REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOHAMED BOUDIAF - M'SILA

FACULTE DES MATHEMATIQUES ET DE L'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

N° :

DOMAINE : Mathématiques et Informatique
FILIERE : INFORMATIQUE

OPTION : RESEAUX ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

Mémoire présenté pour l'obtention
Du diplôme de Master Académique

Par : HERIZI Maria

Intitulé

Simulation des Algorithmes Cryptographiques Basés sur les Courbes Elliptiques en Cooja Contiki

Soutenu devant le jury composé de :

Mr Mezrag Fares	Université de M'sila	Président
Mr Chikouche Noureddine	Université de M'sila	Rapporteur
Mr Tahri Zohir	Université de M'sila	Examinateur

Année universitaire : 2017 /2018

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOHAMED BOUDIAF - M'SILA

FACULTE DES MATHEMATIQUES ET DE L'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

N° :

DOMAINE : Mathématiques et Informatique
FILIERE : INFORMATIQUE

OPTION : RESEAUX ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

Mémoire présenté pour l'obtention
Du diplôme de Master Académique

Par : HERIZI Maria

Intitulé

Simulation des Algorithmes Cryptographiques Basés sur les Courbes Elliptiques en Cooja Contiki

Soutenu devant le jury composé de :

Mr Mezrag Fares	Université de M'sila	Président
Mr Chikouche Noureddine	Université de M'sila	Rapporteur
Mr Tahri Zohir	Université de M'sila	Examinateur

Année universitaire : 2017 /2018

Dédicace

A mon cher mari : Rachid

Remerciements

Je remercie d'abord ALLAH, le bon dieu qui m'a guidé dans le chemin du savoir.

Mes remerciements s'adressent également à mon encadreur Monsieur Chikouche Noureddine, pour avoir accepté de diriger ce travail. Son soutien, sa clairvoyance et ses compétences m'ont été d'une aide inestimable

Je remercie également Diego F. Aranha le développeur de Relic qui ma offert son aide précieux

Je tiens aussi à exprimer l'honneur qui m'est fait par les membres de jury en acceptant d'évaluer mon travail. Qu'il trouve ici ma reconnaissance et mon respect.

Nous tenons également à remercier toutes les personnes qui ont participé, à titre professionnel ou personnel à la réalisation ce travail.

Maria

Table des matières

Chapitre 4 - Evaluation des performances des Algorithmes Cryptographiques

Liste des figures

CHAPITRE 1 - Généralités sur les réseaux de capteurs sans fil

Chapitre 3 - Les Algorithmes Cryptographiques basés sur les courbes Elliptiques

Chapitre 4 - Evaluation des performances des Algorithmes Cryptographiques en RelicTool Kit

Liste des tables

CHAPITRE 1 - Généralités sur les réseaux de capteurs sans fil

Chapitre 3 - Les Algorithmes Cryptographiques basés sur les courbes Elliptiques

Chapitre 4-Evaluation des performances des Algorithmes Cryptographiques en RelicTool Kit

INTRODUCTION GENERALE

Les réseaux de capteurs sans fil (RCSF) sont une thématique de recherche en pleine expansion. Car ils ont un intérêt particulier pour les applications militaires, environnementales, domestiques, médicales, et bien sûr les applications liées à la surveillance des infrastructures critiques. Durant les dix dernières années, de nombreux travaux et études ont été menés pour répondre aux différentes problématiques posées. Parmi les quels le problème de sécurité. La bonne application de sécurité passe donc par une gestion intelligente des ressources présentes au sein de chaque capteur du RCSF. Le besoin d'apporter une solution de sécurité fiable paraît important voir crucial. Or, de par les caractéristiques des RCSFs (absence d'infrastructure topologie dynamique, nombre important de capteurs, sécurité physique limitée, modes et lieux de déploiement offrant de multiples possibilités d'attaques) et des contraintes inhérentes aux nœuds capteurs (l'énergie, les capacités de calcul, de stockage et de transmission limitées), la sécurisation des réseaux de capteurs est à la source, aujourd'hui, de beaucoup de recherches scientifiques et techniques.

Des travaux de recherche antérieurs ont d'ailleurs démontré que les solutions de sécurité proposées pour les réseaux sans fil (mobiles et adhoc), surtout celles basées sur l'utilisation de la cryptographie à clé publique ne peuvent pas être appliquées directement dans les réseaux de capteurs sans fil du fait des couts de calcul importants engendrés.

La solution consiste à augmenter toujours plus la puissance des chiffrements, la problématique posée par la faiblesse de calcul et le besoin d'´économiser l´énergie des capteurs amènent à se poser des questions nouvelles sur les méthodes de sécurité à utiliser. Il est important que les solutions de sécurité à mettre en œuvre soient les moins couteuses en consommation de ressources et visent à réduire les temps, le nombre de transmissions, et l'occupation de la bande passante. En l'occurrence, ce sont les solutions apportant une sécurité maximale tout en préservant la durée de vie du capteur,

Dans ce mémoire, nous nous sommes intéressés aux problèmes de sécurité dans les RCSFs, nous cherchons à définir des mécanismes et solutions peu couteux en

énergie pour les réseaux de capteurs sans fil qui prennent en compte la relative faiblesse de défense d'un réseau autonome.

Dans cette optique nous simulons des algorithmes cryptographiques asymétriques basés sur les courbes elliptiques peu gourmandes en énergie qui sont implémentés dans la bibliothèque Relic Tool Kit dont on a affronté des problèmes pénibles pendant l'intégration de celle-ci en Contiki qui est de référence dans le domaine des RCSFs et par extension de l'Internet des objets (IoT). Et s'il est légèrement plus exigeant en ressources que TinyOS, Contiki reste très léger et particulièrement bien adapté aux nœuds capteurs constituant les RCSFs.

Contiki est distribué avec le simulateur avancé de RCSF : Cooja, ce dernier a été utilisé pour tester la validité des codes produits pour effectuer une étude comparative entre les algorithmes cryptographiques : ECDH, ECDSA, et ECIES concernant l'occupation de l'espace mémoire (RAM et ROM), le temps d'exécution écoulé et l'énergie consommé afin de déduire l'algorithme le plus convenable aux caractéristiques de nœud capteur dans les RCSFs.

Ce mémoire est constitué de quatre chapitres :

Chapitre 1 :

Vision générale sur les réseaux de capteurs sans fil (RCSF). Le point de départ est le capteur qui sera intégré dans un réseau de manière à répondre à une application donnée.

Description des principaux concepts liés aux réseaux de capteurs sans fil tels que : l'architecture, les applications en vedette, les principales caractéristiques et limitations.

Chapitre 2 :

Présentation des aspects fondamentaux de la sécurité des réseaux de capteurs sans fil. Et également les solutions de base, qui sont proposées dans la littérature pour répondre aux besoins en termes de sécurité.

Chapitre 3 :

Les algorithmes cryptographiques basés sur les courbes elliptiques utilisé dans les RCSF ainsi que les librairies de cryptographie (open source) contenants ces algorithmes.

Chapitre 4 :

Résultats des tests réalisées en utilisant la cryptographie a clé public basé sur les courbes elliptiques avec Contiki.

Représentation de l'environnement de simulation et la plateforme sélectionnée.

Explication des différentes méthodes de mesures de : temps d'exécution, la RAM, la ROM et l'énergie consommée par le nœud capteur

Citation des problèmes rencontrées dans le système Contiki et discussion des résultats obtenues.

CHAPITRE 1:

Généralités sur les réseaux de capteurs sans fil

1.1 Introduction

La popularité des réseaux de capteurs sans fil (RCSFs) 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.

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.

L'objectif de ce chapitre est de donner une vision générale sur les réseaux de capteurs sans fil (RCSF). Le point de départ est le capteur qui sera intégré dans un réseau de manière à répondre à une application donnée.

1.2 Définition d'un RCSF ou WSN (Wireless Sensor Network)

Les réseaux de capteurs utilisent un grand nombre de dispositifs très petits, nommés (nœuds capteurs), pour former un réseau sans infrastructure établie. Dans ces réseaux, chaque nœud est capable de détecter son environnement et de traiter l'information au niveau local ou de l'envoyer à un ou plusieurs points de collecte, à l'aide d'une connexion sans fil, la position de ces nœuds n'est pas obligatoirement prédéterminée, ils peuvent être aléatoirement dispersés dans une zone géographique appelée « champ de captage » correspondant au terrain d'intérêt pour le phénomène capté (par exemple : lâchée de capteurs sur un volcan pour étudier les phénomènes vulcanologiques et leurs évolutions). Le réseau possède en général un nœud particulier, le puit (ou sink), connectée avec les autres nœuds et relié à une alimentation électrique ce qui est illustré par la figure suivante [1].

