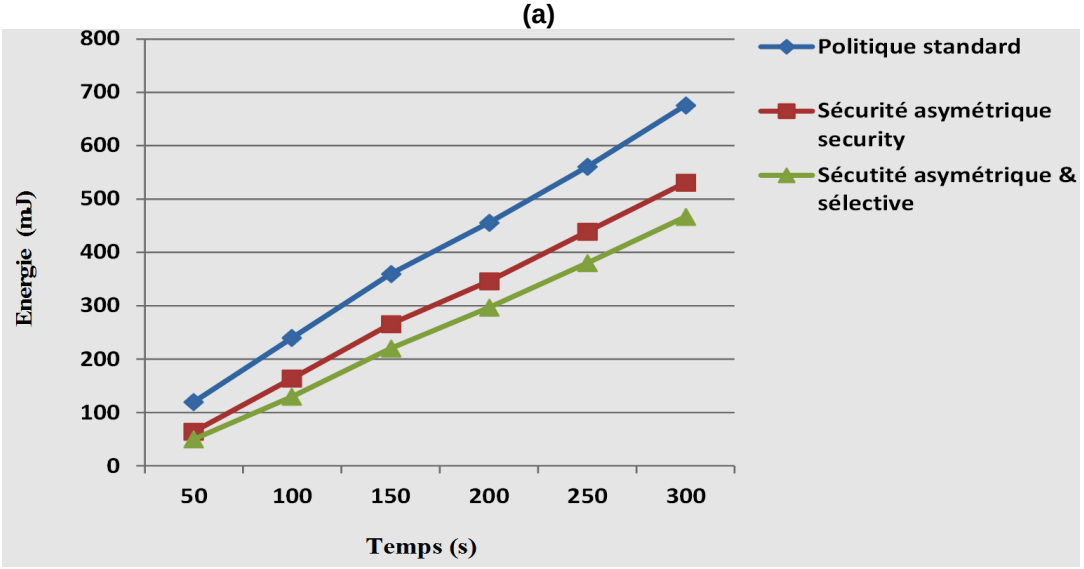
Nous remarquons à partir de la figure ci-dessus qu'avec la politique de sécurité standard où toutes les requêtes et les réponses sont sécurisées sans aucune distinction, les coûts énergétiques sont considérables. Cependant, en adoptant la sécurité asymétrique où la surcharge sécuritaire diminue sensiblement. En outre, l'application de la sélectivité de la sécurité conjointement avec la politique asymétrique permet une optimisation tangible de la consommation énergétique relative à la sécurité au niveau du serveur CoAP.

4.2.2. Résultats obtenus avec l'attaque DoS

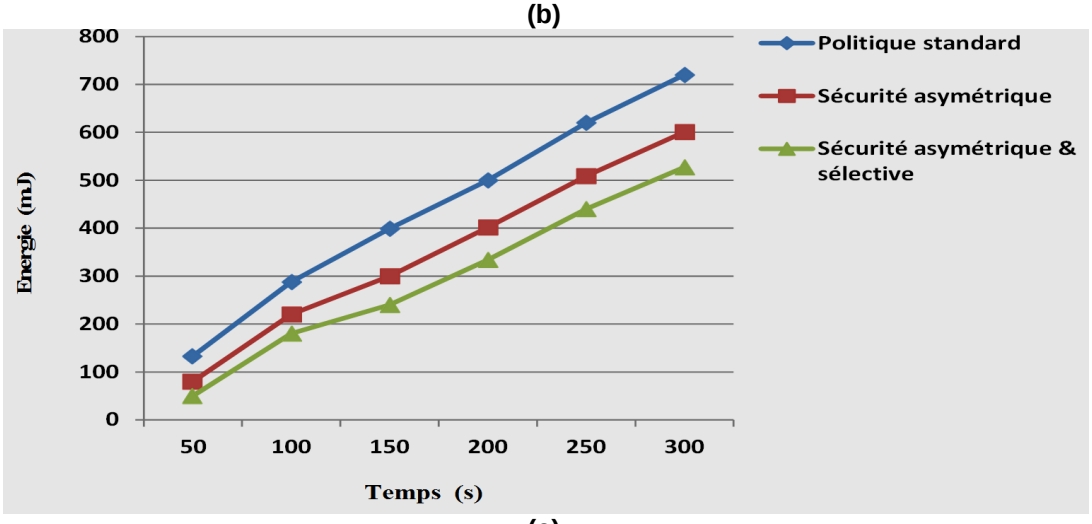
D'autre part, la figure 7.9 montre qu'en cas d'une attaque par déni de service visant le serveur CoAP, la solution proposée permet d'atténuer considérablement la consommation d'énergie induite.



(a)



(b)



(c)

Figure 7.9. L'impact de l'attaque DoS sur le serveur CoAP. **(a)** le client envoie 5 requêtes par seconde. **(b)** le client envoie 10 requêtes par seconde. **(c)** le client envoie 15 requêtes par seconde.

4.2.3. Le coût sécuritaire dans un intervalle de 5 secondes

De plus, nous avons estimé le coût de la sécurité dans un intervalle de 5 secondes avec différentes fréquences d'écoulement des requêtes vers le serveur CoAP dans tous les cas considérés (1 requête par 5 secondes, une requête par 2 secondes, une requête par seconde et 5, 10, 15 requêtes par seconde). La figure 7.10 représente les résultats obtenus qui confirment qu'évidemment avec la politique standard, même si le nombre de requêtes reçues est réduit, les coûts énergétiques sont non-négligeables. En revanche, la stratégie proposée affaiblit considérablement les niveaux de consommation d'énergie nécessaire pour la sécurité des communications humain-à-objet se déroulant entre un client HTTP puissant et un serveur CoAP contraint.