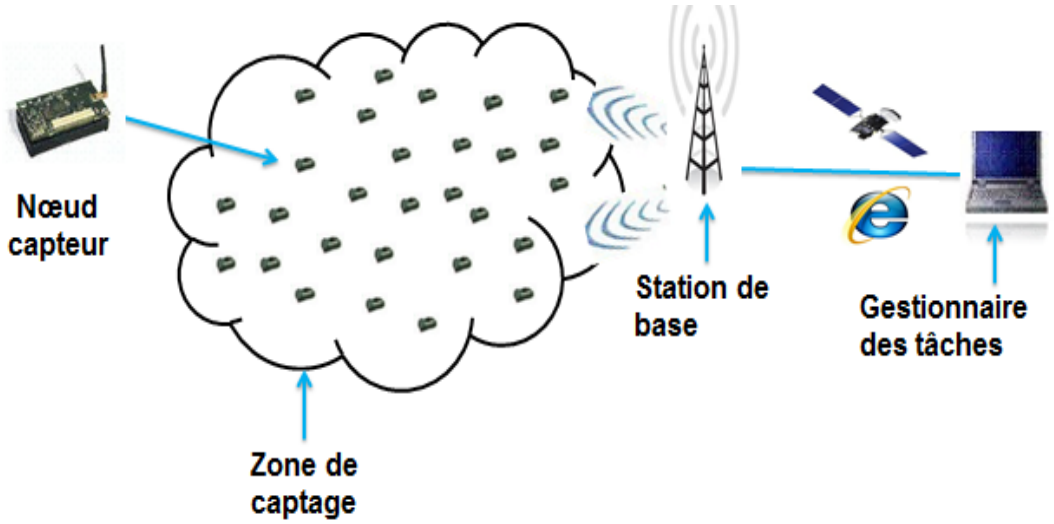

Figure 1.1 Architecture des RCSF [1]

1.3 Domaines d'application des RCSF

Parmi les domaines d'application ou la technologie des réseaux de capteurs est la plus intéressante et efficace, nous citons : [2]

- **Militaire :**

1. Contrôle des forces ennemies.
2. Contrôle des équipements.
3. Détection d'attaques par des armes : nucléaires, chimiques, etc.

- **Santé :**

1. Télésurveillance des signes vitaux et des niveaux d'activité à domicile des personnes âgées ou handicapées.
2. Contrôle à distance des données physiologiques.
3. Suivi et contrôle des médecins et des patients dans les hôpitaux.

- **Environnemental :**

1. Microclimats.
2. Découverte de catastrophes naturelles telles que : détection d'incendie, détection d'inondation, etc...

- **Agriculture :**

1. Contrôle d'irrigation.

- **Maison :**

1. Automatisation.

- **Commercial :**

1. Contrôle environnemental dans les usines et les bureaux.
2. Contrôle d'inventaire.
3. Suivi des véhicules et détection des encombrements.

- **Sécurité :**

1. Détection d'intrusions

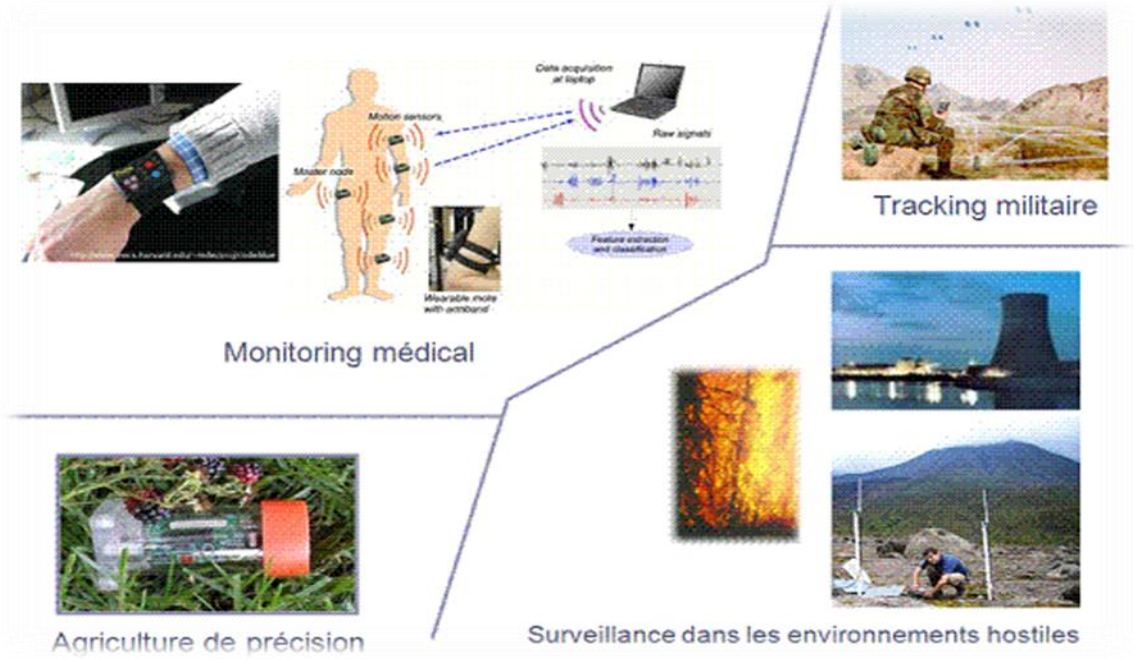

Figure 1.2 Applications des RCSF [3]

1.4 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 (< .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.

1.5 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 : [1]

- **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).

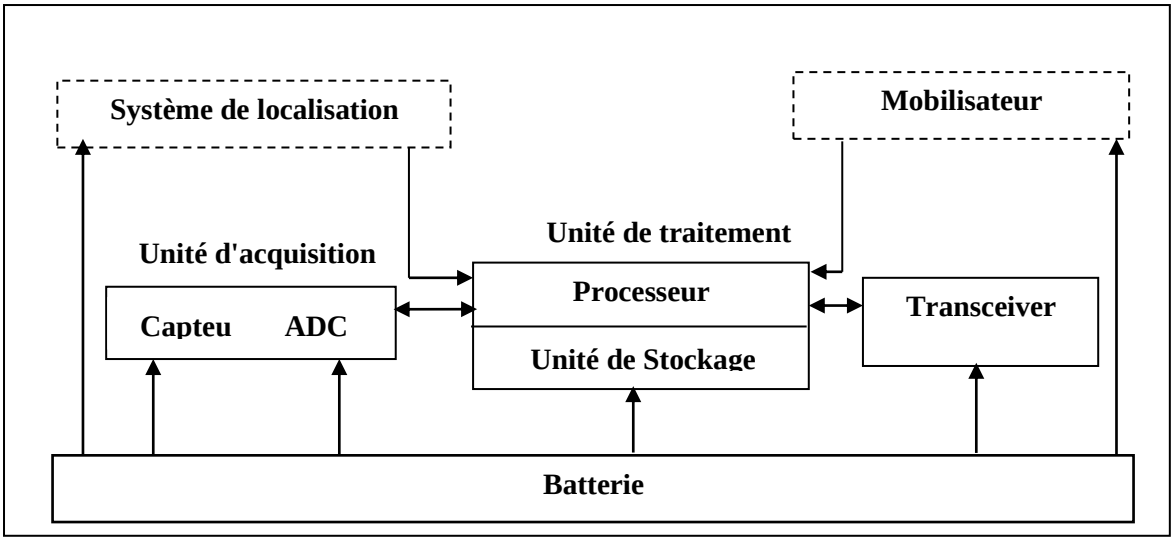

Figure 1.3 Composants du nœud capteur [1]

1.6 Evolution de Capteurs :

Le tableau suivant illustre l'évolution de capteurs [3].

Génération	Période	Taille	Poids	Batterie
1ère	1800 – 1900	Grande boite à chaussures	Kilogrammes	Grosse
2 ème	Entre 2000 et 2003	Boite de catre	Grammes	AA
3 ème	2010	Particule de poussière	Négligeable	Solaire

Table 1.1 Les trios générations des nœuds Capteurs

1.7 Exemple de nœuds Capteurs :

Il existe plusieurs modèles commercialisés des capteurs sans fil, parmi les plus célèbres : Tmote Sky et WiSMote (figure 1.4). Le tableau 1.2 présente les caractéristiques des deux plateformes.

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

Table1.2 Caractéristiques des capteurs WiSMote et Tmote Sky [1]

Capteur WiSMote **Capteur Tmote Sky**

Figure 1.4 Exemple de nœuds Capteurs

1.8 Systèmes d'exploitation pour le capteur

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.

Dans le tableau ci-dessous, on compare Contiki et TinyOS suivant plusieurs critères :

	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

Table 1.3 TinyOS et Contiki [1]

1.9 Contiki

Contiki est un système d'exploitation léger et flexible pour capteurs miniatures en réseau. Ces dernières années, le monde scientifique a porté une attention importante aux réseaux de capteurs sans fil. La miniaturisation des capteurs et leur prix relativement faible, permettent d'imaginer des applications très variées dans les domaines scientifiques, militaires, industriels et domotiques. Pour faciliter le développement de ces applications, une équipe du centre suédois de recherche scientifique SICS (en) a créé Contiki [4]. Destiné à être embarqué dans des capteurs miniatures ne disposant généralement que de ressources limitées, Contiki propose malgré tous les principales caractéristiques et fonctionnalités d'un système d'exploitation tout en favorisant une consommation énergétique et une empreinte mémoire minimales.

Ses principaux atouts sont:

- Le support de protocole 6LoWPAN.
- Sa flexibilité et sa portabilité.

- Disponible gratuitement sous licence BSD.
- Contiki peut être utilisé et modifié, même à des fins commerciales.

1.10 L'architecture hybride du système d'exploitation Contiki

Pour économiser de la mémoire, l'approche basée sur les événements a été, dans un premier temps, privilégiée. Les incertitudes sur la taille de la pile d'exécution et le nombre de processus à prévoir sont ainsi évitées. Cependant, les opérations longues telles que la cryptographie de données s'accordent mal avec l'utilisation des événements par la monopolisation du système pour une durée conséquente. Pour cette raison, dans un second temps, le système Contiki s'est vu ajouter un composant qui lui permet de fonctionner comme un système multitâche. Ce composant est une bibliothèque de fonctions optionnelle appelée explicitement par le programme qui en a besoin. Cette bibliothèque permet la gestion des processus et de leur pile d'exécution respective.

Le cœur du système Contiki est composé de différents éléments :

• le noyau .
• le chargeur de programme .
• les bibliothèques utilisées .
• la pile de communication avec les pilotes pour le matériel .
• le module de gestion des bibliothèques systèmes.

Contiki est distribué avec l'émulateur Cooja pour tester la validité des codes produits.

1.11 Cooja

Cooja est un simulateur de réseaux de capteurs développé en Java.

- Il est fait pour émuler le fonctionnement de capteurs sous le système d'exploitation Contiki.
- Il s'agit d'un simulateur flexible et extensible permettant de modifier ou remplacer tous les niveaux du système.
- Il interagit avec Code Contiki au travers de JNI (Java Native Interface).
- L'interface du simulateur est composée de plusieurs plugins présent sous formes de fenêtres. [4]

1.12 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. 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

2.1 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.

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.

2.3 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, [5]

2.3.1. 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.

2.3.2 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.

2.3.3 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.

2.3.4 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.

2.3.5 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.

2.3.6 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.

2.4 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.

2.5 Classification des 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 [6] :

- **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.

2.6 Les attaques par niveaux 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) [6-7].

2.6.1 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.

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.

2.6.2 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.

2.6.3 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 [6].

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 [8-9], 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.

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 [10] 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.

D. L'attaque d'acheminement sélectif (Selective Forwarding) :

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.

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 de capteur et l'attaquant 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.

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 [8].

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 de 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.

2.6.4. 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.

2.7 Les fondements de la sécurité dans les réseaux de capteurs

L'un des paramètres qui est à la base de la plupart des solutions de sécurité dans les réseaux de capteurs est bien la cryptographie [7] :

2.7.1 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 nécessite l'emplois 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. Les solutions de chiffrement à clés symétriques sont exploitables au sein des réseaux de capteurs et apportent une réelle solution pour la sécurité du réseau. Cependant, si le chiffrement à clé symétrique est possible au sein des réseaux de capteurs, la sécurité totale de ce type de solution reste à démontrer. D'une part parce que le chiffrement à clé symétrique ne garantit pas l'authentification des données comme peut le faire la signature numérique des chiffrements à clés publiques, et d'autre part parce que se pose le problème de la distribution des clés de chiffrement. Dans cette mémoire nous allons nous intéressés uniquement à la cryptographie asymétrique.

. La cryptographie asymétrique

Le mot « cryptographie » est un mot d'origine grecque composé de deux parties :
« Crypto » (kruptos) qui signifie caché et « graphie » (graphein) qui signifie écrire.
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 Algorithme) s'appuient également sur la cryptographie asymétrique pour garantir l'intégrité et l'authentification des messages échangés.

La cryptographie par courbes elliptiques (ECC : Elliptic Curve Cryptography) [11] 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).

2.8 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.

2.8.1. Le protocole TinySec (Tiny Security)

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. [12]

2.8.2. SPINS: Security Protocols for Sensor Networks

SPINS 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). [13]

- **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.
- **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.

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

2.9 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:

Les Algorithmes Cryptographiques Basés sur les courbes Elliptiques

3.1 Introduction

Dans la plupart des mécanismes de sécurisation actuelle, la cryptographie est sans doute la technique la plus utilisée dans le cadre des réseaux filaires et des réseaux sans fil traditionnels disposant d'une capacité de calcul et de mémoire conséquente. Les solutions de cryptographie sont réputées comme des solutions sûres qui répondent à l'ensemble des problèmes liés à la sécurité des données. Les spécificités des réseaux de capteurs, à savoir une faible puissance de calcul et une mémoire limitée auxquelles se rajoute la problématique de préservation de l´énergie, sont des freins considérables à l'utilisation des systèmes cryptographiques courant réputes surs (SSL, RSA, ECC etc.).

Les travaux de recherche actuels s'attachent à trouver des solutions dites de cryptographie légères. Ces solutions consistent à adapter des algorithmes de cryptographie classiques pour les réseaux de capteurs, ou à en trouver de nouveaux tout aussi efficaces en termes de sécurité, de temps d'exécution et de consommation énergétique.

Nous énonçons, dans ce chapitre, les algorithmes cryptographiques basés sur les courbes elliptiques utilisé dans les RCSF ainsi que les librairies de cryptographie (open source) contenants ces algorithmes.

3.2 Le cryptosystème RSA

Le principe de la cryptographie à clé publique proposé en 1976 et en 1977 présenté le premier cryptosystème effectif RSA (du nom de ses auteurs Rivest, Shamir et Adleman) [14]. RSA est basé sur le problème de la factorisation des grands nombres entiers, RSA est encore aujourd'hui la primitive la plus utilisée en protocoles de communication**.**

3.2.1 Génération de clés

Il s'agit de générer les clés publiques et privées. Pour la génération de la clé publique, on choisit deux grand nombres premiers p et q dont le produit est N. Ensuite, on génère les nombres entiers e (un nombre premier) et d tels que $1 < e < \varphi$, le plus grand diviseur commun entre e et φ est 1, $\varphi = (p - 1)(q - 1)$, d est tel que $1 < d < \varphi$ et $(ed$

- *1)* est un multiple de φ. *N* et *e* constituent la clé publique, tandis que la clé secrète *d* est construite à partir de *p* et *q* nécessaires pour son calcul [15].

3.2.2 Chiffrement

Avec la clé publique (N, e), on calcule $c=m^e \bmod N$, ou $m \in [0, N-1]$ est le message à chiffrer et c le message chiffré transmis.

Algorithme Génération de clés avec RSA

Entrées : *l* // taille (en bits) qui détermine le niveau de sécurité

Sorties : Clé publique (N,e) et clé privée d générées

begin

1. Sélection au hasard de deux nombres premiers p et q
2. Calculer le produit N : *pq* et $\varphi=(p-1)(q-1)$
3. Selectionner arbitrairement un entier *e* tel que $1 < e < \varphi$ et le PGDC $(e, \varphi)=1$
4. Calculer *d* tel que $1 < d < \varphi$ et $ed \equiv 1 (\bmod \varphi)$
5. Retourner(*N, e, d*)

Algorithme Chiffrement avec RSA

Entrées : *(N,e) et m* // la clé publique et le texte claire $m \in [0, N-1]$

Sorties : *c*, // le texte chiffre

begin

1. Calculer $c=m^e \bmod N$
2. Retourner(*c*)

3.2.3 Déchiffrement

A la réception du message chiffré *c*, on calcule $m=c^d \bmod N$.

Algorithme déchiffrement avec RSA

Entrées : *(N,e), d et c* // la clé publique, la clé privée et le texte chiffre

Sorties : *m*, // le texte claire

begin

1. Calculer $m=c^d \bmod N$
2. Retourner(*m*)

3.2.4 Signature : le signataire calcule un condense *h* sur le message *m* avec une fonction de hachage cryptographique H() : *h=H(m).* Il utilise la clé privée *d* pour calculer la **signatures :** $s=h^d mod\ N$. Le signataire envoie *s* et le message *m* pour vérification.

Algorithme Signature avec RSA
Entrées : *(N,e), d et m* // la clé publique, la clé privée et le message m
Sorties : *s,* // la signature
begin
1. Calculer *h=H(m)*
2. Calculer $s=h^d mod\ N$
3. Retourner(*s*)

3.2.5 Vérification de la signature : à la réception de la signature, l'autre partie calcule *h=H(m),* puis à partir de la clé publique et de la signature *s* reçue, elle calcule $h'=s^e mod\ N$. Enfin elle accepte la signature si *h=h'*.
Algorithme Vérification de la signature
Entrées : *(N,e), m et s* // la clé publique, le message m, la signature s
Sorties : Acceptation ou rejet de la signature
begin
1. Calculer *h=H(m)*
2. Calculer $h'=s^e mod\ N$
3. Accepter si *h=h'* et rejet sinon

3.3 Les courbes elliptiques pour la cryptographie (ECC)

Les courbes elliptiques sont les courbes les plus simples après les droites et les coniques. On peut considérer une courbe elliptique sur un corps fini $\mathbb{F}$, elle intervient ainsi dans certains protocoles cryptographiques. On peut définir une courbe elliptique E sur un corps fini $\mathbb{F}$ noté E ($\mathbb{F}$) par son équation à la forme de Weierstrass [16] :

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 . \quad (1.1)$$

où a_1, a_2, a_3,a_4 et a_6 $\mathbb{F}$.

Les corps sont des systèmes de nombres bien connus tels que les nombres rationnels $\mathbb{Q}$ ou les nombres réels $\mathbb{R}$ et les nombres complexes $\mathbb{C}$. Un corps est un ensemble $\mathbb{F}$ possédant deux opérations élémentaires, l'addition (notée +) et la multiplication (notée .), satisfaisant les propriétés arithmétiques suivantes :

– ($\mathbb{F}$, +) est un groupe abélien (avec l'addition) et l'élément neutre est 0.

– ($\mathbb{F}^*$, .) est un groupe abélien (avec la multiplication) et l'élément neutre est 1.