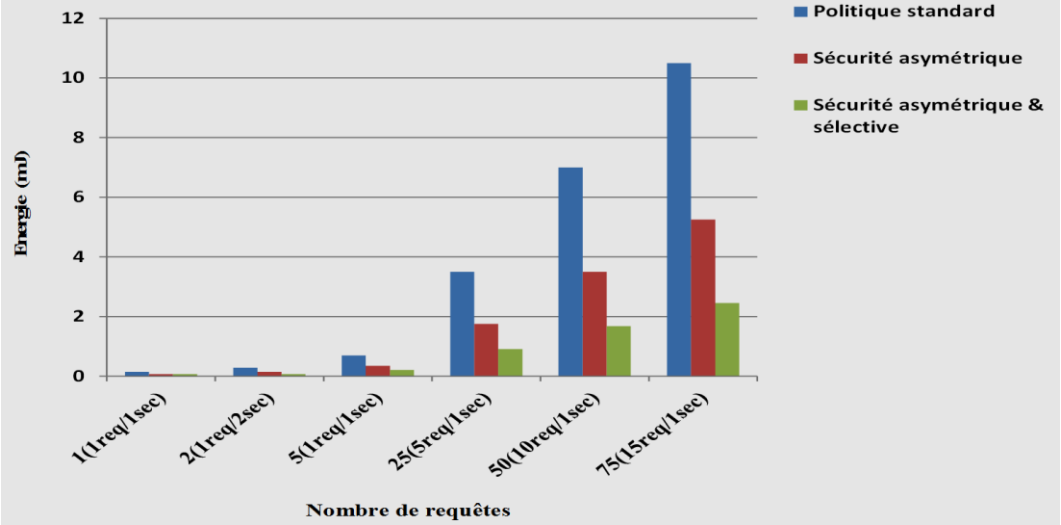


Figure 7.10. Estimation du coût de la sécurité dans une période de 5 secondes par rapport au nombre de requêtes reçues par le serveur CoAP dans chacune des fréquences considérées.

4.2.4. Récapitulation des coûts énergétiques du système proposé

Dans nos évaluations, nous considérons également la surcharge des communications comme un critère non négligeable d'évaluation de la solution proposée pour la sécurité des communications H2T. En effet, dans la solution proposée, les messages qui ne sont pas concernés par la sécurité (toutes les requêtes et les réponses non critiques) sont communiqués sans l'encapsulation de IPsec, ce qui pourrait réduire la surcharge de communication. Le tableau 7.5 présente le coût total des calculs et des communications avec les stratégies proposées et l'approche classique, sous les fréquences d'interaction considérées.

Comme nous utilisons le protocole IPsec compressé (proposé dans [79]) du côté du réseau 6LoWPAN, les différences dans les coûts énergétiques des communications sont faibles avec des interactions moins fréquentes. Par ailleurs, dans le cas des communications intensives entre les

clients HTTP et les serveurs CoAP, le gain en termes de surcharge de communication devient important, comme indiqué dans le tableau 7.5.

Aussi, Tableau ci-dessous présente les estimations des coûts généraux de sécurité dans deux cas : 1) avec une clé de session pré-partagée entre l'hôte HTTP externe et le serveur CoAP, et 2) avec le mécanisme d'établissement dynamique de la sécurité de bout-en-bout que nous avons proposé (CD-HIP) [107-108], qui induit une consommation énergétique supplémentaire d'environ 67 mJ.

D'après les résultats obtenus, nous constatons que le gain énergétique avec la solution proposée (dans les deux cas d'établissement de sécurité statique et automatique) est important, et peut arriver jusqu'à 20% de taux de réservation d'énergie.

Table 7.5. Résumé des résultats des évaluations effectuées.

Fréquence de communication	Solution	Total du coût computationnel [mJ]	Total des coûts de communic-ation [mJ]	Total du coût de la sécurité (clé pré-chargée) [mJ]	Total du coût de la sécurité (CD-HIP) [mJ]
1 Req / 5 secs	Politique standard	83	242	325	392
	Sécurité Asym.	74	231	305	372
	Asym & Sélective	64	224	288	355
1 Req / 2 secs	Politique standard	113	483	596	663
	Sécurité Asym.	104	470	574	641
	Asym & Sélective	92	459	551	618
1 Req / 1 sec	Politique standard	426	867	1293	1360
	Sécurité Asym.	345	839	1184	1251
	Asym & Sélective	312	804	1116	1183
5 Reqs / 1sec	Politique standard	620	1204	1824	1891
	Sécurité Asym.	498	1161	1659	1726
	Asym & Sélective	438	1137	1575	1642
10 Reqs / 1 sec	Politique standard	686	1477	2163	2230
	Sécurité Asym.	531	1419	1950	2017
	Asym & Sélective	465	1398	1863	1930
15 Reqs / 1 sec	Politique standard	729	1652	2381	2448
	Sécurité Asym.	601	1613	2214	2281
	Asym & Sélective	527	1584	2111	2178

4.2.5. Estimation du délai moyen des communications H2T avec le système proposé

Cette fois-ci, il s'agit d'estimer le paramètre RTT qui représente dans notre cas le temps écoulé entre l'envoi de la requête HTTP et la réception de la réponse HTTP traduite localement, à partir de la réponse CoAP générée par le serveur CoAP. La solution proposée peut améliorer ce paramètre car les différentes parties impliquées effectuent moins de tâches sécuritaires par rapport à l'approche standard, comme le montre la figure et le tableau ci-dessous.

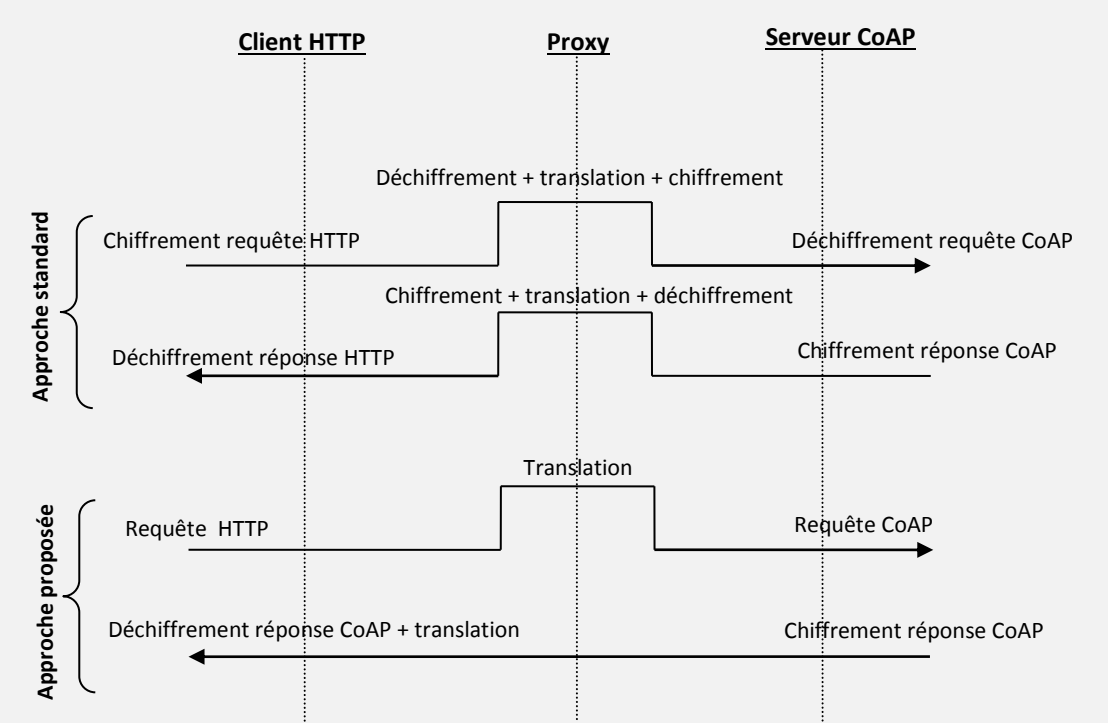


Figure 7.11. Schéma indiquant les différentes opérations effectuées lors d'une communication H2T.

Le tableau suivant présente l'ensemble accumulatif des actions réalisées par chaque entité et le RTT estimé avec la solution standard et la solution de sécurité asymétrique proposée.

Table 7.6. Résumé des opérations réalisées au niveau de chaque partie avec estimation du RTT.

	Client HTTP	**Proxy**	**Serveur CoAP**	**RTT (ms)**
Approche standard	-chiffrer (Req) -déchiffrer (Rép)	-déchiff+translat+chiff (Req) -déchiff+translat+chiff (Rép)	-déchiffrer (Req) -chiffrer (Rép)	1458
Approche asymétrique	-déchiffrer (Rép) -translation (Rép)	- translation (Req)	-chiffrer (Rép)	1247

Pour l'estimation du paramètre RTT, nous avons pris le cas d'un serveur CoAP qui est situé loin à trois sauts du proxy. Le délai de transit entre les deux nœuds est de 551 ms. La taille des messages CoAP échangés varie entre 13 et 18 octets et la taille maximale de la partie à sécuriser dans un datagramme 6LoWPAN encapsulé par IPsec vaut 39 octets. Le temps de chiffrement et de déchiffrement sont estimé à 90 ms et 97 ms, respectivement.

4. Conclusion

Nous avons présenté à travers ce chapitre les résultats d'évaluation de performances des solutions proposées pour assurer la sécurité des réseaux de capteurs 6LoWPAN intégrés à l'Internet des objets. Les résultats obtenus montrent clairement que CD-HIP, qui inclut le premier modèle de compression 6LoWPAN pour HIP et qui est la première solution qui combine la compression avec la

répartition des calculs cryptographiques, est bien adapté pour la mise en place d'une session de sécurité de bout-en-bout avec les nœuds capteurs connectés à Internet.

Les résultats d'évaluation ont également montré l'efficacité du système de sécurité asymétrique et sélective pour une protection de bout-en-bout des communications humain-à-objet, avec un coût nettement raisonnable. La solution proposée considère des communications qui tournent entre des clients HTTP puissants et des serveurs CoAP contraints (nœuds capteurs) dans l'IoT. Elle exploite les hétérogénéités technologiques et matérielles entre les dispositifs impliqués dans tel type d'interactions pour permettre une sécurité de bout-en-bout effective, avec une atténuation considérable de l'impact des attaques par déni de service sur les nœuds capteurs dans le réseau 6LoWPAN. La solution proposée peut même être appliquée pour optimiser la sécurité des communications objet-à-objet entre des clients et des serveurs CoAP.

Conclusion générale

L'Internet des objets est une évolution de l'internet actuel qui est née de la convergence de plusieurs types de réseaux et de technologies, en particulier l'Internet IPv6, les réseaux de capteurs sans fil et la technologie d'identification par radiofréquence RFID. En effet, les réseaux de capteurs représentent la partie la plus intéressante parmi l'ensemble des technologies fondatrices de l'IoT. Ils ont déjà réalisé un succès remarquable dans différents domaines d'applications urbaines, rurales, civiles et militaires. Et avec l'intégration à Internet, leurs avantages et rendements applicatifs sont prévus à prendre un espace beaucoup plus large avec de nouvelles perspectives.