– La distributivité est respectée : $(a+b).c = a.c + b.c$ pour tout $a,b,c \in \mathbb{F}$.

Les corps premiers notés par $\mathbb{F}$p où $p=q^m$ et q un nombre premier (appelé la caractéristique de $\mathbb{F}_p$) sont généralement utilisés en cryptographie. Si m=1 alors $\mathbb{F}_p$ est appelé un *corps premier*. Dans cet mémoire, nous avons travaillé avec les corps premiers $\mathbb{F}$p , où p >3. Pour ces corps premiers, si la caractéristique est supérieure à 3, l'équation de Weierstrass pour une courbe elliptique sur un corps fini premier noté $E(\mathbb{F}_p)$ peut être représentée par :

$$E : y^2=x^3+ax+b \,. \qquad (1.3)$$

où a et b $\in \mathbb{F}_p$.

Pour être utilisée en cryptographie, la condition nécessaire est que le discriminant du polynôme soit égal à 0. Cette condition garantie que, pour tout point de la courbe elliptique, passe une et une seule tangente.

$$f(x) = x^3 + ax + b, \Delta = 4a^3 + 27b^2 \neq 0. \qquad (1.4)$$

L'ensemble des points (x, y), dont les coordonnées x, y vérifient l'équation 1.1 et le point à l'infini noté ∞ sont sur la courbe et forment un groupe abélien additif ($E(\mathbb{F}_p)$,+):

$$(E(\mathbb{F}_p),+) = (x, y) \in \mathbb{F}_p.\mathbb{F}_p : y^2 - x^3 - ax - b = 0 \cup \{\infty\} \qquad (1.5)$$

Ce groupe est principalement constitué de deux opérations de base : le doublement de point (2*P*)et l'addition de points (*P*+*Q*) où *P* et *Q* sont deux points différents de la courbe.

Etant donnés $P=(x_p,y_p)$ et $Q=(x_q,y_q)$ deux points ($\neq\infty$) d'une courbe elliptique sur un corps fini $\mathbb{F}_p$ noté $E(\mathbb{F}_p)$. Géométriquement, l'addition de point (P + Q) avec P≠ Q consiste à prendre le symétrique du troisième point (P*Q) d'intersection de la droite PQ avec la courbe elliptique. Le doublement d'un point est le cas particulier d'addition où

P = Q, on prend alors le symétrique du point d'intersection de la tangente en P avec la courbe elliptique. Si P et Q sont symétriques par rapport à l'axe des *x*, dans ce cas la droite PQ coupe la courbe au point `a l'infini (qui est le zéro du groupe) et donc Q = -P.

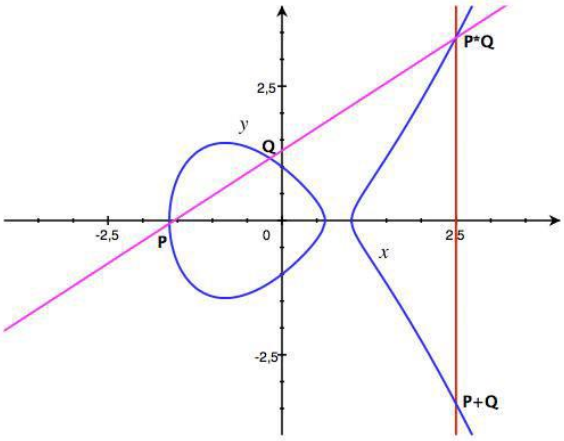

Figure 3.1 Addition de points [15]

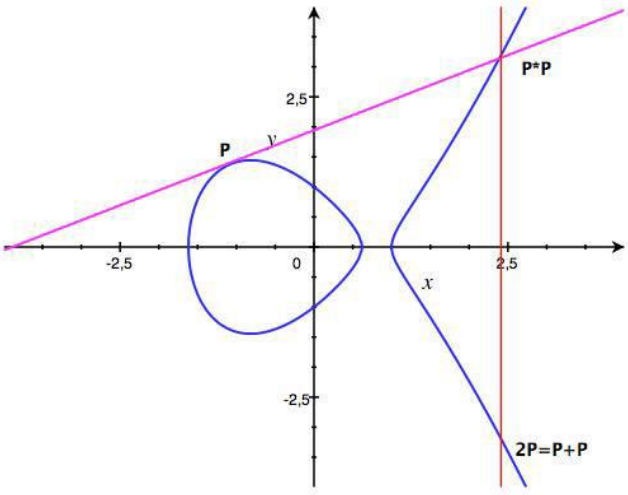

Figure 3.2 Doublement de point [15]

On va citer les trois algorithmes principaux qui se basent sur ECC (ECIES, ECDH, ECDSA).

3.4 ECIES

Le protocole ECIES [17] est un système de chiffrement à clé publique basé sur ECC (de l'anglais Elliptic Curve Integrated Encryption Scheme).

Supposons qu'Alice désire envoyer un message M à Bob d'une manière sécurisée, ils doivent d'abord disposer de toutes les informations suivantes :

- KDF (Fonction Dérivation Key) : une cryptographique fonction de hachage peut générer les clés de toute longueur.
- MAC (Message Authentification Code) : Code transmis avec les données dans le but d'assurer l'intégrité de ces dernières.
- SYM : Algorithme de chiffrement symétrique.
- $E(F_p)$: La courbe elliptique utilisée avec le point de générateur G dont ord1(G) = n.
- K_B : La clé publique de Bob $K_B = k_B.G$ où $k_B \in [1, n-1]$ est sa clé privée.

Pour chiffrer le message M, Alice doit effectuer des opérations suivantes :

1. Choisir un nombre entier $k \in [1, n-1]$ et calculer $R = k.G$.
2. Calculer $Z = k.K_B$.
3. Générer les clés (k_1, k_2) = KDF (abscisse(Z), R).
4. Chiffrer le message $C = SYM(k_1, M)$.
5. Générer le code MAC $t = MAC(k_2, C)$.
6. Envoyer (R, C, t) à Bob.

Pour déchiffrer le message (R, C, t), Bob doit effectuer des calculs ci-dessous :

1. Rejeter le message si $R \notin E(F_p)$.
2. Calculer $Z = k_B.R = k_B.k.G = k.K_B$.
3. Générer les clés (k_1, k_2) = KDF(abscisse(Z), R).
4. Générer le code MAC $t' = MAC(k_2, C)$.
5. Rejeter le message si $t' \neq t$.
6. Déchiffrer le message $M = SYM^{-1}(k_1, C)$.

3.5 ECDH

L'échange de clés Diffie-Hellman basé sur les courbes elliptiques, (de l'anglais Elliptic curve Diffie–Hellman, abrégé ECDH) .Dans le groupe associé à une courbe elliptique, le problème du logarithme discret est considéré comme difficile. On peut définir l'échange de clés Diffie-Hellman basé sur les courbe elliptiques.

Alice et Bob se mettent d'accord (publiquement) sur une courbe elliptique E (a,b,p), c'est-adire qu'ils choisissent une courbe elliptique $y^2 = x^3+ax+b \bmod p$. Ils se mettent aussi d'accord (toujours publiquement) sur un point P situé sur la courbe.

On définit nP = P + P + ….P (n fois) où l'opération + correspond à la somme de 2 points définie par le symétrique du troisième point d'intersection de la droite définie par

les 2 points originaux avec la courbe elliptique. Dans le cas où les deux points à ajouter sont identiques, on considère que la droite qui les joint est la tangente à la courbe elliptique passant par l'un d'entre eux. Un tel point est rationnel.

Secrètement, Alice choisit un entier d_A, et Bob un entier d_B. Alice envoie à Bob le point d_AP, et Bob envoie à Alice d_BP. Chacun de leur côté, ils sont capables de calculer $d_A(d_BP) = d_B(d_AP) = d_Ad_BP$ qui est un point de la courbe, et constitue leur clef secrète commune.

Si Eve a espionné leurs échanges, elle connaît E (a,b,p), P, d_AP, d_BP. Pour pouvoir calculer d_Ad_BP, il faut pouvoir calculer d_A connaissant P et d_AP. C'est ce que l'on appelle résoudre le logarithme discret sur une courbe elliptique. Or, actuellement, si les nombres sont suffisamment grands, on ne connaît pas de méthode efficace pour résoudre ce problème en un temps raisonnable**.**

3.6 ECDSA

Elliptic Curve Digital Signature Algorithm est un algorithme de signature numérique à clé publique, variante de DSA. Il fait appel à la cryptographie sur les courbes elliptiques. L'algorithme a été proposé en 1992 par Scott Vanstone. [18]

3.6.1 Algorithme

Soit une courbe de formule $y^2 = x^3 + ax + b$ c'est une courbe elliptique sur un corps d'entiers fini modulo p avec p un nombre premier et un point G de la courbe. L'ordre de G est n, le plus petit entier tel que nG donne *O* le point à l'infini de la courbe et élément neutre du groupe commutatif sur les points de la courbe.

1. **Préparation des clés**

- Choisir un entier s entre 1 et n − 1 qui sera la clé privée.
- Calculer $Q = sG$ en utilisant l'élément de la courbe elliptique.
- La clé publique est *Q* et la clé privée est s.

2. **Signature**

- Choisir de manière aléatoire un nombre k entre 1 et *n-1*
- Calculer $(i,j) = kG$

- Calculer x = i mod n ; si x = 0, aller à la première étape
- Calculer $y = k^{-1}(H(m) + sx)$ mod n où H(m) est le résultat d'un hachage cryptographique sur le message m à signer,
- Si $y=0$, aller à la première étape
- La signature est la paire (x, y).

3. Vérification

- Vérifier que Q est différent de O (le point à l'infini) et que Q appartient bien à la courbe elliptique
- Vérifier que nQ donne O
- Contrôler que x et y sont bien entre 1 et n-1
- Calculer $(i,j) = (H(m)y^{-1} \bmod n)G + (xy^{-1} \bmod n)Q$
- Vérifier que $x = i \bmod n$

D'après les études sur les systèmes de cryptographies RSA et ECC cités précédemment en conclus que les courbes elliptiques requièrent des clés plus courtes que RSA. Or l'usage de clés de petites tailles confère beaucoup d'avantages : les calculs sont plus rapides, la consommation électrique globale est diminuée, et l'espace mémoire est réduit Contrairement à RSA, les courbes elliptiques sont plus rapides pour les opérations privées que pour les opérations publiques. Ainsi, elles sont pratiques dans les environnements à fortes contraintes de ressources tels que les réseaux de capteurs, les cartes à puces, les téléphones mobiles, etc... Les ECCs sont appelées à remplacer progressivement RSA [19]. Le Tableau 1.1 présente une comparaison en termes de taille de clés des cryptosystèmes ECC et RSA pour le même niveau de sécurité.

Clés RSA (bits)	**Clés ECC(bits)**
1024	160
2048	224
3072	256
7680	384
5360	521

Table 3.1 Comparaison entre ECC et RSA (paramètres de NIST)

3.7 Choix de librairies cryptographiques:

Après l'étude des différents algorithmes de cryptographie l'étape suivante est d'aborder les librairies cryptographiques présentes dans le domaine de l'open source. Plusieurs facteurs sont importants dans le choix des bibliothèques a utilisées dans notre projet. Pour débuter on s'est concentrés à trouver la librairie gratuite avalable, facile à utiliser et peut consomante pour la mémoire, une librairie cryptographique spécialement conçue avec des systèmes embarqués, et implémentée en langage C pour être intégré dans Contiki qui lui-même est écrit en langage C.

Il apparait aussi que les librairies cryptographiques écrites en C sont très rare comparées à celles écrites en C^{++} et en Java . Or seule trois open source librairies conçues pour les systèmes embarqués sont gratuits (RelicToolKit, LibTomCrypt et Miracl).

3.8 Les librairies cryptographiques:

3.8.1 Miracl:

Bibliothèque d'entier multi-précision et d'arithmétique relationnelle écrite en C, elle est considérée comme le SDK open source standard pour la cryptographie à courbe elliptique (ECC) [20].

La bibliothèque MIRACL comprend plus de 100 routines couvrant tous les aspects de l'arithmétique multi-précision.

Toutes les routines ont été entièrement optimisées pour la vitesse et l'efficacité, tout en restant standard, portable C. Cependant, d'autres options de langage d'assemblage rapide pour certaines routines critiques sont également incluses, en particulier pour la populaire gamme de processeurs Intel 80x86. Une interface C++ est également fournie avec le code source complet. Elle a été proposée à l'origine par D. Matula et P. Kornerup.

3.8.2 LibTomCtypt:

LibTomCrypt est une bibliothèque cryptographique ISO C portative destinée à être un ensemble d'outils pour les cryptographes qui conçoivent des cryptosystèmes. Il prend en charge les chiffrements symétriques, les hachages unidirectionnels, pseudo-aléatoires générateurs de nombres, cryptographie à clé publique (via PKCS # 1 RSA, DH ou ECCDH).[21]

La bibliothèque a été conçue de telle sorte que de nouveaux chiffrements / hachages / PRNG peuvent être ajoutés au moment de l'exécution ,et l'API existante (et les fonctions d'API auxiliaires) peuvent utiliser automatiquement les nouvelles conceptions. Elle a été conçue par Tom St Denis.

3.8.3 RelicToolKit:

RELIC est une meta-toolkit à outils cryptographique moderne mettant l'accent sur l'efficacité et la flexibilité. RELIC peut être utilisé pour créer toolkit cryptographiques efficaces et utilisables, adaptées à des niveaux de sécurité spécifiques et à des choix algorithmiques. [22]

RELIC est un projet en cours et des fonctionnalités qui seront ajoutées sur demande. L'objectif est de fournir :

- Facilité de portabilité et inclusion de code dépendant de l'architecture
- Expérimentation simple avec des implémentations alternatives
- Tests et benchmarks pour chaque fonction implémentée
- Configuration flexible
- Efficacité maximale

Quant aux algorithmes, RELIC met en œuvre à ce jour :

- Arithmétique d'entiers à précision multiple
- Arithmétique des champs premiers et binaires

- Courbes elliptiques sur les champs premiers et binaires (courbes NIST et courbes favorables à l'appariement)
- Cartes bilinéaires et champs d'extension associés
- Protocoles cryptographiques

Elle est développée par Diego de Freitas Aranha.

3.9 Comparaison entre les bibliothèques cryptographiques :

Les librairies cryptographiques	Avantages	Inconvénients
Miracl	• référencée comme un outil cryptographique fournissant une facilité à implémenté des algorithmes pour les applications de sécurité dans le monde réel. • Fournis des code compacts, rapides et efficaces . • Présente une performance élevé dans n'importe quel processeur ou platforme	• comme seul inconvénient elle ne supporte pas la platforme Msp430 qui est en base de notre projet. Puisque Cooja, le simulateur de réseaux de capteurs sans-fil du projet Contiki (utilisant MSPSim pour l'émulation des noeuds MSP430 des réseaux),
LibTomCrypt	• Facile à comprendre et à modifier . • offre plus de flexibilité dans l'ajout ou de suppression des différentes parties. • ses bibliothèques mathématiques sont facilement changeable.	• offre des fonctionnalités limitées comparait à Relic. • Librairie ancienne et n'est pas régulièrement mis à jours.
RelicToolKit	• Spécialement écrite pour être utilisé dans les réseaux de	• Documentation détaillé est très insuffisante.

	capteur sans fils. • Très flexible est fournit une large gamme de moderne fonction cryptographique. • Supporte plusieurs librairies mathématiques tel que Relic-easy et GnuMp. • Nouvelle librairie avec de nouveaux algorithmes inclus. • Supporte trois méthodes d'allocation mémoire (Statique, Stack et Dynamique) . • Supporte multithreading. • Supporte le teste et le benchmarking de différents composants.	• Quelques détailles de configuration sont difficiles à comprendre.

Table 3.2 Comparaison entre les bibliothèques cryptographiques

On se basant sur la comparaison faite dans le tableau ci-dessus on a opté pour la bibliothèque RelicToolKit Pour ses avantages et sa capacité de supporter la plateforme Msp430 qui est en base de notre projet.

3.10 Conclusion :

Dans ce chapitre nous avons cités les algorithmes de cryptographie asymétriques basés sur les courbes elliptiques et les bibliothèques de cryptographie qui les implémentes, et nous avons choisis la bibliothèque RelicToolKit pour être utilisé dans les capteurs et évaluer l'énergie consommée par ces derniers, qui sera le travail désigné dans le chapitre suivant.

CHAPITRE 4:

Evaluation des performances des Algorithmes Cryptographiques

4.1 Introduction

Dans ce chapitre on va présenter les résultats des tests réalisées en utilisant la cryptographie a clé public basé sur les courbes elliptiques avec Contiki. Initialement on va présenter l'environnement de simulation et la plateforme sélectionnée, ensuite nous expliquerons les différentes méthodes de mesures de : temps d'exécution, la RAM, la ROM et l'énergie consommée par le nœud capteur et finalement on va citer les problèmes rencontrées dans le système Contiki et discuté les résultats obtenus.

4.2 Environnement de simulation

Pour effectuer nos tests d'évaluation, nous avons utilisé Cooja Contiki 2.7. Cooja (*Contiki os java simulator*) a été inventé en 2002 à l'institut de recherche suédois SICS ICT. Cooja est reconnu pour être un simulateur bien développé, destiné à la simulation des réseaux de capteurs (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*.

Etant destiné à la simulation des réseaux 6LoWPANs, Cooja implémente une pile protocolaire TCP/IP complète et adaptée pour les environnements capteurs (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. [1]

4.3 Sélection de plateforme

Dans les simulations effectuées nous avons utilisé la plateforme TMote Sky [23] pour le nœuds capteur. Cette plateforme est assez célèbre et elle est utilisée très souvent dans les travaux de recherche ciblant les réseaux de capteurs.

Figure 4.1 Capteur Tmote-Sky de Moteiv[23]

Hardware	Sonde Capteur	Batterie
8MHz Texas Instruments MSP430 10k RAM, 48k Flash 250kbps 2.4 GHz IEEE 802.15.4 Radio Antenne intégrée USB interface	Température Humidité Lumière	2 piles AA 21-23 mA en mode réception 19-21 mA en mode émission 5-21 uA deep sleep mode Logiciel de gestion

Table 4.1 Caractéristiques des capteurs Tmote-Sky de Moteiv [23]

4.4 Bibliothèque Relic Tool Kit

Relic n'est une bibliothèque de contiki mais on peut l'intégrer pour être compatible avec Tmote-Sky en procédant par les étapes suivantes : [22]

Après avoir télécharger relic-toolkit-0.4.0 de son site official : [22]

La procédure de construction requis : CMake[24] cross-platform build system et le compilateur MSPGCC .