Un avantage majeur de l'incorporation des réseaux de capteurs dans l'Internet, comme une partie de l'Internet des objets, est que l'écart entre le monde physique composé d'une diversité d'objets et le monde virtuel de l'Internet puisse être comblé. Donc les données de captage (représentant des informations contextuelles ou comportementales concernant les objets ou les endroits physiques) peuvent être désormais accédées d'une manière ubiquitaire (à n'importe quand et de n'importe où). Cette ubiquité d'accès aux services des nœuds capteurs améliore bel et bien la qualité des services et le mode de vie des gens qui se retrouvent entourés par une grande masse de dispositifs intelligents. Cependant, l'omniprésence des nœuds capteurs ouverts à l'Internet comporte une contrepartie préoccupante sur la sécurité des données récoltées (qui sont généralement assez critiques) et la vie privées des utilisateurs de l'IoT.

Avec l'émergence de l'IoT, les risques en termes de sécurité deviennent énormes avec un degré élevé de diversification et de gravité. D'autre part, les contraintes imposées sur les réseaux de capteurs (principalement les limitations de ressources) empêchent la mise en place des mécanismes de sécurité hautement robustes car ils requièrent des dispositifs puissants, ce qui n'est pas le cas pour les capteurs contraints.

L'objet de cette thèse étant de proposer des solutions de sécurité pour la protection des réseaux de capteurs connectés à l'IoT. Dans ce contexte, nous avons proposé une solution efficace pour assurer la sécurité de bout-en-bout des communications entre les hôtes ordinaires sur Internet et les nœuds capteurs. La solution est basée sur le protocole HIP qui est intrinsèquement bénéfique pour l'IoT comme il supporte la mobilité des nœuds terminaux et il garantit un bon niveau de transparence à la localisation des hôtes IP. La solution proposée combine le premier modèle de compression 6LoWPAN pour les messages HIP avec un modèle approprié pour la répartition de la charge sécuritaire incluse dans le processus d'établissement de sécurité de bout-en-bout dans HIP. La solution est également la première qui combine les deux techniques d'adaptation pour les protocoles de sécurité destinés à l'IoT.

La deuxième solution proposée consiste en un système de sécurité asymétrique et sélective pour les communications humain-à-objets dans l'IoT. La solution exploite les différences de capacités entre les clients HTTP ordinaires et les serveurs CoAP contraints (des nœuds capteurs) pour concentrer la

sécurité uniquement sur le sens de communication emprunté par les messages les plus critiques (contenant les données de captage). La solution permet d'équilibrer la charge sécuritaire sur les nœuds capteurs tout en abaissant l'effet des attaques par déni de service.

Les résultats d'évaluation des solutions proposées ont montré l'efficacité et la robustesse des solutions proposées pour une sécurité adaptative des interactions avec les nœuds capteurs connectés à l'IoT.

La concrétisation de l'IoT, en tant que projet, est encore dans ses débuts. Ainsi, le problème de sécurité dans l'IoT est classé parmi les plus grands défis de cette concrétisation. En effet, Les risques et les menaces ne sont pas encore bien définis et il est même prévu que de nouvelles générations d'attaques émergeront avec divers degrés de gravité. Dans ce contexte, nous constatons que le rapport sécurité / interopérabilité est une véritable problématique émergeante dans l'IoT. D'une part la standardisation et l'unification des technologies de communication garantissent une meilleure interopérabilité, mais les problèmes de sécurité deviennent d'autant plus cruciaux. Et d'autre part, la diversification des protocoles de communications assure un bon niveau de protection des réseaux composant l'IoT, mais cette fois-ci l'interopérabilité est sensiblement affectée. Dans tous les cas, il serait nécessaire de développer des solutions sécuritaires qui soient à la fois robustes et averties des différentes contraintes de l'IoT. Donc, des travaux de recherche intensifs restent à entreprendre dans ce cadre pour surmonter les défis soulevés.

Comme perspectives, nous projetons d'intégrer les solutions proposées dans des applications réelles de l'IoT, à titre d'exemple : une application médicale ou une application de maison intelligente. D'autre part, et après avoir traité le cas de la sécurité des communications point-à-point dans l'IoT, nous nous concentrons actuellement sur le développement des solutions efficaces pour la sécurité des communications multicast dans le web d'objets intelligents et les communications du style *publish/subscribe*.

Dans nos prochains travaux de recherche, nous comptons proposer un nouveau modèle d'intégration sécurisée des réseaux de capteurs à l'IoT qui devrait être capable d'allier les avantages des modèles existants, tout en favorisant la sécurité des RCSFs. Nous allons également proposer d'autres solutions qui adressent la protection des réseaux de capteurs connectés à l'IoT contre divers types d'attaques externes, notamment les attaques de la classe déni de service (DoS, DDoS,…) qui représentent un risque majeur menaçant l'Internet des objets.

Bibliographie

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, computer networks 38 (4) (2002) 393-422.

[2] F. Akyildiz, W. S. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Communications, 2002.

[3] W. R. Heinzelman, A. Chandarkasan, H. Balakrishanan, Energy efficient communication protocol for wireless micro sensor networks, 33rd IEEE International Conference on System Sciences, Hawaii, 2000, pp. 1–10.

[4] A. Bilami, D.E. Boubiche, A hybrid energy aware routing algorithm for wireless sensor networks, IEEE Symposium on Computers and Communications ISCC, Marrakech, 2008, pp. 975-980.

[5] L. D. P. Mendes, J. J. P. C. Rodrigues, A survey on cross-layer solutions for wireless sensor networks, Journal of Network and Computer Applications 34 (2) (2011) 523–534.

[6] J. Yick, B. Mukherjee, D. Ghosal, Wireless sensor networ survey, Computer Networks 52 (12) (2008) 2292–2330.

[7] C. Karlof, D. Wagner, Secure routing in wireless sensor networks: attacks and countermeasures Ad Hoc Networks 1 (2) (2003) 293–315.

[8] Y. Wang, G. Attebury, B. Ramamurthy, A survey of security issues in Wireless Sensor Networks, IEEE communications surveys & tutorials 8 (2) (2006) 2-21.