4.4.1 Compilation

Dans le terminal de Contiki on procède ainsi :

tar xzvf relic-toolkit-0.4.0 .tar.gz

ça va créer le répertoire nommé relic-toolkit-0.4.0, ensuite en crée le répertoire cible:

mkdir -p relic-target en exécute cmake dans le répertoire cible en utilisant (MSP430 presets):cd relic-target ../relic-version/preset/msp-pbc-80.sh ../relic-version, le preset va

chercher le compilateur msp430-gcc qui lui aussi doit être installer en contiki à l'aide de la commande suivante: *sudo apt-get install gcc-msp430* dans notre chemin d'accès.

Par défaut la plateforme concerné est msp430f1611 qui le microcontrôleur de TmoteSky.

Finalement on compile la bibliothèque par la commande : *make*.

4.5 Méthode de mesure

En raison de la limitation des ressources de réseaux de capteurs, il est essentiel démesurer l'utilisation de la mémoire, le temps de chiffrement et déchiffrement de la sécurité mis en œuvre. Dans cette section, nous décrivons les paramètres de mesures tels que ROM, RAM et le temps des différents algorithmes cryptographiques, et l'énergie consommé par le nœud de capteur dans lequel sont implémentés ces algorithmes. Ces paramètres sont utilisés dans notre évaluation plus tard.

4.5.1 Délai de traitement

Pour mesurer le temps écoulé entre les événements dans les différents programmes, nous avons utilisés la bibliothèque compteur de temps (counter clock)en temps réel (RTIMER) comme suite:

```
t1=RTIMER_NOW();
cp_ecdsa_gen(d,q);
t2=RTIMER_NOW();
printf("Ticks-gen= %u\n",t2-t1);
```

Sachant que le temps résultant est mesuré en milliseconde.

4.5.2 ROM

En raison des ressources limitées de la mémoire de nœuds de capteurs, la ROM Flash utilisé est un paramètre important qui doit être pris en compte dans la conception d'un logiciel de sécurité dans les réseaux de capteurs. Les bibliothèques cryptographiques classiques sont trop grandes pour les dispositifs de l'embarqué. En fait la taille du code de chiffrement et déchiffrement choisi par la suite doit être de l'ordre de quelques kilo-octets. C'est pourquoi nous sommes intéressés à comparer les tailles des mémoires ROM des différents programmes utilisées.

4.5.3 RAM

La Mémoire RAM est encore plus limitée que la mémoire ROM Flash. La plateforme TmoteSky (qui sera expliqué à la suite) fonctionne à base de microcontrôleur MSP430 avec une mémoire volatile de 10 KB seulement.

Pour connaitre l'exigence de la RAM et Le ROM en utilise l'outil msp430-size qui se trouve en msp430-gcc. En premier temps cet outil est utilisé pour mesurer la taille de code utilisé par contiki. Ensuite le code PKC est intégré avec, et la taille de code est mesurée une deuxième fois.

Finalement la taille de code requis par la fonction cryptographique est estimée, et peut s'écrire mathématiquement comme :

ROM/RAM exigée = la taille de code compilé avec PKC – la taille de code compilé normale.

4.5.4 L'énergie

Comme tout RCSFs a basse consommation l'énergie estimée se mesuré en micro-joules, mais en cryptographie a clé publique complexe avec de longues équations mathématiques la consommation est mesuré en mill-joules, pour se faire Contiki a mis au point un module nommé Energest.

Energest est un logiciel d'estimation de l'énergie consommée par les composants de nœud de capteur qui sont : L'unité de traitement (CPU), l'unité de communication (Transceiver : transmitter-receiver), et le temps pendant lequel le capteur était en mode de faible consommation d'énergie LPM (Low Power Mode).la table ci-dessous démontre les valeurs standards des composantsdu TmoteSky cités précédemment [24]

Composant	Consommation actuelle (mA)
CPU	1.8
LPM	0.0545
Radio TX	17.7
Radio RX	20
Tension d'alimentation	3 V

Table 4.2 Valeurs standards des composants de TmoteSky

On estime l'énergie consommée comme suite :

Energy_Consommée = ((1.8 * diff.cpu + 0.0545 * diff.lpm + 20* diff.listen + 17.7* diff.transmit) * 3 / RTIMER_SECOND)); (4.1)

Où 3 désigne 3 voltes (le voltage de nœud de capteur Tmote Sky).
RTIMER_SECOND représente le nombre de ticks(nombre d'impulsions d'horloge) per seconde, variable utilisé en Contiki.

4.6 Problèmes de Contiki

Comme il a été mentionné dans la thèse de doctorat : Evaluation et amélioration des plates-formes logicielles pour réseaux de capteurs sans-fil, pour optimiser la qualité de service et l'énergie par Kévin Roussel. J'ai rencontré dans mon travail les problèmes suivants [26] :

- À l'exception du code source du système et des exemples d'applications fournis, il n'y a quasiment aucune documentation.
- On regrettera notamment l'absence totale de documents techniques de référence, où l'architecture générale du système, les choix de conception et d'implantation seraient expliqués ; une telle documentation de référence représente un outil essential pour réellement comprendre une plate-forme logicielle, et ainsi permettre aux développeurs débutants de devenir rapidement efficaces dans leur travail.
- Il y a également très peu de documents d'introduction (« tutoriels ») : le seul document d'initiation officiel est la page "get started" du site Web du projet Contiki. Celle-ci montre comment télécharger et utiliser la distribution Linux dédiée au test du système, nommée "Instant Contiki", pour effectuer rapidement des simulations de réseaux de capteurs sans-fil — grâce à l'utilisation du simulateur Cooja fourni par le projet Contiki—puis télécharger des programmes d'exemple sur du matériel (des motes de type Zolertia Z1).
- Aucun document n'est disponible pour montrer et (plus important encore) expliquer aux développeurs les nombreuses fonctionnalités de Cooja.
- Aucun document pour détailler comment programmer des applications avec Contiki .

- Aucune introduction signalant quelles sont les spécificités du développement embarqué sur des systèmes aussi contraints que les motes constituant les réseaux de capteurs sans-fil (par opposition au développement « classique » sur PC).
- Enfin, l'absence de documentation sur le fonctionnement interne et sur les méthodes éventuelles pour adapter le système à ses propres besoins, est particulièrement regrettable pour qui souhaite mener des travaux concrets de recherche et de développement avec, et surtout dans ce système — or, de telles initiatives d'exploitation « avancée » sont appelées à être relativement nombreuses, étant donné le statut d'OS pour WSN de référence que possède actuellement Contiki.
- La principale source de documentation, en matière de développement sous Contiki, est la mailing-list "contiki-developers". Toute personne souhaitant developer sur cette plate-forme logicielle ne peut espérer atteindre un quelconque but sérieux sans souscrire à cette liste, et demander de l'aide et des renseignements à ses membres. Bien que cette liste soit une source riche d'informations, soit réactive, et en général bien disposée à l'égard des nouveaux venus, on peut difficilement considérer qu'elle remplace de façon satisfaisante le manque de documentation technique et de tutoriels.
- Ajoutons également que, si de nombreux exemples d'applications conçues avec le système Contiki sont fournis avec le code source du système, il n'y a par contre aucun exemple de code destiné à s'insérer dans le système lui-même.
- Le développement au sein du système Contiki, par exemple pour l'adapter à ses besoins ou y rajouter des fonctionnalités, n'en est ainsi qu'encore plus difficile à apprendre et à maîtriser.

4.7 Evaluation des performances :

Après avoir mis au point les différentes méthodes de mesure, on va les utilisées pour évaluer les performances des différents cryptosystème : ECDH, ECDSA et ECIES.

4.7.1 Paramètres d'implémentation

La taille de clés utilisée pour chaque cryptosystème est 160 bits qui est utilisée pour le niveau de sécurité 80.

4.7.2 Occupation de mémoire (ROM/RAM)

La table 4.3 représente les résultats obtenus après l'exécution des algorithmes cryptographiques asymétrique : ECDH, ECDSA et ECIES en cas de génération de clés publique et privé par rapport à l'occupation de mémoire (ROM et RAM)

	Génération de clé publique et privé	
	RAM	**ROM**
ECDH	748	21078
ECDSA	748	21078
ECIES	748	21078

Table 4.3 Occupation de mémoire pour ECDH, ECDSA et ECIES (en Octet)

La Figure 4.2 nous montre le résultat d'évaluation de la mémoire pour les des différents cryptosystème : ECDH, ECDSA et ECIES.

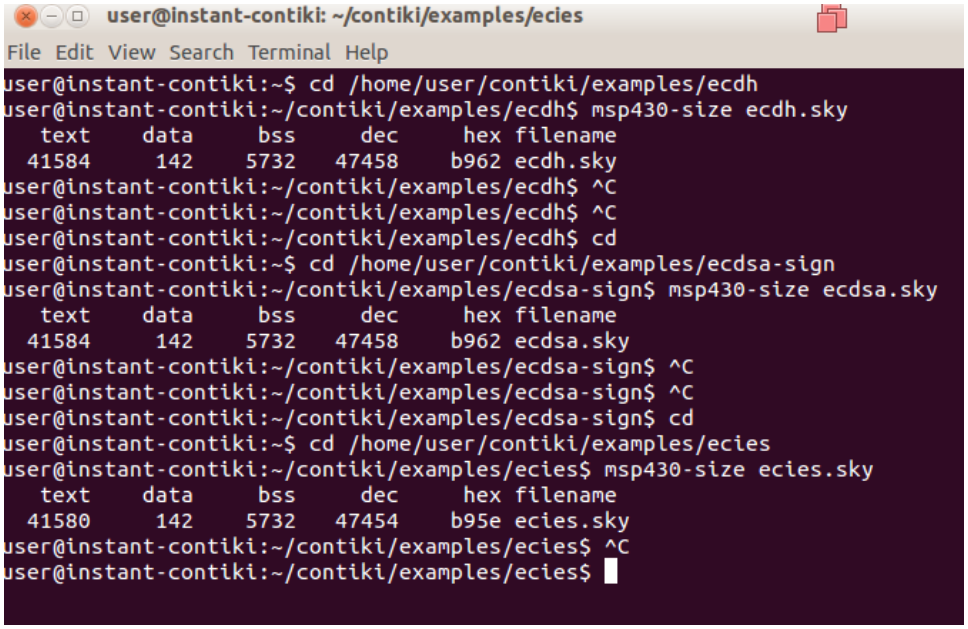

Figure 4.2 Evaluation de l'occupation de mémoire par ECDH, ECDSA et ECIES en cas de génération de clé publique et privé.

La table 4.4 représente les résultats obtenus après l'exécution de l'algorithme ECDH par rapport à l'occupation de mémoire (ROM et RAM)

	RAM	**ROM**
ECDH	748	22124

Table 4.4 Occupation de mémoire pour ECDH (en Octet)

La table 4.5 représente les résultats obtenus après l'exécution de l'algorithme ECDSA, par rapport à l'occupation de mémoire (ROM et RAM) dans les deux opérations : Signature et Vérification.

	Signature		**Vérification**	
	RAM	**ROM**	**RAM**	**ROM**
ECDSA	748	22038	748	22828

Table 4.5 Occupation de mémoire pour ECDSA (en Octet)

La table 4.6 représente les résultats obtenus après l'exécution de l'algorithme ECIES, par rapport à l'occupation de mémoire (ROM et RAM) dans les deux opérations : Chiffrement et Déchiffrement.

	Chiffrement		**Déchiffrement**	
	RAM	**ROM**	**RAM**	**ROM**
ECIES	336	27504	336	27504

Table 4.6 Occupation de mémoire pour ECIES (en Octet)

4.7.3 Consommation d'énergie

Pour estimer l'énergie consommée on utilise l'application powertrace.c par l'ajout de l'instruction *powertrace_start(CLOCK_SECOND * 10);* dans le code source.[27]

La table 4.7 représente les résultats obtenus après l'exécution des algorithmes cryptographiques asymétriques : ECDH, ECDSA et ECIES en cas de génération de clé publique et privé par rapport à la consommation d'énergie.

Unité de calcule en milli joule	**Consommation d'énergie**
	Génération de clé publique et privé
ECDH	71.45
ECDSA	71.45
ECIES	71.45

Table 4.7 Consommation d'énergie pour ECDH, ECDSA, et ECIES

La table 4.8 représente les résultats obtenus après l'exécution de l'algorithme ECDH, par rapport à la consommation d'énergie.

Unité de calcule en milli joule	**Consommation d'énergie**
ECDH	71.49625

Table 4.8 Consommation d'énergie pour ECDH

La table 4.9 représente les résultats obtenus après l'exécution de l'algorithme ECDSA, par rapport à consommation d'énergie des deux opérations : Signature et Vérification.

Unité de calcule en milli joule	**Consommation d'énergie**	
	Signature	**Vérification**
ECDSA	71.49117	71.45245

Table 4.9 Consommation d'énergie pour ECDSA

La table 4.10 représente les résultats obtenus après l'exécution de l'algorithme ECIES, par rapport à consommation d'énergie dans les deux opérations : Chiffrement et Déchiffrement.

Unité de calcule en milli joule	**Consommation d'énergie**	
	Chiffrement	**Déchiffrement**
ECIES	71,49140	71,45145

Table 4.10 Consommation d'énergie pour ECIES

4.7.4 Temps de traitement

La table 4.11 représente les résultats obtenus après l'exécution des algorithmes : ECDH, ECDSA et ECIES en cas de génération de clés publique et privé par rapport au temps écoulé durant l'exécution.

Unité de calcule en milli seconde	Temps Ecoulé
	Génération de clé publique et privé
ECDH	1536.010
ECDSA	1535.43
ECIES	1533.32

Table 4.11 Temps écoulé durant l'exécution de : ECDH, ECDSA, et ECIES

La table 4.12 représente les résultats obtenus après l'exécution de l'algorithmes : ECDH par rapport au temps écoulé durant l'exécution.

Unité de calcule en milli seconde	Temps Ecoulé
ECDH	1495.91

Table 4.12 Temps écoulé durant l'exécution de : ECDH

La table 4.13 représente les résultats obtenus après l'exécution de l'algorithmes : ECDSA par rapport au temps écoulé durant l'exécution des deux opérations : Signature et Vérification.

Unité de calcule en milli seconde	Temps Ecoulé	
	Signature	Vérification
ECDSA	1791.68	1497.37

Table 4.13 Temps écoulé durant l'exécution de : ECDSA

La table 4.14 représente les résultats obtenus après l'exécution de l'algorithmes : ECIES par rapport au temps écoulé durant l'exécution des deux opérations : Chiffrement et Déchiffrement.

Unité de calcule en milli seconde	Temps Ecoulé	
	Chiffrement	Déchiffrement
ECIES	864.99	864.99

Table 4.14 Temps écoulé durant l'exécution de : ECDSA

4.8 Discussion

La problématique qui s'impose dan RCSFs est sans doute : l'énergie, la mémoire et le temps de traitement et la sécurisation des données.

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.
Pour cela, on a évalué chaque algorithme cryptographique en terme de : l'espace de mémoire exigé dans la RAM et la ROM, le temps de traitement pour chaque phase (génération de clé publique et privé, chiffrement, déchiffrement, signature, et vérification), et la consommation d'énergie.

4.8.1 Occupation de la mémoire ROM Flash

- D'après la table 4.3, dans la phase de génération de clé publique et privé on remarque que tous les PKCs exigent la même valeur de ROM.
- D'après les tables 4.4, 4.5, 4.6 on constate que les valeurs de ROM sont comprises entre 22038 et 22828et ainsi c'est le ECDH qui exige la moindre valeur de ROM que ECDSA et ECIES.

4.8.2 Occupation de la mémoire RAM

- D'après la table 4.3, dans la phase de génération de clé publique et privé, on Remarque que tous les PKCs exige la même valeur de RAM.
- D'après les tables 4.4, 4.5, 4.6 on constate ECIES exige la moindre valeur de RAM que ECDSA et ECDH.

4.8.3 La consumation d'énergie

- D'après les tables 4.7, 4.8, 4.9, 4.10 on remarque que la consommation d'énergie par les PKCs est presque similaire.

4.8.4 Le temps de traitement

- D'après la table 4.11, dans la phase de génération de clé publique et privé, on observe que le temps écoulé est presque le même pour tous les PKCs
- D'après les tables 4.12, 4.13, 4.14 on conclus que le ECIES prend le moindre temps d'exécution par rapport aux ECDH et ECDSA.

4.9 Conclusion

Dans ce chapitre on a présentés les résultats des tests réalisées en utilisant la cryptographie a clé public basé sur les courbes elliptiques avec Contiki .Initialement on a présenté l'environnement de simulation et la plateforme sélectionnée ,ensuite nous avons expliqué les différentes méthodes de mesures de : temps d'exécution, la RAM, la ROM et l'énergie consommée par le nœud capteur et on a cité les problèmes rencontrées dans le système Contiki et finalement on a discuté les résultats obtenues. Et d'après les résultats obtenus on a opté pour ECIES comme l'algorithme cryptographique le mieux convenable pour les RCSFs à cause de sa faible exigence de ressources, tel que la RAM, le temps de traitement, et l'énergie, sauf qu'il reste le problème de sa consommation de la ROM.

CONCLUSION GENERALE

Depuis longtemps, l'être humain a cherché à concrétiser son confort, son bien être, sa santé, sa sécurité, et l'innovation dans sa vie. Et parmi les innovations qu'il a atteint les RCSFs qui peuvent contenir des millions voire des milliards de nœud capteur. De nos jours, les capteurs sont partout : dans les rues, dans les maisons, dans les bureaux, dans la voiture, aux frontières d'un pays, sous l'eau, dans les barrages et dans les forets. Les avancées des technologies sans fil et de la micro-électronique ont poussé cette évolution. Actuellement, les nœuds capteurs sans fil génèrent des informations (de surveillance, de commande, etc…) et nécessitent des protocoles pour assurer leur organisation et sécurisation.

Dans ce mémoire, nous nous sommes intéressés aux problèmes de sécurité dans les RCSFs, nous cherchons à définir des mécanismes et solutions peu couteux en énergie pour les ces derniers qui prennent en compte la relative faiblesse de défense d'un réseau autonome.

Ainsi nous avons cités les algorithmes cryptographiques asymétriques basés sur les courbes elliptiques et les bibliothèques cryptographiques qui les implémentent, et nous avons choisis la bibliothèque Relic Tool Kit pour être utilisé dans les capteurs et évaluer l'occupation d'espace mémoire, le temps écoulé et l'énergie consommée par ces derniers.