[9] J. Lee, K. Kapitanova, S. H. Son, The price of security in wireless sensor networks, Computer Networks 54 (17) (2010) 2967–2978.

[10] http://cordis.europa.eu/project/rcn/95511_en.html. (consulté en Décembre 2015).

[11] T. Park, S. Member, K. G. Shin, Soft tamper-proofing via program integrity verification in wireless sensor networks, IEEE Transactions on mobile computing, 4 (3) (2005) 297-308.

[12] R. K. Sundararajan, U. Arumugam, Intrusion detection algorithm for mitigating sinkhole attack on LEACH protocol in wireless sensor networks, Journal of Sensors (2015) 1-12.

[13] S. Sahraoui, S. Bouam, Secure routing optimization in hierarchical cluster-based wireless sensor networks, International Journal of Communication Networks and Information Security (IJCNIS) 5 (3) (2013) 178-185.

[14] Y. Shen, S. Liu, Z. Zhang, Detection of Hello Flood Attack Caused by Malicious Cluster Heads on LEACH Protocol, International Journal of Advancements in Computing Technology 7 (2) (2015) 40-47.

[15] L. Batina, N. Mentens , K. Sakiyama, B. Preneel, I. Verbauwhede, Low-Cost Elliptic Curve Cryptography for Wireless Sensor Networks, Third European Workshop ESAS 2006, Hamburg, Germany, September 20-21, 2006, pp. 6-17.

[16] L Buttyán, L Dóra, WiFi Security–WEP and 802.11i, EURASIP Jthisnal on Wireless Communication and Networking (2006) 1-13.

[17] M. A. Simplício, P. S. L. M. Barreto, C. B. Margi , T. C. M. B. Carvalho, A survey on key management mechanisms for distributed Wireless Sensor Networks, Computer Networks 54 (15) (2010) 2591–2612.

[18] J. Zhang, V. Varadharajan, Wireless sensor network key management survey and taxonomy, Journal of Network and Computer Applications 33 (2) (2010) 63–75.

[19] N. A. Alrajeh, S. Khan, B. Shams, Intrusion Detection Systems in Wireless Sensor Networks: A Review (2013) 1-7.

[20] D. E. Boubiche, A. Bilami, A cross layer intrusion detection system for wireless sensor network, International Journal of Network Security & Its Applications (IJNSA), 4 (2) (2012) 35-52.

[21] C. Karlof, N. Sastry, D. Wagner, TinySec: a link layer security architecture for wireless sensor networks, Proceedings of the 2nd international conference on embedded networked sensor systems, 2004, pp. 162-175.

[22] A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, D.E. Culler, SPINS: Security Protocols for Sensor Networks, Wireless Network 8 (2002) 521–34.

[23] K. Zhang, C. Wang, C. Wang, A secure routing protocol for cluster-based wireless sensor networks using group key management, IEEE Computer Society (2008) 1-5.

[24] D. E Vans, The Internet of things: how the next evolution of the internet is changing every thing, Cisco Internet Business Solutions Group (IBSG), 2011.

[25] A. Hakin, A. Gokhale, P. Berthou, D. C. Schmidt, T. Gayraud, Software-Defined Networking: Challenges and research opportunities for Future Internet, Computer Networks 75 (part A) (2014) 453–471.

[26] L. Atzori, A. Lera, G. Morabioto, The Internet of Things : a survey, Computer Networks 54 (15) (2010) 2787–2805.

[27] D. Miorandi, S. Sicari, F. De-Pellegrini, I. Chlamtac, Internet of things: Vision, applications and research challenges, Ad Hoc Networks 10 (7) (2012) 1497-1516.

[28] X. Jia, Q. Feng, T. Fan, Q. Lei, RFID technology and its applications in Internet of Things (IoT), 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), Yichang, 21-23 April 2012, pp. 1282 – 1285.

[29] S. Duquennoy, G. Grimaud, J. J. Vandewalle, The Web of Things: Interconnecting Devices with High Usability and Performance, Zhejiang, 25-27 May 2009, pp. 323 – 330.

[30] G. Aceto , A. Botta , W. Donato , A. Pescapè, Cloud monitoring: A survey, Computer Networks 57 (9) (2013) 2093–2115.

[31] J. Granjal , E. Monteiro, J. Sá Silva, Security in the integration of low-power Wireless Sensor Networks with the Internet: A survey, Ad Hoc Networks 24 (2015) 264‑287.

[32] O. Garcia-Morchon, S. Keoh, S. Kumar, R. Hummen, R. Struik, Security Considerations in the IP-based Internet of Things, draft-garcia-core-security-04, March 26, 2012.

[33] S. Severi, F. Sottile, G. Abreu, C. Pastrone, M2M technologies: Enablers for a pervasive Internet of Things, 2014 European Conference on networks and communications (EuCNC), Bologna, 23-26 June 2014, pp. 1-5.

[34] http://votplatform.com/ (consulté en Novembre 2015).

[35] http://www.internet-of-things-research.eu/ (consulté en Novembre 2015).

[36] http://www.iot-butler.eu/ (consulté en Novembre 2015).

[37] http://www.hydramiddleware.eu/news.php (consulté en Novembre 2015).

[38] https://www.nitrd.gov/ (consulté en Novembre 2015).

[39] D. Guinard, V. Trifa, T. Pham, O. Liechti, Towards physical mashups in the web of things, In: 2009 Sixth International Conference on Networked Sensing Systems (INSS), Pittsburgh, 2009, pp.1-4.

[40] D. Guinard, V. Trifa, Towards the web of things: Web mashups for embedded devices, In: proceedings of Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), Madrid, Spain, 2009.

[41] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, Hypertext Transfer Protocol -- HTTP/1.1, Request for Comments: 2616, 1999.

[42] V.M. Scuturici, S. Surdu, Y. Gripay, G. M. Petit, Ubiware: Web-based dynamic data & service management platform for AmI. In: Proceedings of Posters and Demo Track (2012), ACM, pp. 1-11.

[43] ABERER, K., AND HAUSWIRTH, M. Middleware support for the internet of things.

[44] D. Conzon, T. Bolognesi, P. Brizzi, A. Lotito, R. Tomasi, M. A. Spirito, The virtus middleware: An xmpp based architecture for secure iot communications, In: Proceedings of Computer Communications and Networks (ICCCN), IEEE 21st International Conference on (2012), pp. 1–6.

[45] M. Eisenhauer, P. Rosengren, P. Antolin, A development platform for integrating wireless devices and sensors into ambient intelligence systems. In: Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2009. SECON Workshops'09, (2009), pp. 1–3.

[46] Suman Nath, Jie Liu, Feng Zhao, Sensormap for wide-area sensor webs, Computer 40 (7) (2007) 0090–0093.

[47] https://datatracker.ietf.org/wg/6lowpan/charter/ (consulté en Décembre 2015)

[48] https://datatracker.ietf.org/wg/roll/charter/ (consulté en Décembre 2015)

[49] https://datatracker.ietf.org/wg/core/charter/ (consulté en Décembre 2015)

[50] IEEE P802.15.4/D18, Draft Standard: Low Rate Wireless Personal Area Networks, Feb. 2003.

[51] S.Deering, R.Hinden, Internet Protocol, Version 6 (IPv6) Specification, Request for Comments: 2460, 1998.

[52] G. Montenegro, N. Kushalnagar, J. Hui, D. Culler, Transmission of IPv6 packets over IEEE 802.15.4 networks, Request for Comments 4944, 2007.

[53] C. Bormann, 6LoWPAN Generic Compression of Headers and Header-Like Payloads, Internet- draft-bormann-6lowpan-ghc-04, 2012.

[54] T. Winter, P. Thubert, A. B randt, J. Hui, R. Kelseky, P. Levis, K. Pister, R. Struik, J.P. Vasseur, R. Alexander, RPL: IPv6 routing protocol for low-power and lossy networks, Request for Comments 6550, 2012.

[55] Z. Shelby, K. Kartke, C. Bormann, and B. Frank, Constrained Application Protocol (CoAP), draft-ietf-core-coap-12, 2012.

[56] http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. (consulté en Aout 2015).

[57] http://www.weightless.org/about/what-is-weightless. (consulté en Aout 2015).

[58] K. T. Nguyen, M. Laurent, N. Oualha, Survey on secure communication protocols for the Internet of Things, Ad Hoc Networks 32 (2015) 17-31.

[59] http://securityaffairs.co/wordpress/21397/cyber-crime/iot-cyberattack-large-scale.html

[60] R. Hummen, J. Hiller, H. Wirtz, M.Henze, H. Shafagh, K. Wehrle, 6LoWPAN fragmentation attacks and mitigation mechanisms, In: Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks, Budapest, Hungary, 17-19 April 2013, pp. 55-66.

[61] S. Sicari, A. Rizzardi, L.A. Grieco, A. Coen-Porisini, Security, privacy and trust in Internet of things: the road ahead, Computer Networks 76 (2015) 146–164.

[62] G. P. Schaffer, Worms and viruses and botnets, oh my!: Rational responses to emerging internet threats. IEEE security & privacy, (3) 2006 52-58.

[63] A. Naumenko, A. Katasonov, V. Terziyan, A security framework for smart ubiquitous industrial resources. In Enterprise Interoperability II. Springer, 2007, pp. 183–194.

[64] R. J. C. Benito , D. Garrido, P. Plaza, R. Roman, N. Sanz, J. L. Serrano, SMEPP: A secure middleware for embedded p2p, In: Proceedings of ICT Mobile and Wireless Communications Summit (ICT-MobileSummit'09), Santander, Spain, 2009, pp. 1-8.

[65] T. Dierks, C. Allen, The TLS protocol, Request for Comments 2246, 1999.

[66] B. Sarikaya, et al, Security Bootstrapping Solution for Resource-Constrained Devices, Technical report IETF Internet Draft draft-sarikaya-coresbootstrapping-05, 2012.

[67] P.D. Khambre, S.S. Simbhare, P.S. Chavan, Secure Data in Wireless Sensor Network via AES (Advanced Encryption Standard), International Journal of Computer Science and Information Technologies (IJCSIT) 3 (2) (2012) 3588-3592.

[68] L. Wallgren,S. Raza,T. Voigt, Routing Attacks and Countermeasures in the RPL-Based Internet of Things, International Journal of Distributed Sensor Networks, 2013.

[69] S.O. Amin, Y. J. yoon, M. S. Siddiqui, C. S. Hong , A novel intrusion detection framework for IP-based Sensor Networks, In: Proceedings of International Conference on Information Networking, Chiang Mai, 21-24 January 2009, pp. 1-3.

[70] A. Le, J. Loo, A. Lasebae, M. Aiash, Y. Luo, 6LoWPAN: a study on QoS security threats and countermeasures using intrusion detection system approach, International Journal of Communication Systems, 2012.

[71] S. Raza, L. Wallgren, T. Voigt, SVELTE: Real-time intrusion detection in the Internet of Things, Ad Hoc Networks 11 (8) (2013) 2661–2674.

[72] E. Rescorla, Diffie-Hellman Key Agreement Method, Request for Comments: 2631, 1999.

[73] R. Roman, Key management systems for sensor networks in the context of the Internet of Things, Computers and Electrical Engineering, 37 (2) (2011) 147–159.

[74] M. R. Abdmeziem, D. Tandjaoui, An end-to-end secure key management protocol for e-health applications, Computers & Electrical Engineering 44 (2015) 184–197.