Dans cette optique nous avons simulé des algorithmes cryptographiques asymétriques basés sur les courbes elliptiques peu gourmandes en énergie qui sont ECDH, ECDSA et ECIES on utilisant le système d'exploitation Contiki qui est le plus adapté au nœud capteur et distribué avec l'émulateur Cooja pour tester la validité des codes produits et faire une étude comparative entre ces derniers et qui a aboutit au choix de l'algorithme ECIES comme l'algorithme le plus adapté et convenable aux caractéristique de nœud capteur.

Notre travail réalisé est une étape initiative pour proposer une nouvelle approche de sécurisation des protocoles de communication dans les RCSFs en utilisant les algorithmes cryptographiques à bas coût, celui-ci est la perspective de notre travail.

.

Bibliographies

[1] Somia Sahraoui,,Mechanisms de sécurité pour l'intégration des RCSFs à l'IoT (Internet of Things),Doctorat, Université de Batna 2, Année 2016.

[2] Cristian Duran-Faundez, thèse " Transmission d'images sur les réseaux de capteurs sans fil sous la contrainte de l'énergie", Université Henri Poincaré, Nancy 1, Année 2009

[3] Applications des RCSF [En ligne] https://moodle.utc.fr/file.php/498/SupportWeb/co/Module_RCSF_34.html consulté le : 04/04/2018.

[4] Contiki [En linge] ,https://fr.wikipedia.org/wiki/Contiki consulté le **:** 04/04/2018.

[5] Y. Wang, G. Attebury, B. Ramamurthy, A survey of security issues in Wireless Sensor Networks, IEEE communications surveys & tutorials 8 (2) (2006) 2-21.

[6] C. Karlof, D. Wagner, Secure routing in wireless sensor networks: attacks and Countermeasures Ad Hoc Networks 1 (2) (2003) 293–315.

[7] J. Lee, K. Kapitanova, S. H. Son, The price of security in wireless sensor networks, Computer Networks 54 (17) (2010) 2967–2978.

[8] 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.

[9] 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.

[10] Manel, Majdoub, and Eltaief Hamdi. "A localization algorithm using a mobile beacon in wireless sensor networks." In *Computer & Information Technology (GSCIT), 2014 Global Summit on*, pp. 1-5. IEEE, 2014.

[11] 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.

[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] A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, D.E. Culler, SPINS: Security Protocols for Sensor Networks, Wireless Network 8 (2002) 521–34.

[14] D. SOW, Courbes elliptiques, Cryptographie à clés publiques et Protocoles d'échange de clés, Doctorale, Dakar, 2013.

[15] Faye. Youssou, Algorithmes d'authentification et de cryptographie efficaces pour les réseaux de capteurs sans fil,(Doctorat) l'Université de Franche-Comté et de Université Cheikh Anta Diop (Sénégal), 18/09/2014.

[16] Faye, Youssou and Guyennet, Hervé and Niang, Ibrahima. A User Authentication-Based Probabilistic Risk Approach for Wireless Sensor Networks.In iWMANET 2012, Int. Workshop on Mobile Ad-Hoc Wireless Networks, conjointly organized with IEEE International Conference on Selected Topics(iCOST) 2012, Avignon, France, pages 124-129, IEEE Computer Society, Juillet 2012 .

[17] Y.SHOU, Cryptographie sur les courbes elliptiques et tolérance aux pannes dans les réseaux de capteurs, Doctorat, l'Université de Franche-Comté, 2014

[18] D. Johnson, A. Menezes, and S. Vanstone. The elliptic curve digital signature algorithm (ECDSA). International Journal of Information Security, 1, pp. 36–63, 2001.

[19] S.Vanstone. Ecc holds key to next-gen cryptography. *Rapport technique, Certicom*, 2004

[20] Miracl [En linge] https://github.com/miracl/MIRACL consulté le:19/03/2018.

[21] Hassan. Razi, Qamar Toheed, Asymmetric-Key Cryptography for Contiki, (Master), University of Gothenburg ,juillet 2010.

[22] Relic Toolkit [En linge] https://code.google.com/archive/p/relic-toolkit/ consulté le : 19/03/2018.

[23] Tmote sky [En linge] http://www.moteiv.com/products/docs/tmoteskydatasheet.pdf. consulté le: 07/06/2018.

[24] Cmake [En linge] https://cmake.org/ consulté le:19/03/2018.

[25] Kevin Roussel. Evaluation et amélioration des plates-formes logicielles pour les réseaux de capteurs sans-fil, pour optimiser la qualité de service et l'énergie, Doctorat, de l'Université de Lorraine, le 3 juin 2016.

[26] Powertrace [En linge] http://thingschat.blogspot.co.uk/2015/04/contiki-os-using-powertrace-and.html consulté le :29/04/2018.

نبذة مختصرة

أصبحت شبكات الاستشعار اللاسلكية (RCSFs) مكونًا أساسيًا في العديد من التطبيقات التي تمتد من مجرد مراقبة درجة حرارة الغابات الى المراقبة في العديد من محطات الطاقة، ومع الاعتماد المتزايد على هذا النوع من الشبكات وكذا ارتباطها بالأنترنت أصبح من الضروري اعطاء اولويات امنية مشددة لضمان السلوك الحسن لعقدة الاستشعار في هذه الشبكات، مما يتطلب استعمال التشفير بالمفتاح العام وبالضبط التي تعتمد على المنحنى الإهليليجي.

في هذه المذكرة انواع مختلفة من الخوارزميات في هذا النوع من التشفير تمت محاكاتها بالاستعمال النظام Contiki الأكثر شيوعا في RCSFs. رغم ان هذا التشفير في RCSFs له جانب سلبي من ناحية استهلاك الطاقة، ويتطلب حجما كبيرا من الذاكرتين الحية والميتة. بعد تحليل مختلف مكتبات التشفير تم اخييار Relic و نقلها الى Contiki و محاكات مختلف الخوارزميات الموجودة بها في Sky لتقييمها. حيث أظهرت النتائج إمكانية استعمال التشفير بالمفتاح العام حيث يتطلب جهدا اقل لاستخدامه في .Msp430 عوامل مثل سرعة المعالج وحجم الذاكرة تؤدي الى نتائج أفضل، كما ان استعمال العديد من المكتبات الرياضة المحسنة مع هذه الخوارزميات يمكن ان يساعد بشكل كبير في زيادة الأداء.

كلمات البحث : شبكات الاستشعار اللاسلكية، الأمن، التشفير بالمفتاح العام.

Abstract

Wireless Sensor Networks have become a core component in much diverse application range which extends from just a forest temperature monitoring to monitoring in many power plants. With this increased dependency on WSN and its association to current internet, hardened security primitives are required to ensure the correct behaviour of sensor nodes on its own and as a whole network. Public-Key Cryptography (PKC) was considered too expensive for WSN but it all changed due to the advancements in software and hardware prototypes.

In this thesis different PKC algorithms based on eliptic curve have been analysed, that can be used with Contiki. Contiki is a new but popular operating system used in WSN. Using PKC in wireless sensor networks has its own negative aspects like more energy utilization, requirement of more RAM and ROM space. To investigate the feasibility of public key cryptography, different cryptographic libraries were analysed. After analyse of differents cryptographic libraries . We selected Relic that was ported to Contiki,. For evaluation of algorithms used in this library with Contiki, simulation based Tmote Sky is used. Results have shown that PKC is possible, and fewer efforts are required to use it with Msp430. Factors like processor speed and RAM size lead to better results in case of such integrations. It is also observed that use of many mathematical optimizations provided with these algorithms can significantly help in performance increase.

Key words: wireless sensor networks, security, PKC.

Résumé

Les réseaux de capteurs sans fil sont devenus un composant essentiel dans des différentes applications, allant de la simple surveillance de la température des forêts à la surveillance dans de nombreuses centrales électriques. Avec cette dépendance accrue sur WSN et son association à l'Internet, des primitives de sécurité renforcées sont nécessaires pour assurer le comportement correct des nœuds de capteurs sur son propre réseau et sur l'ensemble du réseau. La cryptographie à clé publique (PKC) était considérée comme trop chère pour WSN, mais tout a changé en raison des progrès réalisés dans les prototypes de logiciels et de matériel.

Dans ce mémoire différentes algorithmes PKC basés sur les courbes elliptiques ont été analysées, qui peuvent être utilisées avec Contiki. Contiki est un nouveau système d'exploitation populaire et utilisé dans WSN. L'utilisation de la PKC dans les réseaux de capteurs sans fil a ses propres aspects négatifs, comme l'utilisation de plus d'énergie, l'exigence de plus d'espace RAM et ROM. Pour étudier la faisabilité de la PKC, différentes bibliothèques cryptographiques ont été analysées, parmi lesquelles, Relic, a été sélectionnées pour évaluation, cette bibliothèque a été portées sur Contiki. Pour l'évaluation de cette librairie avec Contiki, on utilise la simulation Tmote Sky. Les résultats ont montré que la PKC est possible et que moins d'efforts sont nécessaires pour l'utiliser avec Msp430. Des facteurs tels que la vitesse du processeur et la taille de la RAM conduisent à de meilleurs résultats dans le cas de telles intégrations. Il est également observé que l'utilisation de nombreuses optimisations mathématiques fournies avec ces algorithmes peut considérablement aider à augmenter les performances.

Mots clés : réseaux de capteurs sans fil, sécurité, PKC.