[75] M. R. Abdmeziem, D. Tandjaoui, I. Romdhani, A Decentralized Batch-based Group Key Management Protocol for Mobile Internet of Things (DBGK), In: Proceedings of the 14th IEEE International Conference on Ubiquitous Computing and Communications (IUCC-2015), Liverpool, 2015, pp. 1109 – 1117.

[76] S. Frankel, S. Kishnan, IP Security (IPsec) and Internet Key Exchange (IKE) document roadmap, Request for Comments: 6071, 2011.

[77] S. Kent, IP Authentication Header, Request for Comments: 4302, 2005.

[78] S. Kent, IP Encapsulating Security Payload (ESP), Request for Comments: 4303, 2005.

[79] S. Raza, T. Voigt, U. Roedig, 6LoWPAN Extension for IPsec, the Interconnecting Smart Objects with the Internet Workshop, 2011.

[80] S. Raza, T. Voigt, V. Jutvik, Lightweight IKEv2: A Key ManagementSolution for both the Compressed IPsec and the IEEE 802.15.4 Security, In: the IETF Workshop on Smart Object Security, Paris, 2012.

[81] R. Bonetto, N. Bui, V. Lakkundi, A. Olivereau, Secure communication for smart IoT objects: Protocol stacks, use cases and practical examples, In: World of Wireless, Mobile and Multimedia Networks (WoWMoM) IEEE International Symposium , San Francisco, 2012, pp. 1-7.

[82] Y. Ben-Saied, A. Olivereau, D. Zeghlache, M. Laurent, Lightweight collaborative key establishment scheme for the Internet of Things, Computer Networks 64 (2014) 273–295.

[83] R. Moskowitz, P. Nikander, P. Jokela, T. Henderson, Host Identity Protocol, IETF RFC 5201, 2008.

[84] R. Moskowitz, T. Heer, P. Jokela, T. Henderson, Host Identity Protocol Version 2 (HIPv2), Request for Comments: 7401, 2015.

[85] R. Moskowitz, R. Hummen, HIP Diet EXchange (DEX), draftmoskowitz-hip-dex-05, 2016.

[86] T. Heer, LHIP Lightweight Authentication Extension for HIP, draft-heer-hip-lhip-00, 2007.

[87] Y. Ben Saied, A. Olivereau, D-HIP: A Distributed Kkey Exchange Scheme for HIP-Based Internet of Things, In: IEEE International Symposium on World of Wireless Mobile and Multimedia Networks (WoWMoM), San Francisco, 2012, pp. 1-7.

[88] R. Hummen, J. Hiller, M. Henze, K. Wehrle, Slimfit – A HIP DEX Compression Layer for the IP-based Internet of Things, In: IEEE WiMob 2013 Workshop IoT, Lyon, 2013, pp. 259-266.

[89] F.V. Meca, J.H. Ziegeldorf, P.M. Sanchez, O.G. Morchon, HIP security architecture for the IP-based Internet of Things, In: 2013 27th International Conference on Advanced Information Networking and Applications Workshops, Barcelona, 2013, pp. 1331-1336

[90] R. Alexander, T. Tsao, Adapted Multimedia Internet KEYing (AMIKEY): An extension of Multimedia Internet KEYing (MIKEY) Methods for Generic LLN Environments, IETF, Internet-Draft, 2012.

[91] S. Fouladgar, B. Mainaud, K. Masmaudi, H. Afifi, Tiny 3-TLS: a trust delegation protocol for wireless sensor networks, Security and Privacy in Ad-Hoc and Sensor Networks, Hamburg, Germany, 2006, pp. 32-42.

[92] E. Rescorla, N. Modadugu, Datagram Transport Layer Security, Request for Comments: 4347, 2006.

[93] T. Kothmayr, W. Hu, C. Schmitt, M. Brunig, G. Carle, Poster: Securing the Internet of Things with DTLS, In: the 9th ACM Conference on Embedded Networked Sensor Systems, 2011, pp. 345-346.

[94] S. Raza, D. Trabalza, T. Voigt, 6LoWPAN compressed DTLS for CoAP, In: The 8th IEEE International Conference on Distributed Computing in Sensor Systems, 2012, pp. 287 – 289.

[95] S. Raza, H. Shafagh, K. Hewag, R. Hummen, Lithe: Lightweight Secure CoAP for the Internet of Things, IEEE sensors journal 13 (10) (2013) 3711 – 3720.

[96] T. Kothmayr, C. Schmitt, W. Hu, M. Brunig, G. Carle, DTLS based Security and Two-Way Authentication for the Internet of Things, Ad Hoc Networks 11 (8) (2013) 2710‑2723.

[97] H. Shafagh, A. Hithnawi, Poster Abstract: Security Comes First, A Public-Key cryptography framework for the Internet of things, In: The 10th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'14), 2014, pp. 135-136.

[98] A. Yegin, Z. Shelby, CoAP Security Options, draft-yegin-coap-security-options-00, 2011.

[99] H. Slimane, H. Selmi, approche pour la sécurisation des communications de l'Internet des objets, mémoire de Master, encadré par Maamar Sedrati et Somia Sahraoui, Université de Batna 2, 2015.

[100] R. Hummen, J. H. Ziegeldorf, H. Shafagh, S. Raza, K. Wehrle, Towards Viable Certificate-based Authentication for the Internet of Things, In: Proceedings of the 2nd ACM workshop on Hot topics on wireless network security and privacy (HotWiSec'13), Budapest, Hungary 2013, pp. 37-42.

[101] J. Liu, Y. Xiao, C. L. Philip-Chen, Authentication and Access Control in the Internet of Things, In: Proceedings of the 32nd International Conference on Distributed Computing Systems Workshops, 2012, pp. 588-592.

[102] J. Singh, T. Pasquier, J. Bacon, H. Ko, D. Eyers, Twenty security considerations for cloud-supported internet of things, IEEE Internet of Things Journal (2015) 1-16.

[103] http://www.sensorcloud.com/ (consulté en Septembre 2015).

[104] http://www.sensatrack.com/ (consulté en Septembre 2015).

[105] D. Kuptsov, B. Nechaev, A. Gurtov, Securing Medical Sensor Network with HIP, Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (2012) 150-157.

[106] A. Khurri, D. Kuptsov, A. Gurtov, On Application of Host Identity Protocol in Wireless Sensor Networks, In: IEEE 7th International Conference on Mobile Adhoc and Sensor Systems (MASS), San Francisco, CA, 2010, pp. 345-358.

[107] S. Sahraoui, A. Bilami, Compressed and distributed host identity protocol for end-to-end security in the IoT, In: Proceedings of the 2014 IEEE Fifth International Conference on Next Generation Networks and Services (NGNS), Casablanca, Moroco, 28-30 May 2014, pp. 295 – 301.

[108] S. Sahraoui, A. Bilami, Efficient HIP-based approach to ensure lightweight end-to-end security in the internet of things, Science Direct, Elsevier Computer Networks 91 (2015) 26–45.

[109] G. Camarillo, J. Melen, Host Identity Protocol (HIP) Immediate Carriage and Conveyance of Upper-Layer Protocol Signaling (HICCUPS), IETF RFC 6078, 2011.

[110] V.B. Misic, J. Fang, J. Misic, MAC Layer Security of 802.15.4-Compliant Networks, In: IEEE International Conference on Mobile Adhoc and Sensor Systems, Washington, 2005, pp. 847-854.

[111] J. Großschadl, A. Szekely, S. Tillich, The Energy Cost of Cryptographic Key Establishment in Wireless Sensor Networks, In: the 2nd ACM Symposium on Information, Computer and Communications Security (ASIACCS), Singapore, 2007, pp. 380–382.

[112] O. Briante, M. Amadeo, C. Campolo, A. Molinaro, S. Y. Paratore, G. Ruggeri , eDomus: User-home interactions through Facebook and Named Data Networking, In: 2014 Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), Singapore, June 30 -July 3 2014, pp. 155 – 157.

[113] A. Zanella, A. Vangelista, Internet of Things for Smart Cities, IEEE Internet of Things Journal 1 (1) (2014) 22-32.

[114] M. S. H. Talpur, The Appliance of Pervasive Internet of Things in Healthcare Systems, ArXiv preprint (2013).

[115] S. Sahraoui, A. Bilami, Asymmetric End-to-End Security for Human-to-Thing Communications in the Internet of Things, In: Proceedings of the 4th International Symposium on Modeling and Implementation of Complex Systems (MISC 2016), Constantine, Algeria, 7-8 May 2016, pp. 249-260.

[116] Asymmetric Digital Subscriber Line (ADSL), AG Communication Systems, pp. 1-14.

[117] A. Castellani, S. Loreto, A. Rahman, T. Fossati, E. Dijk, Guidelines for HTTP-CoAP Mapping Implementations, draft-ietf-core-http-mapping-06, July 2015.

[118] A. Sehgal, Using the Contiki Cooja Simulator, 2013. (consulté en Décembre 2015)

[119] https://www.sics.se/ (consulté en Janvier 2015).

[120] A. Dunkels, F. Osterlind, N. Tsiftes, Z. He , Software-based on-line energy estimation for sensor nodes, In: 4th Workshop Embedded Netw. Sensors, New York, 2007, pp. 28-32.

[121] https://github.com/ejoerns/contiki-inga/wiki/Cooja-Border-Router-Setup (consulté en Novembre 2015).

[122] T. Kivinen, M. Kojo, More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE), Request for Comments: 3526, 2003.

[123] T. Pornin, Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA), Request for Comments: 6979, 2013.

[124] S. Turner, Using SHA2 Algorithms with Cryptographic Message Syntax, Request for Comments: 5754, 2010.

[125] P. Kasinathan, C. Pastrone, M. A. Spirito, M. Vinkovits, Denial-of-Service detection in 6LoWPAN based Internet of Things, 2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2013, pp. 600-607.

[126] Tmote Sky <http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf>

[127] F. Medjek, D. Tandjaoui, M. R. Abdmeziem, N. Djedjig, Analytical evaluation of the impacts of Sybil attacks against RPL under mobility, In: Proceedings of the 12th International Symposium on Programming and Systems (ISPS), Algeries-Algeria, 2015, pp. 1-9.

[128] A. Dunkels, O. Schmidt, T. Voigt, M. Ali, Protothreads: Simplifying Event-Driven Programming of Memory-Constrained Embedded Systems, In: Proceedings of the 4th international conference on Embedded networked sensor systems, USA, 2006, pp. 29-42.

[129] http://www.opex360.com/2015/05/15/larmee-chinoise-met-en-garde-les-objets-connectes-internet/ (consulté en Novembre 2015).

[130] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, B. Moeller, Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS), Request for Comments: 4492.

[131] A. Bachir, D. Barthel, M. Heusse, A. Duda, O(1)-Reception routing for sensor networks, 30 (2007) 2603–2614.

[132] M. Kovatsch, Demo Abstract : Human-CoAP Interaction with Copper, In: Proceedings of the International conference on Distributed Computing in Sensor Systems and Workshops, Barcelona, 2011, pp. 1-2.

[133] Y. Medjadba, S. Sahraoui, Intrusion Detection System to Overcome a Novel Form of Replay Attack (Data Replay) in Wireless Sensor Networks, International Journal of Computer and Information Secuity (IJCNIS), 8 (7) (2016) 50-60.

[134] P. S. Andre, Extensible Messaging and Presence Protocol (XMPP): Core, Request for Comments: 6120, 2011.